

---

## Low Power 32-bit Microcontroller with Embedded Flash

---

- 3.3V Operation
- ACPI Compliant
- LPC Interface
  - LPC I/O Cycles Decoded
- VTR (standby) and VBAT (Power Planes)
  - Low Standby Current in Sleep Mode
- Configuration Register Set
  - Compatible with ISA Plug-and-Play Standard
  - EC-Programmable Base Address
- ARC-625D Embedded Controller (EC)
  - 16 KB Single Cycle 32-bit Wide Dual-ported SRAM, Accessible as Closely Coupled Data Memory and Instruction Memory
  - 32 x 32 x 64 Fast Multiply
  - Divide Assist and Saturation Arithmetic
  - Maskable Interrupt Aggregator/Accelerator Interface
  - Maskable Hardware Wake-Up Events
  - Sleep mode
  - JTAG Debug Port, Includes JTAG Master
  - MCU Serial Debug Port
  - 8-Channel DMA Interface Supports SMBus Controllers and EC/Host GP-SPI Controllers
  - Delay Register
  - Boot ROM
- Embedded Flash
  - 192 KB user space 32-bit Access, 30 ns Access Time, 10 K Cycles Endurance
  - 1 KB EEPROM Emulation, 40 ns Access Time, 250 K Cycles Endurance
  - Programmable by LPC, EC and JTAG Interfaces
  - Flash Security Enhancements
    - 4K Boot Block Protection
    - Direct JTAG and Direct LPC-protected (2) Pages at or Near Top of Memory for Password Protection
- Legacy Support
  - Fast GATEA20 & Fast CPU\_RESET
- System to EC Message Interface
  - 8042 Style Host Interface
  - Embedded Memory Interface
    - Host Serial or Parallel IRQ Source
    - Provides Two Windows to On-Chip SRAM for Host Access
- Two Register Mailbox Command Interface
- Host Access of Virtual Registers Without EC Intervention
- Mailbox Registers Interface
  - Thirty-two 8-Bit Scratch Registers
  - Two Register Mailbox Command Interface
  - Two Register SMI Source Interface
- ACPI Embedded Controller Interface
  - Four Instances
  - 1 or 4 Byte Full Duplex Bidirectional Data Transfer Capable
- ACPI Power Management Interface
  - SCI Event-Generating Functions
- BIOS Debug Port
  - ISA Port 80 Plug-in Card Emulation
  - 2 Instances
  - Time Stamping Option
- Battery Backed Resources
  - Power-Fail Status Register
  - 32 kHz Clock Generator
  - Week Alarm Timer Interface with Programmable Wake-up from 1ms to 45 Days
  - VBAT-Powered Control Interface
    - 6 Latched Inputs
    - GPIO Capable
  - VBAT-Backed 64 Byte Memory
- Three EC-based SMBus 2.0 Host Controllers
  - Allows Master or Dual Slave Operation
  - Controllers are Fully Operational on Standby Power
  - DMA-driven I<sup>2</sup>C Network Layer Hardware
  - I<sup>2</sup>C Datalink Compatibility Mode
  - Multi-Master Capable
  - Supports Clock Stretching
  - Programmable Bus Speeds
  - 400 kHz Capable
  - Hardware Bus Access “Fairness” Interface
  - SMBus Time-outs Interface
  - 12 Port Flexible Multiplexing
  - Port Isolation
- PECI Interface 3.0
- Keyboard Matrix Scan Interface
  - 18 x 8 Interrupt/Wake Capable Multiplexed Keyboard Scan Matrix
  - Row Predrive Option

# MEC1618/MEC1618i

---

- Three independent Hardware Driven PS/2 Ports
  - Fully functional on Main and/or Suspend Power
  - PS/2 Edge Wake Capable
- 133 General Purpose I/O Pins
  - 8 GPIO Pass-Through Port (GTPP)
- 3-pin LED Interface
  - Programmable Blink Rates
  - Piecewise Linear Breathing LED Output Controller
  - Operational in EC Sleep States
- Programmable 16-bit Counter/Timer Interface
  - Four Wake-capable 16-bit Auto-reloading Counter/Timer Instances
  - Four Operating Modes per Instance: Timer, One-shot, Event and Measurement.
  - 4 External Inputs, 4 External Outputs
- Hibernation Timer Interface
  - Two 32.768 kHz Driven Timers
  - Programmable Wake-up from 0.5ms to 128 Minutes
- System Watch Dog Timer (WDT)
- Input Capture and Compare Timer
  - 32-bit Free-running timer
  - Six 32-bit Capture Registers
  - Two 32-bit Compare Registers
  - Capture, Compare and Overflow Interrupts
- BC-Link Interconnection Bus
  - Two High Speed and one Low Speed Bus Masters Controllers
- Two General Purpose Serial Peripheral Interface Controllers (ECGP-SPI)
  - One 3-pin EC-driven Full Duplex Serial Communication Interface
  - One 4-pin EC/Host-driven Full Duplex Serial Communication Interface to SPI Flash Interface
  - Flexible Clock Rates
  - SPI Burst Capable
- FAN Support
  - 16 Programmable Pulse-Width Modulator Outputs
    - Multiple Clock Rates
    - 16-Bit 'On' & 16-Bit 'Off' Counters
  - 6 Fan Tachometers
  - 6 x 2 Capture/Compare Timer Interface
- ADC Interface
  - 10-bit Conversion in 10 $\mu$ s
  - 16 Channels
  - Integral Non-Linearity of  $\pm 0.5$  LSB; Differential Non-Linearity of  $\pm 0.5$  LSB
- HDMI-CEC Interface Controller
- Two Pin Debug Port with Standard 16C550A Register Interface
  - Accessible from Host and EC
  - Programmable Input/output Pin Polarity Inversion
  - Programmable Main Power or Standby Power Functionality
  - Standard Baud Rates to 115.2 Kbps, Custom Baud Rates to 2 Mbps
- Resistor/Capacitor Identification Detection (RC\_ID)
  - Single Pin Interface to External Inexpensive RC Circuit
  - Replacement for Multiple GPIO's
  - Provides 8 Quantized States on One Pin
- Integrated Standby Power Reset Generator
- Gang Programmer Interface
  - JTAG Enabled
  - Supports Mass Programming and Mass Verify
  - JTAG Mass Erase
- Clock Generator
  - VBAT powered 32.768 kHz Oscillator  $\pm 2\%$  Accuracy
    - VBAT powered 32.768 kHz external input
    - External Clock Auto Detect Option
  - Operational on Suspend Power
  - Programmable Clock Power Management Control & Distribution
  - 20.27 MHz (nom) Oscillator
- Package
  - 156 Pin LFBGA RoHS Compliant package

## Tool Requirements

Metaware version 8.7 or newer must be used.

## TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at [docerrors@microchip.com](mailto:docerrors@microchip.com). We welcome your feedback.

### Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

<http://www.microchip.com>

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

### Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; <http://www.microchip.com>
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include -literature number) you are using.

### Customer Notification System

Register on our web site at [www.microchip.com](http://www.microchip.com) to receive the most current information on all of our products.

# MEC1618/MEC1618i

---

## Table of Contents

1.0 General Description .....	5
2.0 MEC1618 Pin Configuration .....	8
3.0 Bus Hierarchy .....	47
4.0 Logical Device Configuration .....	56
5.0 Power, Clocks, and Resets .....	74
6.0 Host Interface .....	135
7.0 Embedded Memory Interface .....	153
8.0 ACPI Embedded Controller Interface (ACPI-ECI) .....	177
9.0 8042 Emulated Keyboard Controller .....	199
10.0 ACPI PM1 Block Interface .....	216
11.0 MailBox Register Interface .....	225
12.0 Two Pin Serial Port (UART) .....	234
13.0 Embedded Flash Subsystem .....	256
14.0 ARC 625D Embedded Controller .....	288
15.0 EC Interrupt Aggregator .....	296
16.0 Watchdog Timer Interface .....	349
17.0 HDMI-CEC Interface Controller .....	356
18.0 16-Bit Timer Interface .....	372
19.0 Hibernation Timer .....	389
20.0 Week Alarm Interface .....	393
21.0 GPIO Interface .....	398
22.0 Input Capture and Compare Timer .....	421
23.0 DMA Controller .....	436
24.0 SMB Device Interface .....	449
25.0 PECl Interface .....	454
26.0 Analog to Digital Converter .....	457
27.0 TACH Monitor .....	469
28.0 PWM Controller .....	478
29.0 RC Identification Detection (RC_ID) .....	485
30.0 General Purpose Serial Peripheral Interface (GP-SPI) .....	494
31.0 VBAT-Powered Control Interface .....	519
32.0 VBAT Powered RAM .....	526
33.0 Blinking/Breathing PWM .....	528
34.0 PS/2 Device Interface .....	548
35.0 Keyboard Matrix Scan Support .....	558
36.0 BC-Link Master .....	567
37.0 BIOS Debug Port .....	576
38.0 Serial Debug Port .....	584
39.0 Boot ROM .....	589
40.0 Gang Programmer Interface .....	592
41.0 JTAG and XNOR .....	604
42.0 Electrical Specifications .....	628
43.0 Timing Diagrams .....	637
44.0 Reference Documents .....	657
The Microchip Web Site .....	659
Customer Change Notification Service .....	659
Customer Support .....	659
Product Identification System .....	660

## 1.0 GENERAL DESCRIPTION

The MEC1618/MEC1618i is the mixed signal base component of a multi-device advanced I/O controller architecture. The MEC1618/MEC1618i incorporates a high-performance 32-bit ARC 625D embedded microcontroller with a 192 Kilobyte [Embedded Flash Subsystem](#), 16 Kilobytes of SRAM and a 1 Kilobyte EEPROM Emulation. The MEC1618/MEC1618i communicates with the system host using the Intel® Low Pin Count bus.

The MEC1618/MEC1618i is the EC Base Component of a split-architecture Advanced I/O Controller system that uses BC-Link communication protocol to access up to three companion components. The BC-Link protocol is peer-to-peer providing communication between the MEC1618/MEC1618i embedded controller and registers located in a companion.

The MEC1618/MEC1618i is directly powered by two separate suspend supply planes ([VBAT](#) and [VTR](#)) and senses a third runtime power plane (VCC) to provide “instant on” and system power management functions. The MEC1618/MEC1618i also contains an integrated [VTR Reset Interface](#) and a system [Power Management Interface](#) that supports low-power states and can drive state changes as a result of hardware wake events as defined by the MEC1618/MEC1618i [Wake Interface](#).

The MEC1618/MEC1618i defines a software development system interface that includes an MCU Serial Debug Port, a two pin serial debug port with a 16C550A register interface that is accessible to the EC or to the LPC host and can operate up to 2 MB/s, a flexible Flash programming interface, a [BIOS Debug Port](#), [Gang Programmer Interface](#), and a JTAG interface. The EC can also drive the JTAG interface as a master.

A top-level block diagram of the MEC1618/MEC1618i is shown below in [Figure 1-1](#). An example of system level connection is shown in [Figure 1-2](#). A detailed description of the [Bus Hierarchy](#) can be found in [Section 3.0, "Bus Hierarchy," on page 47](#).

# MEC1618/MEC1618i

FIGURE 1-1: MEC1618/MEC1618i TOP-LEVEL BLOCK DIAGRAM

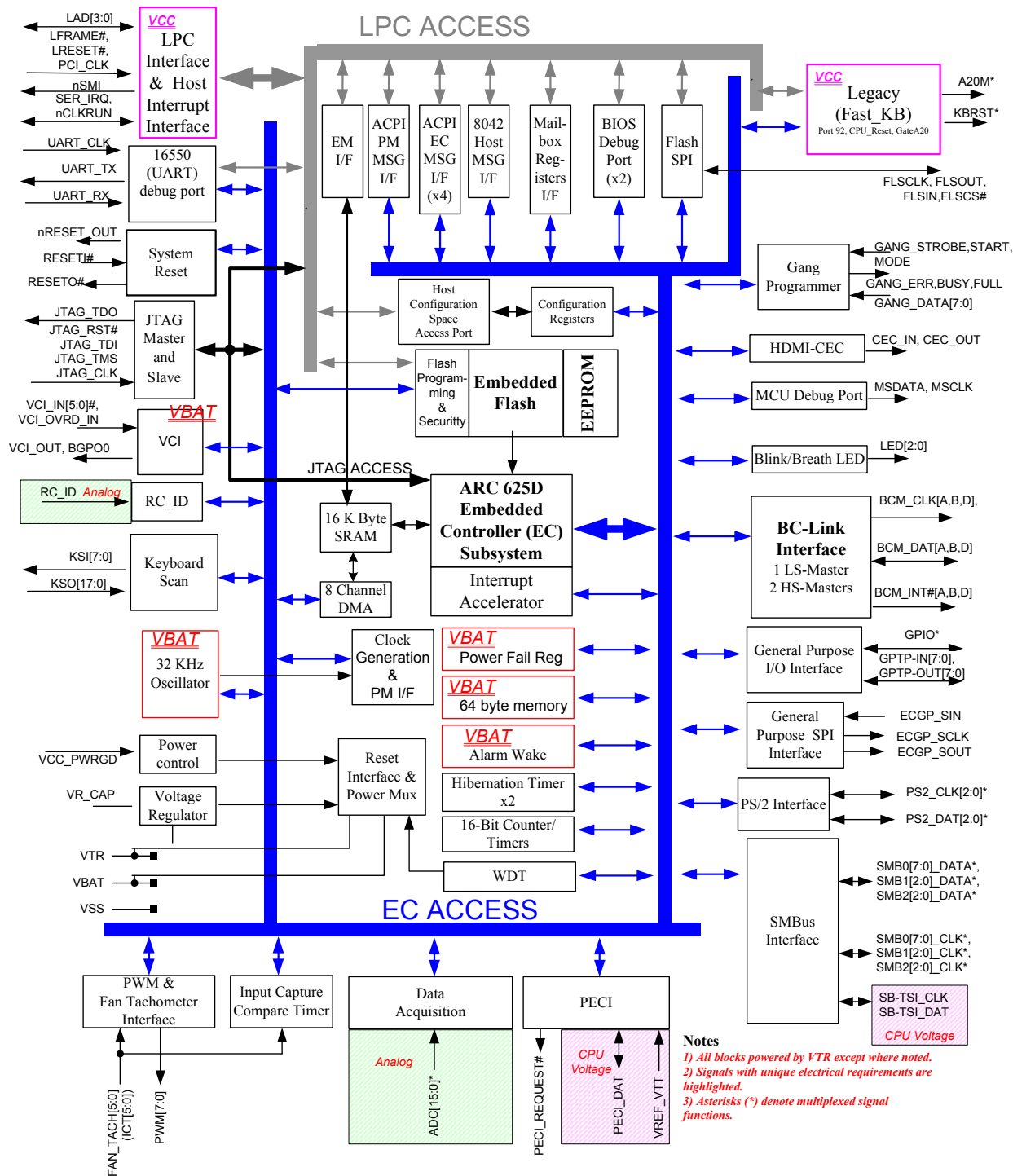
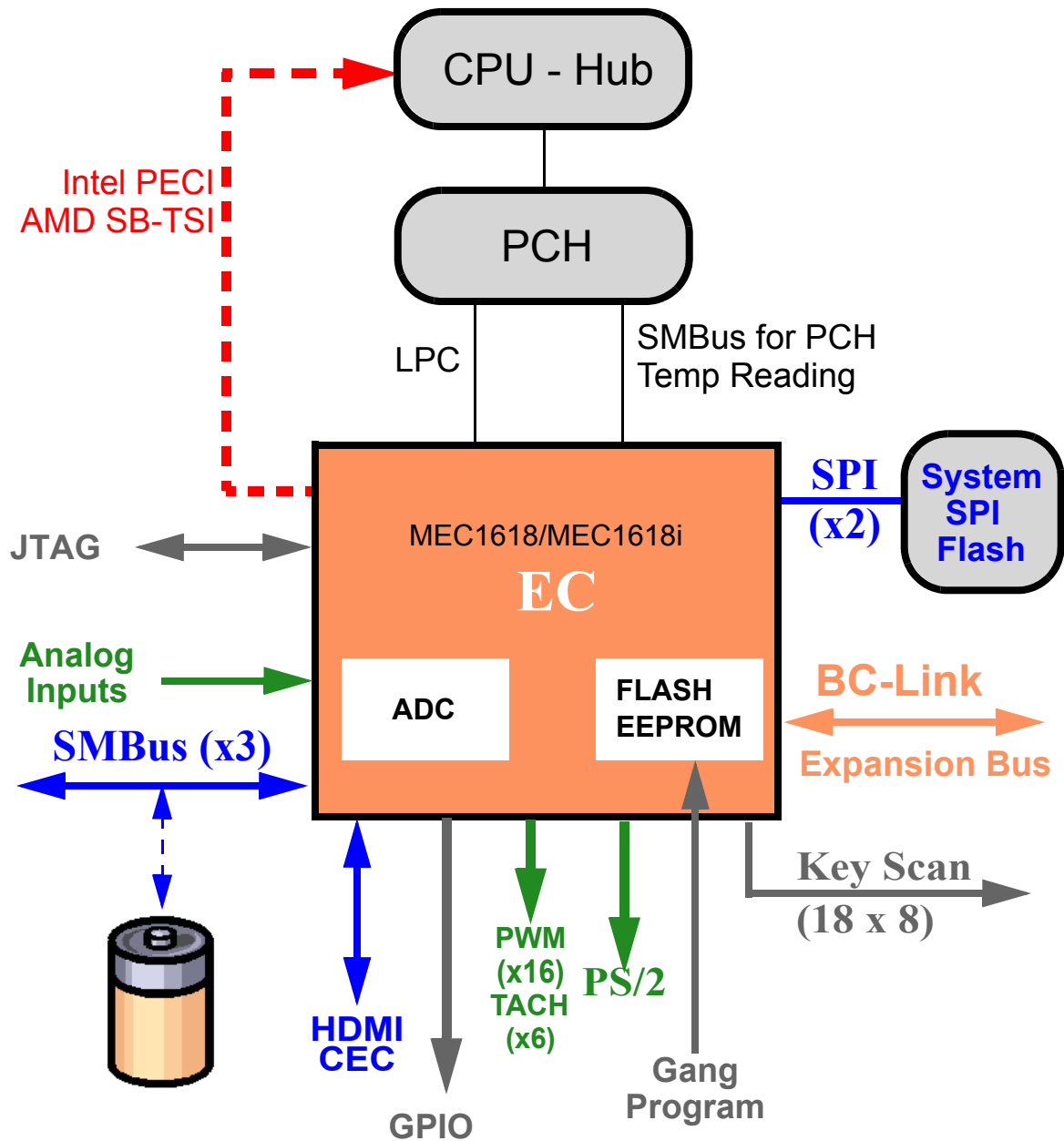


FIGURE 1-2: EXAMPLE OF MEC1618/MEC1618I CONNECTIONS TO SYSTEM COMPONENTS



# MEC1618/MEC1618i

## 2.0 MEC1618 PIN CONFIGURATION

### 2.1 Description

The MEC1618 Pin Configuration chapter includes a [Pin List](#), [General System/Layout Issues](#), [Pin Description](#), [Pin Multiplexing](#), [Notes for Tables in this Chapter](#), [Strapping Options](#) and [Package Outline](#).

bit Note that unless otherwise noted ball numbers shown in the tables throughout this document refer to the [156-Ball LFBGA 11 mm x 11 mm x 0.8 mm Package Outline](#).

### 2.2 Pin List

The MEC1618/MEC1618i [Pin List](#) is illustrated below in [Table 2-1](#). The BGA package ball mapping to MEC1618/MEC1618i Pin Names is shown in [Figure 2-1](#).

**TABLE 2-1: PRELIMINARY MEC1618/MEC1618I PIN CONFIGURATION**

Pin Ref. Number	Ball Number	Pin Name	Pin Ref. Number	Ball Number	Pin Name
1	D4	GPIO165/32KHZ_IN	40	M3	ADC6/GPIO206
2	E3	BGND	41	M4	ADC14/GPIO216
3	C4	VBAT	42	N3	ADC7/GPIO207
4	C1	BGPO0	43	N4	ADC15/GPIO217
5	D1	VCI_OUT	44	K4	VSS_ADC
6	C2	VCI_IN2#/GPIO161	45	M5	LRESET#
7	G1	VCI_IN1#/GPIO162	46	L5	CLKRUN#
8	D2	VCI_IN0#/GPIO163	47	L6	LFRAME#
9	E2	VCI_OV_RD_IN/GPIO164	48	N5	SER_IRQ
10	E1	VCI_IN3#/GPIO000	49	K6	VTR1
11	C3	VCI_IN4#/GPIO234	50	N6	PCI_CLK
12	H2	VCI_IN5#/GPIO235	51	M6	LAD0
13	F4	RESET#	52	M7	LAD1
14	G2	GPIO062/RESET0#/GANG_START	53	L7	LAD2
15	F3	VCC_PWRGD/GANG_MODE	54	K7	LAD3
16	F2	GPIO106/nRESET_OUT	55	K5	GPIO100/nEC_SCI
17	H5	VSS_RO	56	N7	GPIO011/nSMI
18	G4	VTR0	57	L8	GPIO061/LPCPD#
19	J4	VSS0	58	M8	GPIO050/FAN_TACH0
20	H1	GPIO033/RC_ID/GANG_STROBE	59	N9	GPIO051/FAN_TACH1
21	H3	GPIO021/KSI2/GANG_FULL	60	M9	GPIO052/FAN_TACH2
22	J5	VTR_REG	61	L9	GPIO016/GPTP-IN7/FAN_TACH3
23	H4	VR_CAP	62	J9	GPIO230/ECGP_SCLK
24	F1	GPIO060/KBRST/GANG_ERROR	63	N13	GPIO053/PWM0
25	J1	AVTR_ADC	64	K9	GPIO231/ECGP_SOUT
26	J3	VREF_ADC	65	M10	GPIO054/PWM1
27	K3	ADC0/GPIO200	66	K8	GPIO233/ECGP_SIN
28	J2	ADC8/GPIO210	67	M11	GPIO055/PWM2
29	K1	ADC1/GPIO201	68	L13	GPIO056/PWM3
30	K2	ADC9/GPIO211	69	M12	GPIO001/PWM4
31	G3	VTR3	70	L10	GPIO002/PWM5
32	L2	ADC2/GPIO202	71	L12	GPIO014/GPTP-IN6/PWM6
33	L1	ADC10/GPIO212	72	K12	GPIO015/GPTP-OUT6/PWM7
34	M1	ADC3/GPIO203	73	N12	GPIO151/GPTP-IN3/FAN_TACH4/KSO15
35	L3	ADC11/GPIO213	74	N10	GPIO152/GPTP-OUT3/FAN_TACH5/KSO16
36	L4	ADC4/GPIO204	75	J8	VTR2
37	M2	ADC12/GPIO214	76	K13	GPIO003/SMB00_DATA
38	N1	ADC5/GPIO205	77	H13	GPIO004/SMB00_CLK
39	N2	ADC13/GPIO215	78	K11	GPIO005/SMB01_DATA



Pin Ref. Number	Ball Number	Pin Name	Pin Ref. Number	Ball Number	Pin Name
79	J12	GPIO006/SMB01_CLK	118	B12	GPIO146/SMB09_CLK/JTAG_TDO
80	J11	GPIO012/SMB07_DATA	119	B11	GPIO147/SMB08_DATA/JTAG_CLK
81	H11	GPIO013/SMB07_CLK	120	A12	GPIO150/SMB08_CLK/JTAG_TMS
82	H12	GPIO130/SMB10_DATA	121	A11	JTAG_RST#
83	G10	GPIO131/SMB10_CLK	122	C11	GPIO104/UART_TX
84	G11	GPIO132/SMB06_DATA/KSO14	123	A10	GPIO105/UART_RX
85	G12	GPIO140/SMB06_CLK/PWM13/GANG_DATA7	124	B10	GPIO025/UART_CLK/TIN0/EM_INT
86	H9	VTR_FLASH	125	B9	GPIO026/GPTP-IN0/TIN1/KSI3
87	J10	GPIO141/SMB05_DATA/PWM14/FLSCLK	126	D9	GPIO027/GPTP-OUT0/TIN2/KSI4
88	K10	GPIO142/SMB05_CLK/PWM15/FLSOUT/GANG_DATA6	127	C10	GPIO030/GPTP-IN1/TIN3/KSI5
89	H10	GPIO143/SMB04_DATA/PWM12/FLSIN/GANG_BUSY	128	A9	GPIO107/KSO4
90	L11	GPIO144/SMB04_CLK/Reserved/FLSCS#/GANG_DATA5	129	C9	GPIO120/KSO7
91	F11	GPIO007/SMB03_DATA/PS2_CLK0B	130	B8	GPIO124/GPTP-OUT4/KSO11
92	F10	GPIO100/SMB03_CLK/PS2_DAT0B/GANG_DATA4	131	A8	GPIO125/GPTP-IN4/KSO12
93	F12	GPIO154/SMB02_DATA/PS2_CLK1B	132	D6	GPIO031/GPTP-OUT1/TOUT0/KSI6
94	E12	GPIO155/SMB02_CLK/PS2_DAT1B/GANG_DATA3	133	B7	GPIO032/GPTP-IN2/TOUT1/KSI7
95	E11	GPIO110/PS2_CLK2/GPTP-IN5	134	C8	GPIO040/GPTP-OUT2/TOUT2/KSO0
96	E10	GPIO111/PS2_DAT2/GPTP-OUT5/GANG_DATA2	135	A7	GPIO017/GPTP-OUT7/TOUT3/KSI0
97	D12	GPIO112/PS2_CLK1A/KSO5	136	D5	GPIO022/BCM_B_CLK
98	N11	GPIO113/PS2_DAT1A/KSO6/GANG_DATA1	137	C5	GPIO023/BCM_B_DAT
99	M13	GPIO114/PS2_CLK0A	138	E4	GPIO024/BCM_B_INT#
100	J13	GPIO115/PS2_DAT0A/GANG_DATA0	139	C7	GPIO127/A20M
101	J6	VSS1	140	F5	GPIO034/CEC_OUT
102	G13	GPIO070	141	E5	GPIO036/CEC_IN
103	F13	GPIO071	142	A6	GPIO045/LSBCEM_D_INT#/KSO1
104	E13	GPIO072	143	E6	GPIO046/LSBCEM_D_DAT/KSO2
105	D13	GPIO073	144	B6	GPIO047/LSBCEM_D_CLK/KSO3
106	C13	GPIO074	145	C6	GPIO121/BCM_A_INT#/KSO8
107	B13	GPIO075	146	A5	GPIO122/BCM_A_DAT/KSO9
108	F9	GPIO041	147	B5	GPIO123/BCM_A_CLK/KSO10
109	A13	GPIO076	148	A3	GPIO042/Reserved/PECI_DAT/SB-TSI_DAT
110	E9	GPIO220	149	A2	GPIO043/Reserved/Reserved/SB-TSI_CLK
111	E8	GPIO035/PWM8	150	A1	GPIO044/VREF_VTT
112	D8	GPIO170/MCLK	151	B1	GPIO126/KSO13
113	D7	GPIO171/MSDATA	152	A4	GPIO020/KSI1
114	N8	GPIO133/PWM9	153	B2	GPIO156/LED0
115	D11	GPIO134/PWM10	154	B3	GPIO157/LED1
116	D10	GPIO135/PWM11	155	B4	GPIO153/LED2
117	C12	GPIO145/SMB09_DATA/JTAG_TDI	156	D3	GPIO175/32KHZ_OUT/KSO17

# MEC1618/MEC1618i

**FIGURE 2-1: MEC1618/MEC1618i PIN NAME TO 156-PIN LFBGA BALL MAPPING (TOP)**

1	2	3	4	5	6	7	8	9	10	11	12	13	
GPIO044/VREF_VTT	GPIO043/Reserved/Reserved/SB-TSI_CLK	GPIO042/Reserved/PECL_DAT/SB-TSI_DAT	GPIO020/KSI1	GPIO122/BCM_A_DAT/KSO9	GPIO045/LSBCM_D_INT/KSO1	GPIO017/GPTP-OUT7/TOUT3/KSI0	GPIO125/GPTP-IN4/KSO12	GPIO107/KSO4	GPIO105/UART_RX	JTAG_RST#	GPIO150/SMB08_CLK/JTAG_TMS	GPIO076	A
GPIO126/KSO13	GPIO156/LED0	GPIO157/LED1	GPIO153/LED2	GPIO123/BCM_A_CLK/KSO10	GPIO047/LSBCM_D_CLK/KSO3	GPIO032/GPTP-IN2/TOUT1/KSI7	GPIO124/GPTP-OUT4/KSO11	GPIO026/GPTP-IN0/TIN1/KSI3	GPIO025/UART_CLK/TIN0/EM_INT	GPIO147/SMB08_DATA/JTAG_CLK	GPIO146/SMB09_CLK/JTAG_TDO	GPIO075	B
BGPO0	VCL_IN2#/GPIO161	VCL_IN4#/GPIO234	VBAT	GPIO023/BCM_A_DAT	GPIO121/BCM_A_INT#KSO8	GPIO127/A2OM	GPIO040/GPTP-OUT2/TOUT2/KSO0	GPIO120/KSO7	GPIO030/GPTP-IN1/TIN3/KSI5	GPIO104/UART_TX	GPIO145/SMB09_DATA/JTAG_TDI	GPIO074	C
VCL_OUT	VCL_IN0#/GPIO163	GPIO175/32KHZ_OUT/KSO17	GPIO165/32KHZ_IN	GPIO022/BCM_B_CLK	GPIO031/GPTP-OUT1/TOUT0/KSI6	GPIO171/MSDAT_A	GPIO170/MSCLK	GPIO027/GPTP-OUT0/TIN2/KSI4	GPIO135/PWM11	GPIO134/PWM10	GPIO112/PS2_CLK1A/KSO5	GPIO073	D
VCL_IN3#/GPIO00	VCL_OVRD_IN#/GPIO164	BGND	GPIO024/BCM_B_INT#	GPIO036/CEC_IN	GPIO046/LSBCM_D_DAT/KSO2	No Ball	GPIO035/PWM8	GPIO220	GPIO111/PS2_DATA2/GPTP-OUT5/GANG_DATA2A	GPIO110/PS2_CLK2/GPTP-IN5	GPIO155/SMB02_CLK/PS2_DATA1B/GANG_DATA3	GPIO072	E
GPIO060/KBRST/GANG_ERROR	GPIO106/mRESET_OUT	VCC_PWRGD/GANG_MODE	RESET#	GPIO034/CEC_OUT	No Ball	No Ball	No Ball	GPIO041	GPIO101/SMB03_CLK/PS2_DATA0B/GANG_DATA4	GPIO007/SMB03_DATA/PS2_CLK0B	GPIO154/SMB02_DATA/PS2_CLK1B	GPIO071	F
VCL_IN1#/GPIO162	GPIO062/RESET_OUT#GANG_START	VTR3	VTR0	No Ball	No Ball	No Ball	No Ball	No Ball	GPIO131/SMB10_CLK	GPIO132/SMB06_DATA/KSO14	GPIO140/SMB06_CLK/PWM13/GANG_DATA7	GPIO070	G
GPIO033/RC_ID/GANG_STROBE	VCL_IN5#/GPIO235	GPIO021/KSI2/GANG_FULL	VR_CAP	VSS_RO	No Ball	No Ball	No Ball	VTR_FLASH	GPIO143/SMB04_DATA/PWM12/FLSIN/GANG_BUS_Y	GPIO013/SMB07_CLK	GPIO130/SMB10_DATA	GPIO004/SMB00_CLK	H
AVTR_ADC	ADC8/GPIO210	VREF_ADC	VSS0	VTR_REG	VSS1	No Ball	VTR2	GPIO230/ECGP_SCLK	GPIO141/SMB05_DATA/PWM14/FLSCLK	GPIO012/SMB07_DATA	GPIO008/SMB01_CLK	GPIO115/PS2_DATA0A/GANG_DATA0	J
ADC1/GPIO201	ADC9/GPIO211	ADC0/GPIO200	VSS_ADC	GPIO100/mEC_SCI	VTR1	LAD3	GPIO233/ECGP_SIN	GPIO231/ECGP_SOUT	GPIO142/SMB05_CLK/PWM15/FLSOUT/GANG_DATA6	GPIO005/SMB01_DATA	GPIO015/GPTP-OUT6/PWM7	GPIO003/SMB00_DATA	K
ADC10/GPIO212	ADC2/GPIO202	ADC11/GPIO213	ADC4/GPIO204	CLKRUN#	LFRAME#	LAD2	GPIO061/LPCPD#	GPIO016/GPTP-IN7/FAN_TACH3	GPIO002/PWM5	GPIO144/SMB04_CLK/Reserved/FLSCS#GANG_DATA5	GPIO014/GPTP-IN6/PWM6	GPIO056/PWM3	L
ADC3/GPIO203	ADC12/GPIO214	ADC6/GPIO206	ADC14/GPIO216	LRESET#	LAD0	LAD1	GPIO050/FAN_TACH0	GPIO052/FAN_TACH2	GPIO054/PWM1	GPIO055/PWM2	GPIO001/PWM4	GPIO114/PS2_CLK0A	M
ADC5/GPIO205	ADC13/GPIO215	ADC7/GPIO207	ADC15/GPIO217	SER_IRQ	PCI_CLK	GPIO011/nSMI	GPIO133/PWM9	GPIO051/FAN_TACH1	GPIO152/GPTP-OUT3/FAN_TACH5/KSO16	GPIO113/PS2_DATA1A/KSO6/GANG_DATA1	GPIO151/GPTP-IN3/FAN_TACH4/KSO15	GPIO053/PWM0	N

## 2.3 General System/Layout Issues

### 2.3.1 PIN DEFAULT STATE THROUGH POWER TRANSITIONS

The power state and power state transitions illustrated in [Table 2-2](#) are defined in [Section 5.0, "Power, Clocks, and Resets"](#). Pin behavior in this table assumes no specific programming to change the pin state. All GPIO default pins have the same behavior described in [Table 2-2](#) as generic GPIOXX, except where noted.

**TABLE 2-2:** Pin Default State Through Power Transitions

	Signal	VBAT applied	VBAT STABLE	VTR applied	nSYS_RST de-asserted	VCC_PWRGD asserted	VCC_PWRGD de-asserted	nSYS_RST asserted	VTR un-powered	VBAT un-powered	Notes
	GPIO042	unpowered	unpowered	low	In	In	In	Z	glitch	unpowered	
	GPIO043	unpowered	unpowered	low	In	In	In	Z	glitch	unpowered	
	GPIO062	unpowered	unpowered	low	Out=0	Out	Out	Out	glitch	unpowered	
	GPIOXX	unpowered	unpowered	Z	In	In	In	Z	glitch	unpowered	Note E
	SER_IRQ	unpowered	unpowered	glitch	In	Z>I/O (P)>Z	In	In	glitch	unpowered	Note A
	LRESET#	unpowered	unpowered	glitch	In	In	In	Z	glitch	unpowered	Note A
	PCI_CLK	unpowered	unpowered	glitch	In	In	In	Z	glitch	unpowered	
	LFRAME#	unpowered	unpowered	glitch	In	In	In	Z	glitch	unpowered	
	LAD0	unpowered	unpowered	glitch	In	In>I/O (P)>In	In	Z	glitch	unpowered	
	LAD1	unpowered	unpowered	glitch	In	In>I/O (P)>In	In	Z	glitch	unpowered	
	LAD2	unpowered	unpowered	glitch	In	In>I/O (P)>In	In	Z	glitch	unpowered	
	LAD3	unpowered	unpowered	glitch	In	In>I/O (P)>In	In	Z	glitch	unpowered	
	CLKRUN#	unpowered	unpowered	glitch	In	Z>I/O (P)>Z	In	Z	glitch	unpowered	
	BGPO0	Out=0	Out=0	Retain	Retain	Retain	Retain	Retain	Retain	unpowered	Note B
	VCI_INx#	In	In	In	In	In	In	In	In	unpowered	
	VCI_OUT	Out logic	Out logic	Out logic	Out logic	Out logic	Out logic	Out logic	Out logic	unpowered	Note C
	VCI_OVRD_IN	In	In	In	In	In	In	In	In	unpowered	
<b>Legend</b>					<b>Notes</b>						
(P) = I/O state is driven by protocol while power is					<b>Note A:</b> Pin exhibits "VCC" power domain emulation.						
Z = Tristate					<b>Note B:</b> Pin is programmable by the EC and retains its value through a VTR power cycle.						
					<b>Note C:</b> Pin is programmable by the EC and affected by other VBAT inputs pins.						
					<b>Note D:</b> Pin exhibits "VTR" power domain emulation.						
					<b>Note E:</b> Does not include GPIO042, GPIO043, and GPIO062						

### 2.3.2 ALTERNATE FUNCTION PIN STATE THROUGH POWER TRANSITIONS

The power state and power state transitions illustrated in [Table 2-3](#) are defined in [Section 5.0, "Power, Clocks, and Resets"](#). Pin behavior in this table assumes that the EC programs alternate function pin state (see [Section 2.5, "Pin Multiplexing," on page 23](#)).

**TABLE 2-3:** Alternate Function Pin State Through Power Transitions

	Signal	VBAT applied	VBAT STABLE	VTR applied	nSYS_RST de-asserted	VCC_PWRGD asserted	VCC_PWRGD de-asserted	nSYS_RST asserted	VTR un-powered	VBAT un-powered	Notes
	nSMI	unpowered	unpowered	glitch	In>OD(1)	1>OD(P)>1	OD(1)	In	glitch	unpowered	
	KBRST	unpowered	unpowered	glitch	In>Z	1>OD(P)>1	Z	Z>In	glitch	unpowered	Note F
	A20M	unpowered	unpowered	glitch	In>Z	1>OD(P)>1	Z	Z	glitch	unpowered	Note F
	LPCPD#	unpowered	unpowered	glitch	In>Z	In	Z	Z	glitch	unpowered	Note F
<b>Legend</b>					<b>Notes</b>						
(P) = I/O state is driven by protocol while power is					<b>Note E:</b> Transition occurs due to EC selecting alternate function.						
Z = Tristate					<b>Note F:</b> Pin is programmable by the EC and retains its value through a VTR power cycle.						
OD = Open Drain Output Undriven (1) or driven (0)											

# MEC1618/MEC1618i

## 2.3.3 NON 5 VOLT TOLERANT PINS

Table 2-4 lists all signal pins that are not 5.5 Volt tolerant; all other signal pins are 5 Volt tolerant. Signals in Table 2-4 refer to Pin Reference Numbers as defined in Table 2-1.

**TABLE 2-4: NON 5 VOLT TOLERANT PINS**

Ball	MEC1618 Pin Reference Number	Pin Name
K3	27	ADC0/GPIO200
J2	28	ADC8/GPIO210
K1	29	ADC1/GPIO201
K2	30	ADC9/GPIO211
L2	32	ADC2/GPIO202
L1	33	ADC10/GPIO212
M1	34	ADC3/GPIO203
L3	35	ADC11/GPIO213
L4	36	ADC4/GPIO204
M2	37	ADC12/GPIO214
N1	38	ADC5/GPIO205
N2	39	ADC13/GPIO215
M3	40	ADC6/GPIO206
M4	41	ADC14/GPIO216
N3	42	ADC7/GPIO207
N4	43	ADC15/GPIO217
M5	45	LRESET#
L5	46	CLKRUN#
L6	47	LFRAME#
N5	48	SER_IRQ
N6	50	PCI_CLK
M6	51	LAD0
M7	52	LAD1
L7	53	LAD2
K7	54	LAD3
E9	110	GPIO220

## 2.3.4 NON GLITCH PROTECTED PINS

Table 2-5 lists pins that do not have POR output glitch protection. POR output glitch protection guarantees that pins will have a steady-state output during VTR POR. Pins without POR output glitch protection may be susceptible to transitory changes as VTR power is applied. Signals in Table 2-5 refer to Pin Reference Numbers as defined in Table 2-1.

**TABLE 2-5: NON GLITCH PROTECTED PINS**

Ball	MEC1618 Pin Reference Number	Pin Name
K3	27	ADC0/GPIO200
J2	28	ADC8/GPIO210
K1	29	ADC1/GPIO201
K2	30	ADC9/GPIO211
L2	32	ADC2/GPIO202
L1	33	ADC10/GPIO212
M1	34	ADC3/GPIO203
L3	35	ADC11/GPIO213
L4	36	ADC4/GPIO204
M2	37	ADC12/GPIO214
N1	38	ADC5/GPIO205
N2	39	ADC13/GPIO215
M3	40	ADC6/GPIO206
M4	41	ADC14/GPIO216
N3	42	ADC7/GPIO207
N4	43	ADC15/GPIO217
M5	45	LRESET#
L5	46	CLKRUN#
L6	47	LFRAME#
N5	48	SER_IRQ
N6	50	PCI_CLK
M6	51	LAD0
M7	52	LAD1
L7	53	LAD2
K7	54	LAD3
E9	110	GPIO220

# MEC1618/MEC1618i

## 2.3.5 NON BACKDRIVE PROTECTED PINS

Table 2-6 lists pins that do not have backdrive protection. Signals in Table 2-6 refer to Pin Reference Numbers as defined in Table 2-1.

**TABLE 2-6: NON BACKDRIVE PROTECTED PINS**

Ball	MEC1618 Pin Reference Number	Pin Name
K3	27	ADC0/GPIO200
J2	28	ADC8/GPIO210
K1	29	ADC1/GPIO201
K2	30	ADC9/GPIO211
L2	32	ADC2/GPIO202
L1	33	ADC10/GPIO212
M1	34	ADC3/GPIO203
L3	35	ADC11/GPIO213
L4	36	ADC4/GPIO204
M2	37	ADC12/GPIO214
N1	38	ADC5/GPIO205
N2	39	ADC13/GPIO215
M3	40	ADC6/GPIO206
M4	41	ADC14/GPIO216
N3	42	ADC7/GPIO207
N4	43	ADC15/GPIO217
M5	45	LRESET#
L5	46	CLKRUN#
L6	47	LFRAME#
N5	48	SER_IRQ
N6	50	PCI_CLK
M6	51	LAD0
M7	52	LAD1
L7	53	LAD2
K7	54	LAD3
E9	110	GPIO220

## 2.4 Pin Description

### 2.4.1 OVERVIEW

The following tables describe the signal functions in the MEC1618/MEC1618i pin configuration. See [Section 2.6, "Notes for Tables in this Chapter," on page 44](#) for notes that are referenced in the [Pin Description](#) tables.

### 2.4.2 HOST INTERFACE

**TABLE 2-7: HOST INTERFACE**

HOST INTERFACE				(13 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
48	N5	SER_IRQ	Serial IRQ	Note 4, Note 5, Note 2
45	M5	LRESET#	LPC Reset. LRESET# is the same as the system PCI reset, PCIRST#	Note 5, Note 2
50	N6	PCI_CLK	PCI Clock	Note 5, Note 2
47	L6	LFRAME#	Frame signal. Indicates start of new cycle and termination of broken cycle	Note 5, Note 3
51	M6	LAD0	LPC Multiplexed command, address and data bus Bit 0.	Note 4, Note 5, Note 2
52	M7	LAD1	LPC Multiplexed command, address and data bus Bit 1.	Note 4, Note 5, Note 2
53	L7	LAD2	LPC Multiplexed command, address and data bus Bit 2.	Note 4, Note 5, Note 2
54	K7	LAD3	LPC Multiplexed command, address and data bus Bit 3.	Note 4, Note 5, Note 2
46	L5	CLKRUN#	PCI Clock Control	Note 5, Note 2
55	K5	nEC_SCI	Power Management Event	Note 3
57	L8	LPCPD#	The LPC Bus Powerdown Signal.	
56	N7	nSMI	SMI Output	Note 2
124	B10	EM_INT	EM Interface Interrupt Output	

### 2.4.3 BC-LINK INTERFACE

**TABLE 2-8: BC-LINK INTERFACE**

BC-Link Interface				(9 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
147	B5	BCM_A_CLK	BC-Link Master clock	
146	A5	BCM_A_DAT	BC-Link Master data I/O	Note 9
145	C6	BCM_A_INT#	BC-Link Master interrupt	
136	D5	BCM_B_CLK	BC-Link Master clock	
137	C5	BCM_B_DAT	BC-Link Master data I/O	Note 9
138	E4	BCM_B_INT#	BC-Link Master interrupt	
143	E6	LSBCM_D_DAT	BC-Link Master data I/O	Note 9
144	B6	LSBCM_D_CLK	BC-Link Master clock	
142	A6	LSBCM_D_INT#	BC-Link Master interrupt	

**Note 2-1** For ribbon cable applications, the Low Speed BC-Link Master maximum clock frequency is 3 MHz. The High Speed BC-Link Master maximum clock frequency is 21.5 MHz. The clock frequency is set with the [BC Clock Select](#) register.

**Note 2-2** the BCM DAT pins require a weak pull up resistor (100 K Ohms).

# MEC1618/MEC1618i

## 2.4.4 JTAG INTERFACE

**TABLE 2-9: JTAG INTERFACE**

JTAG Interface				(5 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
119	B11	JTAG_CLK	JTAG Test Clock	
121	A11	JTAG_RST#	JTAG Test Reset (active low)	Note 14
117	C12	JTAG_TDI	JTAG Test Data In	
118	B12	JTAG_TDO	JTAG Test Data Out	
120	A12	JTAG_TMS	JTAG Test Mode Select	

## 2.4.5 MASTER CLOCK INTERFACE

**TABLE 2-10: MASTER CLOCK INTERFACE**

Master Clock Interface				(2 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
1	D4	32KHZ_IN	32.768 KHz Crystal Input	
156	D3	32KHZ_OUT	32.768 KHz Digital Output	

## 2.4.6 ANALOG DATA ACQUISITION INTERFACE

**TABLE 2-11: ANALOG DATA ACQUISITION**

Analog Data Acquisition Interface				(17 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
27	K3	ADC0	ADC channel 1	
29	K1	ADC1	ADC channel 2	
32	L2	ADC2	ADC channel 3	
34	M1	ADC3	ADC channel 4	
36	L4	ADC4	ADC channel 5	
38	N1	ADC5	ADC channel 6	
40	M3	ADC6	ADC channel 7	
42	N3	ADC7	ADC channel 8	
28	J2	ADC8	ADC channel 9	
30	K2	ADC9	ADC channel 10	
33	L1	ADC10	ADC channel 11	
35	L3	ADC11	ADC channel 12	
37	M2	ADC12	ADC channel 13	
39	N2	ADC13	ADC channel 14	
41	M4	ADC14	ADC channel 15	
43	N4	ADC15	ADC channel 16	
26	J3	VREF_ADC	ADC Voltage Reference Pin	

**Note:** The voltage on the pins in [Table 2-11](#) must not exceed 3.6 V or damage to the device will occur.



## 2.4.7 FAN TACHOMETER, PWM AND INPUT CAPTURE TIMER INTERFACE

**TABLE 2-12: FAN PWM & TACHOMETER INTERFACE**

FAN PWM & TACHOMETER				(22 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
58	M8	FAN_TACH0	Fan Tachometer Input 1/Input Capture Timer Input 0	
59	N9	FAN_TACH1	Fan Tachometer Input 2/Input Capture Timer Input 1	
60	M9	FAN_TACH2	Fan Tachometer Input 3/Input Capture Timer Input 2	
61	L9	FAN_TACH3	Fan Tachometer Input 4/Input Capture Timer Input 3	
73	N12	FAN_TACH4	Fan Tachometer Input 5/Input Capture Timer Input 4	
74	N10	FAN_TACH5	Fan Tachometer Input 6/Input Capture Timer Input 5	
63	N13	PWM0	Pulse Width Modulator Output 0	
65	M10	PWM1	Pulse Width Modulator Output 1	
67	M11	PWM2	Pulse Width Modulator Output 2	
68	L13	PWM3	Pulse Width Modulator Output 3	
69	M12	PWM4	Pulse Width Modulator Output 4	
70	L10	PWM5	Pulse Width Modulator Output 5	
71	L12	PWM6	Pulse Width Modulator Output 6	
72	K12	PWM7	Pulse Width Modulator Output 7	
111	E8	PWM8	Pulse Width Modulator Output 8	
114	N8	PWM9	Pulse Width Modulator Output 9	
115	D11	PWM10	Pulse Width Modulator Output 10	
116	D10	PWM11	Pulse Width Modulator Output 11	
89	H10	PWM12	Pulse Width Modulator Output 12	
85	G12	PWM13	Pulse Width Modulator Output 13	
87	J10	PWM14	Pulse Width Modulator Output 14	
88	K10	PWM15	Pulse Width Modulator Output 15	

# MEC1618/MEC1618i

## 2.4.8 GENERAL PURPOSE I/O INTERFACE

**TABLE 2-13: GPIO INTERFACE**

GPIO Interface				(133 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
1, 6, 7, 8, 9, 10, 11, 12, 14, (15), 16, 20, 21, 24, 27, 28, 29, 30, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, (45), (46), (47), (48), (50), 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156	D4, C2, G1, D2, E2, E1, C3, H2, G2, (F3), F2, H1, H3, F1, K3, J2, K1, K2, L2, L1, M1, L3, L4, M2, N1, N2, M3, M4, N3, N4, (M5), (L5), (L6), (N5), (N6), K5, N7, L8, M8, N9, M9, L9, J9, N13, K9, M10, K8, M11, L13, M12, L10, L12, K12, N12, N10, K13, H13, K11, J12, J11, H11, H12, G10, G11, G12, J10, K10, H10, L11, F11, F10, F12, E12, E11, E10, D12, N11, M13, J13, G13, F13, E13, D13, C13, B13, F9, A13, E9, E8, D8, D7, N8, D11, D10, C12, B12, B11, A12, C11, A10, B10, B9, D9, C10, A9, C9, B8, A8, D6, B7, C8, A7, D5, C5, E4, C7, F5, E5, A6, E6, B6, C6, A5, B5, A3, A2, A1, B1, A4, B2, B3, B4, D3	GPIO	General Purpose Input Output Pins (pin numbers in parentheses represent interrupt-only or non-functional GPIOs)	Note 8, Note 11

## 2.4.9 GENERAL PURPOSE PASS-THROUGH PORTS INTERFACE

TABLE 2-14: GPIO PASS-THROUGH PORTS

General Purpose Pass-Through Ports				(16 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
125	B9	GPTP-IN0	General Purpose Pass Through Port Input 0	Note 8
127	C10	GPTP-IN1	General Purpose Pass Through Port Input 1	
133	B7	GPTP-IN2	General Purpose Pass Through Port Input 2	Note 8
73	N12	GPTP-IN3	General Purpose Pass Through Port Input 3	Note 8
131	A8	GPTP-IN4	General Purpose Pass Through Port Input 4	Note 8
95	E11	GPTP-IN5	General Purpose Pass Through Port Input 5	Note 8
71	L12	GPTP-IN6	General Purpose Pass Through Port Input 6	Note 8
61	L9	GPTP-IN7	General Purpose Pass Through Port Input 7	Note 8
126	D9	GPTP-OUT0	General Purpose Pass Through Port Output 0	
132	D6	GPTP-OUT1	General Purpose Pass Through Port Output 1	
134	C8	GPTP-OUT2	General Purpose Pass Through Port Output 2	
74	N10	GPTP-OUT3	General Purpose Pass Through Port Output 3	
130	B8	GPTP-OUT4	General Purpose Pass Through Port Output 4	
96	E10	GPTP-OUT5	General Purpose Pass Through Port Output 5	
72	K12	GPTP-OUT6	General Purpose Pass Through Port Output 6	
135	A7	GPTP-OUT7	General Purpose Pass Through Port Output 7	

## 2.4.10 MISCELLANEOUS FUNCTIONS

TABLE 2-15: MISCELLANEOUS FUNCTIONS

MISC Functions				(17 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
139	C7	A20M	KBD GATEA20 Output	Note 2
24	F1	KBRST	CPU_RESET	Note 2
153	B2	LED0	LED Output 1	
154	B3	LED1	LED Output 2	
155	B4	LED2	LED Output 3	
112	D8	MSCLK	SMSC Proprietary EC debug port	Note 10
113	D7	MSDATA	SMSC Proprietary EC debug port	Note 10
124	B10	UART_CLK	UART CLK input	
123	A10	UART_RX	UART RX Input	Note 13
122	C11	UART_TX	UART TX Output	Note 13
15	F3	VCC_PWRGD	System Main Power Indication	
20	H1	RC_ID	RC Identification Detection	
13	F4	RESETI#	System Reset Input	
14	G2	RESETO#	System Reset Output	Note 20
141	E5	CEC_IN	HDMI-CEC control bus input	
140	F5	CEC_OUT	HDMI-CEC control bus output	
16	F2	nRESET_OUT	EC-driven External System Reset	

**Note 2-3** The KBRST pin function is the output of CPU\_RESET described in [Section 9.14, "CPU\\_RESET Hardware Speed-Up,"](#) on page 214.

**Note 2-4** When the CLK\_SRC bit is '1' in the [Configuration Select](#) register (pg. 254), the baud clock is externally sourced from the UART\_CLK pin. UART\_CLK requires a frequency of 1.8432 MHz  $\pm$  2%.

# MEC1618/MEC1618i

**Note 2-5** The nRESET\_OUT pin function is an external output signal version of the internal signal nSIO\_RESET. See the iRESET\_OUT bit in the [Block Sleep Enable Registers on page 118](#).

## 2.4.11 PS/2 INTERFACE

**TABLE 2-16: PS/2 INTERFACE**

PS/2 Interface				(10 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
91	F11	PS2_CLK0B	PS/2 clock output	Note 15
92	F10	PS2_DAT0B	PS/2 data	Note 15
95	E11	PS2_CLK2	PS/2 clock output	
96	E10	PS2_DAT2	PS/2 data	
97	D12	PS2_CLK1A	PS/2 clock output	Note 15
98	N11	PS2_DAT1A	PS/2 data	Note 15
99	M13	PS2_CLK0A	PS/2 clock output	Note 15
100	J13	PS2_DAT0A	PS/2 data	Note 15
93	F12	PS2_CLK1B	PS/2 clock output	Note 15
94	E12	PS2_DAT1B	PS/2 data	Note 15

## 2.4.12 POWER INTERFACE

**TABLE 2-17: POWER INTERFACE**

Power Interface				(14 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
2	E3	BGND	VBAT associated ground	
3	C4	VBAT	VBAT supply	
23	H4	VR_CAP	Internal Voltage Regulator Output (Capacitor Required)	
19	J4	VSS0	VTR associated ground 1	
101	J6	VSS1	VTR associated ground 2	
17	H5	VSS_RO	VTR associated ground used for ring oscillator.	
18	G4	VTR0	VTR supply 1	
49	K6	VTR1	VTR supply 2	
75	J8	VTR2	VTR supply 3	
31	G3	VTR3	VTR supply 4	
44	K4	VSS_ADC	Analog ADC VTR associated ground	
22	J5	VTR_REG	VTR Internal Voltage Regulator Supply	
86	H9	VTR_FLASH	VTR Internal Flash Supply	
25	J1	AVTR_ADC	Analog ADC VTR associated Supply	

**APPLICATION NOTE:** VBAT to VTR switching must be done externally as described in [Section 5.6.9, "Power MUX," on page 109](#).

## 2.4.13 SMBUS INTERFACE

**TABLE 2-18: SMBUS INTERFACE**

SMBus Interface				(24 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
77	H13	SMB00_CLK	SMBus Controller Port 0 Clock	
76	K13	SMB00_DATA	SMBus Controller Port 0 Data	
79	J12	SMB01_CLK	SMBus Controller Port 1 Clock	
78	K11	SMB01_DATA	SMBus Controller Port 1 Data	
94	E12	SMB02_CLK	SMBus Controller Port 2 Clock	
93	F12	SMB02_DATA	SMBus Controller Port 2 Data	
92	F10	SMB03_CLK	SMBus Controller Port 3 Clock	
91	F11	SMB03_DATA	SMBus Controller Port 3 Data	
90	L11	SMB04_CLK	SMBus Controller Port 4 Clock	
89	H10	SMB04_DATA	SMBus Controller Port 4 Data	
88	K10	SMB05_CLK	SMBus Controller Port 5 Clock	
87	J10	SMB05_DATA	SMBus Controller Port 5 Data	
85	G12	SMB06_CLK	SMBus Controller Port 6 Clock	
84	G11	SMB06_DATA	SMBus Controller Port 6 Data	
81	H11	SMB07_CLK	SMBus Controller Port 7 Clock	
80	J11	SMB07_DATA	SMBus Controller Port 7 Data	
120	A12	SMB08_CLK	SMBus Controller Port 8 Clock	
119	B11	SMB08_DATA	SMBus Controller Port 8 Data	
118	B12	SMB09_CLK	SMBus Controller Port 9 Clock	
117	C12	SMB09_DATA	SMBus Controller Port 9 Data	
83	G10	SMB10_CLK	SMBus Controller Port 10 Clock	
82	H12	SMB10_DATA	SMBus Controller Port 10 Data	
149	A2	SB-TSI_CLK	SMBus Controller AMD-TSI Port Clock	
148	A3	SB-TSI_DAT	SMBus Controller AMD-TSI Port Data	

## 2.4.14 PECI INTERFACE

**TABLE 2-19: PECI INTERFACE**

PECI Interface				(2 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
148	A3	PECI_DAT	PECI Bus	
150	A1	VREF_VTT	Processor Interface Voltage Reference	

## 2.4.15 16-BIT COUNTER/TIMER INTERFACE

**TABLE 2-20: 16-BIT COUNTER/TIMER INTERFACE**

16-Bit Counter/Timer Interface				(8 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
124	B10	TIN0	16-Bit Counter/Timer Input 1	
125	B9	TIN1	16-Bit Counter/Timer Input 2	
126	D9	TIN2	16-Bit Counter/Timer Input 3	
127	C10	TIN3	16-Bit Counter/Timer Input 4	
132	D6	TOUT0	16-Bit Counter/Timer Output 1	
133	B7	TOUT1	16-Bit Counter/Timer Output 2	
134	C8	TOUT2	16-Bit Counter/Timer Output 3	
135	A7	TOUT3	16-Bit Counter/Timer Output 4	

# MEC1618/MEC1618i

## 2.4.16 KEYBOARD SCAN INTERFACE

TABLE 2-21: KEYBOARD SCAN INTERFACE

Keyboard Scan Interface				(26 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
135	A7	KSI0	Keyboard Scan Matrix Input 0	
152	A4	KSI1	Keyboard Scan Matrix Input 1	
21	H3	KSI2	Keyboard Scan Matrix Input 2	
125	B9	KSI3	Keyboard Scan Matrix Input 3	
126	D9	KSI4	Keyboard Scan Matrix Input 4	
127	C10	KSI5	Keyboard Scan Matrix Input 5	
132	D6	KSI6	Keyboard Scan Matrix Input 6	
133	B7	KSI7	Keyboard Scan Matrix Input 7	
134	C8	KSO0	Keyboard Scan Matrix Output 0	
142	A6	KSO1	Keyboard Scan Matrix Output 1	
143	E6	KSO2	Keyboard Scan Matrix Output 2	
144	B6	KSO3	Keyboard Scan Matrix Output 3	
128	A9	KSO4	Keyboard Scan Matrix Output 4	
97	D12	KSO5	Keyboard Scan Matrix Output 5	
98	N11	KSO6	Keyboard Scan Matrix Output 6	
129	C9	KSO7	Keyboard Scan Matrix Output 7	
145	C6	KSO8	Keyboard Scan Matrix Output 8	
146	A5	KSO9	Keyboard Scan Matrix Output 9	
147	B5	KSO10	Keyboard Scan Matrix Output 10	
130	B8	KSO11	Keyboard Scan Matrix Output 11	
131	A8	KSO12	Keyboard Scan Matrix Output 12	
151	B1	KSO13	Keyboard Scan Matrix Output 13	
84	G11	KSO14	Keyboard Scan Matrix Output 14	
73	N12	KSO15	Keyboard Scan Matrix Output 15	
74	N10	KSO16	Keyboard Scan Matrix Output 16	
156	D3	KSO17	Keyboard Scan Matrix Output 17	

## 2.4.17 VCI

TABLE 2-22: VCI INTERFACE

VBAT-Powered Control Interface				(9 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
9	E2	VCI_OVRD_IN	Input can cause wakeup or interrupt event	
5	D1	VCI_OUT	OUTPUT from combinational logic and/or EC	
4	C1	BGPO0	VBAT driven GPO	
8	D2	VCI_IN0#	Input can cause wakeup or interrupt event	
7	G1	VCI_IN1#	Input can cause wakeup or interrupt event	
6	C2	VCI_IN2#	Input can cause wakeup or interrupt event	
10	E1	VCI_IN3#	Input can cause wakeup or interrupt event	
11	C3	VCI_IN4#	Input can cause wakeup or interrupt event	
12	H2	VCI_IN5#	Input can cause wakeup or interrupt event	

## 2.4.18 SPI CONTROLLERS INTERFACE

TABLE 2-23: SPI CONTROLLERS INTERFACE

SPI Controllers Interface				(7 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
77	J11	ECGP_SCLK	General Purpose SPI Clock	
79	L10	ECGP_SOUT	General Purpose SPI Output	
81	L9	ECGP_SIN	General Purpose SPI Input	
104	K11	FLSCLK	Flash Interface SPI Clock	
105	L11	FLSOUT	Flash Interface SPI Output	
106	K10	FLSIN	Flash Interface SPI Input	
107	M12	FLSCS#	Flash Interface SPI Chip Select	

**Note 2-6** For [General Purpose Serial Peripheral Interface \(GP-SPI\)](#) interface pins with 8 mA buffers the maximum SPCLK pin clock frequency is 16.13 MHz for all modes. Limited functionality is available at higher frequencies, but performance is not guaranteed (see [Table 30-14, "SPI\\_CLK Frequencies,"](#) on page 513 and [Section 30.9.5.5, "Limits of SPI configurations,"](#) on page 506).

## 2.4.19 GANG PROGRAMMING INTERFACE

TABLE 2-24: GANG PROGRAMMING INTERFACE

Gang Programmer Interface				(14 Pins)
Pin Ref. Number	Ball	Signal Name	Description	Notes
15	F3	GANG_MODE	Gang Programmer operating mode control	Note 21
14	G2	GANG_START	Gang Programmer start control	Note 21
20	H1	GANG_STROBE	Gang Programmer data latch strobe	Note 21
21	H3	GANG_FULL	Gang Programmer data flow control	Note 21
89	H10	GANG_BUSY	Gang Programmer operational status	Note 21
24	F1	GANG_ERROR	Gang Programmer error status	Note 21
100	J13	GANG_DATA0	Gang Programmer Flash program data 0	Note 21
98	N11	GANG_DATA1	Gang Programmer Flash program data 1	Note 21
96	E10	GANG_DATA2	Gang Programmer Flash program data 2	Note 21
94	E12	GANG_DATA3	Gang Programmer Flash program data 3	Note 21
92	F10	GANG_DATA4	Gang Programmer Flash program data 4	Note 21
90	L11	GANG_DATA5	Gang Programmer Flash program data 5	Note 21
88	K10	GANG_DATA6	Gang Programmer Flash program data 6	Note 21
85	G12	GANG_DATA7	Gang Programmer Flash program data 7	Note 21

## 2.5 Pin Multiplexing

Multifunction [Pin Multiplexing](#) in the MEC1618/MEC1618i is controlled by the [GPIO Interface](#) and illustrated in the [Multiplexing Tables](#) that follow. See [Section 2.6, "Notes for Tables in this Chapter,"](#) on page 44 for notes that are referenced in the [Pin Multiplexing](#) tables. See [Section 21.9.1, "Pin Control Register,"](#) on page 406 for [Pin Multiplexing](#) programming details. See also [Section 21.5, "Pin Multiplexing Control,"](#) on page 401.

Pin signal functions that exhibit power domain emulation (see [VCC Power Domain Emulation](#) and [VTR Power Domain Emulation](#) below) have a different power supply designation in the "Emulated Power Well" column and "Signal Power Well" columns of the [Multiplexing Tables](#) in [Section 2.5.3](#). See also [Section 2.3.1, "Pin Default State Through Power Transitions,"](#) on page 11 for a description of pin states through power transitions.

## 2.5.1 VCC POWER DOMAIN EMULATION

Pin signal functions that exhibit [VCC Power Domain Emulation](#) are documented in the [Multiplexing Tables](#) as "Signal Power Well" = VTR and "Emulated Power Well" = VCC. The System Runtime Supply power is not connected to the MEC1618/MEC1618i. The VCC\_PWRGD signal is used to indicate when power is applied to the System Runtime Supply. All pin signal functions that exhibit VCC power domain emulation are powered by VTR and controlled by the VCC\_PWRGD signal input. VCC power domain emulation pin signal functions are tri-stated when VCC\_PWRGD is not asserted and are functional when VCC\_PWRGD is active.

# MEC1618/MEC1618i

## 2.5.2 VTR POWER DOMAIN EMULATION

Pin Signal Functions that exhibit [VTR Power Domain Emulation](#) are documented in the [Multiplexing Tables](#) as “Signal Power Well”= VBAT and “Emulated Power Well” = VTR. All pin signal functions that exhibit [VTR Power Domain Emulation](#) are powered by VBAT and controlled by the internal VTR POR. VTR power domain emulation pin signal functions are tristated when VTR power is not applied and are functional when VTR power is applied.

## 2.5.3 MULTIPLEXING TABLES

**TABLE 2-25: MULTIPLEXING TABLE (1 OF 20)**

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
1	D4	Default: 0	GPIO165	(I/O/OD)-4 mA	VTR	VTR/VCC	Note 5
1	D4	1	32KHZ_IN	IS	VBAT	VBAT	
1	D4	2	Reserved	Reserved	Reserved	Reserved	
1	D4	3	Reserved	Reserved	Reserved	Reserved	
2	E3		BGND	PWR	PWR	PWR	
2	E3						
2	E3						
2	E3						
3	C4		VBAT	PWR	PWR	PWR	
3	C4						
3	C4						
3	C4						
4	C1	Default: 0	BGPO0	0-4 mA	VBAT	VBAT	
4	C1	1	Reserved	Reserved	Reserved	Reserved	
4	C1	2	Reserved	Reserved	Reserved	Reserved	
4	C1	3	Reserved	Reserved	Reserved	Reserved	
5	D1	Default: 0	VCI_OUT	0-4 mA	VBAT	VBAT	
5	D1	1	Reserved	Reserved	Reserved	Reserved	
5	D1	2	Reserved	Reserved	Reserved	Reserved	
5	D1	3	Reserved	Reserved	Reserved	Reserved	
6	C2	0	GPIO161	(I/O/OD)-4 mA	VTR	VTR/VCC	
6	C2	Default: 1	VCI_IN2#	IS	VBAT	VBAT	
6	C2	2	Reserved	Reserved	Reserved	Reserved	
6	C2	3	Reserved	Reserved	Reserved	Reserved	
7	G1	0	GPIO162	(I/O/OD)-4 mA	VTR	VTR/VCC	
7	G1	Default: 1	VCI_IN1#	IS	VBAT	VBAT	
7	G1	2	Reserved	Reserved	Reserved	Reserved	
7	G1	3	Reserved	Reserved	Reserved	Reserved	
8	D2	0	GPIO163	(I/O/OD)-4 mA	VTR	VTR/VCC	
8	D2	Default: 1	VCI_IN0#	IS	VBAT	VBAT	
8	D2	2	Reserved	Reserved	Reserved	Reserved	
8	D2	3	Reserved	Reserved	Reserved	Reserved	



TABLE 2-26: MULTIPLEXING TABLE (2 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
9	E2	0	GPIO164	(I/O/OD)-4 mA	VTR	VTR/VCC	
9	E2	Default: 1	VCI_OVRD_IN	IS	VBAT	VBAT	
9	E2	2	Reserved	Reserved	Reserved	Reserved	
9	E2	3	Reserved	Reserved	Reserved	Reserved	
10	E1	0	GPIO000	(I/O/OD)-4 mA	VTR	VTR/VCC	
10	E1	Default: 1	VCI_IN3#	IS	VBAT	VBAT	
10	E1	2	Reserved	Reserved	Reserved	Reserved	
10	E1	3	Reserved	Reserved	Reserved	Reserved	
11	C3	0	GPIO234	(I/O/OD)-4 mA	VTR	VTR/VCC	
11	C3	Default: 1	VCI_IN4#	IS	VBAT	VBAT	
11	C3	2	Reserved	Reserved	Reserved	Reserved	
11	C3	3	Reserved	Reserved	Reserved	Reserved	
12	H2	0	GPIO235	(I/O/OD)-4 mA	VTR	VTR/VCC	
12	H2	Default: 1	VCI_IN5#	IS	VBAT	VBAT	
12	H2	2	Reserved	Reserved	Reserved	Reserved	
12	H2	3	Reserved	Reserved	Reserved	Reserved	
13	F4	Default: 0	RESETI#	I	VTR	Reserved	
13	F4	1	Reserved	Reserved	Reserved	Reserved	
13	F4	2	Reserved	Reserved	Reserved	Reserved	
13	F4	3	Reserved	Reserved	Reserved	Reserved	
14	G2	Default: 0	GPIO062	(I/O/OD)-4 mA	VTR	VTR/VCC	Note 22
14	G2	1	Reserved	Reserved	Reserved	Reserved	
14	G2	2	Reserved	Reserved	Reserved	Reserved	
14	G2	3	Reserved	Reserved	Reserved	Reserved	
15	F3	0	Reserved	Reserved	Reserved	Reserved	
15	F3	Default: 1	VCC_PWRGD	I	VTR	VTR	
15	F3	2	Reserved	Reserved	Reserved	Reserved	
15	F3	3	Reserved	Reserved	Reserved	Reserved	
16	F2	Default: 0	GPIO106	(I/O/OD)-4 mA	VTR	VTR/VCC	
16	F2	1	nRESET_OUT	O-4 mA	VTR	VTR	
16	F2	2	Reserved	Reserved	Reserved	Reserved	
16	F2	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-27: MULTIPLEXING TABLE (3 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
17	H5		VSS_RO	PWR	PWR	PWR	
17	H5						
17	H5						
17	H5						
18	G4		VTR0	PWR	PWR	PWR	
18	G4						
18	G4						
18	G4						
19	J4		VSS0	PWR	PWR	PWR	
19	J4						
19	J4						
19	J4						
20	H1	Default: 0	GPIO033	(I/O/OD)-12 mA	VTR	VTR/VCC	
20	H1	1	RC_ID	I_AN-OD12 mA	VTR	VTR	
20	H1	2	Reserved	Reserved	Reserved	Reserved	
20	H1	3	Reserved	Reserved	Reserved	Reserved	
21	H3	Default: 0	GPIO021	(I/O/OD)-4 mA	VTR	VTR/VCC	
21	H3	1	KSI2	IS	VTR	VTR	
21	H3	2	Reserved	Reserved	Reserved	Reserved	
21	H3	3	Reserved	Reserved	Reserved	Reserved	
22	J5		VTR_REG	PWR	PWR	PWR	
22	J5						
22	J5						
22	J5						
23	H4		VR_CAP	PWR	PWR	PWR	
23	H4						
23	H4						
23	H4						
24	F1	Default: 0	GPIO060	(I/O/OD)-4 mA	VTR	VTR/VCC	
24	F1	1	KBRST	OD-4 mA	VTR	VCC	Note 2
24	F1	2	Reserved	Reserved	Reserved	Reserved	
24	F1	3	Reserved	Reserved	Reserved	Reserved	

TABLE 2-28: MULTIPLEXING TABLE (4 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
25	J1		AVTR_ADC	PWR	PWR	PWR	
25	J1						
25	J1						
25	J1						
26	J3		VREF_ADC	PWR	PWR	PWR	
26	J3						
26	J3						
26	J3						
27	K3	0	GPIO200	(I/O/OD)-4 mA	VTR	VTR/VCC	
27	K3	Default: 1	ADC0	ADC_IN	AVTR_ADC	AVTR_ADC	
27	K3	2	Reserved	Reserved	Reserved	Reserved	
27	K3	3	Reserved	Reserved	Reserved	Reserved	
28	J2	0	GPIO210	(I/O/OD)-4 mA	VTR	VTR/VCC	
28	J2	Default: 1	ADC8	ADC_IN	AVTR_ADC	AVTR_ADC	
28	J2	2	Reserved	Reserved	Reserved	Reserved	
28	J2	3	Reserved	Reserved	Reserved	Reserved	
29	K1	0	GPIO201	(I/O/OD)-4 mA	VTR	VTR/VCC	
29	K1	Default: 1	ADC1	ADC_IN	AVTR_ADC	AVTR_ADC	
29	K1	2	Reserved	Reserved	Reserved	Reserved	
29	K1	3	Reserved	Reserved	Reserved	Reserved	
30	K2	0	GPIO211	(I/O/OD)-4 mA	VTR	VTR/VCC	
30	K2	Default: 1	ADC9	ADC_IN	AVTR_ADC	AVTR_ADC	
30	K2	2	Reserved	Reserved	Reserved	Reserved	
30	K2	3	Reserved	Reserved	Reserved	Reserved	
31	G3		VTR3	PWR	PWR	PWR	
31	G3						
31	G3						
31	G3						
32	L2	0	GPIO202	(I/O/OD)-4 mA	VTR	VTR/VCC	
32	L2	Default: 1	ADC2	ADC_IN	AVTR_ADC	AVTR_ADC	
32	L2	2	Reserved	Reserved	Reserved	Reserved	
32	L2	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-29: MULTIPLEXING TABLE (5 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
33	L1	0	GPIO212	(I/O/OD)-4 mA	VTR	VTR/VCC	
33	L1	Default: 1	ADC10	ADC_IN	AVTR_ADC	AVTR_ADC	
33	L1	2	Reserved	Reserved	Reserved	Reserved	
33	L1	3	Reserved	Reserved	Reserved	Reserved	
34	M1	0	GPIO203	(I/O/OD)-4 mA	VTR	VTR/VCC	
34	M1	Default: 1	ADC3	ADC_IN	AVTR_ADC	AVTR_ADC	
34	M1	2	Reserved	Reserved	Reserved	Reserved	
34	M1	3	Reserved	Reserved	Reserved	Reserved	
35	L3	0	GPIO213	(I/O/OD)-4 mA	VTR	VTR/VCC	
35	L3	Default: 1	ADC11	ADC_IN	AVTR_ADC	AVTR_ADC	
35	L3	2	Reserved	Reserved	Reserved	Reserved	
35	L3	3	Reserved	Reserved	Reserved	Reserved	
36	L4	0	GPIO204	(I/O/OD)-4 mA	VTR	VTR/VCC	
36	L4	Default: 1	ADC4	ADC_IN	AVTR_ADC	AVTR_ADC	
36	L4	2	Reserved	Reserved	Reserved	Reserved	
36	L4	3	Reserved	Reserved	Reserved	Reserved	
37	M2	0	GPIO214	(I/O/OD)-4 mA	VTR	VTR/VCC	
37	M2	Default: 1	ADC12	ADC_IN	AVTR_ADC	AVTR_ADC	
37	M2	2	Reserved	Reserved	Reserved	Reserved	
37	M2	3	Reserved	Reserved	Reserved	Reserved	
38	N1	0	GPIO205	(I/O/OD)-4 mA	VTR	VTR/VCC	
38	N1	Default: 1	ADC5	ADC_IN	AVTR_ADC	AVTR_ADC	
38	N1	2	Reserved	Reserved	Reserved	Reserved	
38	N1	3	Reserved	Reserved	Reserved	Reserved	
39	N2	0	GPIO215	(I/O/OD)-4 mA	VTR	VTR/VCC	
39	N2	Default: 1	ADC13	ADC_IN	AVTR_ADC	AVTR_ADC	
39	N2	2	Reserved	Reserved	Reserved	Reserved	
39	N2	3	Reserved	Reserved	Reserved	Reserved	
40	M3	0	GPIO206	(I/O/OD)-4 mA	VTR	VTR/VCC	
40	M3	Default: 1	ADC6	ADC_IN	AVTR_ADC	AVTR_ADC	
40	M3	2	Reserved	Reserved	Reserved	Reserved	
40	M3	3	Reserved	Reserved	Reserved	Reserved	

TABLE 2-30: MULTIPLEXING TABLE (6 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
41	M4	0	GPIO216	(I/O/OD)-4 mA	VTR	VTR/VCC	
41	M4	Default: 1	ADC14	ADC_IN	AVTR_ADC	AVTR_ADC	
41	M4	2	Reserved	Reserved	Reserved	Reserved	
41	M4	3	Reserved	Reserved	Reserved	Reserved	
42	N3	0	GPIO207	(I/O/OD)-4 mA	VTR	VTR/VCC	
42	N3	Default: 1	ADC7	ADC_IN	AVTR_ADC	AVTR_ADC	
42	N3	2	Reserved	Reserved	Reserved	Reserved	
42	N3	3	Reserved	Reserved	Reserved	Reserved	
43	N4	0	GPIO217	(I/O/OD)-4 mA	VTR	VTR/VCC	
43	N4	Default: 1	ADC15	ADC_IN	AVTR_ADC	AVTR_ADC	
43	N4	2	Reserved	Reserved	Reserved	Reserved	
43	N4	3	Reserved	Reserved	Reserved	Reserved	
44	K4		VSS_ADC	PWR	PWR	PWR	
44	K4						
44	K4						
44	K4						
45	M5	0	Reserved	Reserved	Reserved	Reserved	
45	M5	Default: 1	LRESET#	PCI_I	VTR	VCC	Note 5, Note 2
45	M5	2	Reserved	Reserved	Reserved	Reserved	
45	M5	3	Reserved	Reserved	Reserved	Reserved	
46	L5	0	Reserved	Reserved	Reserved	Reserved	
46	L5	Default: 1	CLKRUN#	PCI_OD	VTR	VCC	Note 5, Note 2
46	L5	2	Reserved	Reserved	Reserved	Reserved	
46	L5	3	Reserved	Reserved	Reserved	Reserved	
47	L6	0	Reserved	Reserved	Reserved	Reserved	
47	L6	Default: 1	LFRAME#	PCI_I	VTR	VCC	Note 5, Note 3
47	L6	2	Reserved	Reserved	Reserved	Reserved	
47	L6	3	Reserved	Reserved	Reserved	Reserved	
48	N5	0	Reserved	Reserved	Reserved	Reserved	
48	N5	Default: 1	SER_IRQ	PCI_IO	VTR	VCC	Note 4, Note 5, Note 2
48	N5	2	Reserved	Reserved	Reserved	Reserved	
48	N5	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-31: MULTIPLEXING TABLE (7 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
49	K6		VTR1	PWR	PWR	PWR	
49	K6						
49	K6						
49	K6						
50	N6	0	Reserved	Reserved	Reserved	Reserved	
50	N6	Default: 1	PCI_CLK	PCI_ICLK	VTR	VCC	Note 5, Note 2
50	N6	2	Reserved	Reserved	Reserved	Reserved	
50	N6	3	Reserved	Reserved	Reserved	Reserved	
51	M6	Default: 0	LAD0	PCI_IO	VTR	VCC	Note 4, Note 5, Note 2
51	M6	1	Reserved	Reserved	Reserved	Reserved	
51	M6	2	Reserved	Reserved	Reserved	Reserved	
51	M6	3	Reserved	Reserved	Reserved	Reserved	
52	M7	Default: 0	LAD1	PCI_IO	VTR	VCC	Note 4, Note 5, Note 2
52	M7	1	Reserved	Reserved	Reserved	Reserved	
52	M7	2	Reserved	Reserved	Reserved	Reserved	
52	M7	3	Reserved	Reserved	Reserved	Reserved	
53	L7	Default: 0	LAD2	PCI_IO	VTR	VCC	Note 4, Note 5, Note 2
53	L7	1	Reserved	Reserved	Reserved	Reserved	
53	L7	2	Reserved	Reserved	Reserved	Reserved	
53	L7	3	Reserved	Reserved	Reserved	Reserved	
54	K7	Default: 0	LAD3	PCI_IO	VTR	VCC	Note 4, Note 5, Note 2
54	K7	1	Reserved	Reserved	Reserved	Reserved	
54	K7	2	Reserved	Reserved	Reserved	Reserved	
54	K7	3	Reserved	Reserved	Reserved	Reserved	
55	K5	Default: 0	GPIO100	(I/O/OD)-4 mA	VTR	VTR/VCC	
55	K5	1	nEC_SCI	OD-4 mA	VTR	VTR	Note 3
55	K5	2	Reserved	Reserved	Reserved	Reserved	
55	K5	3	Reserved	Reserved	Reserved	Reserved	
56	N7	Default: 0	GPIO011	(I/O/OD)-4 mA	VTR	VTR/VCC	
56	N7	1	nSMI	OD-4 mA	VTR	VTR	Note 2
56	N7	2	Reserved	Reserved	Reserved	Reserved	
56	N7	3	Reserved	Reserved	Reserved	Reserved	

TABLE 2-32: MULTIPLEXING TABLE (8 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
57	L8	Default: 0	GPIO061	(I/O/OD)-4 mA	VTR	VTR/VCC	
57	L8	1	LPCPD#	I	VTR	VCC	
57	L8	2	Reserved	Reserved	Reserved	Reserved	
57	L8	3	Reserved	Reserved	Reserved	Reserved	
58	M8	Default: 0	GPIO050	(I/O/OD)-4 mA	VTR	VTR/VCC	
58	M8	1	FAN_TACH0	I	VTR	VTR	
58	M8	2	Reserved	Reserved	Reserved	Reserved	
58	M8	3	Reserved	Reserved	Reserved	Reserved	
59	N9	Default: 0	GPIO051	(I/O/OD)-4 mA	VTR	VTR/VCC	
59	N9	1	FAN_TACH1	I	VTR	VTR	
59	N9	2	Reserved	Reserved	Reserved	Reserved	
59	N9	3	Reserved	Reserved	Reserved	Reserved	
60	M9	Default: 0	GPIO052	(I/O/OD)-4 mA	VTR	VTR/VCC	
60	M9	1	FAN_TACH2	I	VTR	VTR	
60	M9	2	Reserved	Reserved	Reserved	Reserved	
60	M9	3	Reserved	Reserved	Reserved	Reserved	
61	L9	Default: 0	GPIO016	(I/O/OD)-4 mA	VTR	VTR/VCC	
61	L9	1	GPTP-IN7	I	VTR	VTR/VCC	Note 8
61	L9	2	FAN_TACH3	I	VTR	VTR	
61	L9	3	Reserved	Reserved	Reserved	Reserved	
62	J9	Default: 0	GPIO230	(I/O/OD)-4 mA	VTR	VTR/VCC	
62	J9	1	ECGP_SCLK	0-4 mA	VTR	VTR	
62	J9	2	Reserved	Reserved	Reserved	Reserved	
62	J9	3	Reserved	Reserved	Reserved	Reserved	
63	N13	Default: 0	GPIO053	(I/O/OD)-4 mA	VTR	VTR/VCC	
63	N13	1	PWM0	(O/OD)-4 mA	VTR	VTR	
63	N13	2	Reserved	Reserved	Reserved	Reserved	
63	N13	3	Reserved	Reserved	Reserved	Reserved	
64	K9	Default: 0	GPIO231	(I/O/OD)-8 mA	VTR	VTR/VCC	
64	K9	1	ECGP_SOUT	(I/O/OD)-8 mA	VTR	VTR	
64	K9	2	Reserved	Reserved	Reserved	Reserved	
64	K9	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-33: MULTIPLEXING TABLE (9 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
65	M10	Default: 0	GPIO054	(I/O/OD)-4 mA	VTR	VTR/VCC	
65	M10	1	PWM1	(O/OD)-4 mA	VTR	VTR	
65	M10	2	Reserved	Reserved	Reserved	Reserved	
65	M10	3	Reserved	Reserved	Reserved	Reserved	
66	K8	Default: 0	GPIO233	(I/O/OD)-8 mA	VTR	VTR/VCC	
66	K8	1	ECGP_SIN	(I/O/OD)-8 mA	VTR	VTR	
66	K8	2	Reserved	Reserved	Reserved	Reserved	
66	K8	3	Reserved	Reserved	Reserved	Reserved	
67	M11	Default: 0	GPIO055	(I/O/OD)-4 mA	VTR	VTR/VCC	
67	M11	1	PWM2	(O/OD)-4 mA	VTR	VTR	
67	M11	2	Reserved	Reserved	Reserved	Reserved	
67	M11	3	Reserved	Reserved	Reserved	Reserved	
68	L13	Default: 0	GPIO056	(I/O/OD)-4 mA	VTR	VTR/VCC	
68	L13	1	PWM3	(O/OD)-4 mA	VTR	VTR	
68	L13	2	Reserved	Reserved	Reserved	Reserved	
68	L13	3	Reserved	Reserved	Reserved	Reserved	
69	M12	Default: 0	GPIO001	(I/O/OD)-4 mA	VTR	VTR/VCC	
69	M12	1	PWM4	(O/OD)-4 mA	VTR	VTR	
69	M12	2	Reserved	Reserved	Reserved	Reserved	
69	M12	3	Reserved	Reserved	Reserved	Reserved	
70	L10	Default: 0	GPIO002	(I/O/OD)-4 mA	VTR	VTR/VCC	
70	L10	1	PWM5	(O/OD)-4 mA	VTR	VTR	
70	L10	2	Reserved	Reserved	Reserved	Reserved	
70	L10	3	Reserved	Reserved	Reserved	Reserved	
71	L12	Default: 0	GPIO014	(I/O/OD)-4 mA	VTR	VTR/VCC	
71	L12	1	GPTP-IN6	I	VTR	VTR/VCC	Note 8
71	L12	2	PWM6	(O/OD)-4 mA	VTR	VTR	
71	L12	3	Reserved	Reserved	Reserved	Reserved	
72	K12	Default: 0	GPIO015	(I/O/OD)-4 mA	VTR	VTR/VCC	
72	K12	1	GPTP-OUT6	(O/OD)-4 mA	VTR	VTR/VCC	
72	K12	2	PWM7	(O/OD)-4 mA	VTR	VTR	
72	K12	3	Reserved	Reserved	Reserved	Reserved	



TABLE 2-34: MULTIPLEXING TABLE (10 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
73	N12	Default: 0	GPIO151	(I/O/OD)-4 mA	VTR	VTR/VCC	Note 8
73	N12	1	GPTP-IN3	I	VTR	VTR/VCC	
73	N12	2	FAN_TACH4	I	VTR	VTR	
73	N12	3	KSO15	(O/OD)-4 mA	VTR	VTR	
74	N10	Default: 0	GPIO152	(I/O/OD)-4 mA	VTR	VTR/VCC	
74	N10	1	GPTP-OUT3	(O/OD)-4 mA	VTR	VTR/VCC	
74	N10	2	FAN_TACH5	I	VTR	VTR	
74	N10	3	KSO16	(O/OD)-4 mA	VTR	VTR	
75	J8		VTR2	PWR	PWR	PWR	
75	J8						
75	J8						
75	J8						
76	K13	Default: 0	GPIO003	(I/O/OD)-8 mA	VTR	VTR/VCC	
76	K13	1	SMB00_DATA	(I/OD)-8 mA	VTR	VTR	
76	K13	2	Reserved	Reserved	Reserved	Reserved	
76	K13	3	Reserved	Reserved	Reserved	Reserved	
77	H13	Default: 0	GPIO004	(I/O/OD)-8 mA	VTR	VTR/VCC	
77	H13	1	SMB00_CLK	(I/OD)-8 mA	VTR	VTR	
77	H13	2	Reserved	Reserved	Reserved	Reserved	
77	H13	3	Reserved	Reserved	Reserved	Reserved	
78	K11	Default: 0	GPIO005	(I/O/OD)-8 mA	VTR	VTR/VCC	
78	K11	1	SMB01_DATA	(I/OD)-8 mA	VTR	VTR	
78	K11	2	Reserved	Reserved	Reserved	Reserved	
78	K11	3	Reserved	Reserved	Reserved	Reserved	
79	J12	Default: 0	GPIO006	(I/O/OD)-8 mA	VTR	VTR/VCC	
79	J12	1	SMB01_CLK	(I/OD)-8 mA	VTR	VTR	
79	J12	2	Reserved	Reserved	Reserved	Reserved	
79	J12	3	Reserved	Reserved	Reserved	Reserved	
80	J11	Default: 0	GPIO012	(I/O/OD)-8 mA	VTR	VTR/VCC	
80	J11	1	SMB07_DATA	(I/OD)-8 mA	VTR	VTR	
80	J11	2	Reserved	Reserved	Reserved	Reserved	
80	J11	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-35: MULTIPLEXING TABLE (11 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
81	H11	Default: 0	GPIO013	(I/O/OD)-8 mA	VTR	VTR/VCC	
81	H11	1	SMB07_CLK	(I/OD)-8 mA	VTR	VTR	
81	H11	2	Reserved	Reserved	Reserved	Reserved	
81	H11	3	Reserved	Reserved	Reserved	Reserved	
82	H12	Default: 0	GPIO130	(I/O/OD)-8 mA	VTR	VTR/VCC	
82	H12	1	SMB10_DATA	(I/OD)-8 mA	VTR	VTR	
82	H12	2	Reserved	Reserved	Reserved	Reserved	
82	H12	3	Reserved	Reserved	Reserved	Reserved	
83	G10	Default: 0	GPIO131	(I/O/OD)-8 mA	VTR	VTR/VCC	
83	G10	1	SMB10_CLK	(I/OD)-8 mA	VTR	VTR	
83	G10	2	Reserved	Reserved	Reserved	Reserved	
83	G10	3	Reserved	Reserved	Reserved	Reserved	
84	G11	Default: 0	GPIO132	(I/O/OD)-8 mA	VTR	VTR/VCC	
84	G11	1	SMB06_DATA	(I/OD)-8 mA	VTR	VTR	
84	G11	2	KSO14	(O/OD)-8 mA	VTR	VTR	
84	G11	3	Reserved	Reserved	Reserved	Reserved	
85	G12	Default: 0	GPIO140	(I/O/OD)-8 mA	VTR	VTR/VCC	
85	G12	1	SMB06_CLK	(I/OD)-8 mA	VTR	VTR	
85	G12	2	PWM13	(O/OD)-8 mA	VTR	VTR	
85	G12	3	Reserved	Reserved	Reserved	Reserved	
86	H9		VTR_FLASH	PWR	PWR	PWR	
86	H9						
86	H9						
86	H9						
87	J10	Default: 0	GPIO141	(I/O/OD)-8 mA	VTR	VTR/VCC	
87	J10	1	SMB05_DATA	(I/OD)-8 mA	VTR	VTR	
87	J10	2	PWM14	O-8 mA	VTR	VTR	
87	J10	3	FLSCLK	O-8 mA	VTR	VTR	
88	K10	Default: 0	GPIO142	(I/O/OD)-8 mA	VTR	VTR/VCC	
88	K10	1	SMB05_CLK	(I/OD)-8 mA	VTR	VTR	
88	K10	2	PWM15	(O/OD)-8 mA	VTR	VTR	
88	K10	3	FLSOUT	(I/O/OD)-8 mA	VTR	VTR	

**TABLE 2-36: MULTIPLEXING TABLE (12 OF 20)**

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
89	H10	Default: 0	GPIO143	(I/O/OD)-8 mA	VTR	VTR/VCC	
89	H10	1	SMB04_DATA	(I/OD)-8 mA	VTR	VTR	
89	H10	2	PWM12	(O/OD)-8 mA	VTR	VTR	
89	H10	3	FLSIN	(I/O/OD)-8 mA	VTR	VTR	
90	L11	Default: 0	GPIO144	(I/O/OD)-8 mA	VTR	VTR/VCC	
90	L11	1	SMB04_CLK	(I/OD)-8 mA	VTR	VTR	
90	L11	2	Reserved	(O/OD)-8 mA	VTR	VTR	
90	L11	3	FLSCS#	O-8 mA	VTR	VTR	
91	F11	Default: 0	GPIO007	(I/O/OD)-12 mA	VTR	VTR/VCC	
91	F11	1	SMB03_DATA	(I/OD)-12 mA	VTR	VTR	
91	F11	2	PS2_CLK0B	(I/OD)-12 mA	VTR	VTR	Note 16
91	F11	3	Reserved	Reserved	Reserved	Reserved	
92	F10	Default: 0	GPIO010	(I/O/OD)-12 mA	VTR	VTR/VCC	
92	F10	1	SMB03_CLK	(I/OD)-12 mA	VTR	VTR	
92	F10	2	PS2_DAT0B	(I/OD)-12 mA	VTR	VTR	Note 16
92	F10	3	Reserved	Reserved	Reserved	Reserved	
93	F12	Default: 0	GPIO154	(I/O/OD)-12 mA	VTR	VTR/VCC	
93	F12	1	SMB02_DATA	(I/OD)-12 mA	VTR	VTR	
93	F12	2	PS2_CLK1B	(I/OD)-12 mA	VTR	VTR	Note 16
93	F12	3	Reserved	Reserved	Reserved	Reserved	
94	E12	Default: 0	GPIO155	(I/O/OD)-12 mA	VTR	VTR/VCC	
94	E12	1	SMB02_CLK	(I/OD)-12 mA	VTR	VTR	
94	E12	2	PS2_DAT1B	(I/OD)-12 mA	VTR	VTR	Note 16
94	E12	3	Reserved	Reserved	Reserved	Reserved	
95	E11	Default: 0	GPIO110	(I/O/OD)-12 mA	VTR	VTR/VCC	
95	E11	1	PS2_CLK2	(I/OD)-12 mA	VTR	VTR	
95	E11	2	GPTP-IN5	I	VTR	VTR/VCC	Note 8
95	E11	3	Reserved	Reserved	Reserved	Reserved	
96	E10	Default: 0	GPIO111	(I/O/OD)-12 mA	VTR	VTR/VCC	
96	E10	1	PS2_DAT2	(I/OD)-12 mA	VTR	VTR	
96	E10	2	GPTP-OUT5	(O/OD)-12 mA	VTR	VTR/VCC	
96	E10	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-37: MULTIPLEXING TABLE (13 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
97	D12	Default: 0	GPIO112	(I/O/OD)-12 mA	VTR	VTR/VCC	
97	D12	1	PS2_CLK1A	(I/O/OD)-12 mA	VTR	VTR	Note 16
97	D12	2	KSO5	(O/OD)-12 mA	VTR	VTR	
97	D12	3	Reserved	Reserved	Reserved	Reserved	
98	N11	Default: 0	GPIO113	(I/O/OD)-12 mA	VTR	VTR/VCC	
98	N11	1	PS2_DAT1A	(I/O/OD)-12 mA	VTR	VTR	Note 16
98	N11	2	KSO6	(O/OD)-12 mA	VTR	VTR	
98	N11	3	Reserved	Reserved	Reserved	Reserved	
99	M13	Default: 0	GPIO114	(I/O/OD)-12 mA	VTR	VTR/VCC	
99	M13	1	PS2_CLK0A	(I/O/OD)-12 mA	VTR	VTR	Note 16
99	M13	2	Reserved	Reserved	Reserved	Reserved	
99	M13	3	Reserved	Reserved	Reserved	Reserved	
100	J13	Default: 0	GPIO115	(I/O/OD)-12 mA	VTR	VTR/VCC	
100	J13	1	PS2_DAT0A	(I/O/OD)-12 mA	VTR	VTR	Note 16
100	J13	2	Reserved	Reserved	Reserved	Reserved	
100	J13	3	Reserved	Reserved	Reserved	Reserved	
101	J6		VSS1	PWR	PWR	PWR	
101	J6						
101	J6						
101	J6						
102	G13	Default: 0	GPIO070	(I/O/OD)-4 mA	Reserved	VTR/VCC	
102	G13	1	Reserved	Reserved	Reserved	Reserved	
102	G13	2	Reserved	Reserved	Reserved	Reserved	
102	G13	3	Reserved	Reserved	Reserved	Reserved	
103	F13	Default: 0	GPIO071	(I/O/OD)-4 mA	Reserved	VTR/VCC	
103	F13	1	Reserved	Reserved	Reserved	Reserved	
103	F13	2	Reserved	Reserved	Reserved	Reserved	
103	F13	3	Reserved	Reserved	Reserved	Reserved	
104	E13	Default: 0	GPIO072	(I/O/OD)-4 mA	Reserved	VTR/VCC	
104	E13	1	Reserved	Reserved	Reserved	Reserved	
104	E13	2	Reserved	Reserved	Reserved	Reserved	
104	E13	3	Reserved	Reserved	Reserved	Reserved	

**TABLE 2-38: MULTIPLEXING TABLE (14 OF 20)**

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
105	D13	Default: 0	GPIO073	(I/O/OD)-4 mA	Reserved	VTR/VCC	
105	D13	1	Reserved	Reserved	Reserved	Reserved	
105	D13	2	Reserved	Reserved	Reserved	Reserved	
105	D13	3	Reserved	Reserved	Reserved	Reserved	
106	C13	Default: 0	GPIO074	(I/O/OD)-4 mA	Reserved	VTR/VCC	
106	C13	1	Reserved	Reserved	Reserved	Reserved	
106	C13	2	Reserved	Reserved	Reserved	Reserved	
106	C13	3	Reserved	Reserved	Reserved	Reserved	
107	B13	Default: 0	GPIO075	(I/O/OD)-4 mA	Reserved	VTR/VCC	
107	B13	1	Reserved	Reserved	Reserved	Reserved	
107	B13	2	Reserved	Reserved	Reserved	Reserved	
107	B13	3	Reserved	Reserved	Reserved	Reserved	
108	F9	Default: 0	GPIO041	(I/O/OD)-4 mA	VTR	VTR	
108	F9	1	Reserved	Reserved	Reserved	Reserved	
108	F9	2	Reserved	Reserved	Reserved	Reserved	
108	F9	3	Reserved	Reserved	Reserved	Reserved	
109	A13	Default: 0	GPIO076	(I/O/OD)-4 mA	Reserved	VTR/VCC	
109	A13	1	Reserved	Reserved	Reserved	Reserved	
109	A13	2	Reserved	Reserved	Reserved	Reserved	
109	A13	3	Reserved	Reserved	Reserved	Reserved	
110	E9	Default: 0	GPIO220	(I/O/OD)-4 mA	VTR	VTR/VCC	
110	E9	1	Reserved	Reserved	Reserved	Reserved	
110	E9	2	Reserved	Reserved	Reserved	Reserved	
110	E9	3	Reserved	Reserved	Reserved	Reserved	
111	E8	Default: 0	GPIO035	(I/O/OD)-4 mA	VTR	VTR/VCC	
111	E8	1	PWM8	(O/OD)-4 mA	VTR	VTR	
111	E8	2	Reserved	Reserved	Reserved	Reserved	
111	E8	3	Reserved	Reserved	Reserved	Reserved	
112	D8	Default: 0	GPIO170	(I/O/OD)-8 mA	VTR	VTR/VCC	
112	D8	1	MSCLK	O-8 mA	VTR	VTR	Note 10
112	D8	2	Reserved	Reserved	Reserved	Reserved	
112	D8	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-39: MULTIPLEXING TABLE (15 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
113	D7	Default: 0	GPIO171	(I/O/OD)-8 mA	VTR	VTR/VCC	
113	D7	1	MSDATA	O-8 mA	VTR	VTR	Note 10
113	D7	2	Reserved	Reserved	Reserved	Reserved	
113	D7	3	Reserved	Reserved	Reserved	Reserved	
114	N8	Default: 0	GPIO133	(I/O/OD)-4 mA	VTR	VTR/VCC	
114	N8	1	PWM9	(O/OD)-4 mA	VTR	VTR	
114	N8	2	Reserved	Reserved	Reserved	Reserved	
114	N8	3	Reserved	Reserved	Reserved	Reserved	
115	D11	Default: 0	GPIO134	(I/O/OD)-4 mA	VTR	VTR/VCC	
115	D11	1	PWM10	(O/OD)-4 mA	VTR	VTR	
115	D11	2	Reserved	Reserved	Reserved	Reserved	
115	D11	3	Reserved	Reserved	Reserved	Reserved	
116	D10	Default: 0	GPIO135	(I/O/OD)-4 mA	VTR	VTR/VCC	
116	D10	1	PWM11	(O/OD)-4 mA	VTR	VTR	
116	D10	2	Reserved	Reserved	Reserved	Reserved	
116	D10	3	Reserved	Reserved	Reserved	Reserved	
117	C12	Default: 0	GPIO145	(I/O/OD)-8 mA	VTR	VTR/VCC	
117	C12	1	SMB09_DATA	(I/OD)-8 mA	VTR	VTR	
117	C12	2	JTAG_TDI	I	VTR	VTR	
117	C12	3	Reserved	Reserved	Reserved	Reserved	
118	B12	Default: 0	GPIO146	(I/O/OD)-8 mA	VTR	VTR/VCC	
118	B12	1	SMB09_CLK	(I/OD)-8 mA	VTR	VTR	
118	B12	2	JTAG_TDO	O-8 mA	VTR	VTR	
118	B12	3	Reserved	Reserved	Reserved	Reserved	
119	B11	Default: 0	GPIO147	(I/O/OD)-8 mA	VTR	VTR/VCC	
119	B11	1	SMB08_DATA	(I/OD)-8 mA	VTR	VTR	
119	B11	2	Reserved	Reserved	Reserved	VTR	
119	B11	3	JTAG_CLK	I	VTR	VTR	
120	A12	Default: 0	GPIO150	(I/O/OD)-8 mA	VTR	VTR/VCC	
120	A12	1	SMB08_CLK	(I/OD)-8 mA	VTR	VTR	
120	A12	2	Reserved	Reserved	Reserved	VTR	
120	A12	3	JTAG_TMS	I	VTR	VTR	

TABLE 2-40: MULTIPLEXING TABLE (16 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
121	A11	Default: 0	JTAG_RST#	IS	VTR	VTR	Note 14
121	A11	1	Reserved	Reserved	Reserved	Reserved	
121	A11	2	Reserved	Reserved	Reserved	Reserved	
121	A11	3	Reserved	Reserved	Reserved	Reserved	
122	C11	Default: 0	GPIO104	(I/O/OD)-8 mA	VTR	VTR/VCC	
122	C11	1	UART_TX	O-8 mA	VTR	VTR/VCC	Note 13
122	C11	2	Reserved	Reserved	Reserved	Reserved	
122	C11	3	Reserved	Reserved	Reserved	Reserved	
123	A10	Default: 0	GPIO105	(I/O/OD)-4 mA	VTR	VTR/VCC	
123	A10	1	UART_RX	I	VTR	VTR	Note 13
123	A10	2	Reserved	Reserved	Reserved	Reserved	
123	A10	3	Reserved	Reserved	Reserved	Reserved	
124	B10	Default: 0	GPIO025	(I/O/OD)-4 mA	VTR	VTR/VCC	
124	B10	1	UART_CLK	I	VTR	VTR	
124	B10	2	TIN0	I	VTR	VTR	
124	B10	3	EM_INT	(O/OD)-4 mA	VTR	VCC	
125	B9	Default: 0	GPIO026	(I/O/OD)-4 mA	VTR	VTR/VCC	
125	B9	1	GPTP-IN0	I	VTR	VTR/VCC	Note 8
125	B9	2	TIN1	I	VTR	VTR	
125	B9	3	KSI3	IS	VTR	VTR	
126	D9	Default: 0	GPIO027	(I/O/OD)-4 mA	VTR	VTR/VCC	
126	D9	1	GPTP-OUT0	(O/OD)-4 mA	VTR	VTR/VCC	
126	D9	2	TIN2	I	VTR	VTR	
126	D9	3	KSI4	IS	VTR	VTR	
127	C10	Default: 0	GPIO030	(I/O/OD)-4 mA	VTR	VTR/VCC	
127	C10	1	GPTP-IN1	I	VTR	VTR/VCC	
127	C10	2	TIN3	I	VTR	VTR	
127	C10	3	KSI5	IS	VTR	VTR	
128	A9	Default: 0	GPIO107	(I/O/OD)-4 mA	VTR	VTR/VCC	
128	A9	1	KSO4	(O/OD)-4 mA	VTR	VTR	
128	A9	2	Reserved	Reserved	Reserved	Reserved	
128	A9	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-41: MULTIPLEXING TABLE (17 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
129	C9	Default: 0	GPIO120	(I/O/OD)-4 mA	VTR	VTR/VCC	
129	C9	1	KSO7	(O/OD)-4 mA	VTR	VTR	
129	C9	2	Reserved	Reserved	Reserved	Reserved	
129	C9	3	Reserved	Reserved	Reserved	Reserved	
130	B8	Default: 0	GPIO124	(I/O/OD)-4 mA	VTR	VTR/VCC	
130	B8	1	GPTP-OUT4	(O/OD)-4 mA	VTR	VTR/VCC	
130	B8	2	KSO11	(O/OD)-4 mA	VTR	VTR	
130	B8	3	Reserved	Reserved	Reserved	Reserved	
131	A8	Default: 0	GPIO125	(I/O/OD)-4 mA	VTR	VTR/VCC	
131	A8	1	GPTP-IN4	I	VTR	VTR/VCC	Note 8
131	A8	2	KSO12	(O/OD)-4 mA	VTR	VTR	
131	A8	3	Reserved	Reserved	Reserved	Reserved	
132	D6	Default: 0	GPIO031	(I/O/OD)-4 mA	VTR	VTR/VCC	
132	D6	1	GPTP-OUT1	(O/OD)-4 mA	VTR	VTR/VCC	
132	D6	2	TOUT0	O-4 mA	VTR	VTR	
132	D6	3	KSI6	IS	VTR	VTR	
133	B7	Default: 0	GPIO032	(I/O/OD)-4 mA	VTR	VTR/VCC	
133	B7	1	GPTP-IN2	I	VTR	VTR/VCC	Note 8
133	B7	2	TOUT1	O-4 mA	VTR	VTR	
133	B7	3	KSI7	IS	VTR	VTR	
134	C8	Default: 0	GPIO040	(I/O/OD)-4 mA	VTR	VTR/VCC	
134	C8	1	GPTP-OUT2	(O/OD)-4 mA	VTR	VTR/VCC	
134	C8	2	TOUT2	O-4 mA	VTR	VTR	
134	C8	3	KSO0	(O/OD)-4 mA	VTR	VTR	
135	A7	Default: 0	GPIO017	(I/O/OD)-4 mA	VTR	VTR/VCC	
135	A7	1	GPTP-OUT7	(O/OD)-4 mA	VTR	VTR/VCC	
135	A7	2	TOUT3	O-4 mA	VTR	VTR	
135	A7	3	KSI0	IS	VTR	VTR	
136	D5	Default: 0	GPIO022	(I/O/OD)-16 mA	VTR	VTR/VCC	
136	D5	1	BCM_B_CLK	O-16 mA	VTR	VTR	
136	D5	2	Reserved	Reserved	Reserved	Reserved	
136	D5	3	Reserved	Reserved	Reserved	Reserved	



TABLE 2-42: MULTIPLEXING TABLE (18 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
137	C5	Default: 0	GPIO023	(I/O/OD)-16 mA	VTR	VTR/VCC	
137	C5	1	BCM_B_DAT	IO-16 mA	VTR	VTR	Note 9
137	C5	2	Reserved	Reserved	Reserved	Reserved	
137	C5	3	Reserved	Reserved	Reserved	Reserved	
138	E4	Default: 0	GPIO024	(I/O/OD)-4 mA	VTR	VTR/VCC	
138	E4	1	BCM_B_INT#	IO-4 mA	VTR	VTR	
138	E4	2	Reserved	Reserved	Reserved	Reserved	
138	E4	3	Reserved	Reserved	Reserved	Reserved	
139	C7	Default: 0	GPIO127	(I/O/OD)-4 mA	VTR	VTR/VCC	
139	C7	1	A20M	O-4 mA	VTR	VCC	Note 2
139	C7	2	Reserved	Reserved	Reserved	Reserved	
139	C7	3	Reserved	Reserved	Reserved	Reserved	
140	F5	Default: 0	GPIO034	(I/O/OD)-8 mA	VTR	VTR/VCC	
140	F5	1	CEC_OUT	(O/OD)-8 mA	VTR	VTR	
140	F5	2	Reserved	Reserved	Reserved	Reserved	
140	F5	3	Reserved	Reserved	Reserved	Reserved	
141	E5	Default: 0	GPIO036	(I/O/OD)-4 mA	VTR	VTR/VCC	
141	E5	1	CEC_IN	I	VTR	VTR	
141	E5	2	Reserved	Reserved	Reserved	Reserved	
141	E5	3	Reserved	Reserved	Reserved	Reserved	
142	A6	Default: 0	GPIO045	(I/O/OD)-4 mA	VTR	VTR/VCC	
142	A6	1	LSBCM_D_INT#	I	VTR	VTR	
142	A6	2	KSO1	(O/OD)-4 mA	VTR	VTR	
142	A6	3	Reserved	Reserved	Reserved	Reserved	
143	E6	Default: 0	GPIO046	(I/O/OD)-8 mA	VTR	VTR/VCC	
143	E6	1	LSBCM_D_DAT	I/O-8 mA	VTR	VTR	Note 9
143	E6	2	KSO2	(O/OD)-8 mA	VTR	VTR	
143	E6	3	Reserved	Reserved	Reserved	Reserved	
144	B6	Default: 0	GPIO047	(I/O/OD)-8 mA	VTR	VTR	
144	B6	1	LSBCM_D_CLK	O-8 mA	VTR	VTR	
144	B6	2	KSO3	(O/OD)-8 mA	VTR	VTR	
144	B6	3	Reserved	Reserved	Reserved	Reserved	

# MEC1618/MEC1618i

TABLE 2-43: MULTIPLEXING TABLE (19 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
145	C6	Default: 0	GPIO121	(I/O/OD)-4 mA	VTR	VTR/VCC	
145	C6	1	BCM_A_INT#	I	VTR	VTR	
145	C6	2	KSO8	(O/OD)-4 mA	VTR	VTR	
145	C6	3	Reserved	Reserved	Reserved	Reserved	
146	A5	Default: 0	GPIO122	(I/O/OD)-16 mA	VTR	VTR/VCC	
146	A5	1	BCM_A_DAT	IO-16 mA	VTR	VTR	Note 9
146	A5	2	KSO9	(O/OD)-16 mA	VTR	VTR	
146	A5	3	Reserved	Reserved	Reserved	Reserved	
147	B5	Default: 0	GPIO123	(I/O/OD)-16 mA	VTR	VTR/VCC	
147	B5	1	BCM_A_CLK	O-16 mA	VTR	VTR	
147	B5	2	KSO10	(O/OD)-16 mA	VTR	VTR	
147	B5	3	Reserved	Reserved	Reserved	Reserved	
148	A3	Default: 0	GPIO042	(I/O/OD)-8 mA	VTR	VTR/VCC	
148	A3	1	Reserved	Reserved	Reserved	Reserved	
148	A3	2	PECI_DAT	IO-PECI	VREF_VTT	VREF_VTT	
148	A3	3	SB-TSI_DAT	(I/OD)-SB-TSI	VREF_VTT	VREF_VTT	
149	A2	Default: 0	GPIO043	(I/O/OD)-8 mA	VTR	VTR/VCC	
149	A2	1	Reserved	Reserved	Reserved	Reserved	
149	A2	2	Reserved	Reserved	Reserved	Reserved	
149	A2	3	SB-TSI_CLK	(I/OD)-SB-TSI	VREF_VTT	VREF_VTT	
150	A1	Default: 0	GPIO044	(I/O/OD)-4 mA	VTR	VTR/VCC	
150	A1	1	VREF_VTT	I	VTR	VTR	
150	A1	2	Reserved	Reserved	VREF_VTT	VREF_VTT	
150	A1	3	Reserved	Reserved	Reserved	Reserved	
151	B1	Default: 0	GPIO126	(I/O/OD)-4 mA	VTR	VTR/VCC	
151	B1	1	KSO13	(O/OD)-4 mA	VTR	VTR	
151	B1	2	Reserved	Reserved	Reserved	Reserved	
151	B1	3	Reserved	Reserved	Reserved	Reserved	
152	A4	Default: 0	GPIO020	(I/O/OD)-4 mA	VTR	VTR/VCC	
152	A4	1	KSI1	I	VTR	VTR	
152	A4	2	Reserved	Reserved	Reserved	Reserved	
152	A4	3	Reserved	Reserved	Reserved	Reserved	

TABLE 2-44: MULTIPLEXING TABLE (20 OF 20)

Pin Ref. Number	Ball	MUX	Signal	Buffer Type	Signal Power Well	Emulated Power Well	Notes
153	B2	Default: 0	GPIO156	(I/O/OD)-8 mA	VTR	VTR/VCC	
153	B2	1	LED0	(O/OD)-8 mA	VTR	VTR	
153	B2	2	Reserved	Reserved	Reserved	Reserved	
153	B2	3	Reserved	Reserved	Reserved	Reserved	
154	B3	Default: 0	GPIO157	(I/O/OD)-8 mA	VTR	VTR/VCC	
154	B3	1	LED1	(O/OD)-8 mA	VTR	VTR	
154	B3	2	Reserved	Reserved	Reserved	Reserved	
154	B3	3	Reserved	Reserved	Reserved	Reserved	
155	B4	Default: 0	GPIO153	(I/O/OD)-8 mA	VTR	VTR/VCC	
155	B4	1	LED2	(O/OD)-8 mA	VTR	VTR	
155	B4	2	Reserved	Reserved	Reserved	Reserved	
155	B4	3	Reserved	Reserved	Reserved	Reserved	
156	D3	Default: 0	GPIO175	(I/O/OD)-4 mA	VTR	VTR/VCC	
156	D3	1	32KHZ_OUT	O-4 mA	VTR	VTR	
156	D3	2	KSO17	(O/OD)-4 mA	VTR	VTR	
156	D3	3	Reserved	Reserved	Reserved	Reserved	

## 2.6 Notes for Tables in this Chapter

<b>Note 1</b>	Buffer modes are described per signal function. On multiplexed pins buffer modes are separated by a slash "/"; e.g., a pin with two multiplexed functions where the primary function is an input and the secondary function is an 8mA bi-directional driver is represented as "I/O-8". Buffer modes in parentheses represent multiple buffer modes for a single pin function. The number following the "-" represents the balanced output sink/source capability of the buffer in milliamps.
<b>Note 2</b>	This pin function exhibits "VCC" power domain emulation. The System Runtime Supply power is not connected to this device. The VCC_PWRGD signal is used to indicate when power is applied to the System Runtime Supply. All input and output signals that require "VCC" power domain functionality are powered by VTR and controlled by the VCC_PWRGD signal input. See the Pin Default State Transitions from Powered to Unpowered Table.
<b>Note 3</b>	The nEC_SCI pin can be controlled by hardware and EC firmware. The nEC_SCI pin can drive either the ACPI Run-time GPE Chipset input or the Wake GPE Chipset input. Depending how the nEC_SCI pin is used, other ACPI-related SCI functions may be best supplied by other general purpose outputs that can be configured as open-drain drivers.
<b>Note 4</b>	These pins require an external weak pull-up resistors of 10k-100k ohms.
<b>Note 5</b>	These pins are 3.3V only (non-5V tolerant).
<b>Note 6</b>	This pin is tristated when PWRGD is inactive and the pin is configured as a VCC2-powered alternate function.
<b>Note 7</b>	This pin function exhibits "VTR" power domain emulation. This pin is powered by VBAT power. All input and output signal functions that require "VTR" power domain emulation functionality are powered by VBAT and controlled internally by the application of VTR power. This pin is tristated when VTR power is not applied and the pin is configured as a VTR emulation powered signal function.
<b>Note 8</b>	Signals on this pin can be configured to generate a wake-up event to the EC on selected edges.
<b>Note 9</b>	A pull-up is not needed on this BC-Link DATA pin as long as the voltage remains above the logic-high threshold during the second turnaround cycle.
<b>Note 10</b>	This pin may require a weak pull-up.
<b>Note 11</b>	Most GPIO pins are (I/O/OD). See the 'Pin Multiplexing' tables and associated notes for specific exceptions.
<b>Note 12</b>	This pin has EC wakeup and interrupt capability controlled by the corresponding Pin Control Register. A GPIO assignment is documented in the GPIO chapter to provide interrupt and wakeup capability. The GPIO should not be used for I/O. See Detailed Pin Multiplexing Assignments section in the GPIO chapter and lookup this pin and see the associated note.
<b>Note 13</b>	The two pin debug port UART can be used by the Host or EC. This pin can be VCC protected or not VCC protected under program control by the POWER bit in the Configuration Select Register in Host configuration space (also accessible by the EC).
<b>Note 14</b>	When the JTAG_RST# pin is not asserted (logic'1'), the JTAG_TDI, JTAG_TDO, JTAG_CLK, JTAG_TMS signal functions in the JTAG interface are unconditionally routed to the interface; the Pin Control register for these pins has no effect. When the JTAG_RST# pin is asserted (logic'0'), the JTAG_TDI, JTAG_TDO, JTAG_CLK, JTAG_TMS signal functions in the JTAG interface are not routed to the interface and the Pin Control Register for these pins controls the muxing. The pin control registers can not route the JTAG interface to the pins. System Board Designer should terminate this pin in all functional state using jumpers and pull-up or pull down resistors, etc .

<b>Note 15</b>	PS/2 ports ending with signal functions ending with "A" or "B" are muxed to a single controller. Only one set of clock and data are intended to be used at a time (either "A" or "B" not both). The unused port segment should have its associated pin control register's, Mux Control Field programmed away from the PS2 controller.
<b>Note 16</b>	Most GPIO's are (I/O/OD). See Multiplexing tables below and associated notes for specific exceptions.
<b>Note 17</b>	The GPIO assignment on this pin only provides interrupt and wakeup capability. This is provided by the Interrupt Detection field in the Pin Control register. The Mux control field in the Pin Control Register should <u>not</u> be set to '00' = GPIO or undesirable results may occur.
<b>Note 18</b>	This pin is also used as a JTAG TAP controller select strap option. There is a weak pullup enabled on this pin by default.
<b>Note 19</b>	Reserved
<b>Note 20</b>	The RESET0# pin defaults to an output 'low' following a VTR power up, or assertion of the RESET1# input pin. Firmware de-asserts RESET0# by making the GPIO output pin 'high'
<b>Note 21</b>	Signal multiplexing for this pin is enabled when the Gang Programmer Interface is activated; it is not controlled by a GPIO Pin Configuration register.

## 2.7 Strapping Options

GPIO171 is used for the [TAP Controller Select Strap Option](#) (see [Section 41.2.1](#) on page 604). If any of the MEC1618/MEC1618i JTAG TAP controllers are used, GPIO171 must only be configured as an output to a VTR powered external function. GPIO171 may only be configured as an input when the JTAG TAP controllers are not needed or when an external driver does not violate the [Slave Select Timing](#) as defined in [Section 41.2.2](#) on page 604.

See also [Section 5.6.2, "Strap Options,"](#) on page 104.



## 3.0 BUS HIERARCHY

### 3.1 General Description

All devices in the MEC1618/MEC1618i are maintained in a common address space. All communication with on-chip functions is through registers that have addresses in this common address space. The ARC Embedded Controller (EC) can reference all devices through the address space, while the Host can only reference a subset.

### 3.2 Block Diagram

[FIGURE 3-1: MEC1618/MEC1618i Bus Hierarchy on page 48](#) shows, in graphic form, the inter connectivity of devices on the MEC1618/MEC1618i, including the EC, the principal lower buses and most of the peripherals.

\_\_\_\_\_



Data Memory. These memories are shown in [Figure 3-2](#). Software can change the address mapping so that the Closely Coupled Data Memory is mapped to 6\_0000h through 6\_0FFFh, where it can be used for both instructions and data, and the [Embedded Flash Subsystem](#) becomes accessible only through a register interface. Addresses greater or equal to 80\_1000h are propagated through the AHB interface on the ARC processor. References, by either the EC or the Host via the LPC bus, to addresses that are not mapped to any device register or memory will cause a bus error; see [Section 3.4.3, "AHB Bus Errors," on page 55](#).

## 3.3.1 ARC ADDRESS SPACE

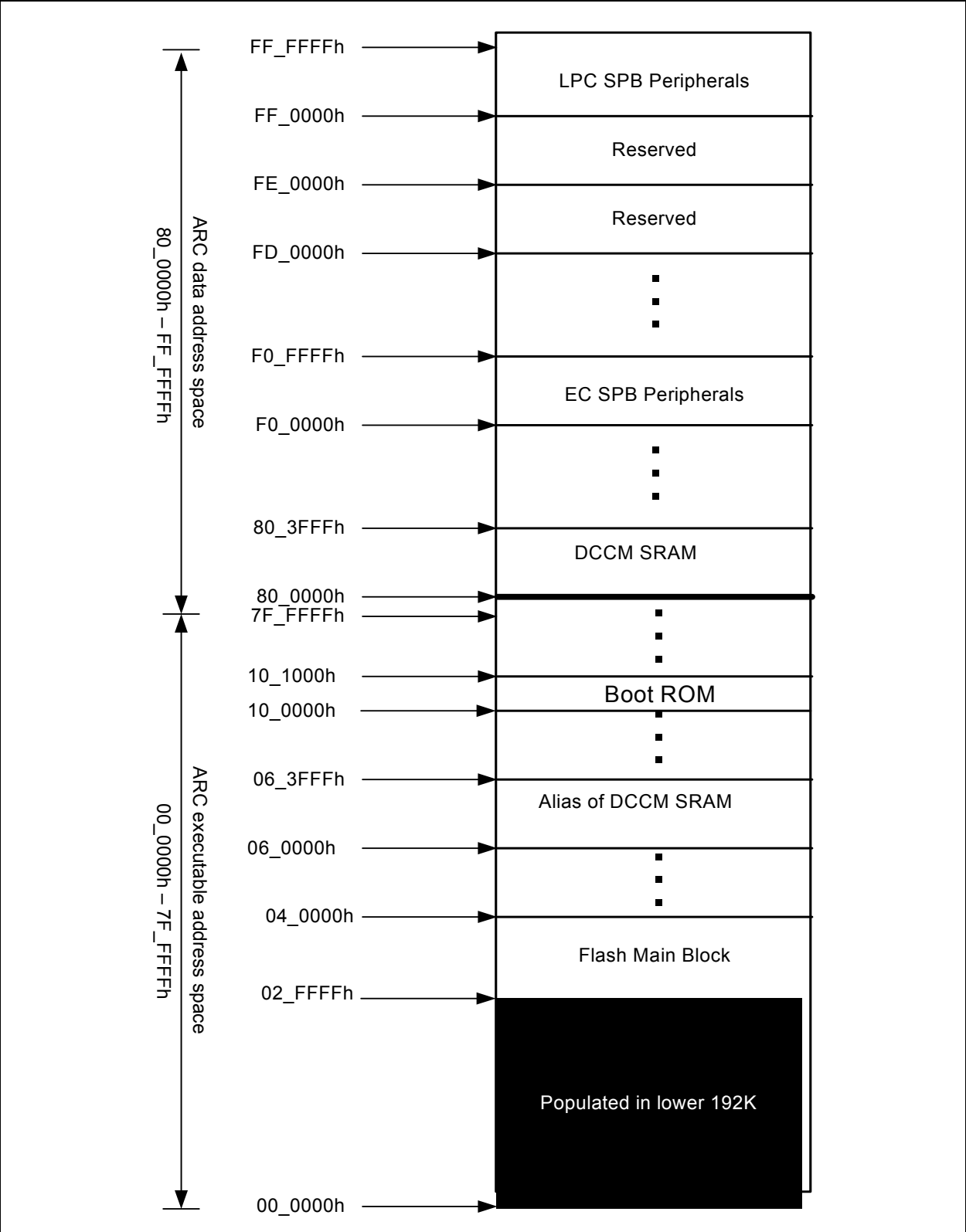
The [ARC Address Space](#) is illustrated in Figure 3-2, "MEC1618/MEC1618i EC Memory Map" & Figure 3-1, "MEC1618/MEC1618i Bus Hierarchy". [EC Instruction Memory](#) occupies the lower half of the address space, 00\_0000h through 7F\_FFFFh. The ARC processor can only execute instructions that are located in the Instruction Memory portion of the address space. Only a portion of the Instruction space is populated.

The upper half of the address space is used solely for data references. The region contains a general-purpose [EC Data Memory](#) SRAM, as well as the address space of the two SPB peripheral buses.

The contents of the Instruction memory can be read and written by the ARC processor through regular load and store instructions. However, there may be a one cycle instruction fetch penalty whenever a data load or store instruction to Instruction memory is executed.

# MEC1618/MEC1618i

FIGURE 3-2: MEC1618/MEC1618i EC MEMORY MAP



## 3.3.2 AHB ADDRESS SPACE

The components on the AHB subsystem and the two SPB bridges define a set of addresses that are accessible by the EC. This address space is shown in [Table 3-1, "MEC1618/MEC1618i Peripheral Address Space"](#). As shown in the table, the Host can access much of the address space, but not all.

**TABLE 3-1: MEC1618/MEC1618i PERIPHERAL ADDRESS SPACE**

Address Range	Device	Accessible by EC	Accessible by Host
F0_0000h - F0_FFFFh	ARC-only SPB Bridge	Yes	No
FD_0000h - FE_FFFFh	Reserved	No	No
FF_0000h - FF_BFFFh and FF_D000h - FF_FFFFh	LPC SPB Bridge	Yes	Yes (limited by LPC interface map)

The address range of FD\_0000h through FE\_FFFFh is reserved in MEC1618/MEC1618i.

The 64KB address space of an SPB Bridge is divided into 1KB Frames. Peripherals are grouped into Logical Devices; each Logical Device corresponds to a 1KB Frame. Logical Devices addressable by the Host are listed in [Table 3-2, "Host Logical Devices in the MEC1618/MEC1618i"](#). Logical device numbers not listed in [Table 3-2](#) are reserved.

The 1KB Address Frame of a Host Logical Device is divided into four subregions, as described in [Section 3.4.2, "Address Framing," on page 54](#).

**TABLE 3-2: HOST LOGICAL DEVICES IN THE MEC1618/MEC1618i**

Logical Device Number	Logical Devices	AHB Address of Frame	Runtime Registers	Configuration Registers	EC-only Registers
0h	Mailbox Interface	FF_0000h	yes	yes	yes
1h	Keyboard Controller (8042)	FF_0400h	yes	yes	no
2h	ACPI EC Channel 0	FF_0800h	yes	yes	no
3h	ACPI EC Channel 1	FF_0C00h	yes	yes	no
4h	ACPI EC Channel 2	FF_1000h	yes	yes	no
5h	ACPI EC Channel 3	FF_1400h	yes	yes	no
6h	ACPI PM1	FF_1800h	yes	yes	no
7h	UART	FF_1C00h	yes	yes	no
8h	Legacy (Fast KB)	FF_2000h	yes	yes	no
Ch	LPC Interface	FF_3000h	no	yes	yes
Dh	Reserved	FF_3400h	no	yes	no
Eh	Embedded Flash Interface	FF_3800h	yes	yes	yes

# MEC1618/MEC1618i

**TABLE 3-2: HOST LOGICAL DEVICES IN THE MEC1618/MEC1618I (CONTINUED)**

Logical Device Number	Logical Devices	AHB Address of Frame	Runtime Registers	Configuration Registers	EC-only Registers
Fh	Flash SPI	FF_3C00h	yes	yes	no
10h	EM Interface	FF_4000h	yes	yes	yes
20h	BIOS Debug Port 0	FF_8000h	yes	yes	no
21h	BIOS Debug Port 1	FF_8400h	yes	yes	yes
3Fh	Global Configuration	FF_FC00h	no	no	yes

The 64KB address space of the ARC-only SPB Bridge is divided into 1KB Frames. Peripherals are grouped into Logical Devices; each Logical Device corresponds to a 1KB Frame. These EC Logical Devices addressable by the ARC-only are listed in [Table 3-3, "EC Logical Devices on MEC1618/MEC1618i"](#). Multiple instantiations of the same block are in a single 1KB Frame. Each instantiation is separated by 128 bytes.

**Note 3-1** All VBAT powered registers are in a Single 1KB Frame separated by 128 bytes (see [Table 3-3, Logical Device Number 33h.](#))

Note that the ARC-only SPB Bridge address space uses the terminology of Logical Device; however, there is no host access to these blocks.

**TABLE 3-3: EC LOGICAL DEVICES ON MEC1618/MEC1618I**

Logical Device Number	AHB Address for Base of Frame	Logical Devices	Notes
0h	F0_0000h	Hibernation Timer	
1h	F0_0400h	Watchdog Timer	
2h	F0_0800h	Input Capture and Compare Timer	
3h	F0_0C00h	16-bit Timer	
4h	F0_1000h	RC ID	
5h	F0_1400h	BC Bus Master	
6h	F0_1800h	SMBus	
7h	F0_1C00h	EC GP-SPI	
8h	F0_2000h	Keyscan Interface	
9h	F0_2400h	DMA	
Ah - Fh	F0_2C00h - F0_3FFFh	Reserved	
10h	F0_4000h	CEC Interface Controller	
11h	F0_4400h	Reserved	
12h	F0_4800h	Reserved	
13h	F0_4C00h	<b>Reserved</b>	
14h - 15h	F0_5000h - F0_57FFh	Reserved	
16h	F0_5800h	PWM(7-0)	

**TABLE 3-3: EC LOGICAL DEVICES ON MEC1618/MEC1618I (CONTINUED)**

Logical Device Number	AHB Address for Base of Frame	Logical Devices	Notes
17h	F0_5C00h	PWM(15-8)	<a href="#">Note 3-2</a>
18h	F0_6000h	TACH	
19h	F0_6400h	PECI	
1Ah	F0_6800h	ADC	
1Bh - 20h	F0_6C00h - F0_83FFh	Reserved	
21h	F0_8400h	LED	
22h	F0_8800h	PS/2	
23h	F0_8C00h	MCU Debug Port	
24h - 2Fh	F0_9000h - F0_BCFFh	Reserved	
30h	F0_C000h	EC Interrupt Aggregator	
31h	F0_C400h	GPIOs	
32h	F0_C800h	Power, Clock & Reset (VTR PWR'ed)	
33h	F0_CC00h	Power, Clock & Reset (VBAT PWR'ed)	
	F0_CC80h	Week Alarm Timer	
	F0_CD00h	VBAT Backed Memory	
34h	F0_D000h	VBAT-Powered Control Interface	
35h - 3Dh	F0_D400h - F0_F7FFh	Reserved	
3Eh	F0_F800h	Reserved	
3Fh	F0_FC00h	System Registers	

**Note 3-2** PWM register addressing within Logical device [17h](#), [PWM\(15-8\)](#), is similar to the addressing within the [PWM\(7-0\)](#) logical address space.

## 3.4 AHB Buses

Addresses and internal buses in the MEC1618/MEC1618i are compatible with ARM Limited's *Advanced Microprocessor Bus Architecture* (AMBA), as specified in *AMBA™ Specification (Rev 2.0)*, 1999.

As seen in [FIGURE 3-1: MEC1618/MEC1618i Bus Hierarchy on page 48](#), there are two separate AHB buses, the EC AHB and the Host AHB. The first has one master, the EC, and two slaves, while the second has two masters, the LPC interface and the AHB-AHB bridge, and four slaves. The bus connections are summarized in [Table 3-4, "MEC1618/MEC1618i AHB Buses"](#) and can be seen in [Figure 3-1](#).

# MEC1618/MEC1618i

**TABLE 3-4: MEC1618/MEC1618i AHB BUSES**

AHB Bus	Master Interfaces	Slave Interfaces
EC AHB	EC	EC SPB bridge AHB-AHB bridge
Host AHB	LPC Interface AHB-AHB bridge	LPC SPB bridge SPI bridge

The AHB-to-AHB bridge is a one-way device: addresses generated on the EC AHB can be propagated to the Host AHB bus, but addresses on the Host AHB bus cannot be propagated to the EC AHB bus. This is the reason that the Host is restricted from accessing the address range F0\_0000h to F0\_FFFFh (the address range of the EC SPB bridge). The bridge maps address from FD\_0000h through FF\_FFFFh. Multi-word block transfers on the EC AHB are converted by the bridge into a series of sequential 4-byte transfers.

Both the EC and the LPC interface can have at most one outstanding AHB bus request at one time. On the Host AHB, the LPC interface always has priority over the AHB-AHB bridge. Both AHB buses support byte, halfword and word AHB transfers. The only Burst mode supported on the EC AHB is an incrementing burst of 4 beats of 4 bytes per beat. Burst mode is not supported on the Host AHB. AHB bus locking or early burst termination are not supported.

## 3.4.1 BUS CLOCKING

The Host AHB runs at the MEC1618/MEC1618i system clock rate ([MCLK](#)). The bus clock and the bus arbiter are shut down when there are no transactions active on the bus. The bus clock and arbiter are restarted as soon as an address is acquired from the LPC bus, or when an EC AHB bus transaction is mapped, via the AHB-to-AHB bridge, to the address space of the Host AHB. The EC AHB clock can be programmed to run at any of the available rates. (See [Section 5.7.4, "Block Sleep Enable Registers," on page 118](#)). The EC bus clock is shut down when the EC is idle.

If the Host AHB clock is idle when an LPC transaction arrives at the LPC interface, the LPC interface will restart the AHB bus clock and arbiter early enough so that as soon as an I/O address is translated to an AHB address the I/O transaction can be placed on the AHB without delay. If the I/O address is not claimed by the MEC1618/MEC1618i then the LPC interface will drop its bus request. If the EC is not requesting the Host AHB at the same time, the Host AHB bus clock will again shut down.

## 3.4.2 ADDRESS FRAMING

The EC can directly address all peripherals on the MEC1618/MEC1618i. The Host, by contrast, is restricted in what it can address. The Host accesses the MEC1618/MEC1618i through the LPC bus, using I/O cycles, Memory Cycles and Firmware Hub cycles (see [Section 6.0, "Host Interface," on page 135](#)). These cycles are mapped into the MEC1618/MEC1618i address space accessible by the LPC interface.

The mapping function forces some of the address bits to preset values, as shown in [Table 3-5, "LPC to MEC1618/MEC1618i Address Mapping"](#). This mapping has the effect of creating four contiguous 256-byte regions: LPC I/O, EC-only, LPC DMA and LPC Configuration. Together, the four regions create a 1024 byte "frame" for each logical device that is accessible to the Host. There is therefore a maximum of 64 Logical Devices, each with a 1KB frame.

LPC I/O cycles for Runtime Registers are mapped into the first 256 bytes of the 1024 byte frame. Configuration Registers, which are accessed through a Configuration portal typically located at addresses 2Eh and 2Fh in the LPC I/O space, are restricted to the highest 256 bytes of the 1024 byte frame.

DMA FIFO addresses are restricted to offsets between 200h and 2FFh within a Logical Device frame. In addition, DMA FIFO addresses are restricted to 32-bit aligned addresses, that is, address bits [1:0] are both 0. The low 10 bits of a DMA FIFO are thus in the form '10xxxxxx00b'.

LPC Firmware Hub cycles and LPC Memory cycles are mapped into the address range 80\_0000h to FF\_FFFFh. This range is mapped into the off-chip SPI Flash memory.

Because the EC does not require the mapping mechanism required for translating LPC Runtime Registers, Configuration Registers and DMA channels, it accesses each 1KB frame uniformly. All Logical Devices located on the EC-only AHB (those in the address range F0\_0000h through F0\_FFFFh) have a flat, 1KB frame.

For details on LPC address mapping to the MEC1618/MEC1618i address space, see [Section 6.0, "Host Interface," on page 135](#).

**TABLE 3-5: LPC TO MEC1618/MEC1618I ADDRESS MAPPING**

Type of Access	Address Bit Mapping
LPC I/O Access Runtime Registers	Address bits[9:8] = 00b Address bits[23:10] set from map
No LPC Access EC-only registers	Address bits[9:8] = 01b
LPC I/O Access through Configuration Access Port Configuration Registers	Address bits[9:8] = 11b Address bits[23:10] set from map
LPC DMA Access	Address bits[9:8] = 10b Address bits[23:10] set from map

### 3.4.3 AHB BUS ERRORS

AHB bus requests by both the Host, through the LPC bus, and the EC, can be terminated with an AHB bus error. The handling of bus errors by the EC is described in the *ARCompact™ Instruction Set Architecture: Programmer's Reference*.

Bus errors may be caused by:

- EC I/O requests that lie outside the range of the Data Closely Coupled Memory, the EC SPB or the AHB-AHB bridge
- I/O requests to either the Host SPB or the EC SPB that map to a non-existent Logical Device
- I/O requests to an invalid register address within a valid Logical Device on either the EC SPB or the Host SPB

## 3.5 Microchip Peripheral Buses (SPB)

The Microchip Peripheral Bus (SPB) is a byte-addressable bus with a 16-bit address space and a 32-bit data path. All accesses must be aligned to the data-size boundaries. The SPB supports 32-bit, 16-bit and 8-bit accesses. All peripheral accesses are 32-bits wide, so that if a peripheral cannot transfer more than 8 bits of a register in one I/O access, register addresses should all be on 32-bit boundaries even though the upper bits (bits 31 through 8) are always zero. The SPB will not assemble a 32-bit word out of multiple 8-bit accesses.

SPB transfers take two cycles, so that a read or write on the AHB to a register on the SPB will take a total of three cycles. The SPB contains an 8-bit word address, four byte lane strobes and up to 64 logical device select strobes. Data on the SPB read and write buses are always 32-bit aligned, with the byte strobes indicating which byte lane or lanes should be active during a transaction. An SPB peripheral must steer bytes, based on the byte strobes, to the correct byte lane.

There are two SPB bridges forwarded to two SPB (busses):

#### LPC SPB

The LPC SPB address range is FF\_0000h - FF\_BFFFh and FF\_C400h - FF\_FFFFh.

#### EC SPB

The EC SPB address range is F0\_0000h through F0\_FFFFh

# MEC1618/MEC1618i

## 4.0 LOGICAL DEVICE CONFIGURATION

### 4.1 Description

The Configuration of the MEC1618/MEC1618i is very flexible and is based on the configuration architecture implemented in typical Plug-and-Play components.

The MEC1618/MEC1618i is designed for motherboard designs in which the resources required by their components are known. With its flexible resource allocation architecture, the MEC1618/MEC1618i allows the BIOS to assign resources at POST.

### 4.2 Location of Configuration Registers

Configuration Registers for Logical Devices accessible by the Host are located on the LPC SPB in the address range FF\_0000h through FF\_FFFFh. All Configuration Registers are located at addresses where address bits 9 and 8 are both '1b' (that is, at offsets 300h through 3FFh from the base of a 1KB address frame). Configuration registers are accessible by the Embedded Controller with 8-bit, 16-bit or 32-bit accesses. The Host can access the registers only with 8-bit accesses.

The Configuration Registers for the LPC Logical Device are located on the LPC SPB in the address range FF\_3300h through FF\_33F0h. The Global Configuration Registers are located within the Global Configuration Logical Device.

### 4.3 Logical Devices

Logical devices described in this section are peripherals that are located on the MEC1618/MEC1618i and are accessible to the Host over the LPC bus.

Each logical device on the MEC1618/MEC1618i can have a set of Runtime Register and a set of Configuration Registers. The distinction between Runtime and Configuration registers is that the Host can access Runtime Registers by a direct I/O address, while it can only access Configuration Registers through a configuration port. The Embedded Controller (EC) can access all Configuration Registers and all Runtime Registers directly. The Logical Device Numbers for the Logical Devices resident in the MEC1618/MEC1618i are listed in [Table 4-14, "MEC1618/MEC1618i Configuration Register Map," on page 69.](#)

**TABLE 4-1: LOGICAL DEVICES**

Logical Device Number	Logical Device	Logical Device CR Map on <a href="#">Table 4-14</a>
0h	<a href="#">Mailbox Interface</a>	<a href="#">on page 69</a>
1h	<a href="#">Keyboard Controller (8042)</a>	<a href="#">on page 69</a>
2h	<a href="#">ACPI EC Channel 0</a>	<a href="#">on page 69</a>
3h	<a href="#">ACPI EC Channel 1</a>	<a href="#">on page 69</a>
4h	<a href="#">ACPI EC Channel 2</a>	<a href="#">on page 69</a>
5h	<a href="#">ACPI EC Channel 3</a>	<a href="#">on page 69</a>
6h	<a href="#">ACPI PM1</a>	<a href="#">on page 69</a>
7h	<a href="#">UART</a>	<a href="#">on page 70</a>
8h	<a href="#">Legacy (Fast KB)</a>	<a href="#">on page 70</a>
Ch	<a href="#">LPC Interface</a>	<a href="#">on page 70</a>
Eh	<a href="#">Embedded Flash Interface</a>	<a href="#">on page 72</a>
Fh	<a href="#">Flash SPI</a>	<a href="#">on page 72</a>



**TABLE 4-1: LOGICAL DEVICES (CONTINUED)**

Logical Device Number	Logical Device	Logical Device CR Map on <a href="#">Table 4-14</a>
10h	EM Interface	<a href="#">on page 72</a>
20h	BIOS Debug Port 0	<a href="#">on page 72</a>
21h	BIOS Debug Port 1	<a href="#">on page 72</a>
3Fh	Global Configuration	<a href="#">on page 72</a>

All Configuration and Runtime Registers in the MEC1618/MEC1618i have an assigned AHB address between FF 0000h and FF FFFFh. Unless indicated otherwise, the EC can issue reads and writes to any register in that AHB address range. The EC can access 8-bit registers with 8-bit reads and writes, 16-bit registers with either 8-bit or 16-bit reads and writes and 32-bit registers with 8-bit, 16-bit and 32-bit reads and writes.

The Host can only access a subset of the AHB address space, and within that space all registers are treated as 8-bit registers, although a register may be implemented as a 32-bit register and accessible to the EC as a 32-bit register. The Host accesses registers in the FF\_0000h through FF\_FFFFh range through LPC I/O cycles. I/O cycles are mapped according to rules described in [Section 4.5, on page 58](#), [Section 4.6, on page 60](#) and [Section 4.7, on page 63](#).

## 4.4 Registers

The [Host Interface](#) has its own Logical Device Number and Base Address as indicated in [Table 4-2](#). The Host LPC I/O addresses for the [Logical Device Configuration](#) are selected via a Base Address Register. LPC access to configuration registers is through the Host Access Configuration Port

The [Logical Device Configuration](#) also has a [Global Configuration](#) block that has a separate Logical Device Number and Base Address Register as indicated in [Table 4-2](#). The Base Address Register for the [Global Configuration](#) has only one writable bit, the Valid Bit, since the only I/O accessible Register has a fixed address.

[Table 4-3](#) is a register summary for the [LPC Interface](#) block and [Table 4-15, “Chip-Level \(Global\) Control/Configuration Registers,” on page 73](#) is a register summary for the [Global Configuration](#) block.

**TABLE 4-2: Host Interface BASE ADDRESS TABLE**

Host Interface Blocks	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
<a href="#">LPC Interface</a>	Ch	<a href="#">FF_3000h</a>
<a href="#">Global Configuration</a>	3Fh	<a href="#">FF_FC00h</a>

**Note:** The Host LPC I/O addresses for this instance are selected via a Base Address Register (see [Section 4.6.2, on page 60](#)). LPC access to configuration registers is through the Host Access Configuration Port (see [Section 4.5.1, on page 58](#).)

[Table 4-3](#) is a register summary for the [Host Access Port](#) block. The LPC I/O address for each Run-Time Register is described below as an offset from its Base Address Register. Each EC address is indicated as an SPB Offset from its AHB base address. Each Configuration register access through the [Host Access Port](#) address via its LDN indicated in [Table 4-2 on page 57](#) and its [Host Access Port](#) index which is described as “Host Config Index” in the tables below.

**TABLE 4-3:**     [Host Access Port](#) REGISTER SUMMARY

Port Name	Host I/O Access			EC Interface			Notes
	Host I/O Index	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
CONFIG PORT	00h	00h	W				
INDEX PORT	00h	00h	R/W				
DATA PORT	01h	01h	R/W				

**Note:**     The EC does not have access to the [Host Access Port](#); however, the EC can access registers with AHB addresses.

## 4.5 Configuration Registers

### 4.5.1 HOST ACCESS PORT

The Host can access Configuration Registers through a port described in [Section 4.5.2, on page 58](#). Host accesses are limited to 8 bits. There are 48 8-bit Global Configuration Registers (at offsets 00h through 2Fh), plus up to 208 8-bit registers associated with each Logical Device. The Logical Device is selected with the [Logical Device Number](#) Register (Global Configuration Register 07h). The INDEX PORT is used to select a specific logical device register. These registers are then accessed through the DATA PORT. The Logical Device registers are accessible only when the device is in the Configuration State.

Only two states are defined (Run and Configuration). In the Run State, the chip will always be ready to enter the Configuration State.

The desired configuration registers are accessed in two steps:

- Write the index of the [Logical Device Number](#) Configuration Register (i.e., 07h) to the INDEX PORT and then write the number of the desired logical device to the DATA PORT
- Write the address of the desired configuration register within the logical device to the INDEX PORT and then write or read the configuration register through the DATA PORT.

**Note:**

- If accessing the Global Configuration Registers, step (a) is not required.
- Any write to an undefined or reserved Configuration register is terminated normally on the LPC bus without any modification of state in the MEC1618/MEC1618i. Any read to an undefined or reserved Configuration register returns FFh.

### 4.5.2 PRIMARY CONFIGURATION ADDRESS DECODER

The logical are configured through three Configuration Access Ports (CONFIG, INDEX and DATA). The BIOS uses these ports to initialize the logical devices at POST ([Table 4-4](#)).

The Base Address of the Configuration Access Ports is determined by the BAR that corresponds to Logical Device [Ch](#), the [LPC Interface](#). This is the first BAR in the table, at AHB address FF\_3360h. The Configuration Access Port BAR is unique in that an LPC I/O access that matches this BAR does not directly generate an AHB read or write. Instead, the Device and Frame values in the BAR indicates that the LPC I/O should be handled locally in the LPC Logical Device. The Configuration map issues an AHB read or write, the results of which are used to complete the LPC access.

**TABLE 4-4: MEC1618/MEC1618i CONFIGURATION ACCESS PORTS**

Port Name	Relative Address	Type	Port Name
CONFIG PORT	Configuration Access Ports Base Address + 0	Write	CONFIG PORT
INDEX PORT	Configuration Access Ports Base Address + 0	Read/Write	INDEX PORT
DATA PORT	Configuration Access Ports Base Address + 1		DATA PORT

## 4.5.2.1 Entering the Configuration State

The INDEX and DATA ports are effective only when the chip is in the Configuration State. The device enters the Configuration State when the Config Entry Key is successfully written to the CONFIG PORT.

**CONFIG ENTRY KEY = < 55H>**

## 4.5.2.2 Exiting the Configuration State

The device exits the Configuration State when the following Config Exit Key is successfully written to the CONFIG PORT address.

**CONFIG EXIT KEY = < AAH>**

## 4.5.2.3 Read Accessing Configuration Port

The data read from the Configuration Port is undefined when not in the Configuration State. Writing the Config Entry Key puts the chip in the Configuration State. After being in the Configuration State, reading the Configuration Port returns the last value written to the Configuration Index. If no value was written the Configuration Port reads 00h.

## 4.5.3 CONFIGURATION SEQUENCE EXAMPLE

To program the configuration registers, the following sequence must be followed:

1. Enter Configuration State
2. Program the Configuration Registers
3. Exit Configuration State.

The following is an example of a configuration program in Intel 8086 assembly language.

```

;-----
; ENTER CONFIGURATION STATE
;-----
MOV DX,CONFIG_PORT_BASE_ADDRESS
MOV AX,055H ;_Config Entry Key
OUT DX,AL

;-----
; CONFIGURE BASE ADDRESS,
; LOGICAL DEVICE 8
;-----
MOV DX,CONFIG_PORT_BASE_ADDRESS
MOV AL,07H
OUT DX,AL ; Point to LD# Config Reg
MOV DX,CONFIG_PORT_BASE_ADDRESS+1
MOV AL, 08H
OUT DX,AL ; Point to Logical Device 8
;
MOV DX,CONFIG_PORT_BASE_ADDRESS
MOV AL,60H
OUT DX,AL ; Point to BASE ADDRESS REGISTER
MOV DX,CONFIG_PORT_BASE_ADDRESS+1
MOV AL,02H
OUT DX,AL ; Update BASE ADDRESS REGISTER
;-----
; EXIT CONFIGURATION STATE
;-----

```

# MEC1618/MEC1618i

```
MOV DX,CONFIG_PORT_BASE_ADDRESS
MOV AX,0AAH ; Config Exit Key
OUT DX,AL.
```

## 4.5.4 CONFIGURATION REGISTER ADDRESS MAPPING

The INDEX PORT defines 256 bytes for configuration. The first 48 of these bytes are Global Configuration registers, which reside in the first 48 bytes of the Configuration part of the address frame for Logical Device 3Fh. Values of INDEX greater than 48 map into registers that are specific to the Logical Device specified in the Global Configuration Logical Device Number Register 7h. These registers reside in upper 20 bytes of the Logical Device address frame. See [Section 4.10.2, on page 68](#) for details.

## 4.6 Configuring Runtime Register Addresses

### 4.6.1 RUNTIME REGISTERS

Runtime Registers are registers that are accessible to the Host within the Host I/O address space. These Host I/O accesses are all mapped into the MEC1618/MEC1618i AHB address space onto devices located on the LPC SPB. Runtime registers all reside within the first 256 bytes of a 1KB Logical Device address frame. The Host accesses these registers with 8-bit LPC I/O accesses. Each 8-bit I/O address is mapped into an 8-bit address in the AHB address space, so the first 256 bytes of the Logical Device frame can accommodate 256 LPC Runtime Registers per Logical Device. The Host I/O addresses are determined by a block of [Base Address Registers](#) located in the LPC Logical Device. The Embedded Controller can access all the Runtime Registers as well, using loads and stores to full AHB addresses.

### 4.6.2 BASE ADDRESS REGISTERS

Each Logical Device has a Base Address Register (BAR). These BARs are located in blocks of Configuration Registers in Logical Device 0Ch, in the AHB address range FF\_3360h through FF\_3384h. On every LPC bus I/O access all Base Address Registers are checked in parallel and if any matches the LPC I/O address the MEC1618/MEC1618i claims the bus cycle.

**Note:** Software should insure that no two BARs map the same LPC I/O address. If two BARs do map to the same address, the [BAR\\_Conflict](#) bit in the [Host Bus Error Register](#) is set when an LPC access targeting the BARConflict address. An EC interrupt can be generated.

Each BAR is 32-bits wide. The format of each BAR is summarized in [Table 4-5, "Base Address Register Format"](#). An LPC I/O request is translated by the BAR into an 8-bit read or write transaction on the AHB bus. The 16-bit LPC I/O address is translated into a 24-bit AHB address

The Base Address Register Table is itself part of the AHB address space. It resides in the Configuration quadrant of Logical Device [Ch](#), the [LPC Interface](#).

**TABLE 4-5: BASE ADDRESS REGISTER FORMAT**

BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24
BIT NAME	LPC Host Address, most significant bits							
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16
BIT NAME	LPC Host Address, least significant bits							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
BIT NAME	Valid	Reserved	Frame					
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
BIT NAME	Reserved	Mask						

## MASK

These 7 bits are used to mask off address bits in the address match between an LPC I/O address and the Host Address field of the BARs, as described in [Section 4.6.3, "Mapping LPC I/O Addresses"](#). A block of up to 128 8-bit registers can be assigned to one base address.

## FRAME

These 6 bits are used to specify a logical device frame number within a bus. This field is multiplied by 400h to provide the frame address within the peripheral bus address. In the MEC1618/MEC1618i Frame values for frames corresponding to logical devices that are not present on the MEC1618/MEC1618i are invalid.

## VALID

If this bit is 1, the BAR is valid and participates in LPC matches. If it is 0 this BAR is ignored

## HOST\_ADDRESS

These 16 bits are used to match LPC I/O addresses

### 4.6.3 MAPPING LPC I/O ADDRESSES

If the following relation holds, a Base Address Register (BAR) will match an LPC I/O address, and the MEC1618/MEC1618i will then claim the LPC bus cycle:

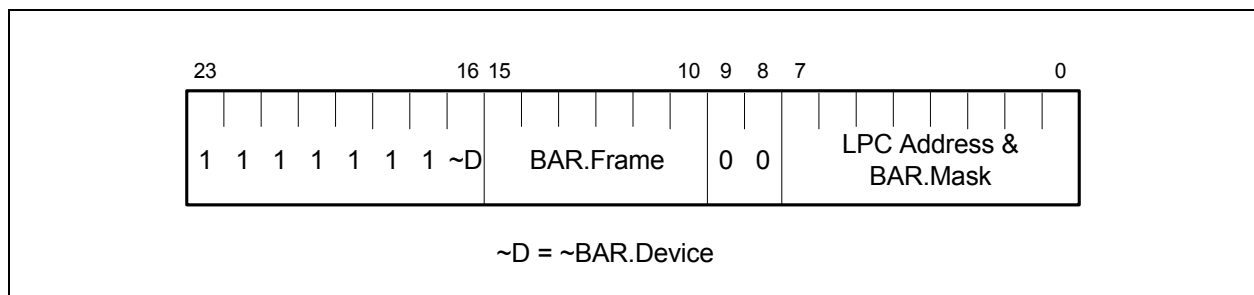
$$(\text{LPC Address} \& \sim \text{BAR.MASK}) == (\text{BAR.LPC\_Address} \& \sim \text{BAR.MASK}) \&\& (\text{BAR.Valid} == 1)$$

If one of the BARs match, the LPC cycle will be claimed and the LPC request will be translated to an AHB address according to the following formula:

$$\text{AHB Address} = \text{FF\_000h} | (\text{BAR.Frame} \ll 10) | (\text{LPC\_Address} \& \text{BAR.MASK})$$

The formula is illustrated in Figure 4-1, "LPC BAR Mapping":

**FIGURE 4-1: LPC BAR MAPPING**



When matching LPC I/O addresses, the MEC1618/MEC1618i ignores address bits that correspond to '1b' bits in the MASK field. When forming the AHB address from the LPC I/O address, the LPC I/O address bits that correspond to the '1b' bits in the MASK are passed through to the AHB address. For example, the [Keyboard Controller \(8042\)](#) Base Address Register has 60h in the LPC Address field, the Frame field is 01h, and the MASK field is 04h. Because of the single '1b' bit in MASK, the BAR matches LPC I/O patterns in the form '00000000011000b', so both 60h and 64h are matched and claimed by the MEC1618/MEC1618i. When forming the AHB address, the Frame number 01h is put in bit positions 15 through 10 and concatenated with FF\_0000h, to form the Keyboard Controller Frame base address of FF\_0400h. If the Host reads from address 60h, a read access will be initiated to AHB address FF\_0400h (the base address OR'd with bit 2 from 60h). If the Host reads from address 64h, a read access will be initiated to AHB address FF\_0404h (again, bit 2 from the LPC I/O address is OR'd into the AHB address).

As another example, if a standard 16550 UART was located at LPC I/O address 238h, then the UART Receive buffer would appear at address 238h and the Line Status register at 23Dh. If the BAR for the UART was set to 0238\_8047h, then the UART will be matched at I/O address 238h, the UART is located in Logical Device 6h, the UART is enabled and the UART device includes 8 registers. The Receive buffer would map to AHB address FF\_1800h and the Line Status register would map to AHB address FF\_1805h.

# MEC1618/MEC1618i

## 4.6.4 BASE ADDRESS REGISTER TABLE

Table 4-6, "Base Address Registers Default Values", lists the Base Address Registers for all logical devices on the MEC1618/MEC1618i. The BAR EC Offsets are relative to the base address of the LPC Logical Device, which is located at AHB address [FF\\_3000h](#). The columns to the right of the heavy line show the field definitions for the default values listed in the column labeled "Reset Default". Shaded fields in [Table 4-6](#) are read-only.

The EC can read and write the BAR table entries with 32-bit, 16-bit or 8-bit accesses. The Host accesses the BAR table using 8-bit reads and writes.

**Note:** The BAR at offset [360h](#) associated with the LPC Interface is different in that LPC I/O accesses that are claimed by this BAR does not translate directly into a AHB addresses. Addresses that are claimed by this BAR, the Configuration Port BAR, are used to manage Configuration Registers. These accesses are described in [Section 4.5, on page 58](#). This BAR is also special in that a byte write to offset 362h (bits[7:0] of the LPC Host Address field) does not directly write into the BAR. Instead, the byte is held in a buffer. A byte write to offset 363h will both write the byte at offset 363h and will copy the buffer into offset 362h. This is done to insure that the 16-bit LPC Host Address field of the Configuration Port BAR is always completely updated in one cycle.

The shaded LPC I/O Address, VALID, FRAME, MASK fields are read-only [Table 4-6](#). The unshaded fields has read/write access.

**TABLE 4-6: BASE ADDRESS REGISTERS DEFAULT VALUES**

BAR EC Offset	LPC Offset	RESET Default	LPC I/O Address	Valid	Frame	Mask	Description
360h	60h	002E_0C01h	002Eh	0	C	1	Logical Device 0Ch: LPC Interface (Configuration Port)
364h	64h	0000_0001h	0000h	0	0	1	Logical Device 00h: Mailbox Register I/F
368h	68h	0060_0104h	0000h	0	1	4	Logical Device 01h: Keyboard Controller (8042 Interface)
36Ch	6Ch	0062_0204h	0062h	0	2	4	Logical Device 02h: ACPI EC Interface 0
370h	70h	0062_0307h	0062h	0	3	7	Logical Device 03h: ACPI EC Interface 1
374h	74h	0062_0407h	0062h	0	4	7	Logical Device 04h: ACPI EC Interface 2
378h	78h	0062_0507h	0062h	0	5	7	Logical Device 05h: ACPI EC Interface 3
37Ch	7Ch	0000_0607h	0000h	0	6	7	Logical Device 06h: ACPI PM1 Interface
380h	80h	0000_0707h	0000h	0	7	7	Logical Device 07h: UART
384h	84h	0092_0800h	0092h	0	8	0	Logical Device 08h: Legacy (GATEA20) I/F

**TABLE 4-6: BASE ADDRESS REGISTERS DEFAULT VALUES (CONTINUED)**

BAR EC Offset	LPC Offset	RESET Default	LPC I/O Address	Valid	Frame	Mask	Description
388h	88h	0000_0E04h	0000h	0	E	4	Logical Device 0Eh: Embedded Flash Interface
38Ch	8Ch	0000_0F07h	0000h	0	F	07h	Logical Device 0Fh: GPSPi for Flash Interface
390h	90h	0000_100Fh	0000h	0	10h	Fh	Logical Device 10h. EM Interface
398h	98h	0000_2000h	0000h	0	20h	0	Logical Device 20h. BIOS Debug Port 0
39Ch	9Ch	0000_2100h	0000h	0	21h	0	Logical Device 21h. BIOS Debug Port 1
3B0h-3ECh	B0h - ECh	0000_0000h	0000h	0	0	0	Reserved

**Note 4-1** In order to avoid address conflict with the [Keyboard Controller \(8042\)](#) LDN 1h at legacy at LPC I/O address 60h/64h, only [ACPI EC Channel 0](#) LDN 2h at AHB base address FF\_0800h should be located at the legacy LPC I/O address 62h/66h. When operating both the [Keyboard Controller \(8042\)](#) and [ACPI EC Channel 0](#) at legacy LPC I/O addresses 60h/62h/64h/66h, the Mask value in [ACPI EC Channel 0](#) BAR should be programmed to 4h and the [ACPI EC Channel 0, Byte Control EC-Register](#) should remain 0. To utilize the [ACPI EC Channel 0](#) in Non Legacy Operation the [ACPI EC Channel 0](#) BAR must be placed on LPC I/O 8 byte boundary. Only bits[3:0] in the Mask field of [ACPI EC Channel 0](#) BAR are programmable, bits[6:4] are read-only '000'.

**Note 4-2** The [ACPI EC Channel 1](#), [ACPI EC Channel 2](#) and [ACPI EC Channel 3](#) (LDN 3h, 4h and 5h) can not be located at the legacy LPC I/O address 62h/66h. The [ACPI EC Channel 1](#), [ACPI EC Channel 2](#) and [ACPI EC Channel 3](#) BAR's must be placed on LPC I/O 8 byte boundaries.

## 4.7 DMA

LPC DMA cycles are mapped to AHB memory addresses by the LPC Logical Device. The addresses reference FIFOs that are associated with the DMA. FIFO addresses are restricted to 32-bit aligned addresses (that is, addresses that are divisible by 4). Configuration for DMA Device access is described in [Section 4.7.1, on page 63](#).

### 4.7.1 DMA CONFIGURATION REGISTERS

The MEC1618/MEC1618i claims an LPC DMA request if the requested channel is listed as valid in the [Table 4-7, "DMA Configuration Register Map"](#). A channel is claimed if the [DMA Configuration Register Format](#) that corresponds to the channel is maps to a Logical Device. In order to execute the DMA operation, the MEC1618/MEC1618i translates the DMA access into a bus read or write of the FIFO that corresponds to channel in question. The address of a DMA FIFO is always one of the first 16 32-aligned addresses within the DMA quadrant of a Logical Device frame.

The mapping in the [DMA Configuration Register Map](#) is used both for mapping LPC DMA I/O requests from the Host to Logical Devices, as well as for mapping DMA requests from Logical Devices to the LPC Bus LDRQ# DMA request signal.

The Host can access the DMA Configuration registers with 8-bit accesses. The EC can access the DMA Configuration registers as four 32-bit registers, eight 16-bit registers or sixteen 8-bit registers.

The mapping of DMA devices into the AHB address space is shown in Figure 4-2, "DMA Address Mapping":

# MEC1618/MEC1618i

FIGURE 4-2: DMA ADDRESS MAPPING

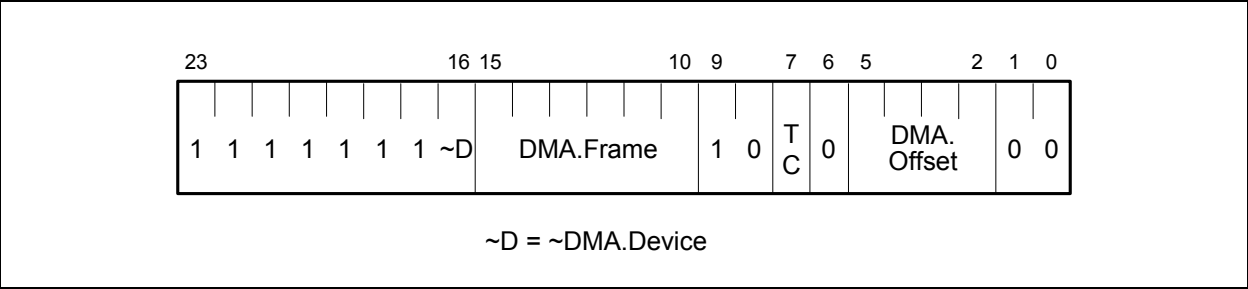


TABLE 4-7: DMA CONFIGURATION REGISTER MAP

Address	Type	Reset	Configuration Register Name
FF_3350h	R/W	0000h	DMA Channel 0
FF_3352h	R/W	0000h	DMA Channel 1
FF_3354h	R/W	0000h	DMA Channel 2
FF_3356h	R/W	0000h	DMA Channel 3
FF_3358h	R	0000h	DMA Channel 4 (Reserved)
FF_335Ah	R/W	0000h	DMA Channel 5
FF_335Ch	R/W	0000h	DMA Channel 6
FF_335Eh	R/W	0000h	DMA Channel 7

**Note 4-3** DMA Channel 4 is reserved in the MEC1618/MEC1618i. LPC Host cycles with DMA channel 4 asserted are unclaimed by the MEC1618/MEC1618i.

TABLE 4-8: DMA CONFIGURATION REGISTER FORMAT

BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
BIT NAME	Valid	Device	Frame					
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
BIT NAME	Reserved				Offset			

**OFFSET**

These 4 bits select which register within the Logical Device address space are to be used for this DMA. The 4 bit field correspond to the following address offsets:

- 0h: +200h
- 1h: +204h
- 2h: +208h
- 3h: +20Ch



4h:	+210h
5h:	+214h
6h:	+218h
7h:	+21Ch
8h:	+220h
9h:	+224h.
Ah:	+228h
Bh:	+22Ch
Ch:	+230h
Dh:	+234h
Eh:	+238h
Fh:	+23Ch

## FRAME

These 6 bits are used to a logical device frame number within a bus. This field is multiplied by 400h to provide the frame address within the peripheral bus address. In the MEC1618/MEC1618i Frame values greater than 0Ah are invalid, since there are only 11 logical devices accessible to the Host.

## VALID

If this bit is 1, the DMA Channel is active on the MEC1618/MEC1618i. If it is 0 this DMA Channel is ignored.

## 4.8 SERIRQ Interrupts

The MEC1618/MEC1618i can routes Logical Device interrupts onto SIRQ stream frames IRQ[0:15]. Routing is controlled by the SIRQ Interrupt Configuration Registers. There is one SIRQ Interrupt Configuration Register for each accessible SIRQ Frame (IRQ); all 16 registers are listed in [Table 4-9, "SIRQ Interrupt Configuration Register Map"](#). Each SIRQ Interrupt Configuration Register controls a series of multiplexors that route to a single Logical Device interrupt as illustrated in [FIGURE 4-3: SIRQ Routing Internal Logical Devices on page 67](#). The format for each SIRQ Interrupt Configuration Register is described in [Table 4-10](#). Each Logical Device can have up to two LPC SERIRQ interrupts. When the MEC1618/MEC1618i is polled by the host, each SIRQ frame routes the level of the Logical Device interrupt (selected by the corresponding SIRQ Interrupt Configuration Register) to the SIRQ stream.

**Note:** Two Logical Devices cannot share a Serial IRQ.

The SIRQ Interrupt Configuration Register The Host can access the Interrupt Configuration registers with 8-bit accesses. The EC can access the Interrupt Configuration registers as four 32-bit registers, eight 16-bit registers or sixteen 8-bit registers.

**Note:** An interrupt is deactivated by setting an entry in the [SIRQ Interrupt Configuration Register Map](#) to FFh, which is the default reset value.

### 4.8.1 SERIRQ CONFIGURATION REGISTERS

**TABLE 4-9: SIRQ INTERRUPT CONFIGURATION REGISTER MAP**

Address	Type	Reset	Configuration Register Name
FF_3340h	R/W	FFh	IRQ0
FF_3341h	R/W	FFh	IRQ1
FF_3342h	R/W	FFh	IRQ2 (nSMI)
FF_3343h	R/W	FFh	IRQ3

# MEC1618/MEC1618i

**TABLE 4-9: SIRQ INTERRUPT CONFIGURATION REGISTER MAP (CONTINUED)**

Address	Type	Reset	Configuration Register Name
FF_3344h	R/W	FFh	IRQ4
FF_3345h	R/W	FFh	IRQ5
FF_3346h	R/W	FFh	IRQ6
FF_3347h	R/W	FFh	IRQ7
FF_3348h	R/W	FFh	IRQ8
FF_3349h	R/W	FFh	IRQ9
FF_334Ah	R/W	FFh	IRQ10
FF_334Bh	R/W	FFh	IRQ11
FF_334Ch	R/W	FFh	IRQ12
FF_334Dh	R/W	FFh	IRQ13
FF_334Eh	R/W	FFh	IRQ14
FF_334Fh	R/W	FFh	IRQ15

**Note 4-4** The SIRQ Interrupt Configuration Registers are through the [Host Access Port](#) as 8-bit accesses. The EC can access the SIRQ Interrupt Configuration Registers as 32-bit, 16-bit across 8-bit boundary or as individual 8-bit accesses.

**TABLE 4-10: SIRQ INTERRUPT CONFIGURATION REGISTER FORMAT**

BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
BIT NAME	Select	Reserved	Frame					

## FRAME

These six bits select the Logical Device as the source for the interrupt.

This field defaults to 3Fh.

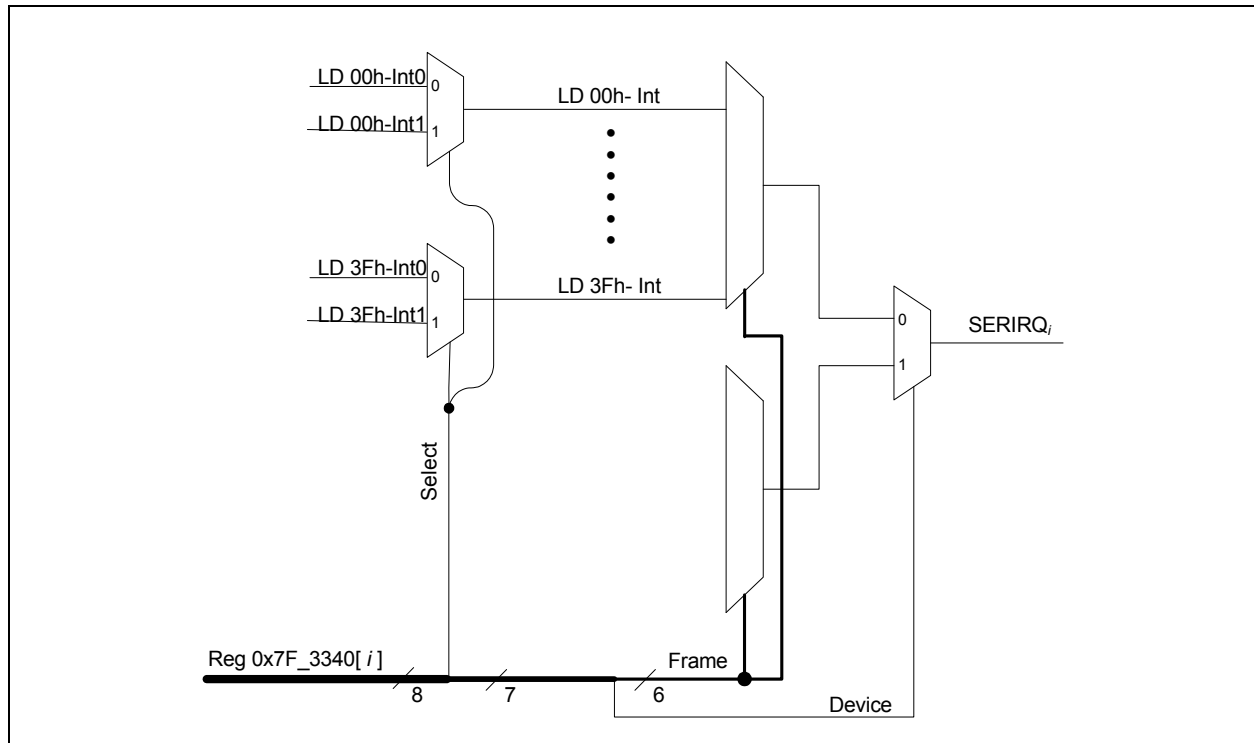
**Note:** The LPC Logical Device (Logical Device Number 0Ch) can be used by the Embedded Controller to generate a Serial Interrupt Request to the Host under software control.

## SELECT

If this bit is 0, the first interrupt signal from the Logical Device is selected for the SERIRQ vector. If this bit is 1, the second interrupt signal from the Logical Device is selected. The KYBD controller is the only Logical Devices on the MEC1618/MEC1618i that has a second interrupt signal. For all other Logical Devices this field is ignored.

This field defaults to 1.

**FIGURE 4-3: SIRQ ROUTING INTERNAL LOGICAL DEVICES**



## 4.8.1.1 MEC1618/MEC1618i SIRQ Routing

**TABLE 4-11: MEC1618/MEC1618i LOGICAL DEVICE SIRQ ROUTING**

SIRQ Interrupt Configuration Register		Logical Device Interrupt Source
SELECT	FRAME	
0	00h	MailBox SIRQ - See <a href="#">Section 11.3.1, on page 226</a>
1	00h	SMI - <a href="#">Section 11.3.1, on page 226</a>
0	01h	Keyboard SIRQ - <a href="#">Section 9.4.1, on page 200</a>
1	01h	Mouse SIRQ - <a href="#">Section 9.4.1, on page 200</a>
0	06h	UART SIRQ - <a href="#">Section 12.3.2, on page 237</a>
0	0Ch	Serial IRQ - <a href="#">Section 6.10.3, on page 150</a>

## 4.9 Configuration Register Reset Conditions

There are two reset conditions that will reset Configuration Registers to default values on the MEC1618/MEC1618i.

- A reset can be caused by a VTR Power On Reset condition or a VCC-Powergood Power on Reset condition.
- Firmware running on the Embedded Controller can reset all Configuration Registers to a default condition through a request to the Embedded Controller sent via the Mailbox interface.

# MEC1618/MEC1618i

## 4.10 Logical Device Configuration/Control Registers

A separate set of control and configuration registers exist for each Logical Device and is selected with the Logical Device # Register (07h). The Logical Devices are listed in [Table 4-1, "Logical Devices," on page 56](#), and the registers within each Logical Device are listed in [Section 4.10.2, on page 68](#).

### 4.10.1 LOGICAL DEVICE ACTIVATION

Many Logical Devices have an Activate register that is used to activate the Logical Device. When a Logical Device is inactive, it is powered down and will not respond to an AHB request. The format for the Activate Register is shown in [Table 4-12, "Activate Register"](#). The Activate Register for the LPC Logical Device is shown in [Table 6-6, "Activate Register," on page 146](#).

**TABLE 4-12: ACTIVATE REGISTER**

HOST OFFSET	BYTE0: 30h			8-bit			HOST SIZE	
EC OFFSET	Frame offset 330h			8-bit			EC SIZE	
POWER	VTR			00b			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R/W
EC TYPE	R	R	R	R	R	R	R	R/W
BIT NAME	Reserved							Activate

#### ACTIVATE

When this bit is 1, the logical device is powered and functional. When this bit is 0, the logical device is powered down and inactive.

### 4.10.2 CONFIGURATION REGISTER MAP

The MEC1618/MEC1618i Configuration register map is shown in [Table 4-14, "MEC1618/MEC1618i Configuration Register Map"](#). Logical Device numbers are in hexadecimal. All Logical Devices are accessible by both the Host and the EC. Logical Devices values between 00h and 3Fh are defined for the MEC1618/MEC1618i; however some are reserved.

The EC has access to all Configuration Registers and all Global Control registers directly on the AHB bus. The address of a Global Control register is FF FF00h+*offset*, where *offset* is the offset listed in Column 2 of [Table 4-15, "Chip-Level \(Global\) Control/Configuration Registers"](#). The other Configuration Registers are accessible at the INDEX ([Table 4-14, Column 1](#)) plus the AHB Address Base ([Table 4-13, "Configuration Register AHB Mapping"](#), Column 2). LDN is the Logical Device Number.

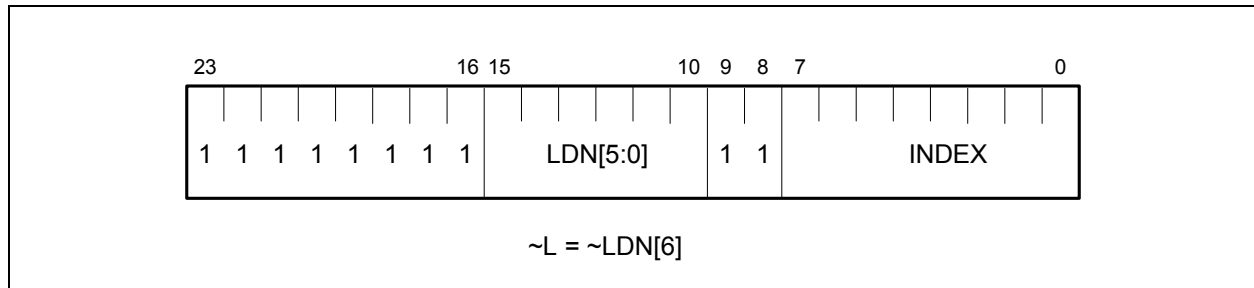
**Note:** The Global Configuration Registers are the first 48 bytes of the configuration space Logical Device 3Fh.

**TABLE 4-13: CONFIGURATION REGISTER AHB MAPPING**

Logical Device Range	AHB Address Base
00h - 3Fh	FF_0300h + (LDN << 10)

The Configuration Register address mapping is illustrated in:

**FIGURE 4-4: CONFIGURATION REGISTER MAPPING**



**TABLE 4-14: MEC1618/MEC1618i CONFIGURATION REGISTER MAP**

LPC CR Index	AHB Offset	Type <a href="#">Note 4-5</a>	Reset <a href="#">Note 4-6</a>	Configuration Register Name
<b>Configuration Registers for LDN 0h (<a href="#">Mailbox Interface</a>) at AHB base address <a href="#">FF_0000h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 1h (<a href="#">Keyboard Controller (8042)</a>) at AHB base address <a href="#">FF_0400h</a></b>				
30h	330h	R/W	00h on <a href="#">nSYS_RST</a>	<a href="#">Activate Register</a>
<b>Configuration Registers for LDN 2h (<a href="#">ACPI EC Channel 0</a>) at AHB base address <a href="#">FF_0800h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 3h (<a href="#">ACPI EC Channel 1</a>) at AHB base address <a href="#">FF_0C00h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 4h (<a href="#">ACPI EC Channel 2</a>) at AHB base address <a href="#">FF_1000h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 5h (<a href="#">ACPI EC Channel 3</a>) at AHB base address <a href="#">FF_1400h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 6h (<a href="#">ACPI PM1</a>) at AHB base address <a href="#">FF_1800h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 7h (<a href="#">UART</a>) at AHB base address <a href="#">FF_1C00h</a></b>				

# MEC1618/MEC1618i

**TABLE 4-14: MEC1618/MEC1618i CONFIGURATION REGISTER MAP (CONTINUED)**

LPC CR Index	AHB Offset	Type <a href="#">Note 4-5</a>	Reset <a href="#">Note 4-6</a>	Configuration Register Name
30h	330h	R/W	00h on nSYS_RST	<a href="#">Activate Register</a>
F0h	3F0h	R/W	00h on nSYS_RST	<a href="#">Configuration Select Register</a>
<b>Configuration Registers for LDN 8h (Legacy (Fast KB)) at AHB base address FF_2000h</b>				
30h	330h	R/W	00h on nSYS_RST	<a href="#">PORT92 Enable Register</a>
<b>Configuration Registers for LDN Ch (LPC Interface) at AHB base address FF_3000h</b>				
30h	330h	R/W	00h on nSYS_RST	<a href="#">Activate Register</a>
40h	340h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ0 Configuration Register</a>
41h	341h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ1 Configuration Register</a>
42h	342h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ2 (nSMI) Configuration Register</a>
43h	343h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ3 Configuration Register</a>
44h	344h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ4 Configuration Register</a>
45h	345h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ5 Configuration Register</a>
46h	346h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ6 Configuration Register</a>
47h	347h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ7 Configuration Register</a>
48h	348h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ8 Configuration Register</a>
49h	349h	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ9 Configuration Register</a>
4Ah	34Ah	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ10 Configuration Register</a>
4Bh	34Bh	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ11 Configuration Register</a>
4Ch	34Ch	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ12 Configuration Register</a>
4Dh	34Dh	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ13 Configuration Register</a>
4Eh	34Eh	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ14 Configuration Register</a>
4Fh	34Fh	R/W	FFh on nSIO_RESET	<a href="#">SIRQ IRQ15 Configuration Register</a>
50h	350h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 0, LSB Configuration Register</a>
51h	351h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 0, MSB Configuration Register</a>
52h	352h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 1, LSB Configuration Register</a>
53h	353h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 1, MSB Configuration Register</a>
54h	354h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 2, LSB Configuration Register</a>
55h	355h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 2, MSB Configuration Register</a>
56h	356h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 3, LSB Configuration Register</a>
57h	357h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 3, MSB Configuration Register</a>
58h	358h	R/W	00h on nSIO_RESET	<a href="#">DMA Channel 4 (Reserved), LSB Configuration Register</a>

**TABLE 4-14: MEC1618/MEC1618i CONFIGURATION REGISTER MAP (CONTINUED)**

LPC CR Index	AHB Offset	Type <a href="#">Note 4-5</a>	Reset <a href="#">Note 4-6</a>	Configuration Register Name
59h	359h	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 4 (Reserved)</a> , MSB Configuration Register
5Ah	35Ah	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 5</a> , LSB Configuration Register
5Bh	35Bh	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 5</a> , MSB Configuration Register
5Ch	35Ch	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 6</a> , LSB Configuration Register
5Dh	35Dh	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 6</a> , MSB Configuration Register
5Eh	35Eh	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 7</a> , LSB Configuration Register
5Fh	35Fh	R/W	00h on <a href="#">nSIO_RESET</a>	<a href="#">DMA Channel 7</a> , MSB Configuration Register
60h - 63h	360h	R/W / R	002E_0C01h on <a href="#">nSIO_RESET</a>	BAR for Configuration Port
64h - 67h	364h	R/W / R	0000_0001h on <a href="#">nSIO_RESET</a>	BAR for Mailbox
68h - 6Bh	368h	R/W / R	0060_0104h on <a href="#">nSIO_RESET</a>	BAR for 8042/Keyboard Interface
6Ch - 6Fh	36Ch	R/W / R	0062_0204h on <a href="#">nSIO_RESET</a>	BAR for ACPI EC Interface 0 (See <a href="#">Note 4-1 on page 63</a> )
70h - 73h	370h	R/W / R	0062_0307h on <a href="#">nSIO_RESET</a>	BAR for ACPI EC Interface 1 (See <a href="#">Note 4-2 on page 63</a> )
74h - 77h	374h	R/W / R	0062_0407h on <a href="#">nSIO_RESET</a>	BAR for ACPI EC Interface 2 (See <a href="#">Note 4-2 on page 63</a> )
78h - 7Bh	378h	R/W / R	0062_0507h on <a href="#">nSIO_RESET</a>	BAR for ACPI EC Interface 3 (See <a href="#">Note 4-2 on page 63</a> )
7Ch - 7Fh	37Ch	R/W / R	0000_0607h on <a href="#">nSIO_RESET</a>	BAR for ACPI PM1 Interface
80h - 83h	380h	R/W / R	0000_0707h on <a href="#">nSIO_RESET</a>	BAR for UART
84h - 87h	384h	R/W / R	0092_0800h on <a href="#">nSIO_RESET</a>	BAR for Legacy (Fast KYBD) Interface
88h - 8Bh	388h	R/W / R	0000_0E04h on <a href="#">nSIO_RESET</a>	BAR for Embedded Flash Interface
8Ch - 8Fh	38Ch	R/W / R	0000_0F07h on <a href="#">nSIO_RESET</a>	BAR for GP-SPI Interface
90h - 93h	390h	R/W / R	0000_100F on <a href="#">nSIO_RESET</a>	BAR for EM Interface
98h - 9Bh	398h	R/W / R	0000_2000 on <a href="#">nSIO_RESET</a>	BAR for <a href="#">BIOS Debug Port 0</a>
9Ch - 9Fh	39Ch	R/W / R	0000_2100 on <a href="#">nSIO_RESET</a>	BAR for <a href="#">BIOS Debug Port 1</a>
B0h - EFh	3B0h - 3EFh	R	0000_0000 on <a href="#">nSIO_RESET</a>	Reserved

# MEC1618/MEC1618i

**TABLE 4-14: MEC1618/MEC1618i CONFIGURATION REGISTER MAP (CONTINUED)**

LPC CR Index	AHB Offset	Type <a href="#">Note 4-5</a>	Reset <a href="#">Note 4-6</a>	Configuration Register Name
<b>Configuration Registers for LDN Eh (Embedded Flash Interface) at AHB base address <a href="#">FF_3800h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN Fh (Flash SPI) at AHB base address <a href="#">FF_3C00h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 10h (EM Interface) at AHB base address <a href="#">FF_4000h</a></b>				
-	-	-	-	None
<b>Configuration Registers for LDN 20h (BIOS Debug Port 0) at AHB base address <a href="#">FF_8000h</a></b>				
30h	330h	R/W	00h on <a href="#">nSYS_RST</a>	Activate Register
<b>Configuration Registers for LDN 21h (BIOS Debug Port 1) at AHB base address <a href="#">FF_8400h</a></b>				
30h	330h	R/W	00h on <a href="#">nSYS_RST</a>	Activate Register
<b>Configuration Registers for LDN 3Fh (Global Configuration) at AHB base address <a href="#">FF_FC00h</a></b>				
00h - 02h	FF00h - FF02h	-		Reserved
03H	FF03	-	-	MCHP Reserved
04h - 06h	FF04h - FF06h	-	00h on <a href="#">nSIO_RESET</a>	Reserved
<a href="#">07h</a>	FF07h	R/W	-	<a href="#">Logical Device Number</a>
08h - 1Fh	FF08h- FF1Fh	-		Reserved
<a href="#">20h</a>	FF20h	R	0Ch	<a href="#">Device ID</a>
<a href="#">21h</a>	FF21h	R	Current Revision hardwired	<a href="#">Device Revision</a> A read-only register that provides device revision information
22h- 23h	FF22h- FF23h	-	04h on <a href="#">nSIO_RESET</a>	MCHP Reserved
<a href="#">24h</a>	FF24h	R/W	00h	<a href="#">Device Mode</a>
25h - 2Fh	FF25h- FF2Fh	-		MCHP Reserved

**Note 4-5** R/W / R means that some parts of a register are read/write and some parts are read-only.

**Note 4-6** Resets are defined in [Section 5.0, "Power, Clocks, and Resets"](#): [nSYS\\_RST](#) on [page 108](#) and [nSIO\\_RESET](#) on [page 76](#).



## 4.11 Chip-Level (Global) Control/Configuration Registers [00h - 2Fh]

The chip-level (global) registers reside in Logical Device 3Fh at AHB addresses FF\_FF00h through FF\_FF2Fh. All unimplemented registers and bits ignore writes and return zero when read. The global registers are accessed in the configuration address range [00h - 2Fh] in all Logical Devices. There is no Activate associated with Logical Device 3Fh: the Global Configuration Registers are always accessible.

As with all Configuration Registers, the INDEX PORT is used to select a Global Configuration Register in the chip. The DATA PORT is then used to access the selected register.

The Host can access all the Global Configuration registers at the offsets listed in [Table 4-15, "Chip-Level \(Global\) Control/Configuration Registers"](#) through the INDEX PORT and the DATA PORT. The EC can access all these registers at the listed offsets from the AHB Base Address shown in [Table 4-14, "MEC1618/MEC1618i Configuration Register Map"](#).

**TABLE 4-15: CHIP-LEVEL (GLOBAL) CONTROL/CONFIGURATION REGISTERS**

Register	Offset	Description
<b>CHIP (GLOBAL) CONTROL REGISTERS</b>		
Reserved	00h-03h	Reserved, Writes are ignored, reads return 0.
Reserved	04h - 06h	Reserved - Writes are ignored, reads return 0.
Logical Device Number	07h	A write to this register selects the current logical device. This allows access to the control and configuration registers for each logical device. <b>Note:</b> The Activate command operates only on the selected logical device.
Reserved	08h - 1Fh	Reserved - Writes are ignored, reads return 0.
Device ID Hard Wired	20h	A read-only register that provides device identification: Bits[7:0] = 0Ch
Device Revision Hard Wired	21h	A read-only register that provides device revision information. Bits[7:0] = current revision when read
Reserved	22h - 23h	Reserved - Writes are ignored, reads return 0.
Device Mode	24h	Bit [1:0] Reserved – writes ignored, reads return “0”. Bit[2] SerIRQ Mode ( <a href="#">Note 4-7</a> ) = 0: Serial IRQ Disabled. = 1: Serial IRQ Enabled (Default). Bit [7:3] Reserved – writes ignored, reads return “0”.
Reserved	25h - 27h	Reserved - Writes are ignored, reads return 0.
Test Register	28h	MCHP Test Mode Register, Reserved for Microchip
Test Register	29h	MCHP Test Mode Register, Reserved for Microchip
Reserved	2Ah - 2Bh	Reserved - Writes are ignored, reads return 0.
Test Register	2Ch	MCHP Test Mode Register, Reserved for Microchip
Reserved	2Dh - 2Fh	Reserved - Writes are ignored, reads return 0.

**Note 4-7** The SerIRQ Mode bit controls the SER\_IRQ pin, the CLKRUN# pin and the affects of LPC DMA requests on CLKRUN# (See [Section 6.7, on page 140](#) and [Section 6.8, on page 143](#)).

# MEC1618/MEC1618i

## 5.0 POWER, CLOCKS, AND RESETS

### 5.1 General Description

The [Power, Clocks, and Resets](#) chapter includes descriptions of the MEC1618/MEC1618i [Clock Generator](#), [Power Configuration](#) and [Reset Interface](#). The [Clock Generator](#), in addition to describing clock sources, also features a [Generic Block Clocking Model](#) and a [Power Management Interface](#). The [Reset Interface](#) description includes internal and external reset sources, as well as descriptions of an internal [1.8V Regulator](#) and [Power MUX](#). Other descriptions in this chapter include [References](#), a [Port List](#), [Interrupt Interface](#) and a [Registers Interface](#).

The [Power Configuration](#), [Clock Generator](#) and Reset circuits have the following features:

#### 5.1.1 Power Configuration

**TABLE 5-1:** [Power Configuration FEATURES](#)

Feature	Page
Description of <a href="#">Power Supplies and Clocks ACPI Context</a>	<a href="#">101</a>
Enumerated <a href="#">Power Supply Configurations</a>	<a href="#">102</a>
<a href="#">Power-Up Sequence</a> Definition	<a href="#">103</a>
<a href="#">1.8V Regulator</a>	<a href="#">108</a>
<a href="#">Power MUX</a>	<a href="#">109</a>

#### 5.1.2 Clock Generator

**TABLE 5-2:** [Clock Generator FEATURES](#)

Feature	Page
Three asynchronous clock sources: <a href="#">20 MHz Oscillator</a> , <a href="#">32.768 kHz Oscillator</a> and <a href="#">PCI_CLK</a>	<a href="#">79</a>
Efficient Logic Design and Controllable <a href="#">Master Clock Trees</a> to Minimize Power Consumption	<a href="#">86</a>
Independent EC-driven <a href="#">Power Management Interface</a>	<a href="#">89</a>
<a href="#">20 MHz Oscillator</a> Optimized for 115.2K baud 16C550A UART Support	–
<a href="#">Generic Block Clocking Model</a>	<a href="#">83</a>
EC-accessible <a href="#">Registers Interface</a>	<a href="#">111</a>

## 5.1.3 Reset Interface

**TABLE 5-3: Reset Interface FEATURES**

Feature	Page
VTR and VBAT Reset Signaling (VTRGD, VBAT_POR, nSYS_RST, nEC_RST)	104
RESET Pin Interface	105
VCC Reset Signaling (VCC Power Good)	109
Watch-Dog Timer Forced Reset	110
Interrupt Interface	134

## 5.2 References

- Advanced Configuration and Power Interface Specification, Revision 1.0b, February 2, 1999.
- Intel® 82801DBM I/O Controller Hub 4 Mobile (ICH4-M), Datasheet, Order Number: 252337-001, Intel Corp., January 2003.
- PCI Mobile Design Guide, Version 1.1, PCI-SIG, December 18, 1998.

## 5.3 Port List

**TABLE 5-4: Power, Clocks, and Resets Port List**

Signal Name	Direction	Source	Destination	Description
ARC_CLK_DISABLE	Input	External - Embedded Controller Core (EC)	Internal - EC Power State Controls	Indication for the Power Management Interface (see Section 5.4.8 on page 89) that an EC Sleep instruction has occurred and the processor is sleeping or halted.
SLEEP_STATE	Output	Internal - EC Power State Controls	Internal - 20 MHz Oscillator Control and External Functions as Needed.	System Sleeping State status indicator as described in "EC Power State Controls," on page 94.
SLEEP_FLAG	Output	Clock Control Register	Internal - 20 MHz Oscillator Control, Block Sleep Enables and External Functions as Needed.	Sleep indicator from the Clock Control Register. See also "EC Power State Controls," on page 94 and "Block Sleep Enables," on page 97.
PCI_CLK	Input	External	Internal Test Functions	33 MHz PCI Clock Input (also TEST_CLK_IN).
LRESET#	Input	External	Internal	PCI Reset. See Section 5.6.11, "LPC RESET," on page 110.
WAKE	Input	External	Internal - Wake Interface	Aggregated Wake indicator from the Section 15.0, "EC Interrupt Aggregator," on page 296 for the Power Management Interface (see Section 5.4.8 on page 89).

# MEC1618/MEC1618i

**TABLE 5-4: Power, Clocks, and Resets Port List (CONTINUED)**

Signal Name	Direction	Source	Destination	Description
WDT_ALRT	Input	External	Internal - <a href="#">Reset Interface</a>	Causes a <a href="#">Watch-Dog Timer Forced Reset</a> as described in <a href="#">Section 5.6.12, "Watch-Dog Timer Forced Reset,"</a> on page 110.
ALL_BLOCK_SLEEP_ENABLE_OUTPUTS	Output	Internal - <a href="#">Block Sleep Enables</a>	External - Sleep-able Blocks	To all blocks as defined in <a href="#">Section 5.7.4, "Block Sleep Enable Registers,"</a> on page 118. See also "Block Sleep Enables," on page 97
<a href="#">MCLK</a> , <a href="#">EC_BUS_CLK_EN</a> , <a href="#">LPC_BUS_CLK_EN</a> , <a href="#">MCLK_DIV1_EN</a> , <a href="#">MCLK_DIV2_EN</a> , <a href="#">MCLK_DIV4_EN</a> , <a href="#">MCLK_DIV8_EN</a> , <a href="#">MCLK_DIV10_EN</a> , <a href="#">MCLK_DIV16_EN</a> , <a href="#">MCLK_DIV20_EN</a> , <a href="#">MCLK_DIV32_EN</a> , <a href="#">MCLK_DIV64_EN</a> , <a href="#">MCLK_DIV128_EN</a> , <a href="#">MCLK_DIV203_EN</a> , <a href="#">MCLK_5HZ_EN</a> , <a href="#">MCLK_DIV20_EN_HST</a> , <a href="#">X32K_CLK</a>	Output	Internal - <a href="#">MCLK Sourced Clocking and 32K Clock Domain</a>	External	See <a href="#">Section 5.4.9, "MCLK Sourced Clocking,"</a> on page 99 and <a href="#">Section 5.4.10, "32K Clock Domain,"</a> on page 100.
32KHZ_OUT	Output	Internal	External	Off-Chip 32.768kHz Oscillator Output (see " <a href="#">32kHz OUTPUT</a> ," on page 116).
ALL_BLOCK "CLOCK REQUIRED" STATUS BITS	Input	External - Sleep-able Blocks	Internal - <a href="#">20 MHz Oscillator Control</a>	see <a href="#">Section 5.7.5, "Clock Required Status Registers,"</a> on page 124.
nSIO_RESET	Output	Internal	Internal/ External	EC-driven SIO Reset and External System Reset (nRESET_OUT-See <a href="#">Table 3.15, "Miscellaneous Functions,"</a> on page 61). (see " <a href="#">iRESET OUT</a> ," on page 114).
VCC_PWRGD	Input	External	Internal - <a href="#">Host Clock Domain</a>	VCC Power Good Input. See <a href="#">Section 5.4.9.4, "Host Clock Domain,"</a> on page 100 and <a href="#">Section 5.6.10, "VCC Power Good,"</a> on page 109. The EC can determine the state of the <a href="#">VCC_PWRGD</a> signal using <a href="#">VCC_PWRGD</a> bit in the <a href="#">Block Sleep Enable Registers</a> . See also <a href="#">Section 5.6.11, "LPC RESET,"</a> on page 110.
VCC_PWRGD_BUFF	Output	Internal	External - Pad Buffers	Buffered <a href="#">VCC_PWRGD</a> output used to tri-state VCC-related Pads.
PCR_INT	Output	Internal	External (" <a href="#">EC Interrupt Aggregator</a> ," on page 296)	see <a href="#">Section 5.9, "Interrupt Interface,"</a> on page 134.
FLASH_PGM	Input	External	Internal - <a href="#">Power-Fail and Reset Status Register</a>	see " <a href="#">FLASH</a> ," on page 132.
LPC_RST#	Output	Internal	External (LPC Interface)	see <a href="#">Section 5.6.11, "LPC RESET,"</a> on page 110.
VTR	Power Well	External	–	Suspend Supply
VBAT	Power Well	External	–	Battery Supply

**TABLE 5-4:** Power, Clocks, and Resets Port List (CONTINUED)

Signal Name	Direction	Source	Destination	Description
VSS	Power Well	External	–	Digital Ground
AGND	Power Well	External	–	Analog Ground for the 32.768 kHz Oscillator.
VTR_1.8	Power Well	Internal	–	Output of the internal 1.8 V regulator (see Section 5.6.8, "1.8V Regulator," on page 108).
VTR1.8_BAT	Power Well	Internal	–	Output of the internal Power MUX for VBAT-backed logic (see Section 5.6.9, "Power MUX," on page 109).
nSYS_RST	Output	Internal	Internal/ External	Synchronized VTR Power Good (Section 5.6.6, "nSYS_RST," on page 108).
nEC_RST	Output	Internal	Internal/ External	Stretched nSYS_RST used for EC reset and Registers Interface (see Section 5.6, "Reset Interface," on page 104).
VBAT_POR	Output	Internal	Internal/ External	VBAT Power On Reset (Section 5.6.5, "VBAT_POR," on page 108)
VR_CAP	Power Well	Internal	–	Capacitor Connection for Internal Voltage Regulator (4.7µF ±20%, ESR 2 Ohms, max.) (see also Section 5.6.8, "1.8V Regulator," on page 108).
RESETI#	Input	External	RESET Pin Interface	See Section 5.6.3, "RESET Pin Interface," on page 105.
AVTRGD	Output	Internal	VBAT-Powered Control Interface	Delayed ALLGD signal in FIGURE 5-17: Reset Interface Block Diagram on page 105.

## 5.4 Clock Generator

### 5.4.1 OVERVIEW

The MEC1618/MEC1618i [Clock Generator](#) includes clock sources as illustrated in Figure 5-1, "Clock Generator Block Diagram": the 20 MHz Oscillator, 32.768 kHz Oscillator and the PCI Clock (PCI\_CLK in Table 5-4). The relationship of these clock sources to the system power supplies is described in Section 5.4.2, "Power Supplies and Clocking," on page 78; their relationship to the ACPI power states is described in Section 5.5.1, "Power Supplies and Clocks ACPI Context," on page 101.

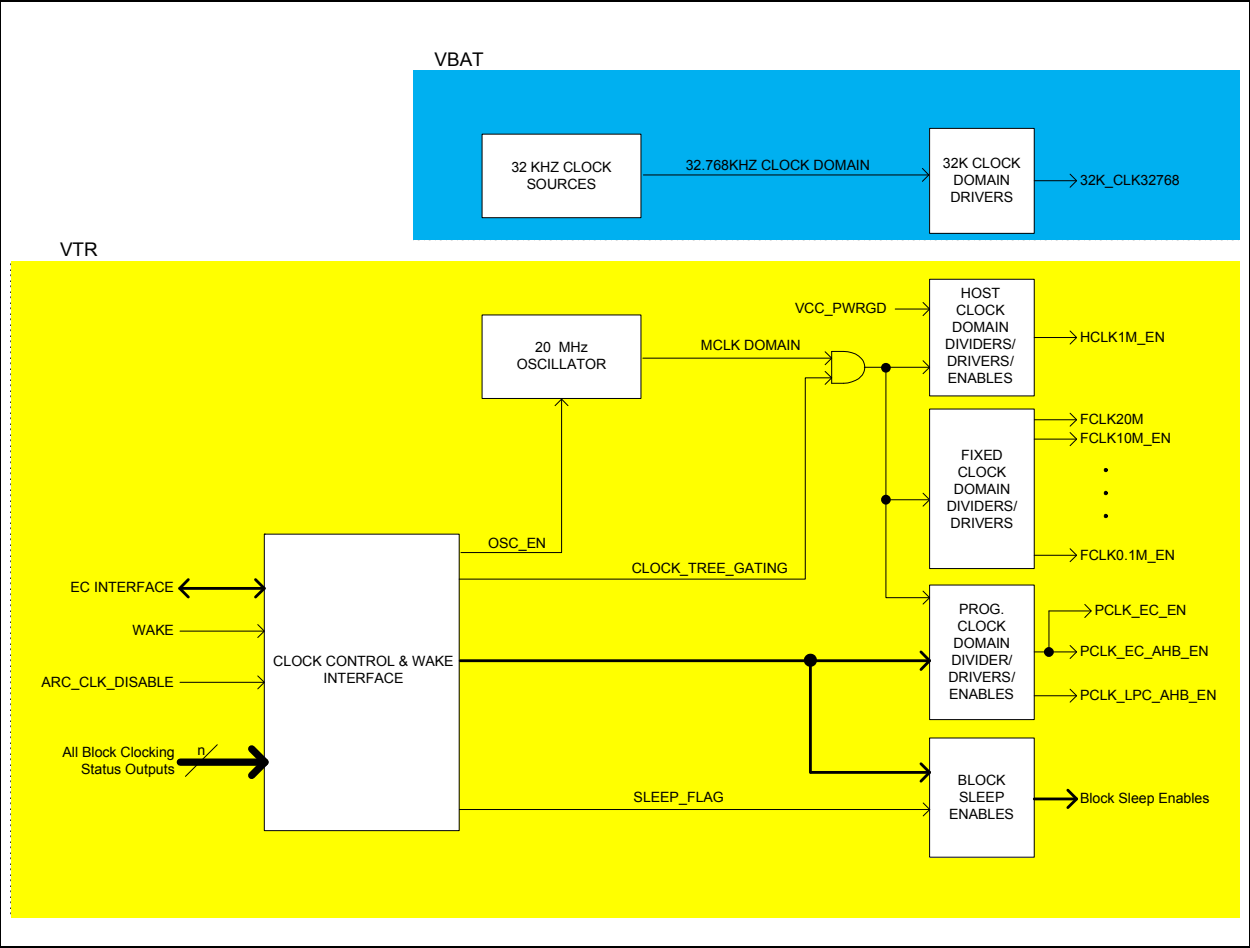
MCLK Sourced Clocking includes [Programmable Clock Domains](#), a [Fixed Clock Domain](#) and a [Host Clock Domain](#). The output from the 32.768 kHz Oscillator, or a 32 kHz external input clock defines a 32K Clock Domain.

The [Clock Generator](#) also includes the definition of a [Generic Block Clocking Model](#) that provides the foundation for a [Power Management Interface](#). This interface defines several [EC Controlled Dynamic Power States](#) that can influence power consumption at the block level and within the [Clock Generator](#).

The [Clock Generator](#) includes an EC accessible [Registers Interface](#).

# MEC1618/MEC1618i

FIGURE 5-1: Clock Generator BLOCK DIAGRAM



## 5.4.2 POWER SUPPLIES AND CLOCKING

Table 5-5 illustrates clocking capabilities versus power supply availability. For more information, see Section 5.5, "Power Configuration," on page 101.

TABLE 5-5: CLOCKS VS. POWER SUPPLIES

Power Supply States (Note 5-1)			Clocks			
VBAT	VTR	VCC	PCI Clock	32.768 kHz Oscillator	32K External	20 MHz Oscillator
OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	OFF	OFF		ON/OFF (Note 5-2) OFF	ON/OFF (Note 5-4) OFF	ON/OFF (Note 5-3)
ON	ON	OFF				
ON	ON	ON	ON/OFF			

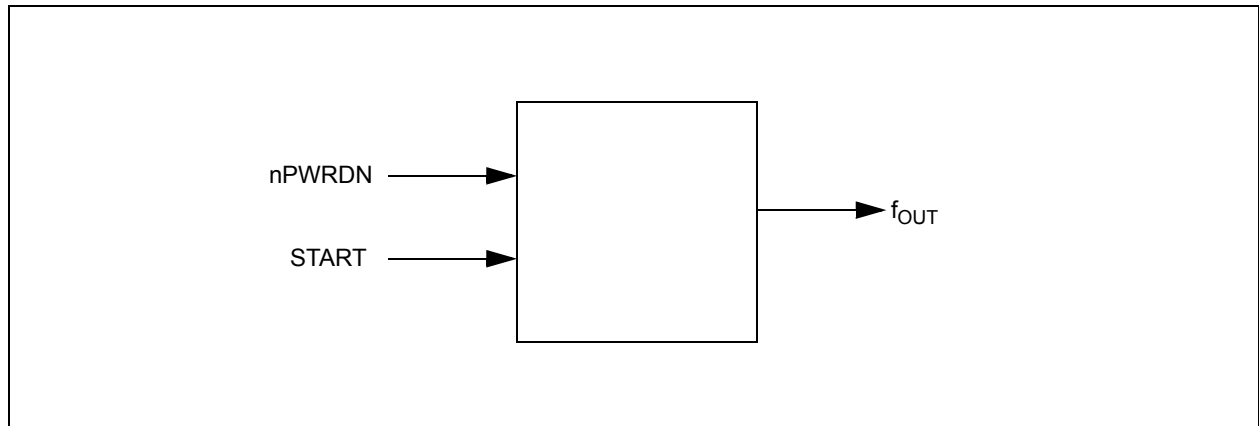
- Note 5-1** Power supply states not illustrated in [Table 5-5](#) are undefined (see also [Section 5.5, "Power Configuration,"](#) on page 101).
- Note 5-2** This is true only after the EC asserts the [32K\\_EN](#) bit in the [Clock Enable Register](#).
- Note 5-3** There is accuracy adjustment latency as described in [Section 5.4.3, "20 MHz Oscillator,"](#) on page 79. The [20 MHz Oscillator](#) can be disabled using the [Power Management Interface](#).
- Note 5-4** An external single-ended 32.768 kHz clock source may be [VTR](#) or [VBAT](#) powered. Note that higher than normal [VBAT](#) current may occur when [VTR](#) transitions from unpowered to powered if the switching threshold on the external single-ended 32.768 kHz clock source is different than the internal [Power MUX](#) switch threshold.

## 5.4.3 20 MHz OSCILLATOR

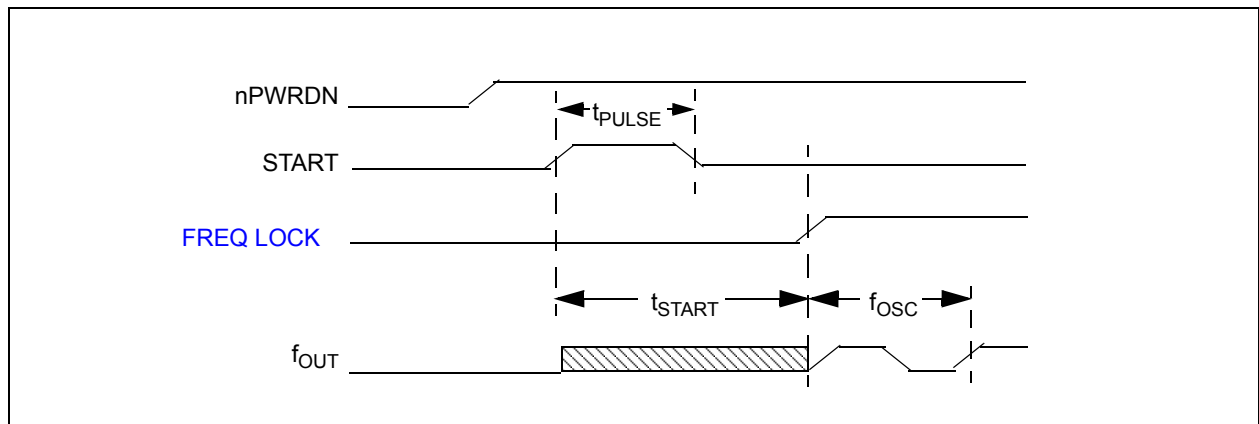
The MEC1618/MEC1618i [Clock Generator](#) includes a high-accuracy, low power, low start-up latency [20 MHz Oscillator](#). The [20 MHz Oscillator](#) is always enabled except during power up and in the [SYSTEM HEAVY SLEEP 3](#) and [SYSTEM DEEPEST SLEEP](#) state when the oscillator is stopped by hardware as described in [Section 5.4.8, "Power Management Interface,"](#) on page 89. The [20 MHz Oscillator](#) timing parameters are shown in [Table 5-6](#).

The [20 MHz Oscillator](#) is reset by [VTRGD](#) as described in [Section 5.6, "Reset Interface,"](#) on page 104 and powered by a dedicated voltage regulator and power cycled as described in [Section 5.4.8.6, "Wake Interface,"](#) on page 98.

**FIGURE 5-2:** [20 MHz Oscillator BLOCK DIAGRAM](#)



**FIGURE 5-3:** [20 MHz Oscillator LOCK TIME](#)



# MEC1618/MEC1618i

**TABLE 5-6:**     20 MHz Oscillator TIMING PARAMETERS

Symbol	Parameter	Limits			Units	Comments
		Min	Nom	Max		
$f_{OSC}$	Oscillator Output Frequency	19.76	20.27	20.78	MHz	$\pm 2\%$ (Commercial) $\pm 2.5\%$ (Industrial) FREQ LOCK = '1' (Note 5-5)
$t_{START}$	Oscillator Lock Time	–	–	300	$\mu s$	
$t_{PULSE}$	Oscillator Start Pulse Time	2	–	–	$\mu s$	

**Note 5-5**     When FREQ LOCK = '0', MCLK Sourced Clocking is defined by the 10 MHz Ring Oscillator.

## 5.4.4     32.768 KHZ OSCILLATOR

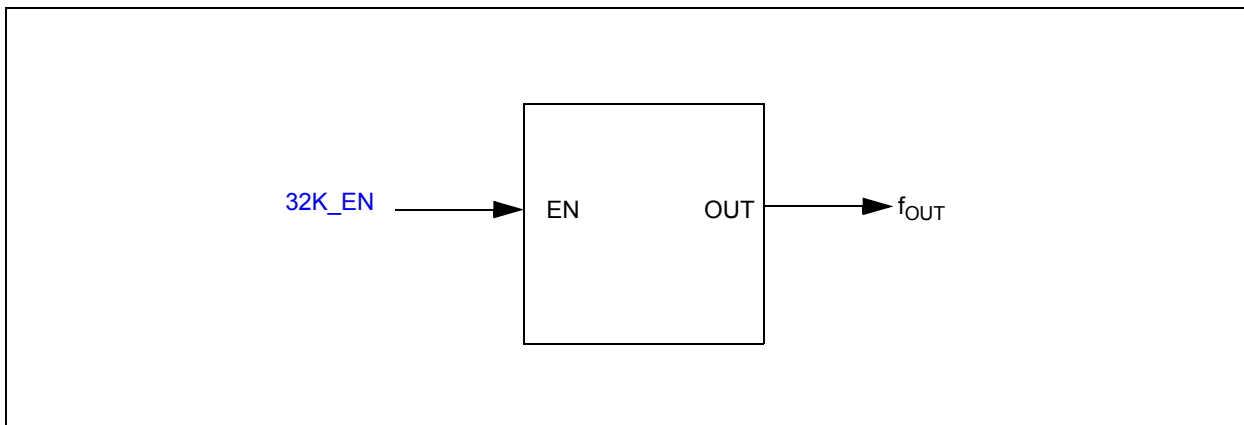
### 5.4.4.1     Description

The 32.768 kHz Oscillator provides a clock source for the 32K Clock Domain (see also Section 5.4.4.2, "32 kHz Clock Domain Switching," on page 81).

The 32.768 kHz Oscillator is controlled by the 32K\_EN bit in the Clock Enable Register (see Section 5.8.2, "Clock Enable Register," on page 133). When 32K\_EN is asserted, there is a start-up delay ( $t_{START}$ ) for the 32.768 kHz Oscillator as shown in Table 5-7.

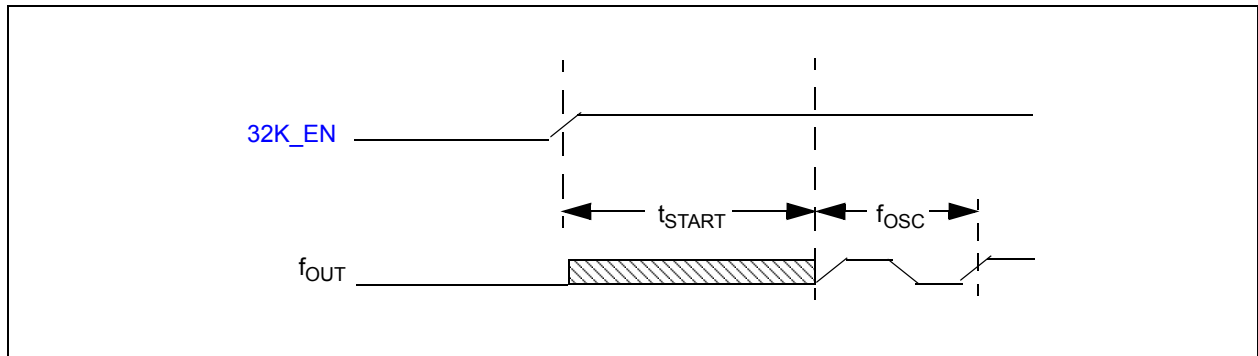
The clocks sourced by the 32.768 kHz Oscillator in the 32K Clock Domain operate as described in Table 5-22, "Typical MEC1618/MEC1618i Clocks vs. ACPI Power States" (see Section 5.5.1, "Power Supplies and Clocks ACPI Context," on page 101).

**FIGURE 5-4:**     32.768 kHz Oscillator BLOCK DIAGRAM





**FIGURE 5-5:** 32.768 kHz Oscillator TIMING



**TABLE 5-7:** 32.768 kHz Oscillator TIMING PARAMETERS

Symbol	Parameter	Limits			Units	Comments
		Min	Nom	Max		
$f_{OSC}$	Oscillator Output Frequency	32.113	32.768	33.423	kHz	
$t_{START}$	Oscillator Lock Time	–	–	300	$\mu s$	

#### 5.4.4.2 32 kHz Clock Domain Switching

Clock sources for the 32K Clock Domain are illustrated in Figure 5-6 includes both an internal 32.768 kHz Oscillator and an external 32 kHz clock source. 32 kHz Clock Domain Switching is controlled as defined in Table 5-8.

The external 32 kHz clock source is configured using the EXT32K\_VBAT bit D0 in the Clock Enable Register.

**TABLE 5-8:** 32KHZ CLOCK DOMAIN CONTROL

AD_EN (Note 5-6)	32K_EN (Note 5-7)	Mode	Description	Notes
0	0	Off	32 kHz clocking (internal/external) disabled (default)	
0	1	Internal	Internal 32.768 kHz Oscillator only	
1	0	External	External 32 kHz Oscillator only	Note 5-8
1	1	Auto Switch	Internal 32 kHz Oscillator used only if external not detected.	

**Note 5-6** AD\_EN is bit D2 in the Clock Enable Register.

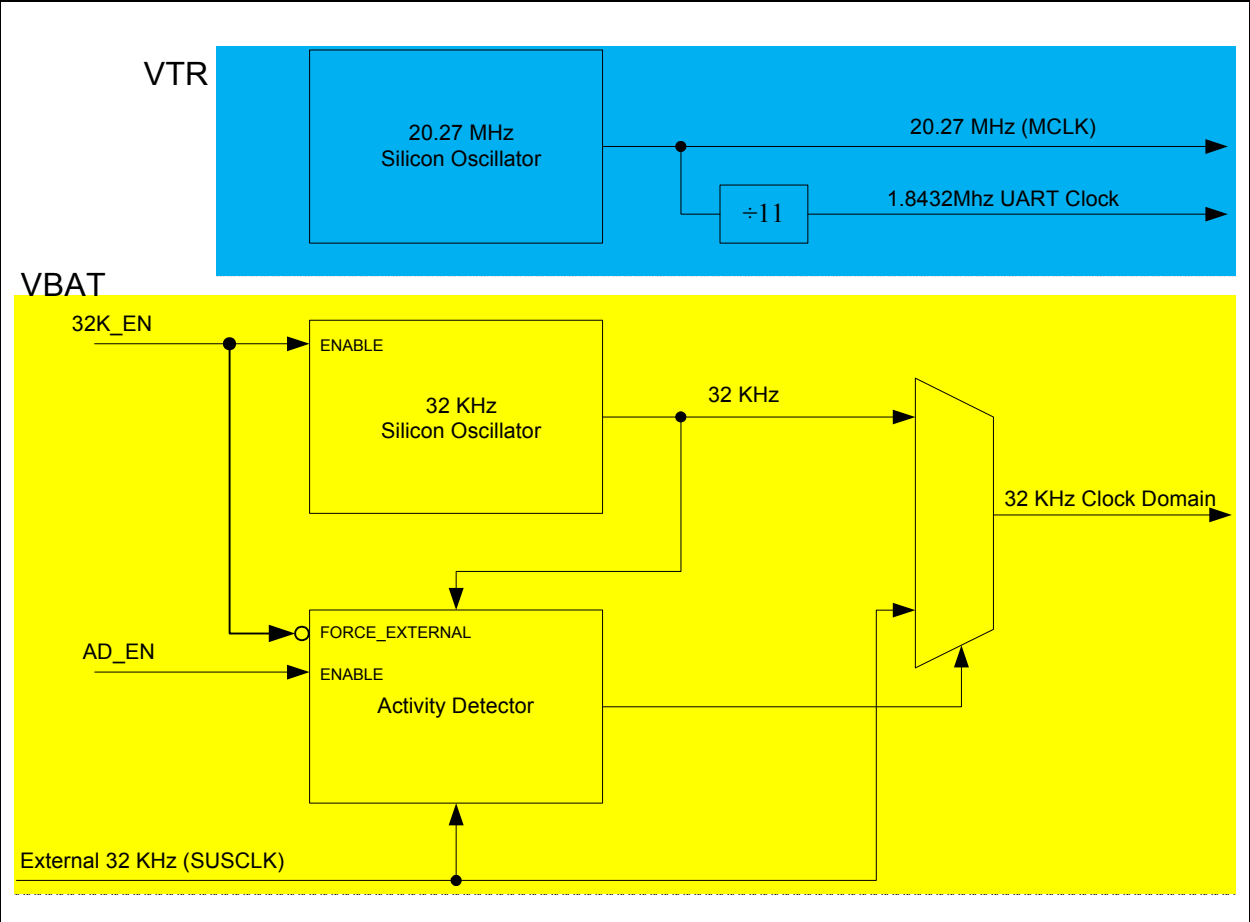
**Note 5-7** 32K\_EN is bit D1 in the Clock Enable Register.

**Note 5-8** If AD\_EN is asserted ('1'), the external 32 kHz Oscillator is selected and the external 32 kHz clock stops (i.e., stays high or low for longer than ~31  $\mu s$ ), the external clock must remain stopped for a minimum of 300  $\mu s$ .

If AD\_EN is asserted ('1'), and the 32K\_EN changes from asserted ('1') to de-asserted, AD\_EN must remain asserted for a minimum of 300  $\mu s$ .

# MEC1618/MEC1618i

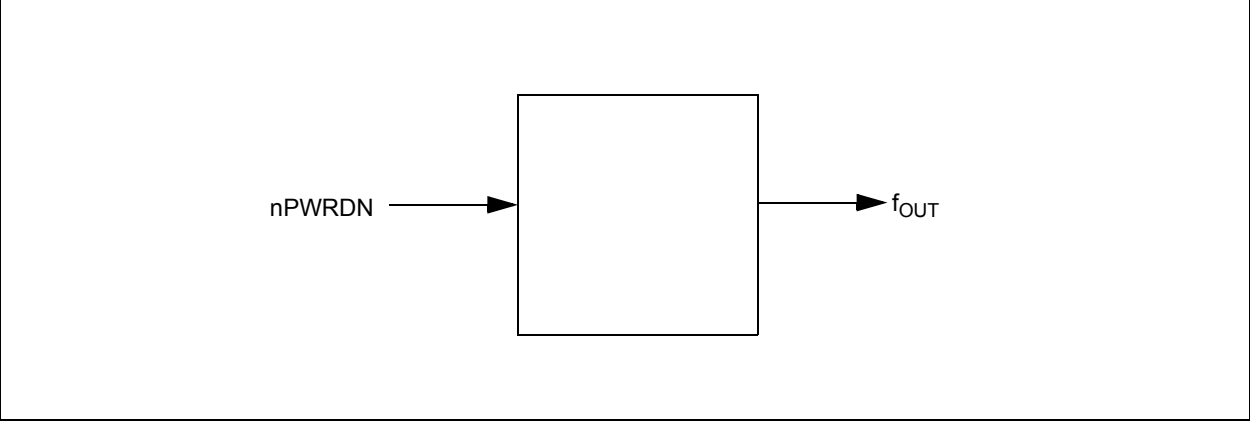
FIGURE 5-6: MEC1618/MEC1618I CLOCKING BLOCK DIAGRAM



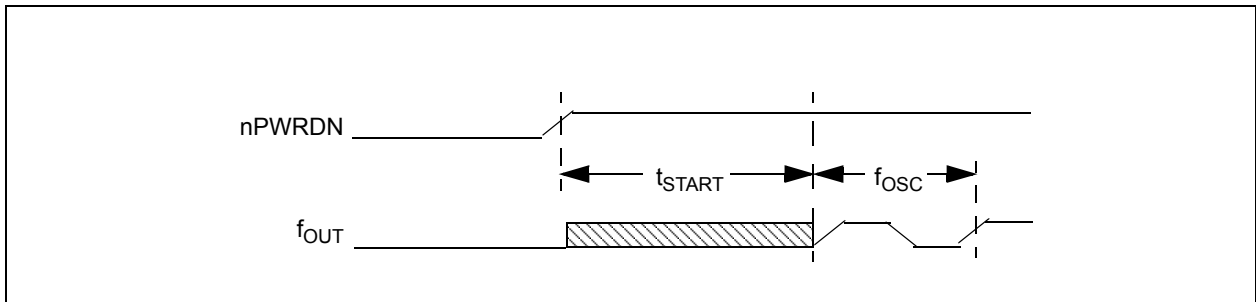
## 5.4.5 10 MHz RING OSCILLATOR

The MEC1618/MEC1618i [Clock Generator](#) includes a low start-up latency [10 MHz Ring Oscillator](#). The [10 MHz Ring Oscillator](#) is enabled during power up. The [10 MHz Ring Oscillator](#) timing parameters are shown in [Table 5-9](#).

FIGURE 5-7: 10 MHz Ring Oscillator BLOCK DIAGRAM



**FIGURE 5-8:** 10 MHz Ring Oscillator LOCK TIME



**TABLE 5-9:** 10 MHz Ring Oscillator TIMING PARAMETERS

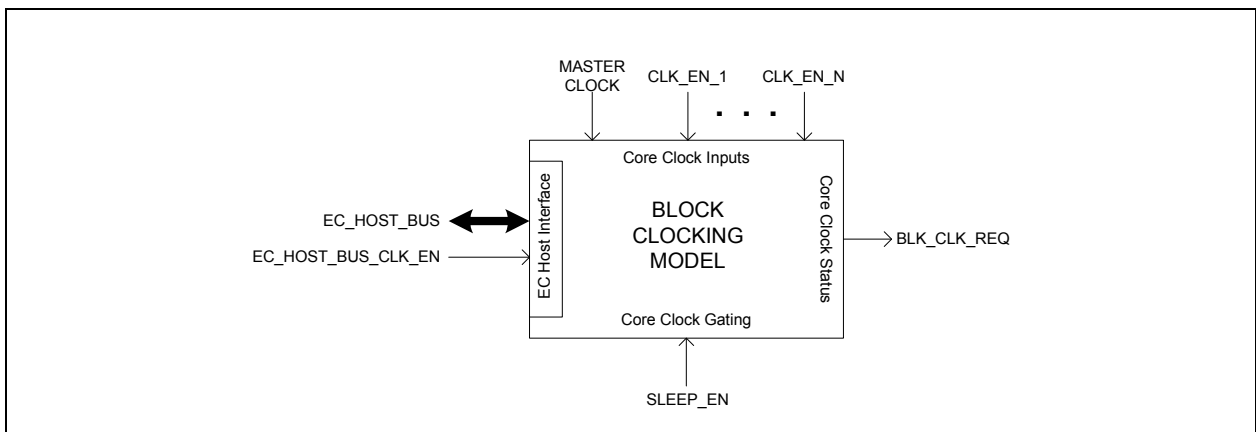
Symbol	Parameter	Limits			Units	Comments
		Min	Nom	Max		
$f_{OSC}$	Oscillator Output Frequency	5	10	15	MHz	
$t_{START}$	Oscillator Startup Time	—	—	10	$\mu$ s	

## 5.4.6 GENERIC BLOCK CLOCKING MODEL

### 5.4.6.1 Overview

The [Generic Block Clocking Model](#) defines the block [Clock Gating](#) interface that is assumed by the [Clock Generator](#) for all of the MEC1618/MEC1618i internal blocks identified in the [Block Sleep Enable Registers](#). [Components](#) of this model are illustrated in [Figure 5-9](#) and described in [Section 5.4.6.2](#). The response of this model to the actions of the [Power Management Interface](#) is described in [Section 5.4.6.3, "Behavior," on page 85](#).

**FIGURE 5-9:** Generic Block Clocking Model ILLUSTRATION



## 5.4.6.2 Components

As shown in [Figure 5-9](#) the external interface for the [Generic Block Clocking Model](#) includes an EC (or other available bus master) host interface, core clock Inputs (which may be clocks or clock enables), a logical core clock gating control and a core clock status output. Not shown in [Figure 5-9](#) is the internal interface for the [Generic Block Clocking Model](#) that includes a block enable bit and may also include a block idle status indicator. Each of the [Generic Block Clocking Model](#) internal and external interface elements and operational states are described in [Table 5-10](#), "Generic Block Clocking Model Components".

When firmware de-asserts the internal block enable bit the block is disabled and in a minimum power consumption state. Depending on the implementation, the host may need to guarantee that the block is not in use before the internal enable bit is de-asserted because it may also function as a reset. Transitions to a minimum block power consumption state while the internal enable bit remains asserted may be requested by the [Power Management Interface](#) using an external sleep enable input (see also [Section 5.4.6.3, "Behavior," on page 85](#)). In both cases (i.e., when the block is disabled or sleeping), the core clock required status indicator output (BLK\_CLK\_REQ in [Figure 5-9](#)) is de-asserted.

When firmware asserts the internal block enable and the external sleep enable input is not asserted, or the external sleep enable input is asserted but the internal idle indicator is not asserted, the block is operational and in a maximum power consumption state. In both of these cases, the core clock required status indicator output is asserted.

**TABLE 5-10: Generic Block Clocking Model COMPONENTS**

Internal Enable Bit ( <a href="#">Note 5-10</a> )	External SLEEP_EN Input ( <a href="#">Note 5-11</a> )	Block Idle Status	Core Clock Required Status Output ( <a href="#">Note 5-9</a> )	State	Power	Description
0	X	X	0	DISABLED	MINIMUM	Block is disabled by firmware and the core clock is not needed and gated 'off' internally. Note: it may be up to the host to guarantee that the block is not in use before the internal enable bit is de-asserted because the internal enable may also function as a reset when not asserted.
1	0	NOT IDLE	1	FULL POWER	MAXIMUM	The full power state identifies the block normal operation mode where the block is neither disabled by firmware nor commanded to sleep by the <a href="#">Power Management Interface</a> .
		IDLE				
	1	NOT IDLE		PREPARING TO SLEEP		A sleep command has been asserted but the core clock is still required because the block is not idle.
		IDLE	0	SLEEPING	MINIMUM	A sleep command has been asserted, the block is idle and the core clocks are stopped ( <a href="#">Note 5-12</a> ).

- Note 5-9** The “Block Clock Required Status Output” (BLK\_CLK\_REQ in [Figure 5-9](#)) only reflects the core clock requirement; i.e. independent of the host interface clock enable (EC\_HOST\_BUS\_CLK\_EN in [Figure 5-9](#)). The MEC1618/MEC1618i [Generic Block Clocking Model](#) assumes that the block may not remain operational without the host interface clock enable which is not stopped unless the [20 MHz Oscillator](#) is disabled (see [Section 5.4.8, “Power Management Interface,”](#) on page 89). The “Clock Required Status” for each block can be seen in the [Clock Required Status Registers](#).
- Note 5-10** The internal enable bit (not shown in [Figure 5-9](#)) is accessible through the EC Host Interface shown in [Figure 5-9](#) and provides a reset to each block. Typically, as soon as the internal enable bit is de-asserted, the block may be immediately reset and held in the lowest power consumption state.
- Note 5-11** The external sleep enables are configured using the [Block Sleep Enables](#) as described in [“Block Sleep Enables,”](#) on page 97.
- Note 5-12** State transitions on the internal enable bit are undefined in the SLEEPING state and may produce undesirable results.

### 5.4.6.3 Behavior

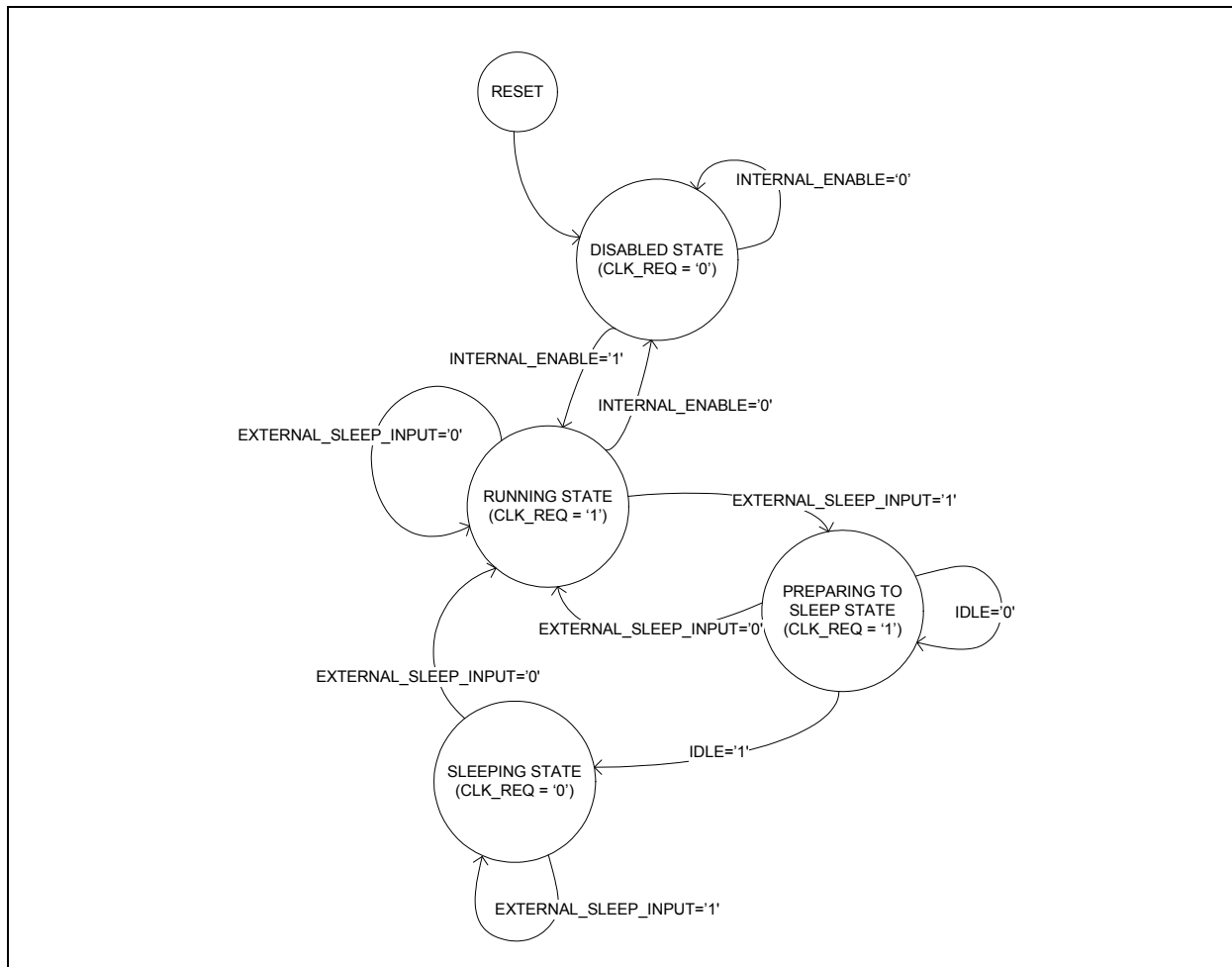
The effects of the [Power Management Interface](#) on the block sleep [Behavior](#) as a result of transitions on the sleep enable input are ignored when the block is disabled. Blocks are typically disabled following a power-on reset ([Figure 5-11, “Mode Transitions for Dynamic Power Management”](#)).

When a block is enabled, a sleep state request as a result of a transition on the sleep enable input must not adversely affect block operation; e.g., by performing illegal operations on an external circuit or corrupting transaction data. As a result, there may be transition latency from the running state to the sleeping state, depending on the nature of the block and its operational state as defined by an internal block ‘idle’ indicator when the sleep enable is asserted. It is possible that a block may never enter the sleeping state if the block does not idle before the sleep enable input is de-asserted as a result of a wake event.

After a block enters the sleeping state, internal clocks are gated ‘off’ and the block is inactive; i.e., outputs are static and the block cannot respond to transitions on inputs, except as defined by the [Wake Interface](#). The transition from the sleeping state to the running state can only occur after the system clocks are running and the sleep enable input is de-asserted.

As described in [Section 5.4.6.2, “Components,”](#) on page 84, transitions from the running state or the preparing to sleep state to the disabled state may occur without latency depending on the implementation (not shown in [Figure 5-11](#)). Transitions from the sleeping state to the disabled state are undefined when the [20 MHz Oscillator](#) is stopped. Transitions from the sleeping state to the disabled state can occur without latency when the [20 MHz Oscillator](#) is running (also not shown in [Figure 5-11](#)).

**FIGURE 5-10:** Generic Block Clocking Model **CLOCK GATING STATE DIAGRAM EXAMPLE**



## 5.4.7 MASTER CLOCK TREES

### 5.4.7.1 Description

The master clock derived from the [20 MHz Oscillator](#) is branched into six internal clock trees in the MEC1618/MEC1618i. These [Master Clock Trees](#) are controlled by register bits in the [Clock Tree Control 0 Register](#) and [Clock Tree Control 1 Register](#) and function as defined in [Section 5.4.7.4, "Control Bit Encoding," on page 89](#) and [Section 5.4.7.2, "Dynamic Clock Tree Gating," on page 86](#). The [Block Allocation Per Clock Tree](#) is defined in [Section 5.4.7.3](#).

Each master clock tree can be forced 'on,' forced 'off,' or dynamically controlled by hardware to help minimize power consumption throughout the [EC Controlled Dynamic Power States](#). See also [Section 5.4.8.3, "Clock Tree Gating in Heavy Sleep," on page 93](#).

### 5.4.7.2 Dynamic Clock Tree Gating

#### 5.4.7.2.1 Overview

[Dynamic Clock Tree Gating](#) specifies that a master clock tree is 'on' if any one of the blocks allocated to that tree requires a clock, or if a bus master is accessing registers in that tree. Clock trees are 'off' when the bits in the [Clock Required Status Registers](#) that are associated with the blocks in a tree are not asserted and a bus master is not accessing registers in the tree.

[Dynamic Clock Tree Gating](#) is enabled as described in [Table 5-12, "Clock Tree Force Control Bits Encoding,"](#) on page 89 and is enabled by default for the [Master Clock Trees](#) (see also [Note 5-14](#)).

The [EC Interrupt Aggregator](#) and the [GPIO Interface](#) are special cases as described in [Section 5.4.7.2.2, "INTERRUPT AGGREGATOR CLOCK TREE"](#) and [Section 5.4.7.2.3, "GPIO CLOCK TREE"](#).

## 5.4.7.2.2 INTERRUPT AGGREGATOR CLOCK TREE

The [EC Interrupt Aggregator](#) does not conform to the [Generic Block Clocking Model](#), but instead dynamically gates its master clock 'off' when all source interrupts and wake-up events are *not* asserted. For the [INTERRUPT AGGREGATOR CLOCK TREE](#) to remain 'off,' all active interrupts must be cleared at the source.

For example, the [PCR\\_INT](#) interrupt is asserted following a [VBAT\\_POR](#) and must be cleared to enable [Dynamic Clock Tree Gating](#) in the [INTERRUPT AGGREGATOR CLOCK TREE](#).

Additionally, when an interrupt from the [GPIO Interface](#) is configured as level-sensitive, this interrupt is continuously asserted when the voltage at the pin changes to the specified level, which also inhibits [Dynamic Clock Tree Gating](#) in this interface. In this instance, [Dynamic Clock Tree Gating](#) can only be achieved by setting the [GPIO Interface](#) interrupt(s) to edge detect mode.

As shown in [Table 5-11](#), the [EC Interrupt Aggregator](#) is part of [EC Clock](#). [Dynamic Clock Tree Gating](#) in [EC Clock](#) cannot occur until the [INTERRUPT AGGREGATOR CLOCK TREE](#) is gated 'off.'

**Note:** The need for clocking within the [EC Interrupt Aggregator](#) is not considered a requirement for keeping the [20 MHz Oscillator](#) running. As a result, the [EC Interrupt Aggregator](#) does not include a clock required output to the [Clock Generator](#). If an interrupt is asserted and enabled, the EC cannot sleep, which by default keeps the [20 MHz Oscillator](#) running. If an interrupt is asserted, but not enabled, the EC may sleep when needed and the [20 MHz Oscillator](#) will stop if no other block requires [MCLK](#).

## 5.4.7.2.3 GPIO CLOCK TREE

The [GPIO Interface](#) does not conform to the [Generic Block Clocking Model](#), but instead dynamically gates its master clock 'off' when all [GPIO Interface](#) interrupts and wake-up events are *not* asserted. To enable [Dynamic Clock Tree Gating](#) for the [GPIO CLOCK TREE](#), follow the procedure described for the [INTERRUPT AGGREGATOR CLOCK TREE](#) to set the GPIO interrupts to edge detect mode.

As shown in [Table 5-11](#), the [GPIO Interface](#) is part of [EC Clock](#). [Dynamic Clock Tree Gating](#) in [EC Clock](#) cannot occur until the [GPIO CLOCK TREE](#) is gated 'off.'

## 5.4.7.3 Block Allocation Per Clock Tree

The [Block Allocation Per Clock Tree](#) is defined in [Table 5-11](#).

Note that clocking for the [ARC 625D Embedded Controller](#) is also dynamically controlled as defined in [Section 5.4.8, "Power Management Interface,"](#) on page 89, but is not considered one of the [Master Clock Trees](#) as described here; however, [EC Clock](#) remains 'on' whenever the [ARC 625D Embedded Controller](#) clock is running.

**TABLE 5-11: MASTER CLOCK TREE BLOCK ALLOCATION**

Clock Tree					
EC Clock Tree 0	EC Clock Tree 1	EC Clock Tree 2	EC Clock Tree 3	Host Clock Tree 0	Host Clock Tree 1
<a href="#">Blinking/Breathing PWM</a> (LEDs)	<a href="#">RC Identification Detection</a> (RC_ID)	<a href="#">PWM Controller</a> 0 - 3	<a href="#">PWM Controller</a> 4 - 15	<a href="#">Logical Device Configuration</a> Global Register Bank	<a href="#">ACPI Embedded Controller Interface</a> (ACPI-ECI) 2 - 3
<a href="#">GPIO Interface</a>	<a href="#">BC-Link Master</a> A & B	<a href="#">TACH Monitor</a> 0 - 1	<a href="#">TACH Monitor</a> 2 - 5	<a href="#">Embedded Flash Subsystem</a> Host Controller	<a href="#">MailBox Register Interface</a>

# MEC1618/MEC1618i

**TABLE 5-11: MASTER CLOCK TREE BLOCK ALLOCATION (CONTINUED)**

Clock Tree					
EC Clock Tree 0	EC Clock Tree 1	EC Clock Tree 2	EC Clock Tree 3	Host Clock Tree 0	Host Clock Tree 1
Power, Clocks, and Resets Register Bank	VBAT Register Bank	SMB Device Interface 0 - 1	SMB Device Interface 2	Host Interface (LPC)	Two Pin Serial Port (UART)
Hibernation Timer Register Bank	EC Interrupt Aggregator	PS/2 Device Interface 0 - 1	PS/2 Device Interface 2	ACPI Embedded Controller Interface (ACPI-ECI) 0 - 1	Host General Purpose Serial Peripheral Interface (GP-SPI)
16-Bit Timer Interface Timer 0	ARC 625D Embedded Controller	16-Bit Timer Interface Timer 1 - 3	Input Capture and Compare Timer	8042 Emulated Keyboard Controller	—
VBAT Powered RAM	SRAM, Flash, EEPROM	Analog to Digital Converter	PECI Interface	Embedded Memory Interface	—
Week Alarm Interface	JTAG Master	—	Keyboard Matrix Scan Support	ACPI PM1 Block Interface	—
VBAT-Powered Control Interface	—	DMA Controller	EC General Purpose Serial Peripheral Interface (GP-SPI)	—	—
—	—	—	HDMI-CEC Interface Controller	—	—
—	—	—	BC-Link Master D	—	—
—	—	—	Watchdog Timer Interface Register Bank	—	—
—	—	—	Serial Debug Port	—	—
—	—	—	JTAG and Boundary Scan	—	—

**Note 5-13** EC Clock remains 'on' whenever the ARC 625D Embedded Controller clock is running. None of the blocks allocated to EC Clock have "Clock Required" Power Management Interface outputs. EC Clock remains 'off' whenever the ARC 625D Embedded Controller is sleeping.



## 5.4.7.4 Control Bit Encoding

**TABLE 5-12: CLOCK TREE FORCE CONTROL BITS ENCODING**

Force 'OFF' Bit	Force 'ON' Bit	Description
0	0	<a href="#">Dynamic Clock Tree Gating</a> enabled (Default) ( <a href="#">Note 5-14</a> ).
0	1	Clock Tree Forced 'On'
1	0	Clock Tree Forced 'Off'
1	1	

**Note 5-14** To shut down [EC Clock](#), [Host Clock Tree 0](#) and [Host Clock Tree 1](#) drive the [VCC\\_PWRGD](#) input pin to ground while the [VCC\\_PWRGD](#) signal function is selected (the default setting for this pin).

## 5.4.8 POWER MANAGEMENT INTERFACE

### 5.4.8.1 Overview

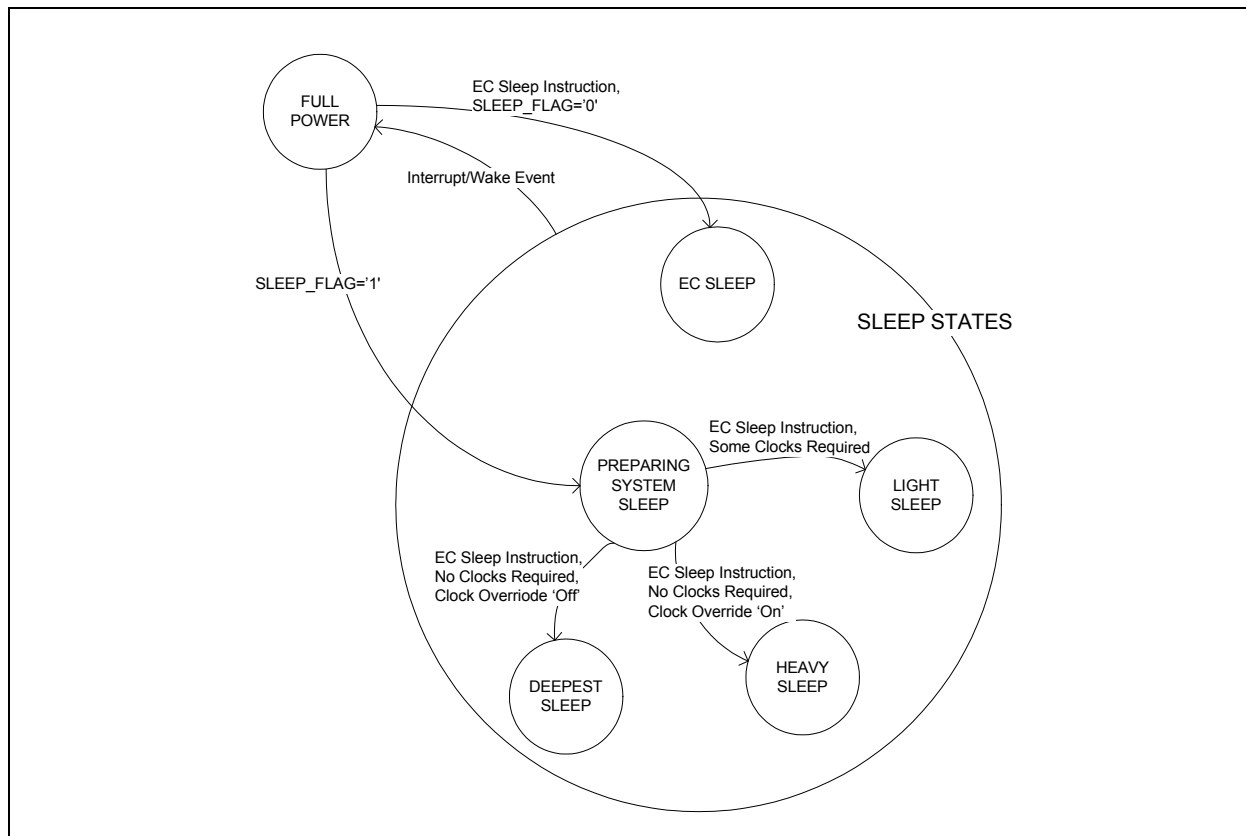
The MEC1618/MEC1618i includes several features to help minimize power consumption, the most intrinsic of which is the application of advanced gate-level low-power design techniques. The EC can also establish the upper run-time power consumption limit by asserting individual internal block enables (i.e., as described in the [Generic Block Clocking Model](#)) only for functions that are absolutely required during normal operation. Finally, the EC can also dynamically minimize power consumption by modulating clocks at the block level and within the [Clock Generator](#) using the clock gating feature of the [Power Management Interface](#).

**APPLICATION NOTE:** Implementing dynamic power management using individual internal block enables alone is complicated by the fact that, depending on the block implementation, block enables may also perform a reset function.

There are six [EC Controlled Dynamic Power States](#) as described in [Section 5.4.8.2, "EC Controlled Dynamic Power States"](#). These states are achieved using clock gating as described in [Section 5.4.8.5, "Clock Gating," on page 94](#), which can also affect the [20 MHz Oscillator](#).

Running and sleeping are the two basic operational modes when considering dynamic power management as defined by this interface ([Figure 5-11](#)). Transitions between these modes are deliberate and persistent. For example, to exit the full power state the EC must issue a sleep command as described in ["EC Controlled Sleep State Activation," on page 96](#). Exiting sleep states requires the [Wake Interface](#).

**FIGURE 5-11: MODE TRANSITIONS FOR DYNAMIC POWER MANAGEMENT**



## 5.4.8.2 EC Controlled Dynamic Power States

Table 5-13 illustrates the [EC Controlled Dynamic Power States](#) that can be achieved using the clock gating feature of [Power Management Interface](#). The [EC Controlled Dynamic Power States](#) closely mirror the system power states defined by the [Generic Block Clocking Model](#), but 1) redefine the “preparing to sleep” state to include the affects of the EC sleep state, 2) define additional implementation-specific sleep states that affect [Wake Interface](#) latency and 3) indicate the aggregated response of all the MEC1618/MEC1618i power-managed blocks. Typically, the higher the state number in Table 5-13, the greater the system power savings. Traversing the [EC Controlled Dynamic Power States](#) requires the [SLEEP\\_FLAG](#), [ARC\\_CLK\\_DISABLE](#) signal, [WAKE](#) signal, the [SLEEP CONFIG](#) field, the [Block Sleep Enable Registers](#) and the [Clock Required Status Registers](#).

See [Section 44.4, “Power Consumption,” on page 920](#) or characterization current values.

TABLE 5-13: EC CONTROLLED DYNAMIC POWER STATES

Device Power State	SLEEP_FLAG (Note 5-15)	ARC_CLK_DISABLE (Note 5-16)	Global Block Clock Status (Note 5-17)	SLEEP_CONFIG (Note 5-18)	State Name	Description
0.	0	0	X	X	FULL POWER	The system is running. This is the highest power consumption state.
1.	0	1	X	X	EC SLEEP	EC has executed a sleep instruction. The rest of the system is unaffected by the EC SLEEP state.
2.	1	0	X	X	PREPARING SYSTEM SLEEP	Sleep commands issued to all sleep-enabled blocks (see "EC Controlled Sleep State Activation," on page 96). The EC can return to the FULL POWER state from PREPARING SYSTEM SLEEP by de-asserting ('0') SLEEP_FLAG (see also Note 5-19).

# MEC1618/MEC1618i

**TABLE 5-13: EC CONTROLLED DYNAMIC POWER STATES (CONTINUED)**

Device Power State	SLEEP_FLAG (Note 5-15)	ARC_CLK_DISABLE (Note 5-16)	Global Block Clock Status (Note 5-17)	SLEEP CONFIG (Note 5-18)	State Name	Description
3.	1	1	Clock Required	X	SYSTEM LIGHT SLEEP	System is in a sleeping mode but the <a href="#">20 MHz Oscillator</a> must remain operating because one or more blocks require a clock.
4.			Clock Not Required	0	SYSTEM HEAVY SLEEP 1	System is in a sleeping mode and no blocks require the clock. There is no clock start up latency following a wake event in <a href="#">SYSTEM HEAVY SLEEP 1</a> (see <a href="#">Table 5-41</a> , “ <a href="#">SLEEP CONFIG Bit Encoding</a> ,” on <a href="#">page 117</a> ).
5.				1	SYSTEM HEAVY SLEEP 2	System is in a sleeping mode and no blocks require the clock. There is minimal clock start up latency following a wake event in <a href="#">SYSTEM HEAVY SLEEP 2</a> (see <a href="#">Table 5-41</a> , “ <a href="#">SLEEP CONFIG Bit Encoding</a> ,” on <a href="#">page 117</a> ).
6.				2	SYSTEM HEAVY SLEEP 3	System is in a sleeping mode and no blocks require the clock. There is more clock start up latency following a wake event in <a href="#">SYSTEM HEAVY SLEEP 3</a> than from <a href="#">SYSTEM HEAVY SLEEP 2</a> (see <a href="#">Table 5-41</a> , “ <a href="#">SLEEP CONFIG Bit Encoding</a> ,” on <a href="#">page 117</a> ).
7.				3	SYSTEM DEEPEST SLEEP	System is in a sleeping mode and no blocks require the clock. Clock start up latency is the longest and power consumption the lowest in <a href="#">SYSTEM DEEPEST SLEEP</a> (see <a href="#">Table 5-41</a> , “ <a href="#">SLEEP CONFIG Bit Encoding</a> ,” on <a href="#">page 117</a> ).

**Note 5-15** [SLEEP\\_FLAG](#) is bit D1 in the [Clock Control Register](#).

**Note 5-16** The [ARC\\_CLK\\_DISABLE](#) signal is described in [Table 5-4](#), “[Power, Clocks, and Resets Port List](#),” on [page 75](#).

**Note 5-17** Includes the sum of all the “Core Clock Required Status” outputs defined in the [Generic Block Clocking Model](#) and aggregated in the [Clock Required Status Registers](#).

**Note 5-18** This column refers to the [SLEEP CONFIG](#) field in the [Clock Control Register](#).

**Note 5-19** In the [PREPARING SYSTEM SLEEP](#) state when the sleep enable to a block is asserted, EC register accesses to that block are undefined and should be avoided.

## 5.4.8.3 Clock Tree Gating in Heavy Sleep

The output of the **20 MHz Oscillator** is gated 'off' in some of the System Heavy Sleep states as defined in [Table 5-14](#) to conserve power beyond gating the **Master Clock Trees** as described in [Section 5.4.7](#).

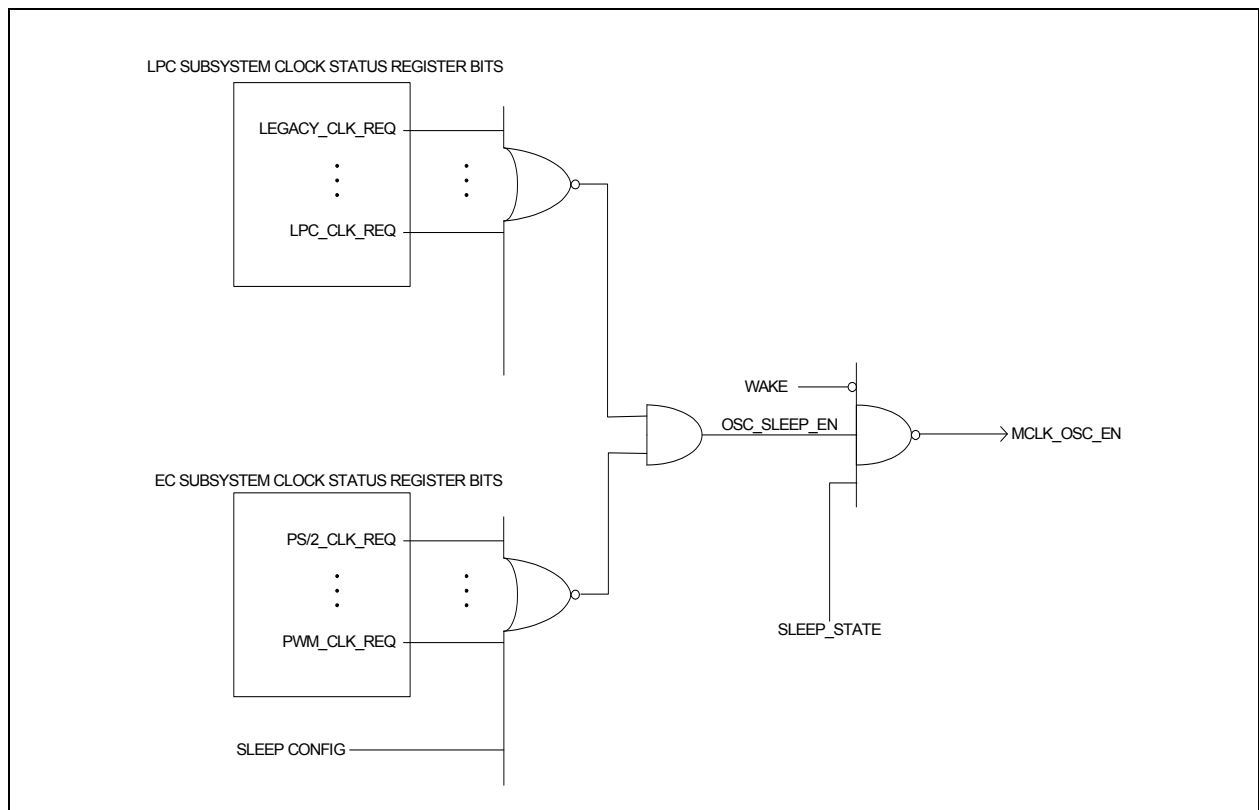
**TABLE 5-14: MEC1618/MEC1618i Clock Tree Gating in Heavy Sleep IN HEAVY SLEEP STATE**

Power State	Clock Required Status Registers	SLEEP CONFIG (Note 5-18)	Description
SYSTEM HEAVY SLEEP 2	No blocks require a clock.	1	Sleep commands issued to all blocks, the EC is sleeping and no blocks require a clock, but the <b>20 MHz Oscillator</b> remains running to limit start-up latency.
SYSTEM HEAVY SLEEP 3		2	

## 5.4.8.4 20 MHz Oscillator Control

The **20 MHz Oscillator** can only be controlled by the EC using the **Power Management Interface**, as illustrated in [FIGURE 5-12: 20 MHz Ring Oscillator Controls on page 93](#) and described in [Table 5-15, "MCLK\\_OSC\\_EN Control"](#).

**FIGURE 5-12: 20 MHZ RING OSCILLATOR CONTROLS**



**TABLE 5-15: MCLK\_OSC\_EN CONTROL**

OSC_SLEEP_Enable (Note 5-20)	SLEEP_STATE (Note 5-21)	WAKE (Note 5-22)	MCLK_OSC_EN (Note 5-23)	Description
0	X	X	1	The 20 MHz Oscillator is always enabled ("on") when at least one block requires the clock or the EC has overridden the 20 MHz Oscillator sleep state using the SLEEP CONFIG field in the Clock Control Register.
X	0	X		The 20 MHz Oscillator is always enabled ("on") when SLEEP_STATE is not asserted.
X	X	1		The 20 MHz Oscillator is always enabled ("on") when WAKE is asserted.
1	1	0	0	20 MHz Oscillator is disabled ("off") when SLEEP_STATE is asserted, all blocks do not require the 20 MHz Oscillator, the EC has not overridden the 20 MHz Oscillator sleep state using the SLEEP CONFIG field and WAKE is not asserted.

**Note 5-20** OSC\_SLEEP\_ENABLE is illustrated in Figure 5-12 and indicates the status of all the "Core Clock Required Status" outputs defined in the Generic Block Clocking Model and aggregated in the Clock Required Status Registers and the SLEEP CONFIG field. For a description of the SLEEP CONFIG signaling, see the SLEEP CONFIG field in Section 5.7.3, "Clock Control Register," on page 115.

**Note 5-21** See "EC Power State Controls," on page 94 and Table 5-4 for a description of the SLEEP\_STATE signal.

**Note 5-22** See Section 5.4.8.6, "Wake Interface," on page 98 and Table 5-4 for a description of the WAKE signal.

**Note 5-23** MCLK\_OSC\_EN is the 20 MHz Oscillator enable control illustrated in Figure 5-12.

## 5.4.8.5 Clock Gating

### 5.4.8.5.1 Overview

Power savings using the Power Management Interface comes from Clock Gating as described above and in the sub-sections that follow. The magnitude of the power savings depends on the configuration of the Block Sleep Enables, the SLEEP CONFIG field in the Clock Control Register, the configuration of the Master Clock Trees and the operational status of the individual blocks as defined by the Clock Required Status Registers.

### 5.4.8.5.2 EC Power State Controls

Assuming WAKE is not asserted, when the SLEEP\_FLAG bit in the Clock Control Register is asserted by the EC (see also "EC Controlled Sleep State Activation," on page 96), a sleep command is sent to all blocks that are configured for sleep as defined by the Block Sleep Enables. Clock Gating occurs at the block level following the sleep command depending on the state of the block clocking requirement (see Section 5.7.5, "Clock Required Status Registers," on page 124).

After the SLEEP\_FLAG bit is asserted, Clock Gating can only occur within the Clock Generator when the ARC\_CLK\_DISABLE signal (see Table 5-4, "Power, Clocks, and Resets Port List," on page 75) is asserted, at which time the SLEEP\_STATE signal that can affect 20 MHz Oscillator Control is asserted (Table 5-16). Timing for the EC Power State Controls is defined in Figure 6.7 and Table 6.11.

If WAKE is asserted when EC Controlled Sleep State Activation is attempted, the 20 MHz Oscillator will remain enabled, the SLEEP\_FLAG will be automatically de-asserted by hardware as described in Section 5.4.8.6, "Wake Interface," on page 98, and an EC interrupt will occur (not shown in Table 5-16).

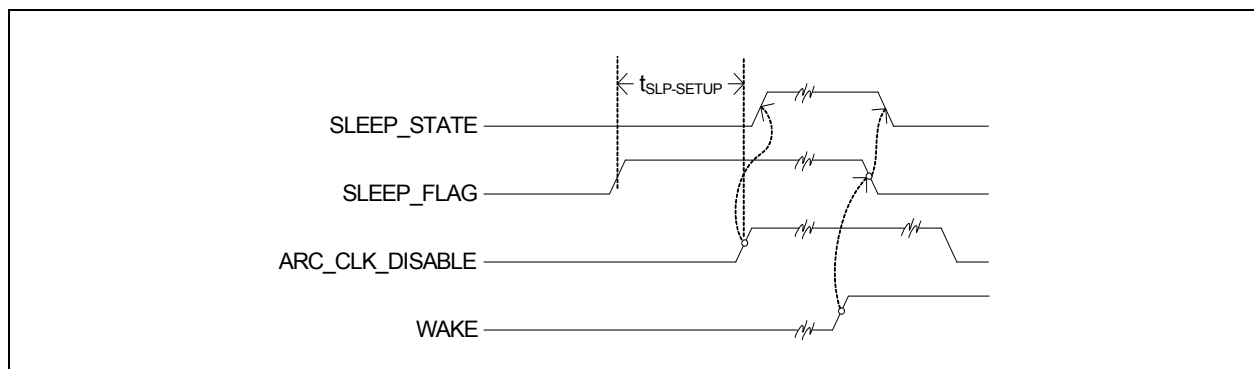
**TABLE 5-16: EC Power State Controls DESCRIPTION**

ARC_CLK_DISABLE (Note 5-16)	SLEEP_FLAG (Note 5-15)	SLEEP_STATE (Note 5-25)	States (Note 5-24)	Description
0	0	0	FULL POWER	–
0	1	0	PREPARING SYSTEM SLEEP	<b>SLEEP_FLAG</b> has been asserted by the EC which asserts sleep enables to blocks that have been enabled for sleeping; i.e., clocks in these blocks are turning 'off' in this state. The EC and the 20 MHz Oscillator clock remain active. The <b>PREPARING SYSTEM SLEEP</b> state may represent a lower than <b>FULL POWER</b> system power consumption state.
1	0	0	EC SLEEP	The <b>Clock Generator</b> is unaffected in the <b>EC SLEEP</b> state. The <b>EC SLEEP</b> state may or may not represent a lower system power state than the <b>PREPARING SYSTEM SLEEP</b> state depending on the <b>Block Sleep Enables</b> and the system operational state.
1	1	1	SYSTEM LIGHT SLEEP, SYSTEM HEAVY SLEEP 1, SYSTEM HEAVY SLEEP 2, SYSTEM HEAVY SLEEP 3, SYSTEM DEEPEST SLEEP	The 20 MHz Oscillator may be stopped as described in <a href="#">Section 5.4.8.4, "20 MHz Oscillator Control,"</a> on page 93 and in <a href="#">Table 6.7</a> .

**Note 5-24** See the system state definitions in [Section 5.4.8.2, "EC Controlled Dynamic Power States,"](#) on page 90.

**Note 5-25** **SLEEP\_STATE** affects the 20 MHz Oscillator as described in [Section 5.4.8.4, "20 MHz Oscillator Control,"](#) on page 93 and appears in [Table 5-4, "Power, Clocks, and Resets Port List,"](#) on page 75.

**FIGURE 5-13: EC Power State Controls TIMING**



**TABLE 5-17: EC Power State Controls TIMING PARAMETERS**

Parameters	Symbol	Min	Typ	Max	Units
System Sleep Setup Time	$t_{\text{SLP-SETUP}}$	1	–	–	EC Clock

## 5.4.8.5.3 EC Controlled Sleep State Activation

### OVERVIEW

EC Controlled Sleep State Activation depends upon the assertion of the [ARC\\_CLK\\_DISABLE](#) signal that occurs when the EC executes a sleep instruction. The dynamic sleep states that can be achieved through [EC Controlled Sleep State Activation](#) in part depend on the state of the [SLEEP\\_FLAG](#) in the [Clock Control Register](#) as defined in the sections below, “[Entering and Exiting EC Sleep State](#)” and “[Entering and Exiting System Sleep States](#).” Simultaneous assertions of the [WAKE](#) input and [EC Controlled Sleep State Activation](#) immediately terminate system sleeps states as defined in [Section 5.4.8.6, “Wake Interface,” on page 98](#).

### ENTERING AND EXITING EC SLEEP STATE

As illustrated in [Table 5-13, “EC Controlled Dynamic Power States,” on page 91](#), transitions to the [EC SLEEP](#) state occur when the [ARC\\_CLK\\_DISABLE](#) signal is asserted (i.e., the EC enters the sleep state) while the [SLEEP\\_FLAG](#) in the [Clock Control Register](#) is not asserted (‘0’). The [EC SLEEP](#) state is terminated when an interrupt to the EC occurs, as described in [Section 5.4.8.6, “Wake Interface,” on page 98](#).

The EC enters the sleep state when the [SLEEP](#) instruction is issued. The [SLEEP](#) instruction halts the CPU pipeline and gates the processor clocks. The clocks to the ARC interrupt logic are not gated. As long as interrupts are enabled in the ARC core STATUS register and at least one interrupt is enabled in the Interrupt Accelerator when the [SLEEP](#) instruction is issued, the processor will wake up and process the interrupt service routine when the interrupt triggers. On return from interrupt, the processor will execute the instruction immediately following the [SLEEP](#) instruction.

“If it is necessary to enable a wakeup interrupt just before entering the sleep state, some care is required to ensure that the interrupt does not fire before the [SLEEP](#) instruction is issued. The [ARC FLAG](#) instruction should be used to enable and disable interrupts and the [FLAG](#) and [SLEEP](#) instructions should be contiguous to insure that the [SLEEP](#) is in the processor pipeline when interrupts are enabled and therefore executes before an interrupt can fire.”

For more information on the [FLAG](#) and [SLEEP](#) instructions, see the *ARCompact™ ISA Instruction Set Architecture Programmer's Reference*.

### ENTERING AND EXITING SYSTEM SLEEP STATES

As illustrated in [Table 5-13, “EC Controlled Dynamic Power States,” on page 91](#), transitions to the system sleep states (sleep states other than [EC SLEEP](#)) occur when the [ARC\\_CLK\\_DISABLE](#) signal is asserted (i.e., the EC enters the sleep state) while the [SLEEP\\_FLAG](#) in the [Clock Control Register](#) is asserted (‘1’). These states include [SYSTEM LIGHT SLEEP](#), [SYSTEM HEAVY SLEEP 1](#) and [SYSTEM DEEPEST SLEEP](#).

The system sleep states are terminated when a wake event occurs as described in [Section 5.4.8.6, “Wake Interface,” on page 98](#). [Entering and Exiting System Sleep States](#) is similar to [Entering and Exiting EC Sleep State](#). Refer to the section [Entering and Exiting EC Sleep State](#) for information about sleeping the EC.

Note that as described in [Section 5.4.8.2, “EC Controlled Dynamic Power States,” on page 90](#), the [PREPARING SYSTEM SLEEP](#) state in [Table 5-13](#) occurs as soon as the [SLEEP\\_FLAG](#) is asserted, independent of the state of [ARC\\_CLK\\_DISABLE](#). The EC can optionally interrogate the [Clock Required Status Registers](#) to estimate the depth of the sleep state, for example, before executing a sleep instruction. The EC can return to the [FULL POWER](#) state from the [PREPARING SYSTEM SLEEP](#) state at any time before executing a sleep instruction by de-asserting the [SLEEP\\_FLAG](#).

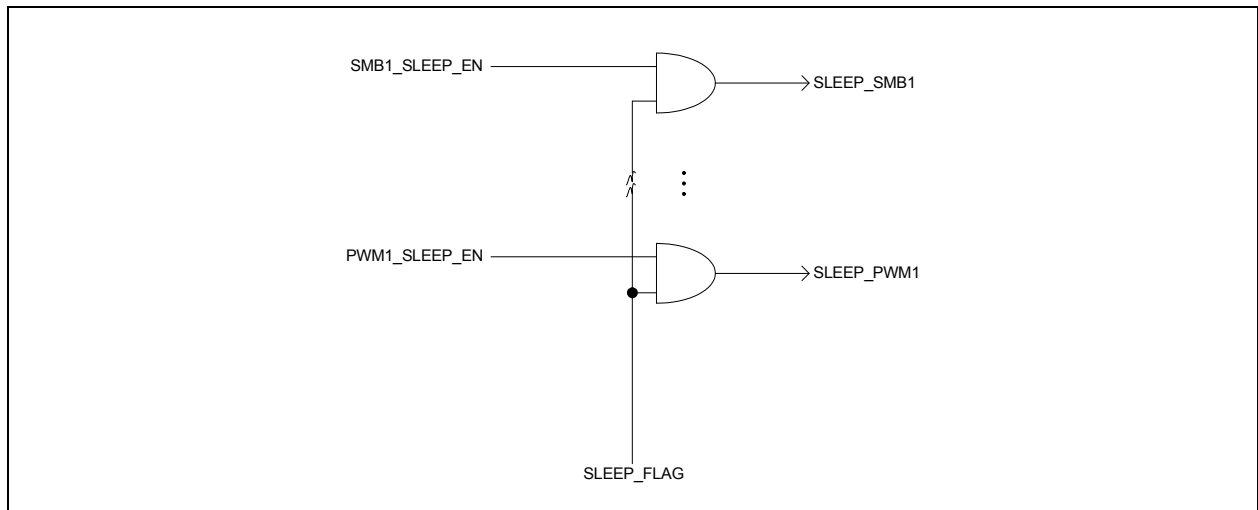
When the ring oscillator is shut down, the oscillator can only be restarted by the [Wake Interface](#) or by power cycling the system. In order for the [Wake Interface](#) to operate, at least one wake capable interrupt must be enabled in the Interrupt Aggregator. This class of interrupts is described in [Section 15.3.5, “Wake Capable Interrupts,” on page 304](#).



## 5.4.8.5.4 Block Sleep Enables

The [Block Sleep Enables](#) allow the EC firmware to determine which blocks will receive sleep commands as a result of [EC Controlled Sleep State Activation](#) (see "Entering and Exiting System Sleep States," on page 96). The Block Sleep Enables are configured using the [Block Sleep Enable Registers](#) (see Section 5.7.4, "Block Sleep Enable Registers," on page 118) and behave as illustrated in [Figure 5-14, "Block Sleep Enables Example"](#) and [Table 5-18, "Block Sleep Enables Definition"](#). There are three [Block Sleep Enable Registers](#): two [EC Blocks Sleep Enables Registers](#) (see Section 5.7.4.2 on page 120) and one [LPC Blocks Sleep Enables Register](#) (Section 5.7.4.1 on page 118).

**FIGURE 5-14:** Block Sleep Enables EXAMPLE



**TABLE 5-18:** Block Sleep Enables DEFINITION

Sleep Enable ( <a href="#">Note 5-26</a> )	SLEEP_FLAG ( <a href="#">Note 5-15</a> )	Block SLEEP_EN Signal ( <a href="#">Note 5-27</a> )	Description
0	0	0	Block not enabled for sleep, system is in <a href="#">FULL POWER</a> or <a href="#">EC SLEEP</a> state.
	1		Block not enabled for sleep, system is in <a href="#">PREPARING SYSTEM SLEEP</a> or <a href="#">SYSTEM LIGHT SLEEP</a> state.
1	0	1	Block enabled for sleep, system is in <a href="#">FULL POWER</a> or <a href="#">EC SLEEP</a> state.
	1		Block enabled for sleep, system is in <a href="#">PREPARING SYSTEM SLEEP</a> , <a href="#">SYSTEM LIGHT SLEEP</a> , <a href="#">SYSTEM HEAVY SLEEP 1</a> , <a href="#">SYSTEM HEAVY SLEEP 2</a> , <a href="#">SYSTEM HEAVY SLEEP 3</a> or <a href="#">SYSTEM DEEPEST SLEEP</a> state.

**Note 5-26** Sleep enable for a single block as defined in [Section 5.7.4, "Block Sleep Enable Registers,"](#) on page 118.

**Note 5-27** [Clock Generator](#) sleep enable output signal to a single block (SLEEP\_EN) as defined in the [Generic Block Clocking Model](#).

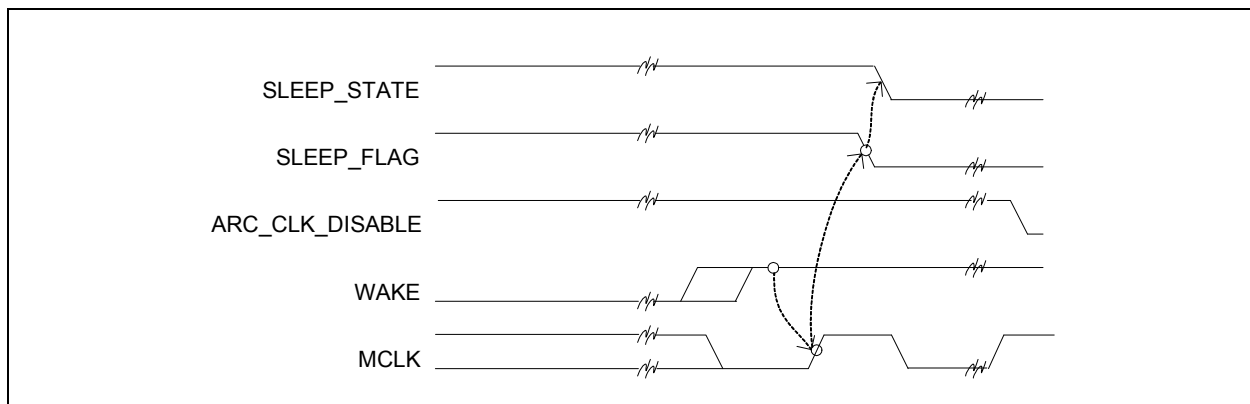
## 5.4.8.6 Wake Interface

The [Wake Interface](#) terminates the sleep states illustrated in [Table 5-13, "EC Controlled Dynamic Power States," on page 911](#) and includes interrupts to the EC to transition from the [EC SLEEP](#) state, as well as wake events that can restart the [20 MHz Oscillator](#) and terminate the [SYSTEM LIGHT SLEEP](#), [SYSTEM HEAVY SLEEP 1](#), [SYSTEM HEAVY SLEEP 2](#), [SYSTEM HEAVY SLEEP 3](#) and [SYSTEM DEEPEST SLEEP](#) states.

The [WAKE](#) signal shown in [Table 5-4, "Power, Clocks, and Resets Port List," on page 75](#) is the only required input to the [Clock Generator](#) for the [Wake Interface](#). [Wake Capable Interrupts](#), although technically part of the [Wake Interface](#), are external to the [Power, Clocks, and Resets](#) function and are not illustrated in this definition. The minimum pulse-width for wake events is characterized in [Table 21-7, "GPIO Interrupt/Wake Event Timing Parameters," on page 404](#).

When the [WAKE](#) signal is asserted while a dynamic system sleep state is enabled, the [SLEEP\\_FLAG](#) in the [Clock Control Register](#) is automatically de-asserted by hardware within the first one or two clocks that occur when the [20 MHz Oscillator](#) is re-enabled as illustrated in [Figure 5-15](#). As described in ["EC Power State Controls," on page 94](#) and in ["EC Controlled Sleep State Activation," on page 96](#), system sleep states are essentially ignored if the [SLEEP\\_STATE](#) signal and the [WAKE](#) signal are simultaneously asserted.

**FIGURE 5-15:** [Wake Interface TIMING](#)



### 5.4.8.6.1 Treatment Of Non-wake Capable Interrupts During Sleep Transitions

There are two categories of Interrupts which effect the [Wake Interface](#) differently: [Wake Capable Interrupts](#) and [Non-Wake Capable Interrupts](#). See [Section 15.3.5, "Wake Capable Interrupts," on page 304](#) in [Section 15.0, "EC Interrupt Aggregator"](#).

The lowest power states in which [Non-Wake Capable Interrupts](#) can be asserted are [PREPARING SYSTEM SLEEP](#) or [SYSTEM LIGHT SLEEP](#). This is because the source of these interrupts require clocks to generate the interrupts. In the event that the [EC Controlled Dynamic Power States](#) enters the [PREPARING SYSTEM SLEEP](#) or [SYSTEM LIGHT SLEEP](#) state by the ARC executing a sleep instruction while a least one non-wake interrupt associated block is active, the [EC Controlled Dynamic Power States](#) will not enter a lower state until a non-wake interrupt occurs. The non-wakeable interrupt changes the ARC from a halt state to a running state to process the interrupt. Firmware needs to toggle the state of the [SLEEP\\_FLAG](#) bit in the [Clock Control Register](#). If these bit were Wake-up events, the firmware would not need to de-assert the [SLEEP\\_FLAG](#) bit since hardware will have done so already.

**APPLICATION NOTE:** If an aggressive sleep policy is implemented and firmware attempts to enter sleep when at least one non-wake interrupt associated block is active, then firmware can treat all interrupt identically (this includes both [Wake Capable Interrupts](#) or [Non-Wake Capable Interrupts](#)). Firmware can service both [Wake Capable Interrupts](#) or [Non-Wake Capable Interrupts](#) by attempting to toggle the [SLEEP\\_FLAG](#) in the [Clock Control Register](#) first '0' then '1' to prepare to sleep.

**APPLICATION NOTE:** if a less aggressive sleep policy is implemented and firmware never attempts to enter sleep when any non-wake interrupt associated block is active, then firmware can treat all [Wake Capable Interrupts](#) identically. Firmware can service all [Wake Capable Interrupts](#) by

attempting setting the [SLEEP\\_FLAG](#) bit to '1' in the [Clock Control Register](#) to prepare to sleep.

## 5.4.9 MCLK SOURCED CLOCKING

### 5.4.9.1 Overview

[MCLK Sourced Clocking](#) includes all of the [Fixed Clock Domain](#), [Host Clock Domain](#) and [Programmable Clock Domains](#) that are derived from the [20 MHz Oscillator](#). [Ring Oscillator Sourced Clocking](#) remains active as long as the [20 MHz Oscillator](#) is running (see [Section 5.4.8.4, "20 MHz Oscillator Control,"](#) on page 93).

All [MCLK Sourced Clocking](#) and [32K Clock Domain](#) clocking is summarized in [Table 5-19](#).

**TABLE 5-19: ALL Clock Generator OUTPUT PORTS SUMMARY**

Symbol	Type (Clock or Enable)	Nominal Frequency	Reference
MCLK	CLOCK	20.27 MHz	<a href="#">Section 5.4.3, "20 MHz Oscillator,"</a> on page 79
EC_BUS_CLK_EN	ENABLE	Programmable	<a href="#">"EC Bus Clock,"</a> on page 99
LPC_BUS_CLK_EN	ENABLE	20.27 MHz	<a href="#">"LPC Bus Clock,"</a> on page 99
MCLK_DIV1_EN	ENABLE	20.27 MHz	<a href="#">Section 5.4.9.3, "Fixed Clock Domain,"</a> on page 100
MCLK_DIV2_EN	ENABLE	10.14 MHz	
MCLK_DIV4_EN	ENABLE	5.07 MHz	
MCLK_DIV8_EN	ENABLE	2.53 MHz	
MCLK_DIV10_EN	ENABLE	2.03 MHz	
MCLK_DIV16_EN	ENABLE	1.27 MHz	
MCLK_DIV20_EN	ENABLE	1.01 MHz	
MCLK_DIV32_EN	ENABLE	0.633 MHz	
MCLK_DIV64_EN	ENABLE	317 kHz	
MCLK_DIV128_EN	ENABLE	158 kHz	
MCLK_DIV203_EN	ENABLE	99.852 kHz	
MCLK_5HZ_EN	ENABLE	5 Hz	
MCLK_DIV20_EN_HST	ENABLE	1.01 MHz	<a href="#">Section 5.4.9.4, "Host Clock Domain,"</a> on page 100
X32K_CLK	CLOCK	32.768 kHz	<a href="#">Section 5.4.4, "32.768 kHz Oscillator,"</a> on page 80

### 5.4.9.2 Programmable Clock Domains

#### 5.4.9.2.1 EC Bus Clock

The [EC Bus Clock](#) ([EC\\_BUS\\_CLK\\_EN](#)) is a programmable clock that is derived from the [EC Clock Divider Register](#) as described in [Section 5.7.1](#) on page 113.

#### 5.4.9.2.2 LPC Bus Clock

The [LPC Bus Clock](#) ([LPC\\_BUS\\_CLK\\_EN](#)) is a fixed clock.

# MEC1618/MEC1618i

## 5.4.9.3 Fixed Clock Domain

The [Fixed Clock Domain](#) represents non-programmable clocks that are derived from the [20 MHz Oscillator](#). The [Fixed Clock Domain](#) outputs as shown in [Table 5-19](#) are [MCLK](#), [MCLK\\_DIV2\\_EN](#), [MCLK\\_DIV4\\_EN](#), [MCLK\\_DIV10\\_EN](#), and [MCLK\\_DIV203\\_EN](#).

## 5.4.9.4 Host Clock Domain

The [Host Clock Domain](#) includes clocking for the [Legacy Port Functions](#). [Host Clock Domain](#) clock gating is controlled by [VCC\\_PWRGD](#) such that when [VCC\\_PWRGD](#) is not asserted ('0'), clocks in the [Host Clock Domain](#) are 'off.'

When [VCC\\_PWRGD](#) is asserted ('1') clocks in the [Host Clock Domain](#) are 'on' and the [Legacy Port Functions](#) may be affected by the [EC Controlled Dynamic Power States](#) as described in [Section 5.4.8, "Power Management Interface," on page 89](#).

The [Host Clock Domain](#) output as shown in [Table 5-19](#) is [MCLK\\_DIV20\\_EN\\_HST](#).

## 5.4.10 32K CLOCK DOMAIN

### 5.4.10.1 Overview

The [32K Clock Domain](#) represents all of the clocking derived from the [32.768 kHz Oscillator](#). The output of the [32.768 kHz Oscillator](#) is synchronized to the [20 MHz Oscillator](#) as described below in [Section 5.4.10.2, "Synchronization"](#).

The [32K Clock Domain](#) remains active as long as the [32.768 kHz Oscillator](#) is running (see ["32K\\_EN," on page 134](#)). The blocks driven by the [32K Clock Domain](#) are summarized in [Table 5-20](#) and are not affected by [Clock Gating](#) in the [Power Management Interface](#). Typically, blocks driven by the [32K Clock Domain](#) can generate wake events, even when the [20 MHz Oscillator](#) is disabled.

### 5.4.10.2 Synchronization

The [32K Clock Domain X32K\\_CLK](#) output is synchronized to the [20 MHz Oscillator](#). Synchronization is disabled under two conditions: 1) when the [20 MHz Oscillator](#) is disabled (e.g., in the [SYSTEM DEEPEST SLEEP](#) state or during a test mode) and 2) when [VTRGD](#) is not asserted (see [Section 5.6.4, "VTRGD," on page 106](#)).

### 5.4.10.3 Summary

The distribution of the [32K Clock Domain](#) throughout the MEC1618/MEC1618i is illustrated in [Table 5-20](#), below.

**TABLE 5-20: 32K Clock Domain DRIVEN BLOCKS**

Block Name	Block Cross References
Week Timer	<a href="#">Section 20.0, "Week Alarm Interface," on page 393</a>
Watch-Dog Timer	<a href="#">Section 16.0, "Watchdog Timer Interface," on page 349</a>
Hibernation Timer 0	<a href="#">Section 19.0, "Hibernation Timer," on page 389</a>
Hibernation Timer 1	
LED Interface	<a href="#">Section 33.0, "Blinking/Breathing PWM," on page 528</a>
Watch-Dog Timer Forced Reset	<a href="#">Section 5.6.12, "Watch-Dog Timer Forced Reset," on page 110</a>
20 MHz Oscillator	<a href="#">Section 5.4.3, "20 MHz Oscillator," on page 79 (Note 5-28).</a>

**Note 5-28** [Synchronization](#) as described in [Section 5.4.10.2](#) does not apply to the output of the [32.768 kHz Oscillator](#) that is applied to the [20 MHz Oscillator](#).

## 5.5 Power Configuration

### 5.5.1 POWER SUPPLIES AND CLOCKS ACPI CONTEXT

The MEC1618/MEC1618i is influenced by three separate power planes, [VBAT](#), [VTR](#), and [VCC](#), as described in [Table 6.15](#). The [VBAT](#) and [VTR](#) power planes provide power directly to the MEC1618/MEC1618i through the [VBAT](#) and [VTR](#) pins shown in [Table 5-4, "Power, Clocks, and Resets Port List"](#). The MEC1618/MEC1618i senses the [VCC](#) power state using the [VCC\\_PWRGD](#) input pin. The [VBAT](#), [VTR](#), and [VCC](#) power sequencing requirements are as follows (see also [Section 5.5.3, "Power-Up Sequence," on page 103](#)):

1. [VCC](#) power can be applied simultaneously with or after [VTR](#) power.
2. [VTR](#) power can be applied simultaneously with or after [VBAT](#).

The typical relationships of the MEC1618/MEC1618i power supplies to the system power states is shown below in [Table 6.15](#). The distribution of the MEC1618/MEC1618i power supplies to the various functional blocks is illustrated in [FIGURE 1-1: MEC1618/MEC1618i Top-level Block Diagram on page 6](#).

The typical relationships of the MEC1618/MEC1618i clocks to the system power states is shown below in [Table 6.16](#). Descriptions of the various clock domains can be found in [Section 5.4, "Clock Generator," on page 77](#).

**TABLE 5-21: TYPICAL MEC1618/MEC1618I POWER SUPPLIES VS. ACPI POWER STATES**

Supply Name	ACPI Power State						Description
	S0 (Full On)	S1 (POS)	S3 (STR)	S4 (STD)	S5 (Soft Off)	G3 (MECH Off)	
<a href="#">VBAT</a>	ON	ON	ON	ON	ON	ON	MEC1618/MEC1618i <a href="#">VBAT</a> Well Supply (assuming a <a href="#">TYPE 2</a> configuration as described in <a href="#">Section 5.5.2, "Power Supply Configurations," on page 102</a> )
<a href="#">VTR</a>	ON	ON	ON	ON/OFF	ON/OFF	OFF	MEC1618/MEC1618i Suspend Supply. ( <a href="#">Note 5-29</a> )
VCC	ON	ON	OFF	OFF	OFF	OFF	MEC1618/MEC1618i Runtime Supply ( <a href="#">Note 5-30</a> )

**Note 5-29** [VTR](#) availability in S4 - S5 may depend, for example, on whether AC power is available.

**Note 5-30** The MEC1618/MEC1618i senses the [VCC](#) power state using the [VCC\\_PWRGD](#) input pin; that is, [VCC](#) power is not directly applied to this device.

**TABLE 5-22: TYPICAL MEC1618/MEC1618I CLOCKS VS. ACPI POWER STATES**

Clock Name	ACPI Power State						Description
	S0 (Full On)	S1 (POS)	S3 (STR)	S4 (STD)	S5 (Soft Off)	G3 (MECH Off)	
<a href="#">32K Clock Domain</a> (External Clock Source)	ON	ON	ON	ON	ON	OFF/ON	<a href="#">32K Clock Domain</a> is driven by an external source, for example, by an Intel ICH4M SUSCLK, the <a href="#">32K Clock Domain</a> is running whenever RSMRST is not asserted. (see <a href="#">Ref[6]</a> ).

# MEC1618/MEC1618i

**TABLE 5-22: TYPICAL MEC1618/MEC1618i CLOCKS VS. ACPI POWER STATES (CONTINUED)**

Clock Name	ACPI Power State						Description
	S0 (Full On)	S1 (POS)	S3 (STR)	S4 (STD)	S5 (Soft Off)	G3 (MECH Off)	
32K Clock Domain (Internal Clock Source)	ON	ON	ON	ON	ON	ON	32K Clock Domain is running whenever VBAT is fully powered except following a VBAT_POR as described in "32K_EN," on page 134.
Host Clock Domain	ON	ON	OFF	OFF	OFF	OFF	The Host Clock Domain is gated by the MEC1618/MEC1618i runtime supply (VCC) as described in Section 5.4.9.4, "Host Clock Domain," on page 100.
PCI_CLK	ON/OFF	ON/OFF	OFF	OFF	OFF	OFF	33MHz LPC Bus clock input powered by the MEC1618/MEC1618i runtime supply (VCC). (Note 5-31)
Programmable Clock Domains and Fixed Clock Domain	ON/OFF	ON/OFF	ON/OFF	OFF/ON	OFF/ON	OFF	These clocks are powered by the MEC1618/MEC1618i suspend supply (VTR) but may start and stop as described in Section 5.4.8, "Power Management Interface," on page 89. (see also Note 5-29)

**Note 5-31** The PCI\_CLK can start and stop in S0/S1 as defined in Ref[7].

## 5.5.2 POWER SUPPLY CONFIGURATIONS

There are two acceptable types of MEC1618/MEC1618i power supply configuration that fundamentally differ based on the need for a backup battery connection to VBAT. In both cases VTR is connected to the suspend supply as described in Section 5.5.1, "Power Supplies and Clocks ACPI Context," on page 101.

### 5.5.2.1 TYPE 1

TYPE 1 configurations do not use a VBAT backup battery connection, VBAT is tied to VTR and the Reset Interface generates a VBAT\_POR whenever VTR powers up as described in Section 5.6.5, "VBAT\_POR," on page 108. In TYPE 1 configurations the VBAT-Powered Control Interface is fully operational when VTR is powered.

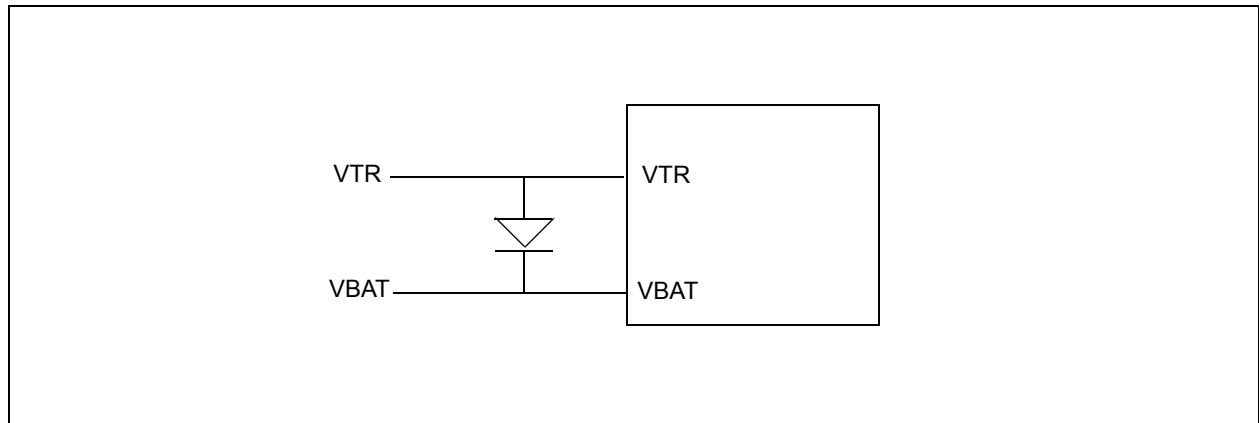
### 5.5.2.2 TYPE 2

TYPE 2 configurations use a VBAT backup battery connection (see Table 44.1, "Operating Conditions," on page 915). Power supply requirements for TYPE 2 configurations are as follows: VBAT is connected to a backup battery that is externally switched through a diode with VTR (Figure 5-16).

In this configuration some internal components that utilize the VBAT power plane are switched internally to VTR using a Power MUX when VTR power is applied as described in Section 5.6.9, "Power MUX," on page 109.

In TYPE 2 configurations, the VBAT-Powered Control Interface can be used to power-on an unpowered system.

**FIGURE 5-16:** TYPE 2 POWER SUPPLY CONFIGURATION



## 5.5.3 POWER-UP SEQUENCE

Table 5-23 summarizes the MEC1618/MEC1618i [Power-Up Sequence](#). For information regarding the typical relationships of the MEC1618/MEC1618i power supplies to the system power states see [Section 5.5, "Power Configuration," on page 101](#).

**TABLE 5-23:** Power-Up Sequence

	VBAT	VTR	VCC	Reset Interface	Description
1.	OFF	OFF	OFF	–	MEC1618/MEC1618i fully unpowered
2.	ON	OFF	OFF	–	<a href="#">32.768 kHz Oscillator</a> may be disabled as described in <a href="#">Section 5.6.5, "VBAT_POR," on page 108</a> .
3.	ON	ON	OFF	<a href="#">VBAT_POR</a> , <a href="#">VTRGD</a> , <a href="#">nSYS_RST</a> , <a href="#">nEC_RST</a> , <a href="#">nSIO_RESET</a>	<a href="#">VBAT</a> -powered registers may be reset as described in <a href="#">Section 5.6.5, "VBAT_POR," on page 108</a> . <a href="#">VTR</a> -powered registers and blocks reset ( <a href="#">nSYS_RST</a> ). EC held in reset as described in <a href="#">Section 5.6.7, "nEC_RST," on page 108</a> .
					<a href="#">nSIO_RESET</a> asserted.
					<a href="#">20 MHz Oscillator</a> enabled.
					EC begins code execution following a delay as described in <a href="#">Section 5.6.7, "nEC_RST," on page 108</a> .
4.	ON	ON	ON	<a href="#">VCC_PWRGD</a> , <a href="#">nSIO_RESET</a>	Registers affected by <a href="#">VCC_PWRGD</a> are reset ( <a href="#">Note 5-32</a> )
					Firmware de-asserts <a href="#">nSIO_RESET</a> as described in <a href="#">"iRESET OUT," on page 114</a> .

**Note 5-32** For [VTR](#)-powered on-chip registers that are reset by [VCC\\_PWRGD](#), it is important that firmware not write to any of these registers until 1 ms following the assertion of [VCC\\_PWRGD](#) ('1').

## 5.6 Reset Interface

### 5.6.1 OVERVIEW

The primary function of the [Reset Interface](#) ([Figure 5-17](#)) is to generate [VTR](#) and [VBAT](#) reset signaling; including, [VTRGD](#), [VBAT\\_POR](#), [nSYS\\_RST](#), [nEC\\_RST](#) and [Watch-Dog Timer Forced Reset](#) ([Table 5-24](#)). The [Reset Interface](#) also includes section regarding a [RESET Pin Interface](#), and the [1.8V Regulator](#). There is other VCC-related [Reset Interface](#) functionality not shown in [Figure 5-17](#) that is described [Section 5.6.10, "VCC Power Good," on page 109](#) and [Section 5.6.11, "LPC RESET," on page 110](#).

Also included in the [Reset Interface](#) are descriptions of the [Power MUX](#) and [Registers Interface](#). These are related [Power, Clocks, and Resets](#) functions that are not described anywhere else in this chapter.

**TABLE 5-24: VTR/VBAT RESET THRESHOLDS**

Parameters	Min	Typ	Max	Units	Notes
<a href="#">VTRGD</a> Reset Threshold	0.5	1.8	2.7	Volts	
<a href="#">VBAT_POR</a> Reset Threshold	0.5	1.25	1.9		

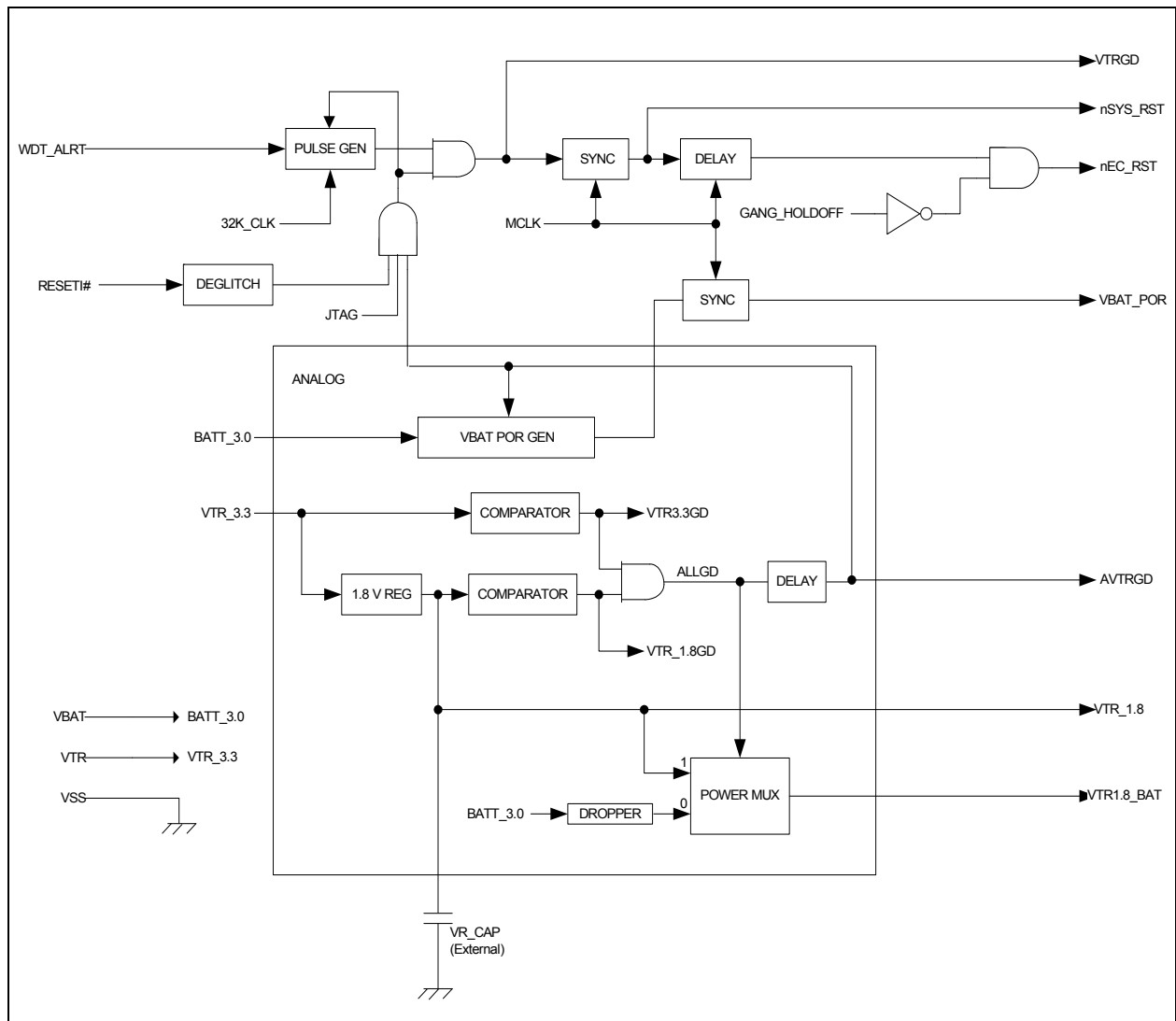
### 5.6.2 STRAP OPTIONS

**TABLE 5-25: STRAP OPTIONS**

Name	Pin	Notes
Boundary Scan	GPIO171/MSDATA	The <a href="#">Boundary Scan</a> strap option is sampled on <a href="#">VTR</a> power up, and is not affected by



**FIGURE 5-17: Reset Interface BLOCK DIAGRAM**



## 5.6.3 RESET PIN INTERFACE

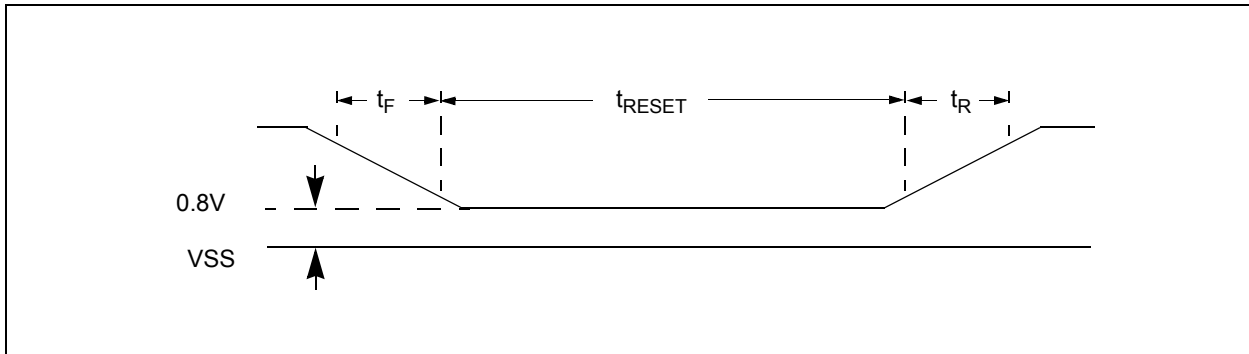
### 5.6.3.1 RESETI#

As illustrated in [Figure 5-18](#), when asserted ('0') the [RESETI#](#) input can force the equivalent of a [VTRGD](#) to the MEC1618/MEC1618i. The [RESETI#](#) input includes an analog glitch filter ([Note 5-33](#)). Timing for the [RESETI#](#) input is shown in [Figure 5-18](#) and [Table 5-26](#).

The [RESETI#](#) input also affects the JTAG interface as defined in [Section 41.7.3.1, "Async JTAG RESET,"](#) on page 611.

# MEC1618/MEC1618i

**FIGURE 5-18: RESETI# TIMING**



**TABLE 5-26: RESETI# TIMING PARAMETERS**

Symbol	Parameter	Limits		Units	Comments
		Min	Max		
$t_F$	RESETI# Fall time	0	10	$\mu\text{s}$	
$t_R$	RESETI# Rise time	0	10	$\mu\text{s}$	
$t_{RESET}$	Minimum Reset Time	1		$\mu\text{s}$	<a href="#">Note 5-33</a>

**Note 5-33** The [RESETI#](#) input can tolerate glitches of no more than 50ns.

## 5.6.3.2 RESETO#

The [RESETO#](#) output pin GPIO062/RESETO# defaults to an output 'low' following a [VTR](#) power up, or assertion of the [RESETI#](#) input pin. Firmware de-asserts [RESETO#](#) by making the GPIO062 output 'high'.

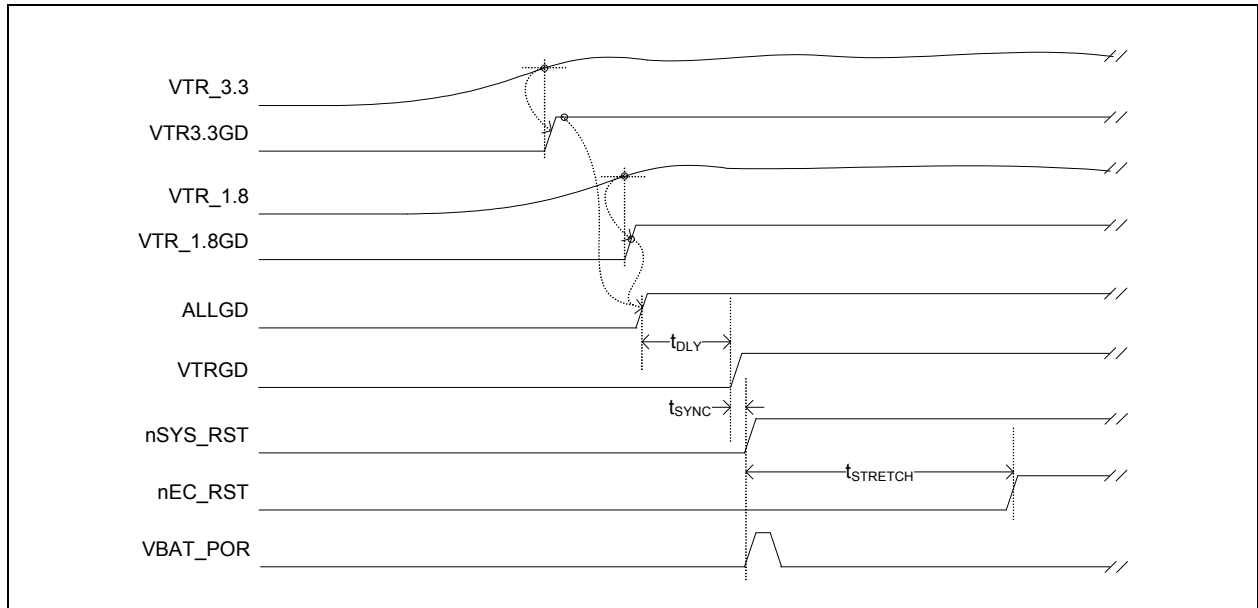
## 5.6.4 VTRGD

[VTRGD](#) is the reset signal for the [20 MHz Oscillator](#) and the source for [nSYS\\_RST](#) and [nEC\\_RST](#).

As shown in [Figure 5-17](#), [Figure 5-19](#) and in [Table 5-27](#), [VTRGD](#) is asserted following a delay after the [VTR](#) and [VTR\\_1.8](#) power supplies exceed preset voltage thresholds as defined in [Table 5-24](#), "[VTR/VBAT Reset Thresholds](#)," on [page 104](#). [VTRGD](#) is de-asserted as soon as either the [VTR](#) or [VTR\\_1.8](#) power supplies drop below these thresholds (see [Figure 5-20](#), "[VTR Power-Down Timing](#)").

[VTRGD](#) can also be asserted as a result of a [Watch-Dog Timer Forced Reset](#) as described in [Section 5.6.12](#), "[Watch-Dog Timer Forced Reset](#)," on [page 110](#), and by the [RESETI#](#) pin in the [RESET Pin Interface](#).

**FIGURE 5-19: VTR POWER-UP TIMING**

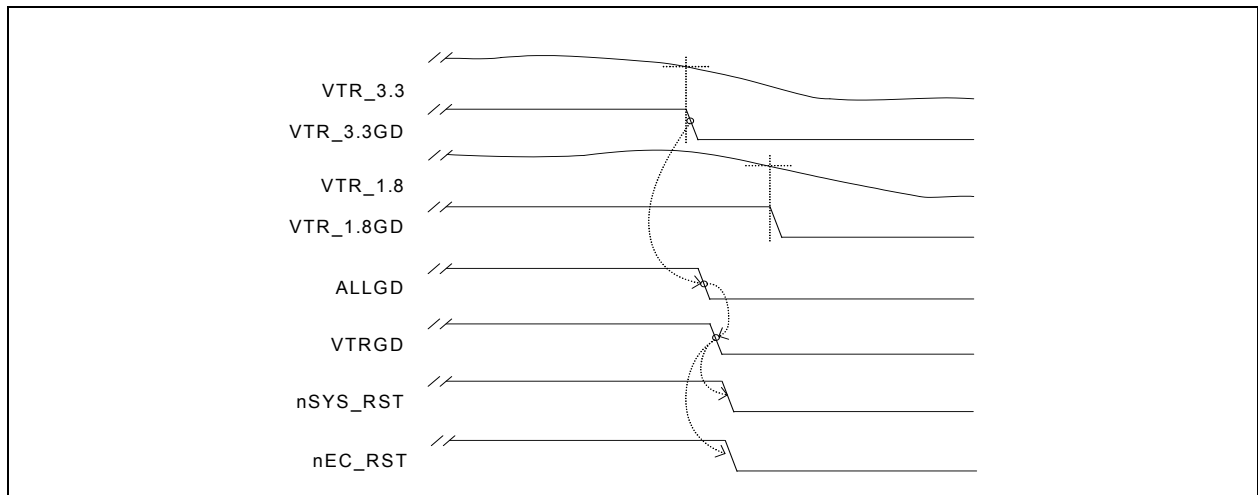


**TABLE 5-27: VTR POWER-UP TIMING**

Parameter	Symbol	Min	Typ	Max	Units	Notes
VTRGD Delay Time	$t_{DLY}$	–	600	800	$\mu$ s	
nSYS_RST Delay Time	$t_{SYNC}$	2	–	3	20 MHz Oscillator Clocks	Note 5-34
nEC_RST Delay Time	$t_{STRETCH}$	–	–	5	ms	

**Note 5-34** This interval is determined using a [Fixed Clock Domain](#) from the [20 MHz Oscillator](#).

**FIGURE 5-20: VTR POWER-DOWN TIMING**



## 5.6.5 VBAT\_POR

**VBAT\_POR** occurs within the [Reset Interface](#) whenever the coin cell is replaced, or the **VBAT** voltage falls below 1.25 V nominal ([Table 5-24](#)).

There is also a **VBAT\_POR** pulse external to the [Reset Interface](#) ([Figure 5-19](#)) that is used to reset **VBAT** powered status bits in **VTR** powered registers. This pulse is asserted at the rising edge of **nSYS\_RST** if a **VBAT\_POR** was detected within the [Reset Interface](#) when **VTRGD** was not asserted.

Note that the [32.768 kHz Oscillator](#) is stopped if the coin cell is replaced, or the **VBAT** voltage falls below 1.25 V nominal while **VTRGD** is not asserted. No action is taken if the coin cell is replaced, or the **VBAT** voltage falls below 1.25 V nominal while **VTRGD** is asserted.

**VBAT\_POR** is used as described throughout this specification to reset registers and functional device blocks. **VBAT\_POR** events are registered in the [Power-Fail and Reset Status Register](#).

## 5.6.6 nSYS\_RST

**nSYS\_RST** is **VTRGD** synchronized to the [20 MHz Oscillator](#). Note that **VTRGD** and **nSYS\_RST** have the same logical sense (uninverted); however, because of nomenclature, the asserted states are opposite. Note that **VTRGD** is defined in [Section 5.6.4, "VTRGD," on page 106](#).

**nSYS\_RST** is de-asserted as defined in [Figure 5-19, "VTR Power-Up Timing"](#) and in [Table 5-27, "VTR Power-Up Timing"](#). **nSYS\_RST** can also be asserted as a result of a [Watch-Dog Timer Forced Reset](#) as described in [Section 5.6.12, "Watch-Dog Timer Forced Reset," on page 110](#), or when the **RESETI#** pin is asserted as described in [Section 5.6.3, "RESET Pin Interface," on page 105](#). **nSYS\_RST** is asserted as soon as either the **VTR** or **VTR\_1.8** power supplies drop below preset voltage thresholds (see [Figure 6.14 VTR Power-Down Timing on page 117](#)).

**nSYS\_RST** is the reset signal for all **VTR**-powered blocks except for the [20 MHz Oscillator](#) and the Embedded Controller. **nSYS\_RST** also affects the [VBAT-Powered Control Interface](#) as described in [Table 31-2, "VCI Output Truth Table," on page 521](#).

## 5.6.7 NEC\_RST

**NEC\_RST** is a delayed version of **nSYS\_RST** that is used to reset the Embedded Controller and for [Registers Interface](#) as described in [Section 5.7, "Registers Interface," on page 111](#).

**NEC\_RST** can be asserted as a result of a [Watch-Dog Timer Forced Reset](#) as described in [Section 5.6.12, "Watch-Dog Timer Forced Reset," on page 110](#), or when the **RESETI#** pin is asserted as described in [Section 5.6.3, "RESET Pin Interface," on page 105](#). Like **nSYS\_RST**, **NEC\_RST** is asserted as soon as either the **VTR** or **VTR\_1.8** power supplies drop below preset voltage thresholds (see [FIGURE 5-20: VTR Power-Down Timing on page 107](#)).

**NEC\_RST** is de-asserted as defined in [Figure 5-19, "VTR Power-Up Timing"](#) and in [Table 5-27, "VTR Power-Up Timing"](#). **NEC\_RST** is held asserted while the [Gang Programmer Interface](#) is activated.

## 5.6.8 1.8V REGULATOR

The [1.8V Regulator](#) generates the MEC1618/MEC1618i core power well. As illustrated in [Figure 5-17, "Reset Interface Block Diagram"](#), the input to the [1.8V Regulator](#) is **VTR**, the output is **VTR\_1.8** (see also [Table 5-4](#)). The [1.8V Regulator](#) is not used when **VTR** is inactive, as described in [Section 5.6.9, "Power MUX" below](#).

The stability of the [1.8V Regulator](#) amplifier depends on an external capacitor, **VR\_CAP** as described in [Table 5-4](#). The choice of capacitor can be either ceramic or low ESR tantalum. Ceramics are the recommended choice due to their superior AC performance (below 100 mΩ ESR), but X5R dielectrics should be used to prevent greater than 20% capacitance variation over temperature and voltage. Low ESR tantalum capacitors will work but care should be taken because the ESR can vary 2x at low temperatures.

**APPLICATION NOTE:** The [1.8V Regulator](#) can be suspended in the [SYSTEM DEEPEST SLEEP](#) state using the **VREG\_SUS** bit in the [VREG Control Register on page 130](#).

## 5.6.9 POWER MUX

To guarantee the highest reliability and lowest possible power consumption, the **Power MUX** switches between the **1.8V Regulator** and a level-shifted **VBAT** voltage to produce the 1.8V internal supply for **VBAT**-backed logic (**VTR1.8\_BAT** in [Table 5-4](#)).

**Power MUX** switching depends on the voltage level of the **1.8V Regulator** and the **VTR** supply. As illustrated in [Figure 5-17](#), the **Power MUX** selects the **1.8V Regulator** after the **VTR** and the **VTR\_1.8** power supplies exceed preset voltage thresholds. The **Power MUX** selects the **VBAT** supply as soon as either the **VTR** or **VTR\_1.8** power supplies drop below these thresholds (see [FIGURE 5-20: VTR Power-Down Timing on page 107](#)).

Note that the **Power MUX** only switches 1.8 volts. To guarantee minimum **VBAT** power consumption for 3.3V **VBAT** powered outputs when **VTR** is fully powered, supply switching from **VBAT** to **VTR** must be done externally.

There is a separate power MUX not shown in [Figure 5-17](#) to guarantee that the **32.768 kHz Oscillator** is powered when **VTR** is powered for **TYPE 1 Power Supply Configurations**.

## 5.6.10 VCC POWER GOOD

**VCC Power Good** is defined by the **VCC\_PWRGD** input pin ([Table 5-4](#)). **VCC\_PWRGD** is also synchronized to the **20 MHz Oscillator** and used for the functions shown in [Table 5-28](#).

The **VCC\_PWRGD** input must always be driven to a '1' or a '0,' even when VCC is 0 V. The minimum **VCC\_PWRGD** pulse width (high and low) is shown below in [Table 5-28](#).

**TABLE 5-28: VCC\_PWRGD INPUT TIMING**

Parameters	Symbol	Min	Typ	Max	Units	Notes
<b>VCC_PWRGD</b> Pulse Width	$t_{VPGPW}$	31	–	–	ns	

**TABLE 5-29: FUNCTIONS AFFECTED BY VCC Power Good**

Name	Reference
Host Clock Domain	see <a href="#">Section 5.4.9.4, "Host Clock Domain," on page 100</a>
LPC RESET	<a href="#">Section 5.6.11, "LPC RESET," on page 110</a>
nSIO_RESET	"iRESET OUT," <a href="#">on page 114</a> .
VCC_PWRGD_BUFF	<a href="#">Table 5-4, "Power, Clocks, and Resets Port List," on page 75</a> .
VCC_PWRGD	"VCC PWRGD," <a href="#">on page 114</a>

# MEC1618/MEC1618i

## 5.6.11 LPC RESET

LPC RESET (LPC\_RST# in Table 5-4) is defined by VCC\_PWRGD, LRESET# and VTRGD as illustrated in Table 5-30. LPC RESET only affects logic that is driven by PCI\_CLK.

**TABLE 5-30: LPC RESET DEFINITION**

VCC_PWRGD (Note 5-35)	LRESET# (Note 5-36)	VTRGD (Note 5-37)	LPC RESET (Note 5-38)
0	X	X	0
1		0	Undefined
	0	1	0
	1		1

**Note 5-35** This is the Table 5-4 VCC\_PWRGD input.

**Note 5-36** This is the Table 5-4 LRESET# input. The EC can determine the state of the LRESET# input using registers in Table 6-8, "LPC Bus Monitor Register," on page 147.

**Note 5-37** See Section 5.6.4, "VTRGD," on page 106.

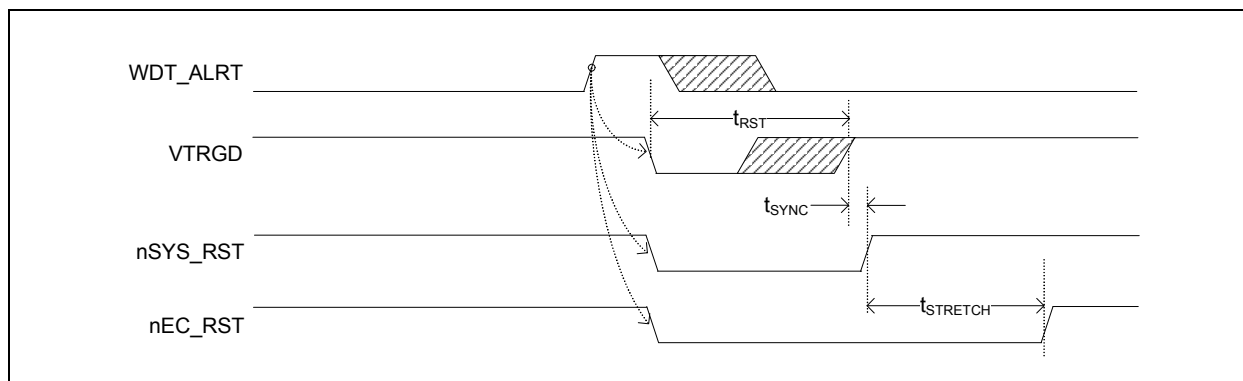
**Note 5-38** LPC RESET is the Table 5-4 LPC\_RST# output. The trailing edge of LPC\_RST# is synchronized to the PCI\_CLK in Table 5-4.

## 5.6.12 WATCH-DOG TIMER FORCED RESET

A Watch-Dog Timer Forced Reset (Figure 5-21) occurs when the WDT\_ALRT input (Table 5-4) is asserted ('1'). As shown in Figure 5-21, VTRGD is de-asserted ('0') and nSYS\_RST and nEC\_RST are asserted ('0') when WDT\_ALRT is asserted. The VTRGD reset time ( $t_{RST}$ ) is determined by the 32.768 kHz Oscillator as shown in Table 5-31. Following the VTRGD reset time, the nSYS\_RST Delay Time ( $t_{SYNC}$ ) and the nEC\_RST Delay Time ( $t_{STRETCH}$ ) are determined using the 20 MHz Oscillator as described in Figure 5-19, "VTR Power-Up Timing" and Table 5-27, "VTR Power-Up Timing".

Note that analog reset signal functions are not shown in Figure 5-21 because it is assumed that the power supplies are fully powered and stable during a Watch-Dog Timer Forced Reset.

**FIGURE 5-21: Watch-Dog Timer Forced Reset TIMING**



**TABLE 5-31: Watch-Dog Timer Forced Reset TIMING**

Parameters	Symbol	Min	Typ	Max	Units
VTRGD Reset Time	$t_{RST}$	1	–	2	32.768 kHz Oscillator Clock Cycles
nSYS_RST Delay Time	$t_{SYNC}$	(see FIGURE 5-19: VTR Power-Up Timing on page 107)			
nEC_RST Delay Time	$t_{STRETCH}$				

## 5.7 Registers Interface

The [Power, Clocks, and Resets](#) registers are located in two address ranges with two Base Address as indicated in [Table 5-32](#). See [Note 3-1 on page 52](#).

**TABLE 5-32: POWER, CLOCKS AND RESETS INTERFACE BASE ADDRESS TABLE**

Power, Clocks and Resets Blocks	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
Power, Clock & Reset (VTR PWR'ed)	32h	F0_C800h
Power, Clock & Reset (VBAT PWR'ed)	33h	F0_CC00h

[Table 5-33](#) is a register summary for the [Power, Clock & Reset \(VTR PWR'ed\)](#) registers. [Table 5-34](#) is a register summary for the [Power, Clock & Reset \(VBAT PWR'ed\)](#) registers

Each EC address is indicated as an SPB Offset from its AHB base address as indicated in [Table 5-32](#).

The following tables summarize the registers allocated for each Instance. The offset field in the following table is the offset from the Embedded Controller (EC) Base Address.

**TABLE 5-33: Power, Clocks, and Resets VTR-POWERED REGISTERS SUMMARY**

Offset (HEX) ( <a href="#">Note 5-39</a> )	Register Name	Access	Page Reference
0	<a href="#">EC Clock Divider Register</a>	R/W	<a href="#">113</a>
4	MCHP Reserved	R/W	–
8	<a href="#">PCR Status and Control Register</a>	R	<a href="#">114</a>
C	<a href="#">Clock Control Register</a>	R/W	<a href="#">115</a>
10	<a href="#">LPC Blocks Sleep Enables Register</a>	R/W	<a href="#">118</a>
14	<a href="#">EC Blocks Sleep Enables Register 1</a>	R/W	<a href="#">120</a>
18	<a href="#">LPC Blocks Clock Required Status Register</a>	R	<a href="#">125</a>
1C	<a href="#">EC Blocks Clock Required Status Register 1</a>	R	<a href="#">120</a>
20	Reserved	R	–
24	MCHP Reserved	R/W	–
28	Reserved	R	–
2C	Reserved	R	–
30	<a href="#">EC Blocks Sleep Enables Register 2</a>	R/W	<a href="#">121</a>

# MEC1618/MEC1618i

**TABLE 5-33: Power, Clocks, and Resets VTR-POWERED REGISTERS SUMMARY (CONTINUED)**

Offset (HEX) (Note 5-39)	Register Name	Access	Page Reference
34	<a href="#">EC Blocks Clock Required Status Register 2</a>	R	<a href="#">127</a>
38	Reserved	R	–
3C	Reserved	R	–
40	<a href="#">Clock Tree Control 0 Register</a>	R/W	<a href="#">128</a>
44	<a href="#">Clock Tree Control 1 Register</a>	R/W	<a href="#">129</a>
48	<a href="#">VREG Control Register</a>	R/W	<a href="#">130</a>
4C	Reserved	R	–
50	Reserved	R	–
54	MCHP Reserved	R/W	–

**Note 5-39** All register addresses are naturally aligned on 32-bit boundaries. Offsets for registers that are smaller than 32 bits are reserved and must not be used for any other purpose.

**TABLE 5-34: Power, Clocks, and Resets VBAT-POWERED REGISTERS SUMMARY**

Offset (HEX) (Note 5-39)	Register Name	Access	Page Reference
0	<a href="#">Power-Fail and Reset Status Register</a>	R/W	<a href="#">131</a>
4	<a href="#">Clock Enable Register</a>	R/W	<a href="#">133</a>
8	MCHP Reserved	R/W	–
C	MCHP Reserved	R/W	–



## 5.7.1 EC CLOCK DIVIDER REGISTER

### 5.7.1.1 Overview

The [EC Clock Divider Register](#) (Table 5-35) contains the [EC\\_CLK\\_DIV](#) bits that are used to program the EC clock and the EC\_AHB clock enable frequency as described in [Table 5-36](#).

**TABLE 5-35: EC Clock Divider Register**

HOST ADDRESS	N/A			N/A		HOST SIZE		
EC OFFSET	00h			32-bit		EC SIZE		
POWER	VTR			01h		nSYS_RST DEFAULT		
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R/W	R/W	R/W	R/W
BIT NAME	Reserved				EC_CLK_DIV			

### EC\_CLK\_DIV

The [EC\\_CLK\\_DIV](#) bits contain the binary encoded divider that determines the EC clock and EC\_AHB clock enable frequency ([Table 5-36](#)). Valid [EC\\_CLK\\_DIV](#) values are 1h - Fh; writing '0' to the [EC Clock Divider Register](#) has no effect. The [EC\\_CLK\\_DIV](#) default is 01h.

**TABLE 5-36: EC\_CLK\_DIV ENCODING**

<a href="#">EC_CLK_DIV</a>	<b>Nominal Frequency (MHz)</b>
0h	No Change
1h	20.27 (default)
2h	10.14
3h - Fh	6.76 – 1.35

**APPLICATION NOTE:** the JTAG clock can't be higher than 1/2 the EC clock as defined by the [EC Clock Divider Register](#).

# MEC1618/MEC1618i

## 5.7.2 PCR STATUS AND CONTROL REGISTER

TABLE 5-37: PCR STATUS AND CONTROL REGISTER

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	08h			32-bit			EC SIZE	
POWER	VTR			00XX_XX1Xb			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R/W	R	R	R	R
BIT NAME	Reserved		FREQ LOCK	iRESET OUT	Reserved	Reserved	VCC PWRGD	Reserved

### VCC PWRGD

The **VCC PWRGD** bit reflects the state of the synchronized VCC\_PWRGD input pin (see [Section 5.6.10, "VCC Power Good," on page 109](#)). The VCC\_PWRGD pin can generate an either-edge interrupt as described in the note associated with the VCC\_PWRGD signal in [Section 21.4, "GPIO Indexing," on page 400](#).

### iRESET OUT

The **iRESET OUT** bit is used by firmware to control the **nSIO\_RESET** signal function ([Table 5-4](#)). Firmware can program the state of **iRESET OUT** except when the **VCC PWRGD** bit is not asserted ('0'), in which case **iRESET OUT** is 'don't care' and **nSIO\_RESET** is asserted ('0') ([Table 5-38](#)). In all other cases, the **nSIO\_RESET** signal function is always the inverse of the **iRESET OUT** bit.

**APPLICATION NOTE:** it should be noted that when the **iRESET OUT** bit is asserted ('1') the internal **nSIO\_RESET** is asserted even if the **nRESET\_OUT** pin is configured as an alternate function.

**Note 5-40** When the **iRESET OUT** bit is set to '1', the falling edge of **VCC PWRGD** causes the **nRESET\_OUT** signal function to be asserted within 15ns. A subsequent rising edge of **VCC PWRGD** will cause the **nRESET\_OUT** signal function to de-assert within 3 **MCLKs**.

**Note 5-41** **nSIO\_RESET** is also the source for the **nRESET\_OUT** signal function ([Table 3.15, "Miscellaneous Functions," on page 61](#)).

**TABLE 5-38: [iRESET OUT BIT BEHAVIOR](#)**

<a href="#">VCC PWRGD</a>	<a href="#">iRESET OUT</a>	<a href="#">nSIO_RESET</a> & <a href="#">nRESET_OUT</a> (See <a href="#">Note 5-41</a> )	Description
0	X	0 (ASSERTED)	The <a href="#">iRESET OUT</a> bit does not affect the state of <a href="#">nSIO_RESET</a> when <a href="#">VCC PWRGD</a> is not asserted.
1	1	0 (ASSERTED)	The <a href="#">iRESET OUT</a> bit can only be written by firmware when <a href="#">VCC PWRGD</a> is asserted.
	0	1 (NOT ASSERTED)	

## FREQ LOCK

[FREQ LOCK](#) is asserted ('1') when the accuracy of the [20 MHz Oscillator](#) is within the tightest tolerance described in [Table 5-6, "20 MHz Oscillator Timing Parameters,"](#) on page 80.

## 5.7.3 CLOCK CONTROL REGISTER

**TABLE 5-39: [Clock Control Register](#)**

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	0Ch			32-bit			EC SIZE	
POWER	VTR			04h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
BIT NAME	SLEEP CONFIG			WAIT FOR LOCK	32kHz OUTPUT	MCHP Reserved	SLEEP_FLAG	Reserved

## SLEEP\_FLAG

The [SLEEP\\_FLAG](#) affects the system power state as described in [Section 5.4.8.2, "EC Controlled Dynamic Power States,"](#) on page 90. The [SLEEP\\_FLAG](#) is R/W. EC firmware asserts [SLEEP\\_FLAG](#) ('1'), which is then typically de-asserted ('0') by hardware as described in [Section 5.4.8.6, "Wake Interface,"](#) on page 98.

# MEC1618/MEC1618i

**APPLICATION NOTE:** Asserting the SLEEP\_FLAG allows the device to be put into lower power states as described in [Section 5.4.8.2, "EC Controlled Dynamic Power States," on page 90](#). If the main oscillator is shut down as a result of asserting the SLEEP\_FLAG, then only the enabled wake capable interrupts can restart the oscillator and bring the part out of its sleep state. When the main oscillator is shut down, non-wake capable interrupts cannot occur. As part of the wake process, the SLEEP\_FLAG is cleared by hardware, re-enabling all blocks by de-asserting all the block SLEEP\_ENABLE signals as described in [Section 5.4.8.5.4, "Block Sleep Enables"](#).

The typical sequence to go to sleep is this:

```
disable_interrupts();
assert_all_sleep_enables();
CLOCK_CONTROL_REGISTER->SLEEP_FLAG = 1;
_SLEEP();
```

where the macro `_SLEEP()` both enables interrupts and issues the ARC sleep instruction.

However, if any interrupt, including non-Wake interrupts, occurs just before the SLEEP\_FLAG is asserted, it will remain pending and will fire as soon as the ARC issues the sleep instruction. The SLEEP\_FLAG will remain asserted, and therefore all blocks are kept in the low power state by their respective SLEEP\_ENABLE signals. As a result, an interrupt service routine may not proceed properly because blocks that it must manipulate remain in their idle state.

If a system uses the sleep interface then every Interrupt Service Routine must manually clear the SLEEP\_FLAG. This guarantees that all blocks that are supposed to be active when the ISR is in process will be active.

## 32KHZ OUTPUT

The [32kHz OUTPUT](#) bit controls the MEC1618/MEC1618i [32KHZ\\_OUT](#) signal function ([Table 5-4](#)). When [32kHz OUTPUT](#) is de-asserted '0,' the 32kHz output clock is disabled and the [32KHZ\\_OUT](#) signal function is driven low. When [32kHz OUTPUT](#) is asserted, the 32kHz output clock is enabled. The [32kHz OUTPUT](#) bit is R/W and disabled by default following [VTRGD](#).

## WAIT FOR LOCK

**TABLE 5-40: WAIT FOR LOCK BIT ENCODING**

WAIT FOR LOCK	Description
0	The system is clocked by the <a href="#">10 MHz Ring Oscillator</a> when a wake event occurs and switches to the <a href="#">20 MHz Oscillator</a> when the <a href="#">FREQ LOCK</a> bit is asserted. (default)
1	The system clocking remains gated when a wake event occurs until the <a href="#">FREQ LOCK</a> bit is asserted.

**SLEEP CONFIG**

The **SLEEP CONFIG** field determines the latency following a wake event until the **20 MHz Oscillator** is locked (**FREQ LOCK**) and clocking the system. This latency is determined as defined in [Table 5-41](#).

**TABLE 5-41: SLEEP CONFIG BIT ENCODING**

SLEEP CONFIG			Wake Latency	Description ( <a href="#">Note 5-42</a> )
D7	d6	d5		
0	0	0	0 ns	<a href="#">Master Clock Trees</a> , the <a href="#">1.8V Regulator</a> , the <a href="#">20 MHz Oscillator</a> , and the <a href="#">20 MHz Oscillator</a> regulator all remain powered and running during sleep cycles ( <a href="#">SYSTEM HEAVY SLEEP 1</a> ) (DEFAULT)
0	0	1	200 ns	<a href="#">Master Clock Trees</a> gated and the <a href="#">1.8V Regulator</a> is suspended during sleep cycles. ( <a href="#">SYSTEM HEAVY SLEEP 2</a> )
0	1	0	200 $\mu$ s	<a href="#">Master Clock Trees</a> gated, the <a href="#">1.8V Regulator</a> is suspended, and the <a href="#">20 MHz Oscillator</a> is stopped during sleep cycles. ( <a href="#">SYSTEM HEAVY SLEEP 3</a> )
0	1	1	2 ms	<a href="#">Master Clock Trees</a> gated, the <a href="#">1.8V Regulator</a> is suspended, the <a href="#">20 MHz Oscillator</a> is stopped, and the <a href="#">20 MHz Oscillator</a> regulator is powered down during sleep cycles. ( <a href="#">SYSTEM DEEPEST SLEEP</a> )
1	X	X	–	Reserved

**Note 5-42** Suspending the [1.8V Regulator](#) also depends on the state of the [VREG SUS](#) bit in the [VREG Control Register](#).

# MEC1618/MEC1618i

## 5.7.4 BLOCK SLEEP ENABLE REGISTERS

The [Block Sleep Enables](#) identified in the [Block Sleep Enable Registers](#) are described below in [Section 5.7.4.1, "LPC Blocks Sleep Enables Register,"](#) on page 118 and [Section 5.7.4.2, "EC Blocks Sleep Enables Registers,"](#) on page 120. The behavior of the [Block Sleep Enables](#) is described in ["Block Sleep Enables,"](#) on page 97.

### 5.7.4.1 LPC Blocks Sleep Enables Register

**TABLE 5-42:** [LPC Blocks Sleep Enables Register](#)

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	10h			32-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved	MCHP Reserved	EMI	FLASH SPI	MCHP Reserved	UART	MCHP Reserved	

**Note 5-43** All writes to the [LPC Blocks Sleep Enables Register](#) should clear bit[0] to '0'.

**TABLE 5-43:** [LPC BLOCKS SLEEP ENABLES/CLOCK REQUIRED REGISTERS BIT NAMES](#)

<b>Bit Name</b>	<b>Block Name</b>	<b>Block Cross References</b>
UART	16C550A UART	<a href="#">Section 12.8, "Sleep Enable/ Clock Request Power State Controls,"</a> on page 255
FLASH SPI	LPC GP-SPI	<a href="#">Section 30.0, "General Purpose Serial Peripheral Interface (GP-SPI),"</a> on page 494
EMI	Embedded Memory Interface	<a href="#">Section 7.0, "Embedded Memory Interface,"</a> on page 153

Some LPC accessible blocks have no Block Sleep Enable bit in the [LPC Blocks Sleep Enables Register](#). [Table 5-44](#) describes these.

**TABLE 5-44: LPC BLOCKS NOT CONTROLLED BY [LPC Blocks Sleep Enables Register](#)**

Block Name	Block Cross References
LPC Interface	<a href="#">Section 6.10.4, "EC Clock Control Register," on page 151</a>
Legacy Port Functions	<a href="#">Legacy Support on page 208</a>
ACPI EC Interface	<a href="#">Section 8.0, "ACPI Embedded Controller Interface (ACPI-ECI)," on page 177</a>
8042 Emulated Keyboard Controller Interface	<a href="#">Section 9.0, "8042 Emulated Keyboard Controller," on page 199</a>
ACPI PM1 Interface	<a href="#">Section 10.0, "ACPI PM1 Block Interface," on page 216</a>
Mailbox Registers Interface	<a href="#">Section 11.0, "MailBox Register Interface," on page 225</a>

# MEC1618/MEC1618i

## 5.7.4.2 EC Blocks Sleep Enables Registers

**TABLE 5-45: EC BLOCKS SLEEP ENABLES REGISTER 1**

HOST ADDRESS	N/A				N/A			HOST SIZE	
EC OFFSET	14h				32-bit			EC SIZE	
POWER	VTR				00_0000h			nSYS_RST DEFAULT	
BUS	EC SPB								
BIT	D31	D30	D28	D28	D27	D26	D25	D24	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R	R	R	R	R	R/W	R/W	R/W	
BIT NAME	Reserved					WDT	HIB1	HIB0	
BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	FLASH	SPI_P	SMB1	SMB0	PWM3	PWM2	PWM1	PWM0	
BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R/W	R/W	R/W	R	R	R	R	R/W	
BIT NAME	PS2_2	PS2_1	PS2_0	Reserved				TACH2	
BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
BIT NAME	TACH1	TACH0	Reserved	RC_ID	C/T3	C/T2	C/T1	C/T0	



**TABLE 5-46: EC BLOCKS SLEEP ENABLES REGISTER 2**

HOST ADDRESS	N/A				N/A			HOST SIZE	
EC OFFSET	30h				32-bit			EC SIZE	
POWER	VTR				0000h			nSYS_RST DEFAULT	
	EC SPB								
BIT	D31	D30	D28	D28	D27	D26	D25	D24	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R	R	R	R	R	R/W	R/W	R/W	
BIT NAME	Reserved					MCHP Reserved	TACH5	TACH4	
BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	PWM15	PWM14	PWM13	PWM12	PWM11	PWM10	PWM9	PWM8	
BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	LED2	LED1	LED0	CEC	Rsrvd	KSC	PECI	ADC	
BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	DMA	CCT	SMB2	TACH3	PWM7	PWM6	PWM5	PWM4	

# MEC1618/MEC1618i

**TABLE 5-47: EC BLOCKS SLEEP ENABLES/CLOCK REQUIRED REGISTERS BIT NAMES**

Bit Name	Block Name (Note 5-44)	Block Cross References
C/T0	16-Bit Counter/Timer 0	<a href="#">Section 18.0, "16-Bit Timer Interface," on page 372</a>
C/T1	16-Bit Counter/Timer 1	
C/T2	16-Bit Counter/Timer 2	
C/T3	16-Bit Counter/Timer 3	
RC_ID	RC ID Interface	<a href="#">Section 29.0, "RC Identification Detection (RC_ID)," on page 485</a>
Reserved	–	–
TACH0	Tachometer 0	<a href="#">Section 27.0, "TACH Monitor," on page 469</a>
TACH1	Tachometer 1	
TACH2	Tachometer 2	
TACH3	Tachometer 3	
TACH4	Tachometer 4	
TACH5	Tachometer 5	
PS2_0	PS/2 Interface 0	<a href="#">Section 34.0, "PS/2 Device Interface," on page 548</a>
PS2_1	PS/2 Interface 1	
PS2_2	PS/2 Interface 2	
PWM0	PWM 0	<a href="#">Section 28.0, "PWM Controller," on page 478</a>
PWM1	PWM 1	
PWM2	PWM 2	
PWM3	PWM 3	
PWM4	PWM 4	
PWM5	PWM 5	
PWM6	PWM 6	
PWM7	PWM 7	
PWM8	PWM8	
PWM9	PWM9	
PWM10	PWM10	
PWM11	PWM11	
PWM12	PWM12	
PWM13	PWM13	
PWM14	PWM14	
PWM15	PWM15	

**TABLE 5-47: EC BLOCKS SLEEP ENABLES/CLOCK REQUIRED REGISTERS BIT NAMES**

Bit Name	Block Name (Note 5-44)	Block Cross References
SMB0	SMBus 0	Section 24.0, "SMB Device Interface," on page 449
SMB1	SMBus 1	
SMB2	SMBus 2	
SPI_P	SPI Peripheral Interface	Section 30.0, "General Purpose Serial Peripheral Interface (GP-SPI)," on page 494
CCT	Capture Compare Timer	Section 22.0, "Input Capture and Compare Timer," on page 421
DMA	DMA	Section 23.0, "DMA Controller," on page 436
ADC	ADC	Section 26.0, "Analog to Digital Converter," on page 457
PECI	PECI	Section 25.0, "PECI Interface," on page 454
KSC	Key scan	Section 35.0, "Keyboard Matrix Scan Support," on page 558
HIB0	Hibernation Timer 0	Section 19.0, "Hibernation Timer," on page 389
HIB1	Hibernation Timer 1	
FLASH	Flash Interface	Section 13.0, "Embedded Flash Subsystem," on page 256
WDT	Watchdog Timer	Section 16.0, "Watchdog Timer Interface," on page 349
LED0	Blinking/Breathing LED 0	Section 33.0, "Blinking/Breathing PWM," on page 528
LED1	Blinking/Breathing LED 1	
LED2	Blinking/Breathing LED 2	
CEC	HDMI CEC Interface	Section 17.0, "HDMI-CEC Interface Controller," on page 356
Reserved	–	–
MCHP Reserved (Note 5-45)	–	–

**Note 5-44** Some EC accessible blocks have no Block Sleep Enable bit in the [EC Blocks Sleep Enables Register 1](#). [Table 5-48](#) describes these.

**Note 5-45** MCHP Reserved Sleep Enable bits must be set to one ('1') to enter System Heavy and Deepest Sleep states defined in [Table 5-13, "EC Controlled Dynamic Power States"](#).

**TABLE 5-48: EC BLOCKS NOT CONTROLLED BY EC Blocks Sleep Enables Registers**

Block Name	Block Cross References
MCU Serial Debug Port	<a href="#">Section 38.0, "Serial Debug Port," on page 584</a>
Master BC Link D	<a href="#">Section 36.0, "BC-Link Master," on page 567</a>
Master BC Link A	
Master BC Link B	
Master BC Link C	
Flash Interface	<a href="#">Section Chapter 14, "Embedded Flash Subsystem," on page 339</a>
EC	<a href="#">Section 14.0, "ARC 625D Embedded Controller," on page 288</a>
Interrupt Aggregator	<a href="#">Section 15.0, "EC Interrupt Aggregator," on page 296 (see also Section 5.4.7.2.2, "INTERRUPT AGGREGATOR CLOCK TREE," on page 87)</a>
Week Alarm Timer	<a href="#">Section 20.0, "Week Alarm Interface," on page 393</a>
GPIO	<a href="#">Section 21.0, "GPIO Interface," on page 398 (see also Section 5.4.7.2.3, "GPIO CLOCK TREE," on page 87)</a>
VCI	<a href="#">Section 31.0, "VBAT-Powered Control Interface," on page 519</a>
VBAT_RAM	<a href="#">Section 32.0, "VBAT Powered RAM," on page 526</a>

## 5.7.5 CLOCK REQUIRED STATUS REGISTERS

The [Clock Required Status Registers](#) indicates the core clock status per block as defined in [Section 5.4.6, "Generic Block Clocking Model," on page 83](#). Like the [Block Sleep Enable Registers](#), there are two types of [Clock Required Status Registers](#): the [LPC Blocks Clock Required Status Register](#) and the [EC Blocks Clock Required Status Registers](#).

When a bit in the [Clock Required Status Registers](#) is asserted ('1'), the block is enabled and requires that the [20 MHz Oscillator](#) remain running as defined in ["EC Power State Controls," on page 94](#).

When a bit in the [Clock Required Status Registers](#) is not asserted ('0'), the block is either not enabled as defined in the [Generic Block Clocking Model](#), or has been commanded to sleep and no longer requires the [20 MHz Oscillator](#).

**Note:** A block that is commanded to sleep will not deassert the clock required signal until the block determines it no longer requires the clock. Each block description that has a sleep interface defines how the individual block determines when it is safe to deassert the clock required signal.

## 5.7.5.1 LPC Blocks Clock Required Status Register

**TABLE 5-49:** LPC Blocks Clock Required Status Register

HOST ADDRESS	N/A				N/A		HOST SIZE	
EC OFFSET	18h				32-bit		EC SIZE	
POWER	VTR				0Xh		nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved						BDP1	BDP0
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved	MCHP Reserved	EMI	FLASH SPI	LEGACY	UART	LPC	Reserved

**Note 5-46** Bit[0] in the LPC Blocks Clock Required Status Register may change state.

# MEC1618/MEC1618i

## 5.7.5.2 EC Blocks Clock Required Status Registers

**TABLE 5-50: EC BLOCKS CLOCK REQUIRED STATUS REGISTER 1**

HOST ADDRESS	N/A				N/A			HOST SIZE	
EC OFFSET	1Ch				32-bit			EC SIZE	
POWER	VTR				XX_XXXXh			nSYS_RST DEFAULT	
BUS	EC SPB								
BIT	D31	D30	D28	D28	D27	D26	D25	D24	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved					WDT	HIB1	HIB0	
BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	FLASH Note 5-47	SPI_P	SMB1	SMB0	PWM3	PWM2	PWM1	PWM0	
BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	PS2_2	PS2_1	PS2_0	MBCLC	MBCLB	MBCLA	MBCLD	TACH2	
BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	–	–	–	–	–	–	–	–	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	TACH1	TACH0	MSDP	RC_ID	C/T3	C/T2	C/T1	C/T0	

**Note 5-47** The Flash Clock Required status bit **FLASH** is asserted whenever the ARC or a JTAG Debug master attempts to access the **EC Blocks Clock Required Status Registers**.

**TABLE 5-51: EC BLOCKS CLOCK REQUIRED STATUS REGISTER 2**

HOST ADDRESS	N/A				N/A		HOST SIZE	
EC OFFSET	34h				32-bit		EC SIZE	
POWER	VTR				0000h		nSYS_RST DEFAULT	
	EC SPB							
BIT	D31	D30	D28	D28	D27	D26	D25	D24
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Rsrvd	Rsrvd	Reserved			MCHP Reserved	TACH5	TACH4
BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	PWM15	PWM14	PWM13	PWM12	PWM11	PWM10	PWM9	PWM8
BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	LED2	LED1	LED0	CEC	Rsrvd	KSC	PECI	ADC
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	DMA	CCT	SMB2	TACH3	PWM7	PWM6	PWM5	PWM4

# MEC1618/MEC1618i

**TABLE 5-52: BLOCKS NOT CONTROLLED BY SLEEP ENABLE BITS WITH CLOCK REQUIRED STATUS OUTPUTS**

Bit Name	Block Name	Block Cross References
LEGACY	Legacy Port Functions	<a href="#">Legacy Support on page 208</a>
UART	16C550A UART	<a href="#">Section 12.8, "Sleep Enable/ Clock Request Power State Controls," on page 255</a>
LPC	LPC Interface	<a href="#">Section 6.10.4, "EC Clock Control Register," on page 151</a>
MBCLD	Master BC Link D	<a href="#">Section 36.0, "BC-Link Master," on page 567</a>
MBCLA	Master BC Link A	
MBCLB	Master BC Link B	
MBCLC	Master BC Link C	
MSDP	MCU Serial Debug Port	<a href="#">Section 38.0, "Serial Debug Port," on page 584</a>
BDP0	BIOS Debug Port 0	<a href="#">Section 37.0, "BIOS Debug Port," on page 576</a>
BDP1	BIOS Debug Port 1	

## 5.7.6 CLOCK TREE CONTROL 0 REGISTER

**TABLE 5-53: [Clock Tree Control 0 Register](#)**

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	40h			32-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	FECT3 OFF	FECT3 ON	FECT2 OFF	FECT2 ON	FECT1 OFF	FECT1 ON	FECT0 OFF	FECT0 ON



## FECT0 ON

Force [EC Clock](#) on.

## FECT0 OFF

Force [EC Clock](#) off.

## FECT1 ON

Force [EC Clock](#) on.

## FECT1 OFF

Force [EC Clock](#) off.

## FECT2 ON

Force [EC Clock](#) on.

## FECT2 OFF

Force [EC Clock](#) off.

## FECT3 ON

Force [EC Clock](#) on.

## FECT3 OFF

Force [EC Clock](#) off.

### 5.7.7 CLOCK TREE CONTROL 1 REGISTER

**TABLE 5-54:** [Clock Tree Control 1 Register](#)

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	44h			32-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	FGPT OFF	FGPT ON	FINTT OFF	FINTT ON	FHT1 OFF	FHT1 ON	FHT0 OFF	FHT0 ON

**Note:** Bits [7:4] have no affect when [EC Clock](#) is off.

# MEC1618/MEC1618i

## FHT0 ON

Force [Host Clock Tree 0](#) on.

## FHT0 OFF

Force [Host Clock Tree 0](#) off.

**Note 5-48** Forcing [Host Clock Tree 0](#) 'off' inhibits access to the [Clock Tree Control 1 Register](#) register. [nSYS\\_RST](#) is required to re-enable access to this register.

## FHT1 ON

Force [Host Clock Tree 1](#) on.

## FHT1 OFF

Force [Host Clock Tree 1](#) off.

## FINTT ON

Force [INTERRUPT AGGREGATOR CLOCK TREE](#) on.

## FINTT OFF

Force [INTERRUPT AGGREGATOR CLOCK TREE](#) off.

## FGPT ON

Force [GPIO CLOCK TREE](#) on.

## FGPT OFF

Force [GPIO CLOCK TREE](#) off.

## 5.7.8 VREG CONTROL REGISTER

**TABLE 5-55:** [VREG Control Register](#)

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	48h			32-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	VREG SUS	MCHP Reserved						

**MCHP RESERVED**

Writing to [MCHP Reserved](#) bits may cause undesirable results.

**VREG SUS****TABLE 5-56:** [VREG SUS](#) BIT ENCODING

<a href="#">VREG SUS</a>	
0	<a href="#">1.8V Regulator</a> is always powered.
1	Suspend <a href="#">1.8V Regulator</a> in <a href="#">SYSTEM HEAVY SLEEP 2</a> , <a href="#">SYSTEM HEAVY SLEEP 3</a> , and <a href="#">SYSTEM DEEPEST SLEEP</a> states. (See <a href="#">Table 5-13</a> , "EC Controlled Dynamic Power States," on page 91.)

**5.8 VBAT Powered Registers****5.8.1 POWER-FAIL AND RESET STATUS REGISTER****5.8.1.1 Overview**

The [Power-Fail and Reset Status Register](#) collects and retains the [VBAT RST](#), [FLASH](#) and [WDT](#) event status when [VTR](#) is unpowered. Asserted events can cause interrupts as described in [Section 5.9](#), "Interrupt Interface," on page 134.

**TABLE 5-57:** [Power-Fail and Reset Status Register](#)

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	00h			32-bit			EC SIZE	
POWER	VBAT			1XX000Xb			VBAT_POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R/WC	R/WC	R/WC	R	R	R	R	R
BIT NAME	VBAT RST	FLASH	WDT	Reserved				DET32K _IN

# MEC1618/MEC1618i

---

DET32K\_IN

**TABLE 5-58: DET32K\_IN BIT ENCODING**

DET32K_IN	Description
0	No clock detected on the 32KHZ_IN pin
1	Clock is detected on the 32KHZ_IN pin

## WDT

The **WDT** bit is asserted ('1') following a [Watch-Dog Timer Forced Reset](#) as described in [Section 5.6.12, "Watch-Dog Timer Forced Reset," on page 110](#). To clear the **WDT** bit EC firmware must write a '1' to this bit; writing a '0' to the **WDT** bit has no affect.

## FLASH

The **FLASH** bit is set to '1' by hardware when **FLASH\_PGM** in [Table 5-4, "Power, Clocks, and Resets Port List," on page 75](#) is asserted. **FLASH\_PGM** is asserted when the [Embedded Flash Subsystem](#) is placed in [Program Mode](#) or [Erase Mode](#). To clear the **FLASH** bit EC firmware must write a '1' to this bit; writing a '0' to the **FLASH** bit has no affect.

## VBAT RST

The **VBAT RST** bit is set to '1' by hardware when a **VBAT\_POR** is detected. This is the register default value. To clear **VBAT RST** EC firmware must write a '1' to this bit; writing a '0' to **VBAT RST** has no affect.

## 5.8.2 CLOCK ENABLE REGISTER

The [Clock Enable Register AD\\_EN](#), [32K\\_EN](#), and [EXT32K\\_VBAT](#) bits should not toggle on the same write; i.e., only one bit should be updated at a time.

**TABLE 5-59:** [Clock Enable Register](#)

HOST ADDRESS	N/A			N/A			HOST SIZE	
EC OFFSET	04h			32-bit			EC SIZE	
POWER	VBAT			00h			VBAT_POR DEFAULT	
				N/A			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	---		D10	D9	D8
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R	R	R	R	R	R/W	R/W	R/W
BIT NAME	Reserved					AD_EN	32K_EN	EXT32K_VBAT

### EXT32K\_VBAT

[EXT32K\\_VBAT](#) selects between a VTR powered, or VBAT powered external 32kHz clock input on the 32KHZ\_IN pin ([Table 5-60](#)).

**TABLE 5-60:** [EXT32K\\_VBAT](#) BIT ENCODING

<a href="#">EXT32K_VBAT</a>	<b>Description</b>
0	VTR powered 32kHz input (default)
1	VBAT powered 32kHz input

# MEC1618/MEC1618i

## 32K\_EN

The [32K\\_EN](#) bit controls the [32.768 kHz Oscillator](#) as defined in [Table 5-61](#).

**TABLE 5-61: 32K\_EN BIT**

32K_EN	32.768 kHz Oscillator	Notes
0	OFF	<a href="#">VBAT_POR</a> (default)
1	ON	The <a href="#">32.768 kHz Oscillator</a> can only be enabled by firmware ( <a href="#">Note 5-49</a> ).

**Note 5-49** [MCLK](#) must not stop before 40  $\mu$ s min after the [32K\\_EN](#) bit is asserted.

## AD\_EN

The Activity Detector bit [AD\\_EN](#) controls [32 kHz Clock Domain Switching](#) as described in [Table 5-8](#).

## 5.9 Interrupt Interface

The [Power, Clocks, and Resets Interrupt Interface](#) inputs include the [VBAT\\_RST](#), [FLASH](#) and [WDT](#) status bits in the [Power-Fail and Reset Status Register](#). The [Interrupt Interface](#) output is [PCR\\_INT](#) in [Table 5-4](#), "Power, Clocks, and Resets Port List," on page 75. The corresponding bit in the [EC Interrupt Aggregator](#) is bit [PFR](#) in [GIRQ23 Source Register](#).

Whenever any [Interrupt Interface](#) input is asserted, [PCR\\_INT](#) is asserted; when all [Interrupt Interface](#) inputs are not asserted, [PCR\\_INT](#) is not asserted. [PCR\\_INT](#) may be masked as described in [Section 15.0](#), "EC Interrupt Aggregator," on page 296.

## 6.0 HOST INTERFACE

### 6.1 General Description

#### 6.1.1 OVERVIEW

The host processor communicates with the MEC1618/MEC1618i via the [LPC Bus Interface](#). The host processor communicates through a series of read/write registers in the MEC1618/MEC1618i. Register access is accomplished through programmed I/O or DMA LPC transfer cycles. All I/O transfer cycles are 8-bits wide. DMA transfer cycles can be 16-bit or 8-bit wide.

The Logical Devices physically located in the MEC1618/MEC1618i are identified in [Table 3-2, "Host Logical Devices in the MEC1618/MEC1618i," on page 51](#) and [Table 4-1, "Logical Devices," on page 56](#). The base addresses of logical devices with registers located in LPC I/O space, including the Keyboard Controller, can be moved via the configuration registers located in the LPC Interface Configuration Register Space.

All configuration register access for the MEC1618/MEC1618i are accessed indirectly through the LPC I/O Configuration Register Port (IOCR-Port.) The default I/O address is 2Eh and 2Fh, but the IOCR-Port can be relocated by either the host or the EC. Detailed description of the MEC1618/MEC1618i Configuration Space is in [Section 4.0, "Logical Device Configuration," on page 56](#).

All LPC transactions that are claimed by the MEC1618/MEC1618i are mapped by the LPC interface to an address in the MEC1618/MEC1618i's AHB address space. All these addresses can also be accessed by the Embedded Controller in the MEC1618/MEC1618i.

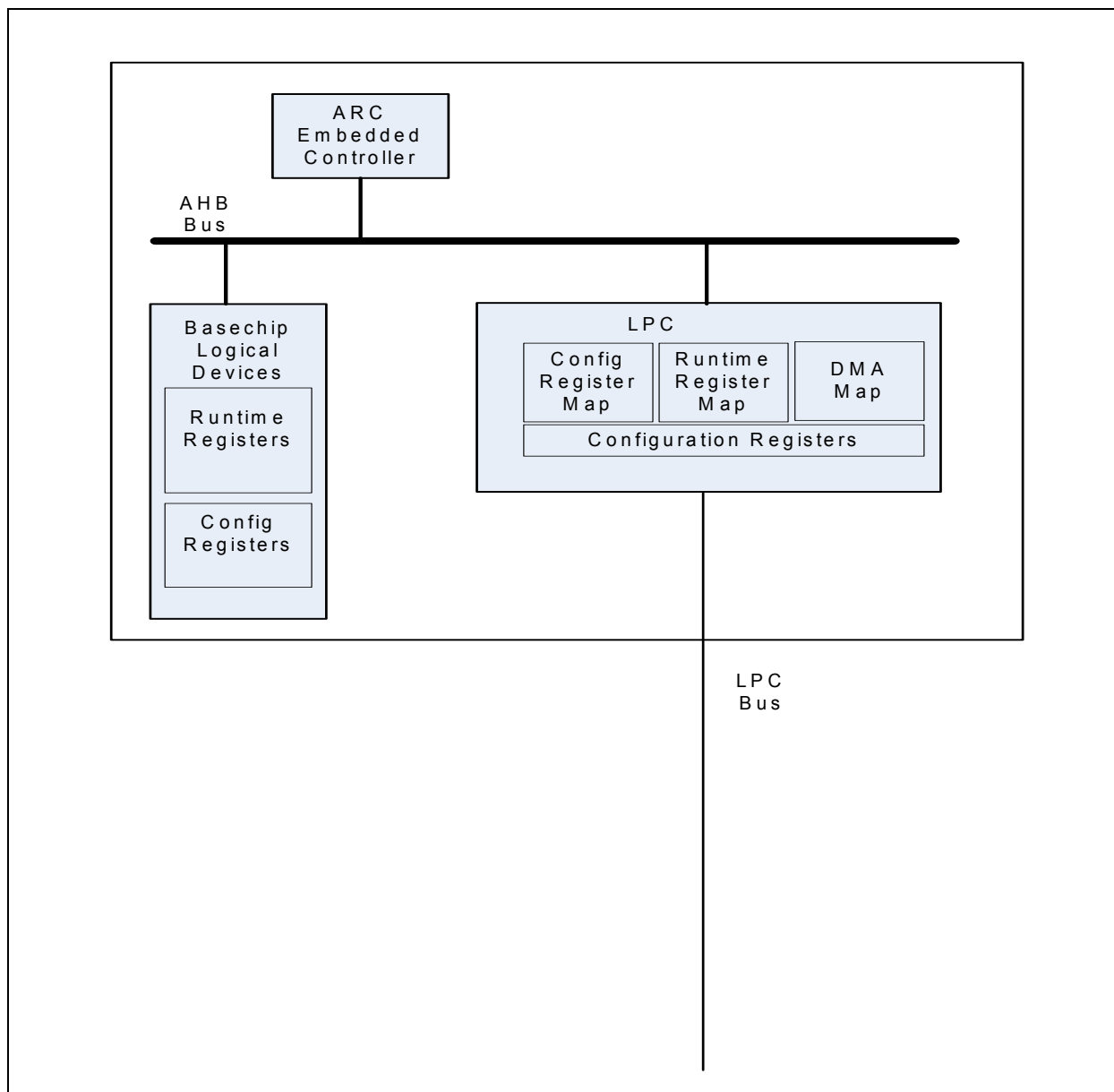
**TABLE 6-1: TARGETS OF LBC CYCLES CLAIMED BY THE MEC1618/MEC1618i**

Target	Acronym	Description	LPC Type
LPC IO Configuration Register Port	<b>IOCR-Port</b>	Standard LPC 2Eh/2Fh Port which permits BIOS access. This port can be relocated by the ARC.	I/O
Logical Devices	<b>LD</b>	Targets physically located in the MEC1618/MEC1618i. The Keyboard Controller Interface uses a Port at 60h/64h	DMA & I/O
Configuration Register	<b>CR</b>	256 byte space per Logical Device accessed by BIOS through the IOCR-Port.	I/O through CR-Port

# MEC1618/MEC1618i

## 6.1.2 BLOCK DIAGRAM

**FIGURE 6-1: LPC INTERFACE IN MEC1618/MEC1618i**



## 6.2 Power, Clocks and Resets

### 6.2.1 POWER DOMAIN

This block is powered by VTR. Although the block is not powered by VCC, the block is also controlled by VCC\_PWRGD. When VCC\_PWRGD is de-asserted, the LPC bus pins are placed in the same state they assume when VTR is off. LAD[3:0] and SERIRQ are tri-stated, LDRQ# is pulled high, CLKRUN# is unpowered and LFRAME#, LRESET# and LPCPD# are gated high; see [Table 6-2, "LPC Bus Pin Behavior on Reset," on page 139](#). The LPC block is also placed in a minimal power state.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.



## 6.2.2 CLOCKS

[LPC Logical Device Configuration Registers](#) and [LPC Logical Device EC-only Registers](#) in this block are clocked by the [LPC Bus Clock](#). The LPC interface itself is clocked by the [PCI\\_CLK](#) clock input.

The clock rate of the [LPC Bus Clock](#) runs at the [MCLK](#) rate.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

## 6.2.3 RESETS

This block is affected by [nSYS\\_RST](#), [VCC Power Good](#) and [LPC RESET](#).

The assertion of [nSYS\\_RST](#) resets the LPC state machine and all registers to their default values. The AHB Master state machine is also reset to its default value.

[VCC Power Good](#) going low resets the LPC state machine. The AHB Master interface that is part of the Host Interface will go to its idle state. Any transaction that is active on the AHB Master when the VCC POR occurs will be terminated in such a way that the AHB subsystem will not be locked up.

The assertion of [LPC RESET#](#) resets the LPC state machine but does not otherwise affect register values. An interrupt to the EC will be generated on either edge of [LRESET#](#). See [Section 6.5, "Host Interrupts to EC," on page 140](#)

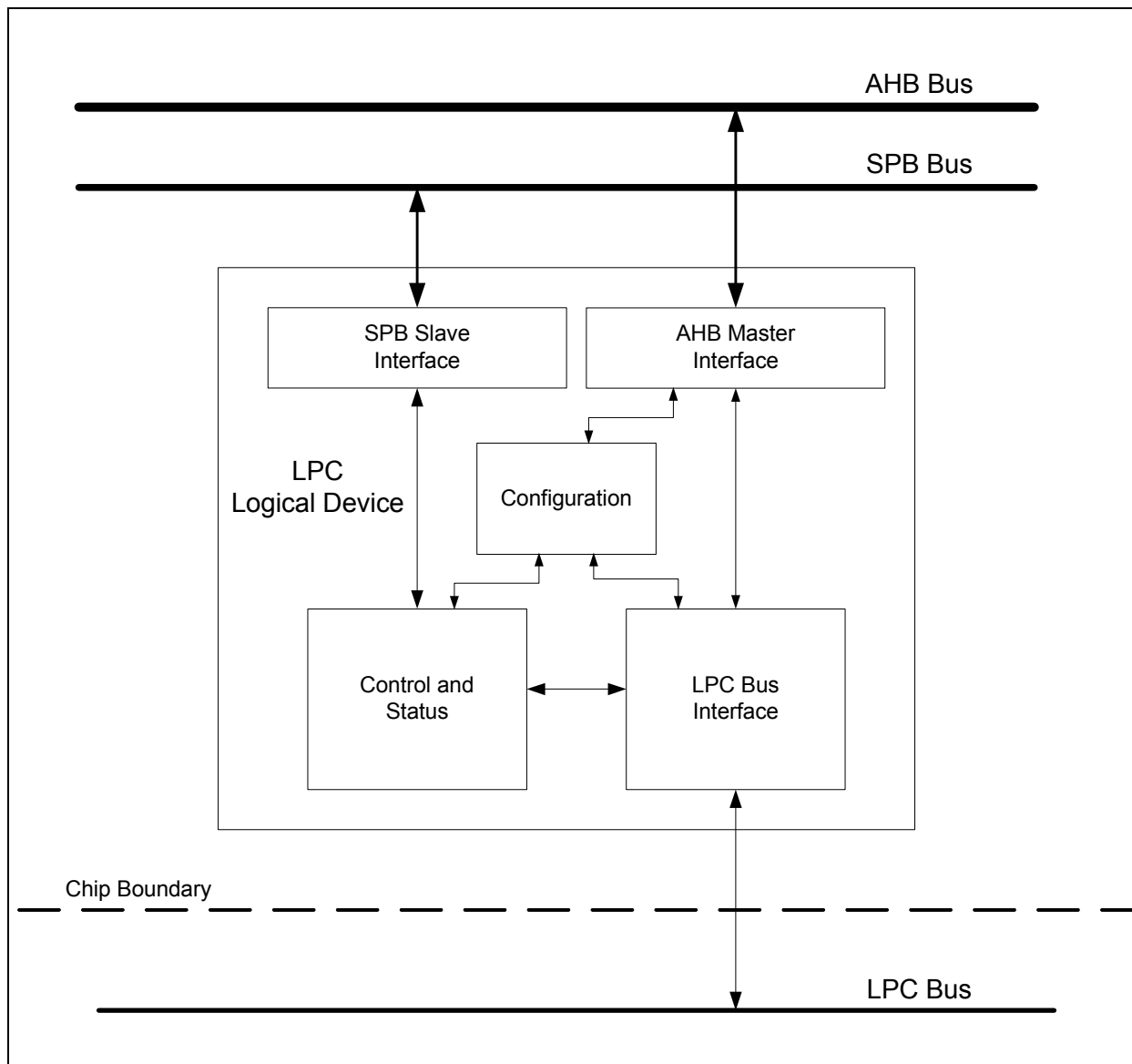
See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

# MEC1618/MEC1618i

## 6.3 LPC Logical Device

The LPC Logical Device structure is illustrated in Figure 6-1, "LPC Interface in MEC1618/MEC1618i".

**FIGURE 6-2: LPC LOGICAL DEVICE**



The LPC Logical Device is directly connected to two internal buses, the 32-bit AHB as well as the SPB. The SPB interface is 32-bits. In addition, it is connected to the external LPC bus.

Host accesses to [Configuration Registers](#) for each Logical Device on the MEC1618/MEC1618i are managed by a Configuration block described in [Section 4.0, "Logical Device Configuration," on page 56](#). Configuration registers are accessed through the LPC IO Configuration Register Port. The LPC Logical Device translates the Configuration address to an AHB address and the Host LPC access is converted into an AHB transaction inside the MEC1618/MEC1618i.

Host I/O accesses to [Configuring Runtime Register Addresses](#) as well as LPC DMA accesses are converted directly to AHB accesses. The LPC address is translated by the LPC Bus Interface to an AHB address inside the MEC1618/MEC1618i and the access becomes an access on the AHB bus.

## 6.3.1 LPC BUS INTERFACE

The MEC1618/MEC1618i communicates with the host over a Low Pin Count (LPC) interface. The LPC interface uses 3.3V signaling. For detailed specifications, see the *Intel Low Pin Count Specification* and the *PCI Local Bus Specification*, Section 4.2.2. The LPC Bus Interface is listed in [Table 3.7, "Host Interface," on page 57](#).

The following cycle types are supported by the LPC Bus protocol.

- 8-bit I/O Read
- 8-bit I/O Write
- 8-bit DMA Read (for Logical Devices which support 8-bit DMA)
- 8-bit DMA Write (for Logical Devices which support 8-bit DMA)
- 16-bit DMA Read (for Logical Devices which support 16-bit DMA)
- 16-bit DMA Write (for Logical Devices which support 16-bit DMA)

LPC transactions that claimed by the MEC1618/MEC1618i require a minimum of two wait SYNCs on the LPC bus. The number of SYNCs may be larger if the AHB bus is in use by the embedded controller, or if the data referenced by the host is not present in a MEC1618/MEC1618i register. The MEC1618/MEC1618i always uses Long Wait SYNCs, rather than Short Wait SYNCs, when responding to an LPC bus request.

[Table 6-2, "LPC Bus Pin Behavior on Reset"](#), shows the behavior of LPC outputs and input/outputs under reset conditions in accordance with the *Intel Low Pin Count Specification* and the *PCI Local Bus*. See [Section 3.4.1, "Pin Default State Through Power Transitions," on page 54](#) for Power transition pin state description and [Section 6.7, "LPC Clock Run and LPC Power Down Behavior," on page 140](#) for LPC protocol dependent pin state transitions requirements.

**TABLE 6-2: LPC BUS PIN BEHAVIOR ON RESET**

Pins	nSYS_RST	VCC POR	LPCPD# Asserted	LRESET# Asserted
LAD[3:0]	Tri-state	Tri-state	Tri-State	Tri-State
LDRQ#	Tri-state	De-asserted (high)	Tri-State	De-asserted (high)
SERIRQ	Tri-state	Tri-state	Tri-State	Tri-State
CLKRUN#	Tri-state	Tri-state	Tri-State	Tri-State

## 6.3.2 LPC I/O CYCLES

LPC 8-bit I/O Read cycles and 8-bit I/O Write cycles are mapped directly to addresses in the MEC1618/MEC1618i AHB address space. The mapping will be to the range FF\_0000h through FF\_FFFFh. For information on how addresses map between the LPC bus and the MEC1618/MEC1618i, see [Section 3.0, "Bus Hierarchy," on page 47](#) and [Section 4.0, "Logical Device Configuration," on page 56](#). For a list of all Configuration Registers accessible to the Host, see [Section 4.0, "Logical Device Configuration," on page 56](#).

## 6.3.3 LPC FIRMWARE HUB AND MEMORY CYCLES

The MEC1618/MEC1618i does not support LPC Firmware Hub cycles and LPC Memory cycles on the LPC Bus.

## 6.3.4 LPC DMA CYCLES

There are no DMA peripherals on the MEC1618/MEC1618i.

## 6.3.5 WAIT SYNC'S ON LPC

LPC cycles, claimed by the MEC1618/MEC1618i are completed with no more than **two** LPC Long WAIT SYNC's.

# MEC1618/MEC1618i

---

## 6.4 LPC Bus Configuration

The mapping from LPC Bus cycles to AHB read/write cycles is managed by the LPC Logical Device. The mapping is defined by a series of configuration registers which are defined in [Section 4.0, "Logical Device Configuration," on page 56](#), in [Section 6.4, "LPC Bus Configuration," on page 140](#).

## 6.5 Host Interrupts to EC

The LRESET# reset signal and the LPCPD# power down signal can be used to generate EC interrupts and wake-up events. The edge detection of the interrupt and wake events are controlled by their associated [Pin Control Register on page 406](#). The interrupts are routed to the LRESET# and LPCPD# bits in the [GIRQ14 Source Register on page 324](#).

The LPC Logical Device can generate an additional interrupt to the EC when a Host access is mapped to the AHB bus. Bit [LPC\\_AHB\\_ERR](#) in the [Host Bus Error Register](#) is set when an LPC-sourced AHB bus access causes an error; it is also routed to the [LPC\\_AHB\\_ERR](#) bit in the [GIRQ14 Source Register on page 324](#). For details see [Section 6.10.2, "Host Bus Error Register," on page 149](#).

## 6.6 EC Interrupts to Host

The Embedded Controller can send an interrupt to the Host on any Serial Interrupt Request channel using the [EC SERIRQ Register](#) in conjunction with the [SERIRQ Configuration Registers](#).

## 6.7 LPC Clock Run and LPC Power Down Behavior

The LPCPD# signal (see the *Intel Low Pin Count Specification*, Section 8.1) and the CLKRUN# signal (see the *Intel Low Pin Count Specification*, Section 8.2) are implemented in the MEC1618/MEC1618i.

### 6.7.1 USING LPCPD#

The MEC1618/MEC1618i tolerates the LPCPD# signal going active and then inactive again without LRESET# going active. This is a requirement for notebook power management functions.

The *LPC Bus Specification, Rev. 1.0*, Section 8.2 states that "After LPCPD# goes back inactive, the LPC I/F will always be reset using LRST#". This text must be qualified for mobile systems where it is possible that when exiting a "light" sleep state (ACPI S1, APM POS), LPCPD# may be asserted but the LPC Bus power may not be removed, in which case LRESET# will not occur. When exiting a "deeper" sleep state (ACPI S3-S5, APM STR, STD, soft-off), LRESET# will occur.

The LPCPD# pin is implemented as a "local" powergood for the LPC bus in the MEC1618/MEC1618i. It is not to be used as a global powergood for the chip. It is used to minimize the LPC power dissipation.

Prior to going to a low-power state, the system asserts the LPCPD# signal. LPCPD# goes active at least 30 microseconds prior to the LCLK signal stopping low and power being shut to the other LPC interface signals. Upon recognizing LPCPD# active, there are no further transactions on the LPC interface. The MEC1618/MEC1618i drives the LDRQ# signal tri-state, and does so until LPCPD# goes active. This prevents the MEC1618/MEC1618i from driving the signals high into a potentially powered-down host.

Upon recognizing LPCPD# inactive, The MEC1618/MEC1618i drives LDRQ# high.

### 6.7.2 USING CLKRUN#

CLKRUN# is used to indicate the PCI clock status as well as to request that a stopped clock be started. See [FIGURE 6-3: CLKRUN# System Implementation Example on page 142](#), an example of a typical system implementation using CLKRUN#.

The CLKRUN# signal in the MEC1618/MEC1618i also supports the LPC LDRQ# DMA protocol since PCI clock is also required to drive the LDRQ# signal active (See [Section 4.7.1, "DMA Configuration Registers," on page 63](#)). If an interrupt or DMA occurs while the PCI clock is stopped, CLKRUN# must be asserted before the interrupt or DMA can be serviced.

PCI Clock Run Support can be enabled and disabled using the [Bit\[2\] SerIRQ Mode](#) in the [Device Mode](#) register, Global Configuration Register [24h](#) (see [Table 4-15, "Chip-Level \(Global\) Control/Configuration Registers," on page 73](#)). When the [Bit\[2\] SerIRQ Mode](#) is '0,' Serial IRQs are disabled, the CLKRUN# pin is disabled, and the affects of Interrupt and DMA requests on CLKRUN# are ignored. When the [Bit\[2\] SerIRQ Mode](#) is '1,' Serial IRQs are enabled, the CLKRUN# pin is enabled, and the CLKRUN# support related to Interrupts and DMA requests as described in the section below is enabled.

The CLKRUN# pin is an open drain output and input. Refer to the *PCI Mobile Design Guide Rev 1.0* for a description of the CLKRUN# function. If CLKRUN# is sampled “high”, the PCI clock is stopped or stopping. If CLKRUN# is sampled “low”, the PCI clock is starting or started (running). CLKRUN# in the MEC1618/MEC1618i supports both Serial IRQ and LPC DMA cycles.

## 6.7.2.1 CLKRUN# Support for Serial IRQ Cycle

If a device in the MEC1618/MEC1618i asserts or de-asserts an interrupt and CLKRUN# is sampled “high”, the MEC1618/MEC1618i can request the restoration of the clock by asserting the CLKRUN# signal asynchronously (Table 6-3). The MEC1618/MEC1618i holds CLKRUN# low until it detects two rising edges of the clock. After the second clock edge, the MEC1618/MEC1618i must disable the open drain driver (Figure 6-4).

The MEC1618/MEC1618i must not assert CLKRUN# if it is already driven low by the central resource; i.e., the PCI CLOCK GENERATOR in Figure 6-4. The MEC1618/MEC1618i will not assert CLKRUN# under any conditions if the Serial IRQs are disabled.

The MEC1618/MEC1618i must not assert CLKRUN# unless the line has been de-asserted for two successive clocks; i.e., before the clock was stopped (Figure 6-4).

## 6.7.2.2 CLKRUN# Support for LPC DMA Cycle

If a device in the MEC1618/MEC1618i requests DMA service while the PCI clock is stopped, CLKRUN# must be asserted to restart the PCI clock so that the LDRQ# signal may be asserted (See Table 6-3). The MEC1618/MEC1618i will not assert CLKRUN# under any conditions if the SerIRQ\_Mode bit is inactive (“0”).

If a device in the MEC1618/MEC1618i asserts a DMA request and CLKRUN# is sampled “high”, the MEC1618/MEC1618i holds CLKRUN# low until it detects two rising edges of the PCI clock. After the second clock edge, the MEC1618/MEC1618i must disable the CLKRUN# open-drain driver (See Figure 6-4).

The MEC1618/MEC1618i will not assert CLKRUN# if it is already driven low by the central resource; i.e., the PCI CLOCK GENERATOR. The MEC1618/MEC1618i also will not assert CLKRUN# unless the signal has been de-asserted for two successive clocks; i.e., before the clock was stopped.

**TABLE 6-3: MEC1618/MEC1618i CLKRUN# FUNCTION**

SIRQ_MODE (Bit[2] SerIRQ Mode in Device Mode Register)	Internal Interrupt or DMA Request	CLKRUN#	Action
0	X	X	None
1	NO CHANGE	X	None
	CHANGE (Note 6-1)	0	None
		1	Assert CLKRUN#

**Note 6-1** “Change” means either-edge change on any or all parallel IRQs routed to the Serial IRQ block. “Assertion” means assertion of DMA request by a device in the MEC1618/MEC1618i. The “change” detection logic must run asynchronously to the PCI Clock and regardless of the Serial IRQ mode; i.e., “continuous” or “quiet”.

# MEC1618/MEC1618i

FIGURE 6-3: CLKRUN# SYSTEM IMPLEMENTATION EXAMPLE

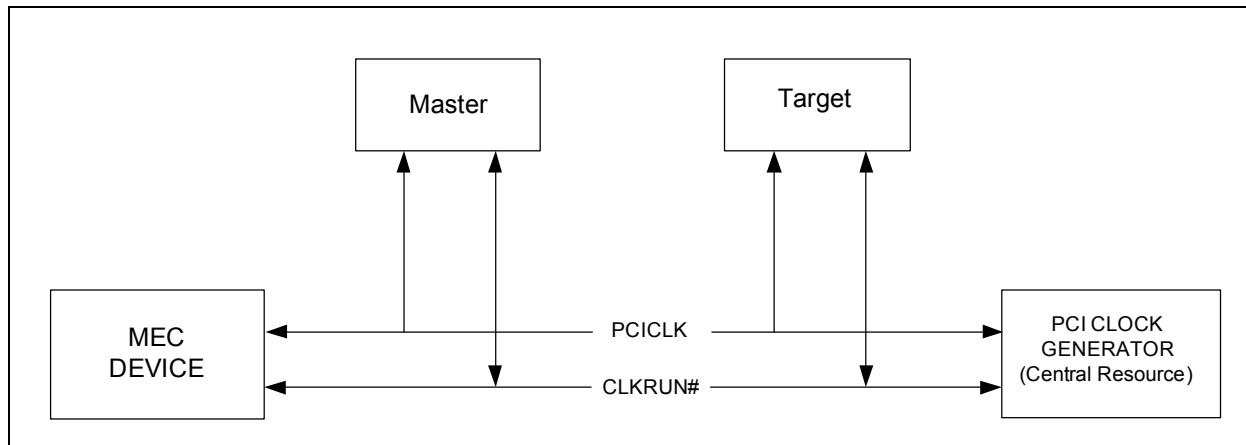
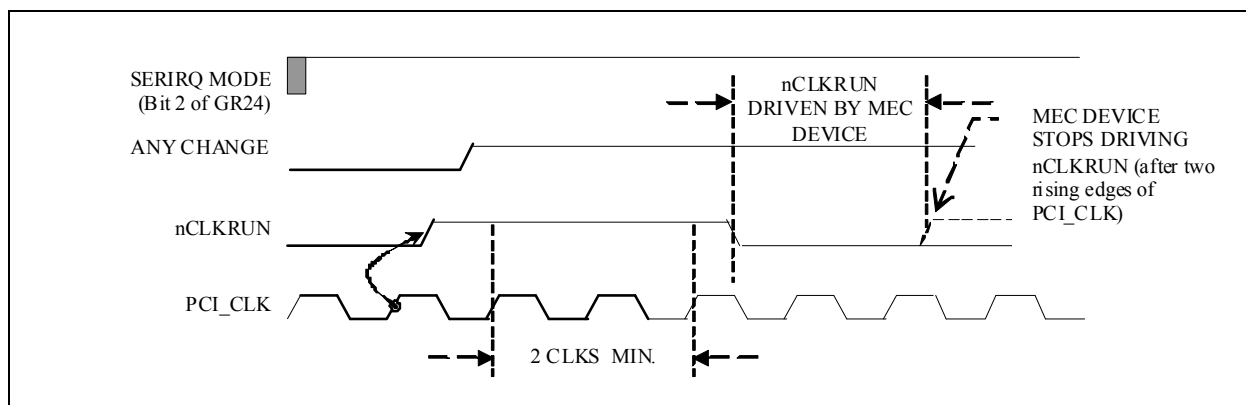


FIGURE 6-4: CLOCK START ILLUSTRATION



**Note:**

- The signal “ANY CHANGE” is the same as “CHANGE/ASSERTION” in [Table 6-3](#).
- The MEC1618/MEC1618i must continually monitor the state of CLKRUN# to maintain the PCI Clock until an active “any IRQ change” condition has been transferred to the host in a Serial IRQ cycle or “any DRQ assertion” condition has been transferred to the host in a DMA cycle. For example, if “any IRQ change or DRQ assertion” is asserted before CLKRUN# is de-asserted (not shown in [Figure 6-4](#)), the MEC1618/MEC1618i must assert CLKRUN# as needed until the Serial IRQ cycle or DMA cycle has completed.

## 6.8 Using Serial Interrupts

The MEC1618/MEC1618i will support the serial interrupt scheme, which is adopted by several companies, to transmit interrupt information to the system. The serial interrupt scheme adheres to the *Serial IRQ Specification for PCI Systems Version 6.0*.

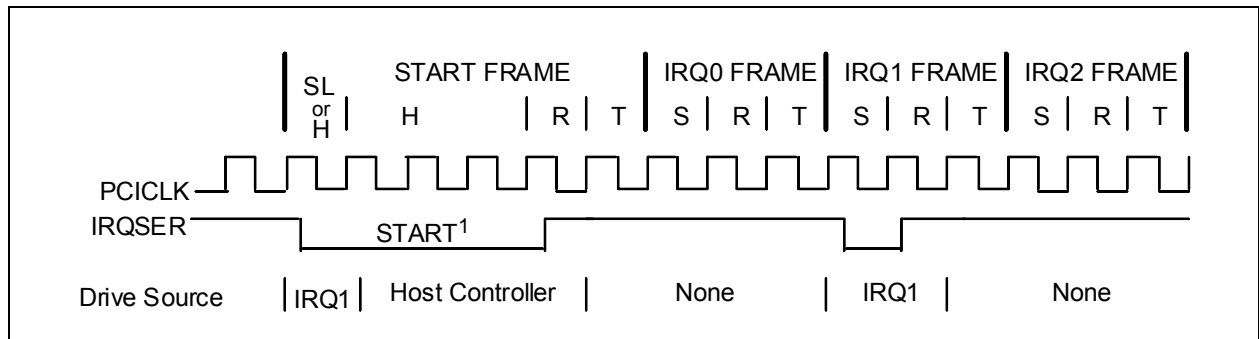
### TIMING DIAGRAMS FOR IRQSER CYCLE

PCICLK = 33 MHz\_IN pin

IRQSER = SIRQ pin

Start Frame timing with source sampled a low pulse on IRQ1.

**FIGURE 6-5: SERIAL INTERRUPTS WAVEFORM “START FRAME”**

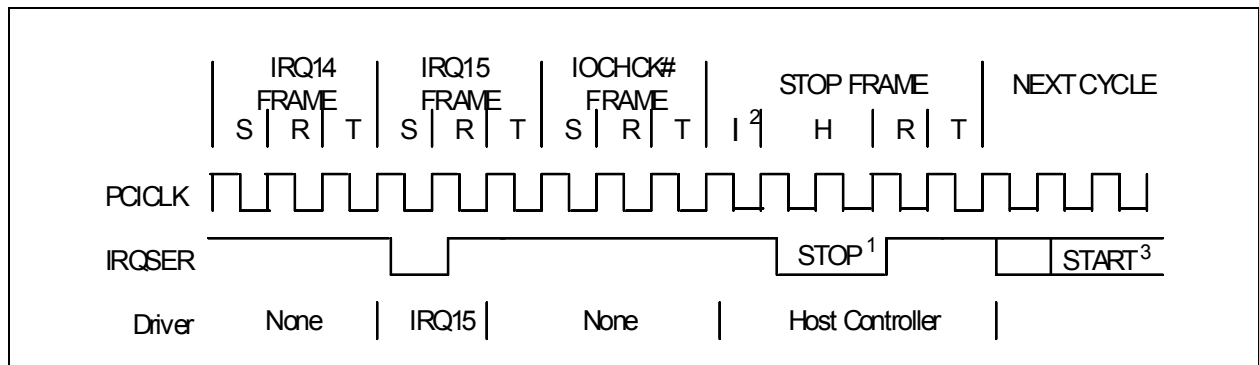


H=Host Control      SL=Slave Control      R=Recovery      T=Turn-around      S=Sample

Start Frame pulse can be 4-8 clocks wide.

Stop Frame Timing with Host using 17 IRQSER sampling period.

**FIGURE 6-6: SERIAL INTERRUPT WAVEFORM “STOP FRAME”**



H=Host Control      R=Recovery      T=Turn-around      S=Sample      I= Idle

Stop pulse is two clocks wide for Quiet mode, three clocks wide for Continuous mode.

There may be none, one, or more Idle states during the Stop Frame.

The next IRQSER cycle's Start Frame pulse may or may not start immediately after the turn-around clock of the Stop Frame.

# MEC1618/MEC1618i

## 6.8.1 SERIRQ MODE BIT FUNCTION

**TABLE 6-4: SERIRQ\_EN CONFIGURATION CONTROL**

CR25 BIT[2]	Name	Description
0	SERIRQ_EN	Serial IRQ Disabled
1		Serial IRQ Enabled (Default)

### 6.8.1.1 IRQSER Cycle Control

There are two modes of operation for the IRQSER Start Frame.

#### QUIET (ACTIVE) MODE

Any device may initiate a Start Frame by driving the IRQSER low for one clock, while the IRQSER is Idle. After driving low for one clock, the IRQSER must immediately be tri-stated without at any time driving high. A Start Frame may not be initiated while the IRQSER is active. The IRQSER is Idle between Stop and Start Frames. The IRQSER is active between Start and Stop Frames. This mode of operation allows the IRQSER to be Idle when there are no IRQ/Data transitions which should be most of the time.

After a Start Frame has been initiated, the host controller will take over driving the IRQSER low in the next clock and will continue driving the IRQSER low for a programmable period of three to seven clocks. This makes a total low pulse width of four to eight clocks. Finally, the host controller will drive the IRQSER back high for one clock then tri-state.

Any IRQSER Device (i.e., The MEC1618/MEC1618i) which detects any transition on an IRQ/Data line for which it is responsible must initiate a Start Frame in order to update the host controller unless the IRQSER is already in an IRQSER Cycle and the IRQ/Data transition can be delivered in that IRQSER Cycle.

#### CONTINUOUS (IDLE) MODE

Only the Host controller can initiate a Start Frame to update IRQ/Data line information. All other IRQSER agents become passive and may not initiate a Start Frame. IRQSER will be driven low for four to eight clocks by host controller. This mode has two functions. It can be used to stop or idle the IRQSER or the host controller can operate IRQSER in a continuous mode by initiating a Start Frame at the end of every Stop Frame.

An IRQSER mode transition can only occur during the Stop Frame. Upon reset, IRQSER bus is defaulted to continuous mode, therefore only the host controller can initiate the first Start Frame. Slaves must continuously sample the Stop Frames pulse width to determine the next IRQSER Cycle's mode.

#### IRQSER DATA FRAME

After a Start Frame has been initiated, the MEC1618/MEC1618i will watch for the rising edge of the Start Pulse and start counting IRQ/Data Frames from there. Each IRQ/Data Frame is three clocks: Sample phase, Recovery phase, and Turn-around phase. During the sample phase, the MEC1618/MEC1618i must drive the IRQSER (SIRQ pin) low, if and only if, its last detected IRQ/Data value was low. If its detected IRQ/Data value is high, IRQSER must be left tri-stated. During the recovery phase, the MEC1618/MEC1618i must drive the SERIRQ high, if and only if, it had driven the IRQSER low during the previous sample phase. During the turn-around phase, the MEC1618/MEC1618i must tri-state the SERIRQ. The MEC1618/MEC1618i drives the IRQSER line low at the appropriate sample point if its associated IRQ/Data line is low, regardless of which device initiated the start frame.

The Sample phase for each IRQ/Data follows the low to high transition of the Start Frame pulse by a number of clocks equal to the IRQ/Data Frame times three, minus one e.g. The IRQ5 Sample clock is the sixth IRQ/Data Frame, then the sample phase is  $\{(6 \times 3) - 1 = 17\}$  the seventeenth clock after the rising edge of the Start Pulse.



**TABLE 6-5: IRQSER SAMPLING PERIODS**

IRQSER Period	Signal Sampled	# of Clocks Past Start
1	Not Used	2
2	IRQ1	5
3	nSMI/IRQ2	8
4	IRQ3	11
5	IRQ4	14
6	IRQ5	17
7	IRQ6	20
8	IRQ7	23
9	IRQ8	26
10	IRQ9	29
11	IRQ10	32
12	IRQ11	35
13	IRQ12	38
14	IRQ13	41
15	IRQ14	44
16	IRQ15	47

The SIRQ data frame will now support IRQ2 from a logical device; previously IRQSER Period 3 was reserved for use by the System Management Interrupt (nSMI). When using Period 3 for IRQ2, the user should mask off the MEC1618/MEC1618i's SMI via the ESMI Mask Register. Likewise, when using Period 3 for nSMI, the user should not configure any logical devices as using IRQ2.

IRQSER Period 14 is used to transfer IRQ13. Each Logical devices will have IRQ13 as a choice for their primary interrupt.

## STOP CYCLE CONTROL

After all IRQ/Data Frames have completed, the host controller will terminate IRQSER activity by initiating a Stop Frame. Only the host controller can initiate the Stop Frame. A Stop Frame is indicated when the IRQSER is low for two or three clocks. If the Stop Frame's low time is two clocks, then the next IRQSER cycle's sampled mode is the Quiet mode; and any IRQSER device may initiate a Start Frame in the second clock or more after the rising edge of the Stop Frame's pulse. If the Stop Frame's low time is three clocks, then the next IRQSER cycle's sampled mode is the continuous mode, and only the host controller may initiate a Start Frame in the second clock or more after the rising edge of the Stop Frame's pulse.

## LATENCY

Latency for IRQ/Data updates over the IRQSER bus in bridge-less systems with the minimum IRQ/Data Frames of 17 will range up to 96 clocks (3.84 $\mu$ S with a 25 MHz PCI Bus or 2.88 $\mu$ S with a 33 MHz PCI Bus). If one or more PCI to PCI Bridge is added to a system, the latency for IRQ/Data updates from the secondary or tertiary buses will be a few clocks longer for synchronous buses, and approximately double for asynchronous buses.

# MEC1618/MEC1618i

## EOI/ISR READ LATENCY

Any serialized IRQ scheme has a potential implementation issue related to IRQ latency. IRQ latency could cause an EOI or ISR Read to precede an IRQ transition that it should have followed. This could cause a system fault. The host interrupt controller is responsible for ensuring that these latency issues are mitigated. The recommended solution is to delay EOIs and ISR Reads to the interrupt controller by the same amount as the IRQSER Cycle latency in order to ensure that these events do not occur out of order.

## AC/DC SPECIFICATION ISSUE

All IRQSER agents must drive/sample IRQSER synchronously related to the rising edge of the PCI bus clock. The IRQSER (SIRQ) pin uses the electrical specification of the PCI bus. Electrical parameters will follow the PCI Specification Section 4, sustained tri-state.

## RESET AND INITIALIZATION

The IRQSER bus uses LRESET as its reset signal and follows the PCI bus reset mechanism. The IRQSER pin is tri-stated by all agents while LRESET is active. With reset, IRQSER slaves and bridges are put into the (continuous) Idle mode. The host controller is responsible for starting the initial IRQSER cycle to collect system's IRQ/Data default values. The system then follows with the Continuous/Quiet mode protocol (Stop Frame pulse width) for subsequent IRQSER cycles. It is the host controller's responsibility to provide the default values to the 8259's and other system logic before the first IRQSER cycle is performed. For IRQSER system suspend, insertion, or removal application, the host controller should be programmed into Continuous (IDLE) mode first. This is to guarantee the IRQSER bus is in Idle state before the system configuration changes.

## 6.9 LPC Logical Device Configuration Registers

The configuration registers in the LPC Logical Device are described in [Section 4.0, "Logical Device Configuration," on page 56](#). These registers control the activity of all the Logical Devices in the MEC1618/MEC1618i. The [Activate Register](#) controls the LPC device itself. The Host can shut down the LPC Logical Device by clearing the Activate bit, but it cannot restart the LPC interface, since after the LPC interface is inactive the Host has no access to any registers on the MEC1618/MEC1618i. The Embedded Controller can set or clear the Activate bit at any time.

**TABLE 6-6:     ACTIVATE REGISTER**

HOST OFFSET	BYTE0: 30h			8-bit			HOST SIZE	
EC ADDRESS	FF_3330h			8-bit			EC SIZE	
POWER	VTR			00b			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R/W
EC TYPE	R	R	R	R	R	R	R	R/W
BIT NAME	Reserved							Activate

## ACTIVATE

When this bit is 1, the LPC Logical Device is powered and functional.

When this bit is 0, the logical device is powered down and inactive. Except for the [Activate Register](#) itself, clocks to the block are gated and the LPC Logical Device will permit the ring oscillator to be shut down (see [Section 6.10.4, "EC Clock Block Register," on page 151](#)). LPC bus output pads will be tri-stated. Serial IRQ activation is separately controlled by the [Device Mode](#) register in the [Chip-Level \(Global\) Control/Configuration Registers](#).

**APPLICATION NOTE:** The [Activate](#) bit in the [Activate Register](#) should not be written '0' to by the Host over LPC.

## 6.10 LPC Logical Device EC-only Registers

Table 6-7, "LPC EC-only Registers" summarizes the registers in the Host Interface block that are only accessible by the EC. In addition to these registers, the Host Interface also contains Configuration registers, described in [Section 4.0](#), "Logical Device Configuration," on page 56.

**TABLE 6-7: LPC EC-ONLY REGISTERS**

AHB Address	Name	VTR POR ( <a href="#">nSYS_RST</a> ) Default
FF_3100h	Reserved	0000_0000h
FF_3104h	<a href="#">LPC Bus Monitor Register</a>	0000_0000h
FF_3108h	<a href="#">Host Bus Error Register</a>	0000_0000h
FF_310Ch	<a href="#">EC SERIRQ Register</a>	0000_0000h
FF_3110h	<a href="#">EC Clock Control Register</a>	0000_0000h
FF_3120h	<a href="#">BAR Inhibit Register</a>	0000_0000_0000_0000h

Because their addresses are in the part of the LPC Logical Device address frame that is not addressable from the LPC bus, the following registers are accessible only to the EC.

### 6.10.1 LPC BUS MONITOR REGISTER

**TABLE 6-8: LPC BUS MONITOR REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	FF_3104h			32-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:1] BIT	D31	D30	D29	...		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved						LRESET_ Status	LPCPD_ Status

## LPCPD\_STATUS

This bit reflects the state of the LPCPD# input pin. The LPCPD\_Status is the inverse of the LPCPD# pin (see [Section 6.7, "LPC Clock Run and LPC Power Down Behavior," on page 140](#)).

When the LPCPD\_Status bit is '0b', the LPCPD# input pin is de-asserted (that is, the pin has the value '1b'). When the LPCPD\_Status bit is '1b', the LPCPD# input pin is asserted (that is, the pin has the value '0b').

An interrupt to the EC will be generated on either edge of LPCPD#. See [Section 6.5, "Host Interrupts to EC," on page 140](#).

## LRESET\_STATUS

This bit reflects the state of the LRESET# input pin. The LRESET\_Status is the inverse of the LRESET# pin (see [Section 6.2.3, "Resets," on page 137](#)).

When the LRESET\_Status bit is '0b', the LRESET# input pin is de-asserted (that is, the pin has the value '1b'). When the LRESET\_Status bit is '1b', the LRESET# input pin is asserted (that is, the pin has the value '0b').

An interrupt to the EC will be generated on either edge of LRESET#. See [Section 6.5, "Host Interrupts to EC," on page 140](#).

## 6.10.2 HOST BUS ERROR REGISTER

**TABLE 6-9: HOST BUS ERROR REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	FF_3108h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	ErrorAddress[23:16]							
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	ErrorAddress[15:8]							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	ErrorAddress[7:0]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/WC	R/WC	R/WC	R/WC	R/WC	R/W	R/WC
BIT NAME	Reserved	Reserved	DMA Err	Config Err	Runtime Err	BAR_Conflict	En_AHB_Err	LPC_AHB_Err

### LPC\_AHB\_ERR

This bit can be used to generate an EC interrupt. It is set whenever either a BAR conflict or an AHB bus error occurs as a result of an LPC access. After being set, it remains set until cleared by being written with a 1.

### EN\_AHB\_ERR

When this bit is 0, only a BAR conflict, which occurs when two BARs match the same LPC I/O address, will cause **LPC\_AHB\_ERR** to be set. When this bit is 1, AHB bus errors will also cause **LPC\_AHB\_ERR** to be set.

# MEC1618/MEC1618i

## BAR\_CONFLICT

This bit is set to 1 whenever a BAR conflict occurs on an LPC address. A Bar conflict occurs when more than one BAR matches the address during of an LPC cycle access. After this bit is set, it remains set until cleared by being written with a 1.

## RUNTIME\_ERR

This bit is set to 1 whenever [En\\_AHB\\_ERR](#) is 1 and an LPC I/O access causes an AHB bus error. This error will only occur if a BAR is misconfigured. After being set, it remains set until cleared by being written with a 1.

## CONFIG\_ERR

This bit is set to 1 whenever [En\\_AHB\\_ERR](#) is 1 and an LPC Configuration access causes an AHB bus error. After being set, it remains set until cleared by being written with a 1.

## DMA\_ERR

This bit is set to 1 whenever [En\\_AHB\\_ERR](#) is 1 and an LPC DMA access causes an AHB bus error. After being set, it remains set until cleared by being written with a 1.

## ERRORADDRESS

This 24-bit field captures the 24-bit AHB address of every LPC transaction whenever the bit [LPC\\_AHB\\_ERR](#) in this register is 0. When [LPC\\_AHB\\_ERR](#) is 1 this register is not updated but retains its previous value. When bus errors occur this field saves the address of the first address that caused an error.

### 6.10.3 EC SERIRQ REGISTER

TABLE 6-10: EC SERIRQ REGISTER

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	FF_310Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:1] BIT	D31	D30	D29	...		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R/W
BIT NAME	Reserved							EC_IRQ

## EC\_IRQ

If the LPC Logical Device is selected as the source for a Serial Interrupt Request by an Interrupt Configuration register (see [Section 4.8, "SERIRQ Interrupts," on page 65](#)), this bit is used as the interrupt source.

## 6.10.4 EC CLOCK CONTROL REGISTER

**TABLE 6-11: EC CLOCK CONTROL REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	FF_3110h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:1] BIT	D31	D30	D29	...		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R/W	R/W
BIT NAME	Reserved						Clock_Control	

### CLOCK\_CONTROL

This field controls when the host interface will permit the internal ring oscillator to be shut down. The choices are as follows:

#### 0h:

The host interface will permit the ring oscillator to be shut down if the LPCPD# signal is asserted (sampled low)

#### 1h:

The host interface will permit the ring oscillator to be shut down if the CLKRUN# signals "CLOCK STOP" and there are no pending serial interrupt request or DMA requests from devices associated with the MEC1618/MEC1618i. The CLKRUN# signals "CLOCK STOP" by CLKRUN# being high for 5 LPCCLK's after the raising edge of CLKRUN#.

#### 2h:

The host interface will permit the ring oscillator to be shut down after the completion of every LPC transaction. It will require the ring oscillator as soon as a START field appears on the LPC bus. This mode may cause an increase in the time to respond to LPC transactions if the ring oscillator has to turn on and adjust its speed after the LPC transaction arrives at the MEC1618/MEC1618i

#### 3h:

The ring oscillator is not permitted to shut down as long as the host interface is active

When the [Activate](#) bit in the [Activate Register on page 146](#) is 0, the Host Interface will permit the ring oscillator to be shut down and the [Clock\\_Control](#) Field is ignored. The [Clock\\_Control](#) Field only effects the Host Interface when the [Activate](#) bit in the [Activate Register](#) is 1.

# MEC1618/MEC1618i

## 6.10.5 BAR INHIBIT REGISTER

TABLE 6-12: BAR INHIBIT REGISTER

HOST OFFSET	n/a				n/a		HOST SIZE	
EC OFFSET	120h				64-bit		EC SIZE	
POWER	VTR				0000_0000_0000_0000h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D63	D62	D61	...		D2	D1	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	BAR_Inhibit[63:0]							

### BAR\_INHIBIT[63:0]

When bit *i* of the [BAR\\_Inhibit\[63:0\]](#) field is asserted ('1'), where *i* is the logical device number of one of the [Host Logical Devices in the MEC1618/MEC1618i](#), the BAR for the associated device is disabled and its addresses will not be claimed on the LPC bus, independent of the value of the Valid bit in the BAR. When bit *i* is not asserted (default), BAR activity for the Logical Device is based on the Valid bit in the BAR.

All of the [BAR\\_Inhibit\[63:0\]](#) bits are R/W and have no affect on reserved logical device numbers. The [Host Logical Devices in the MEC1618/MEC1618i](#) are shown in [Table 3-2](#) in [Section 3.3.2, "AHB Address Space,"](#) on page 51.



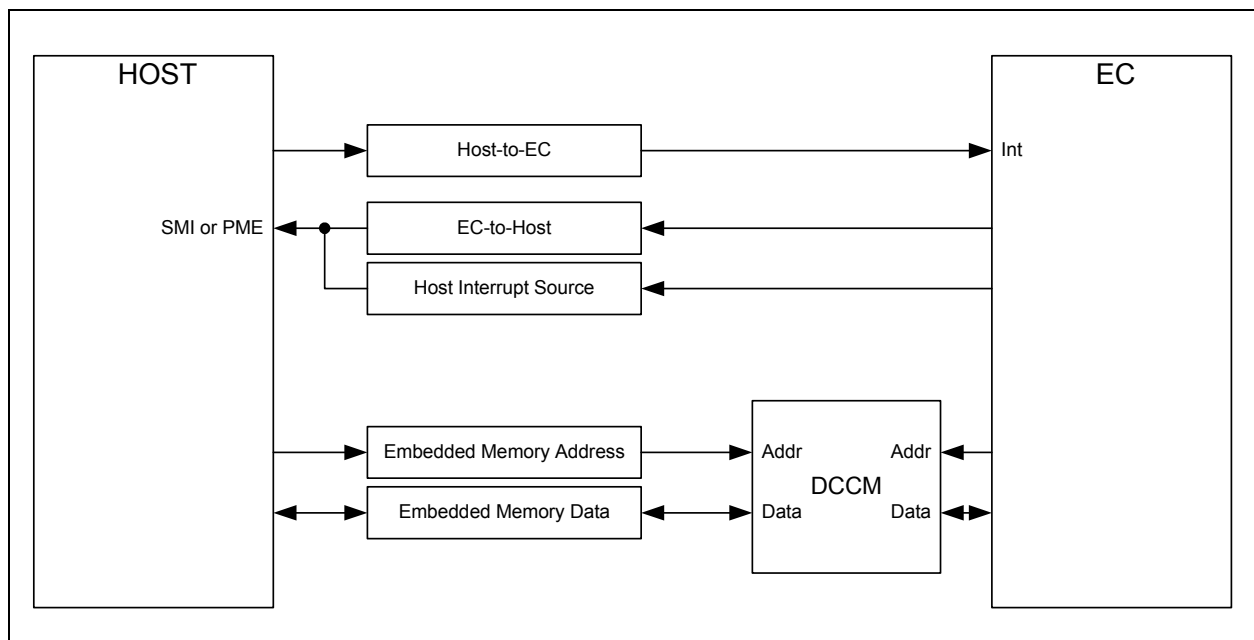
## 7.0 EMBEDDED MEMORY INTERFACE

### 7.1 General Description

The [Embedded Memory Interface](#) provides a standard run-time mechanism for the host to communicate with the Embedded Controller (EC) and other logical components in the MEC1618/MEC1618i ([Figure 7-1](#)). The Embedded Memory Interface includes 12 byte-addressable registers in the Host's I/O address space, as well as 20 bytes of registers that are accessible only by the EC. The Embedded Memory Interface can be used by the Host to read any byte in a region of EC closely-coupled memory, designated by the EC, without requiring any assistance from the EC. A portion of the memory can be written by the Host without any EC assistance as well.

#### 7.1.1 BLOCK DIAGRAM

**FIGURE 7-1: EMBEDDED MEMORY INTERFACE BLOCK DIAGRAM**



### 7.2 Power, Clocks and Reset

#### 7.2.1 POWER DOMAIN

This block is powered by the VTR power supply.

#### 7.2.2 CLOCKS

This block has two clock inputs, the [LPC Bus Clock](#) and the EC clock.

The [Embedded Memory Interface](#) includes support for system-level clock gating. The clock required output is the inversion of the sleep enable input.

#### 7.2.3 RESET

This block is reset when [nSYS\\_RST](#) is asserted.

## 7.3 Interrupts

Each instance of the [Embedded Memory Interface](#) can generate an interrupt event for the HOST-to-EC events. See [HOST-to-EC Mailbox Register on page 160](#). The interrupt source for the EMI is routed onto the [EM\\_MBX](#) bit in the [GIRQ15 Source Register](#) and is edge-sensitive, active high.

When the [EM\\_MBX](#) interrupt status bit is cleared while there is data in the [HOST-to-EC Mailbox Register](#), the interrupt remains not asserted until the host writes another byte. If the host writes a second byte to the [HOST-to-EC Mailbox Register](#) before the EC processes the [EM\\_MBX](#) interrupt resulting from the first byte, the data from the first byte will be overwritten by the second before the EC has a chance to process the first. This can be avoided with a proper hand-shaking protocol between the host and EC (for example, the EC should process a data in the [HOST-to-EC Mailbox Register](#) before signaling the host that the data register is free).

### 7.3.1 EMBEDDED MEMORY INTERFACE SIRQ ROUTING

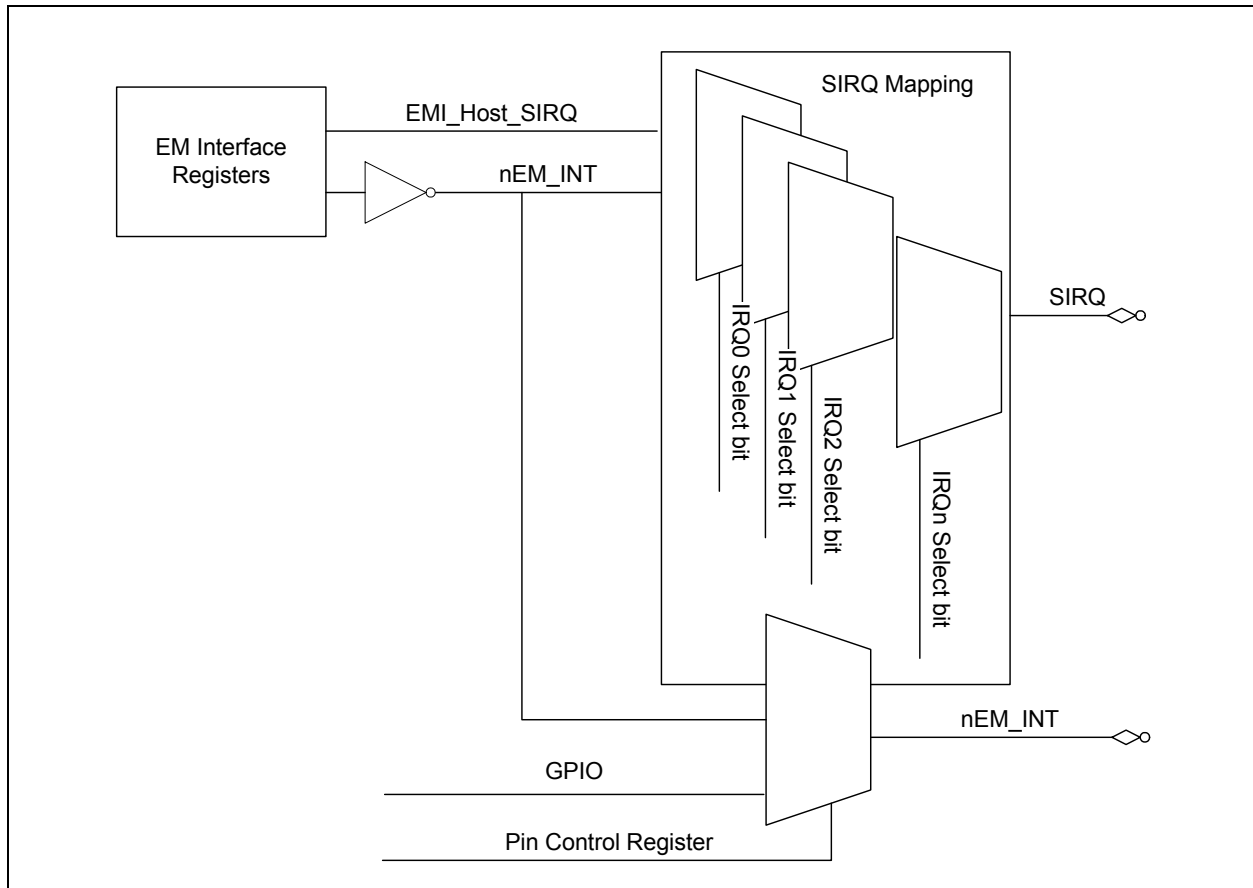
The [Embedded Memory Interface](#) can generate a SIRQ event for the EC-to-HOST EC events. See [HOST-to-EC Mailbox Register on page 160](#) and [Interrupt Source Register on page 166](#). This interrupt is routed to the SIRQ block (see [SERIRQ Configuration Registers on page 65](#)). For this interrupt, the [SELECT on page 66](#) is cleared to '0' in the Interrupt Configuration Register for the selected SIRQ frame.

The [Embedded Memory Interface](#) can also generate an event on an external pin, [nEM\\_INT](#). The pin can be routed to SMI or PME inputs, as required. The EC can cause the event to be generated by either writing the [EC-to-Host Mailbox Register](#) or by setting any of the enabled [EC\\_SWI\[14:0\]](#) bits in the [Interrupt Source Register](#) to 1. The event can be routed to any frame in the SIRQ stream or to the external pin. To enable routing to the SIRQ stream, the bit [SELECT on page 66](#) is set to '1' in the Interrupt Configuration Register for the selected SIRQ frame. The event can be routed to the pin by selecting the [nEM\\_INT](#) function in the associated [Pin Control Register on page 406](#).

The event produces a standard active low on the serial IRQ stream and active low on the open drain [nEM\\_INT](#) pin. See [FIGURE 7-2: Embedded Memory Interface SIRQ and nEM\\_INT routing on page 155](#).

See [Section 4.8.1, "SERIRQ Configuration Registers," on page 65](#).

**FIGURE 7-2: EMBEDDED MEMORY INTERFACE SIRQ AND NEM\_INT ROUTING**



## 7.4 Description

The Embedded Memory Interface contains a Mailbox that enables the Host to send an 8-bit message to the EC and the EC to send an 8-bit message to the Host. When written by the sender, the messages can generate an interrupt at the receiver.

In addition to the messages that can be exchanged, the Embedded Memory Interface permits the Host to read and write a portion of the EC's Data Closely Coupled Memory (DCCM). Host reads and writes take place without intervention or assistance from the EC.

The Embedded Memory Interface occupies 12 bytes in the Host I/O space. Two bytes constitute the Host-to-EC and EC-to-Host message links. Six bytes are used for the interface into the EC DCCM, two for address and four for data. The four data bytes are used for reads and writes to the EC DCCM DMI.

When the Host reads one of the four bytes in the Embedded Memory Interface data register, data from the DCCM at the address defined by the Embedded Memory Interface address register is returned to the Host. Writes to a byte write the corresponding byte in the DCCM. The Embedded Memory Interface can be configured so that, although Host I/O is always byte at a time, transfers between the Embedded Memory Interface data bytes and the DCCM can be configured to occur as single bytes, 2-byte blocks or 4-byte blocks. This is done so that data that the EC treats as 16-bit or 32-bit will be consistent in the Host, even though one byte of the DCCM data may change between two or more 8-bit accesses by the Host.

In addition, there is an auto-increment function for the Embedded Memory Interface address register. When enabled, the Host can read or write blocks of memory in the DCCM by repeatedly accessing the Embedded Memory Interface data register, without requiring Host updates to the Embedded Memory Interface address register.

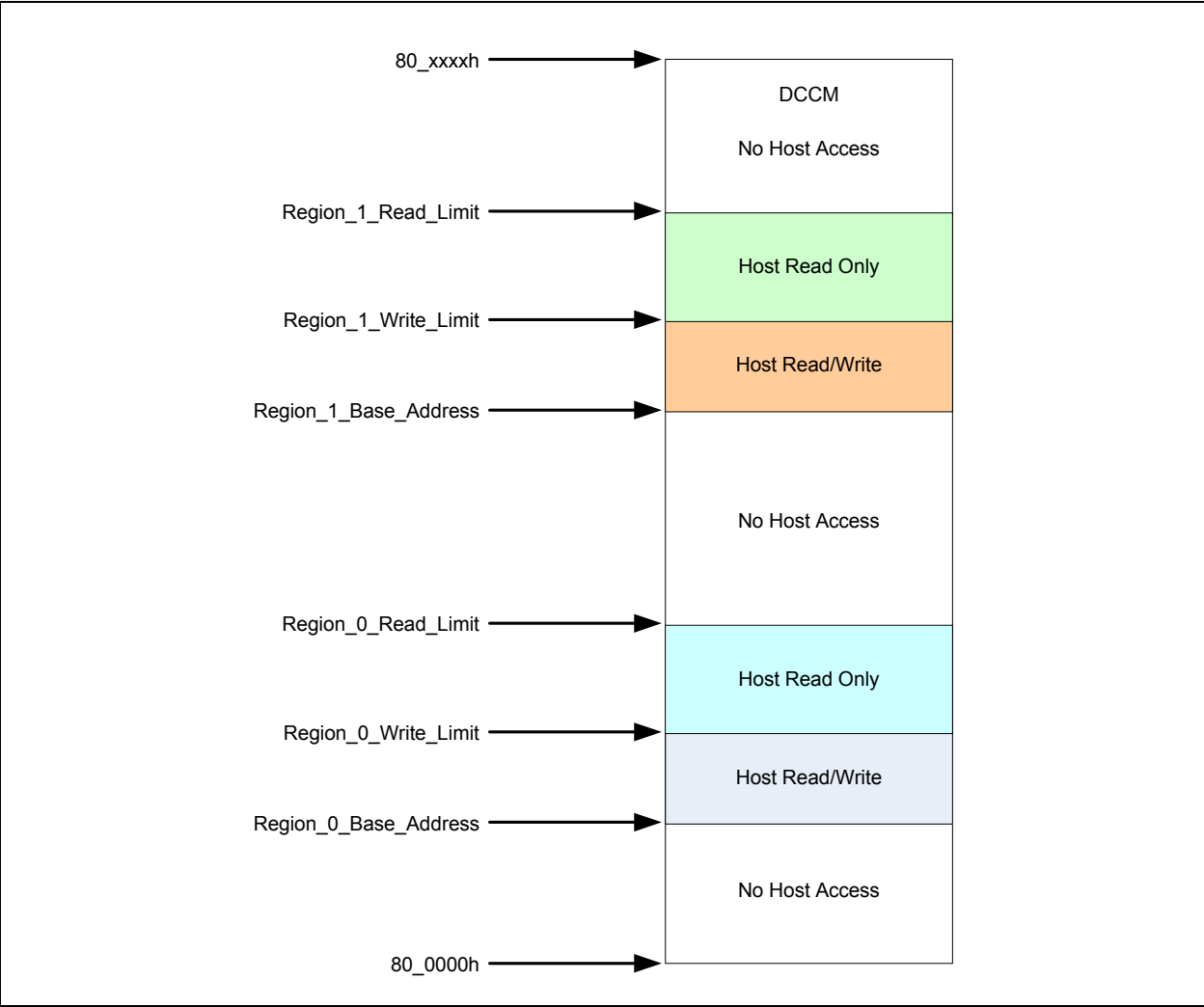
# MEC1618/MEC1618i

**APPLICATION NOTE:** the [RAM\\_Select](#) bit in the [AHB SRAM Configuration Register](#) must be asserted ('1') to properly configure the DCCM for use by the [Embedded Memory Interface](#).

## 7.4.1 EMBEDDED MEMORY MAP

Each Embedded Memory interface provides direct access for the Host into two windows in the EC DCCM SRAM. This mapping is shown in Figure 7-3, "Embedded Memory Addressing":

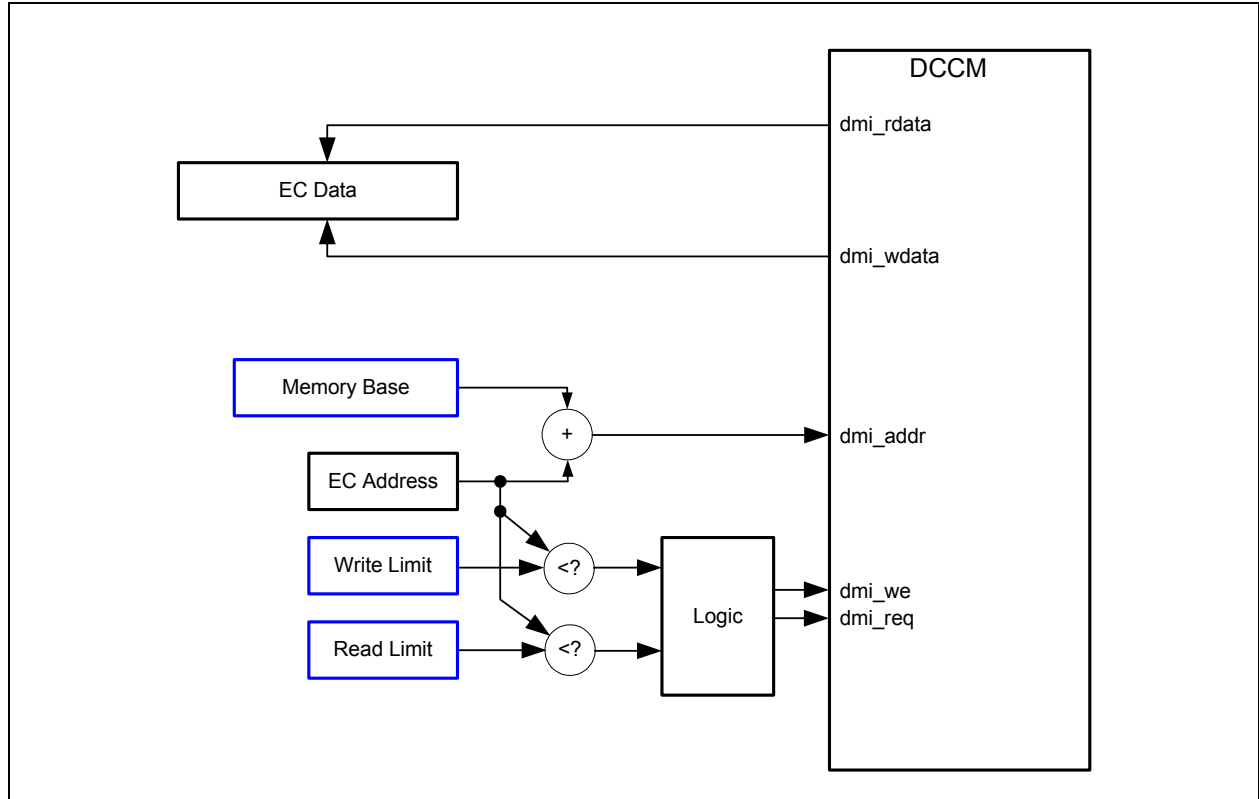
**FIGURE 7-3: EMBEDDED MEMORY ADDRESSING**



The Base addresses, the Read limits and the Write limits are defined by registers that are in the EC address space and cannot be accessed by the Host. In each region, the Read limit need not be greater than the Write limit. The regions can be contiguous or overlapping. For example, if the Region 0 Read limit is set to 0 and the Write limit is set to a positive number, then the Embedded Memory interface defines a region in the EC memory that the EC can read and write but is write-only for the host. This might be useful for storage of security data, which the Host might wish to send to the EC but should not be readable in the event a virus invades the Host.

Each window into the EC memory can be as large as 32K bytes. The Embedded Memory Interface uses the EC's DCCM Direct Memory Interface (DMI) in order to access the memory. Figure 7-4, "Embedded Memory Region Address Control" shows the relationship between one of the regions in the Embedded Memory Interface and the DCCM DMI:

**FIGURE 7-4: EMBEDDED MEMORY REGION ADDRESS CONTROL**



## 7.4.2 EMBEDDED MEMORY INTERFACE USAGE

The Embedded Memory Interface provides a generic facility for communication between the Host and the EC and can be used for many functions. Some examples are:

- **Virtual registers.** A block of read-only memory locations in the DCCM can be used to implement a set of virtual registers. The EC can update these locations with that the Host can later read.
- **Program downloading.** Because the Instruction Closely Coupled Memory is implemented in the same SRAM as the DCCM, the Embedded Memory Interface can be used by the Host to download new program segments for the EC. The Read/Write window would be configured by the Host to point to the beginning of the loadable program region, which could then be loaded by the Host.
- **Data exchange.** The Read/Write portion of the memory window can be used to contain a communication packet. The Host, by default, "owns" the packet, and can write it at any time. When the Host wishes to communicate with the EC, it sends the EC a command, through the Host-to-EC message facility, to read the packet and perform some operations as a result. When it is completed processing the packet, the EC can inform the Host, either through a message in the EC-to-Host channel or by triggering an event such as an SMI directly. If return results are required, the EC can write the results into the Read/Write region, which the Host can read directly when it is informed that the EC has completed processing. Depending on the command, the operations could entail update of virtual registers in the DCCM, reads of any register in the EC address space, or writes of any register in the EC address space. Because there are two regions that are defined by the base registers, the memory used for the communication packet does not have to be contiguous with a set of virtual registers.

Because there are two Embedded Memory Interface memory regions, the Embedded Memory Interface cannot be used for more than two of these functions at a time. The Host can request that the EC switch from one function to another through the use of the Host-to-EC mailbox register.

# MEC1618/MEC1618i

The [Application ID Register](#) is provided to help software applications track ownership of an Embedded Memory Interface. An application can write the [Application ID Register](#) with its Application ID, then immediately read it back. If the read value is not the same as the value written, then another application has ownership of the interface.

**Note 7-1** The protocol used to pass commands back and forth through the Embedded Memory Interface Registers Interface is left to the System designer. Microchip can provide an application example of working code in which the host uses the Embedded Memory Interface registers to gain access to all of the EC registers.

**Note 7-2** The EC must be awake (i.e., the EC must not be in sleep mode with its clocks gated) in order for the EMI to read or write data in the DCCM/ICCM. System software can insure that the EC is awake when the Host is accessing the EMI with an appropriate protocol using the Host-to-EC mailbox register. Before accessing the memory, the Host sends a “Do Not Sleep” command to the EC through the mailbox register. Writing the register generates an EC interrupt, which wakes the EC. The “Do Not Sleep” command sets a state bit that the EC can check before it issues the Sleep command. When the Host has completed accessing the memory, the Host sends a “Sleep Permitted” command to the EC.

## 7.5 Registers

The [Embedded Memory Interface](#) has its own Logical Device Number, and Base Address as indicated in [Table 7-1](#). The Host LPC I/O addresses for the [Embedded Memory Interface](#) are selected via a Base Address Register (see [Section 4.6.2, “Base Address Registers,” on page 60](#)). LPC access to configuration registers is through Host Access Configuration Port (see [Section 4.5.1, “Host Access Port,” on page 58](#).)

[Table 7-2](#) is a register summary for the [Embedded Memory Interface](#) block.

**TABLE 7-1:     [Embedded Memory Interface](#) BASE ADDRESS TABLE**

<a href="#">Embedded Memory Interface Instance</a>	<a href="#">LDN from (Table 3-2 on page 51)</a>	<a href="#">AHB Base Address</a>
<a href="#">EM Interface</a>	10h	<a href="#">FF_4000h</a>

**Note:** The Host LPC I/O addresses for this instance is selected via a Base Address Register (see [Section 4.6.2, “Base Address Registers,” on page 60](#)). LPC access to configuration registers is through the Host Access Configuration Port (see [Section 4.5.1, “Host Access Port,” on page 58](#).)

The [Table 7-2](#) is a register summary for one instance of the [Embedded Memory Interface](#). The LPC I/O address for each Run-Time Register is described below as an offset from its Base Address Register. Each EC address is indicated as an SPB Offset from its AHB base address. Each Configuration register is accessed through the [Host Access Port](#) is via its LDN indicated in [Table 7-1 on page 158](#) and its [Host Access Port](#) index which is described as “Host Config Index” in the tables below.

**TABLE 7-2: Embedded Memory Interface REGISTER SUMMARY**

Register Name	Host I/O Access			EC Interface			Notes
	Host I/O Offset	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
HOST-to-EC Mailbox Register	00h	00h	R/W	00h	0	R/W	Note 7-3
EC-to-Host Mailbox Register	01h	01h	R/WC	01h	1	R/WC	Note 7-4
EC Address Register	02h 03h	02h 03h	R/W	02h	2-3	R/W	
EC Data Register	04h 05h 06h 07h	04h 05h 06h 07h	R/W	04h	0-3	R/W	
Interrupt Source Register	08h 09h	08h 09h	Table 7-9	08h	0-1	Table 7-9	
Interrupt Mask Register	0Ah 0Bh	0Ah 0Bh	R/W	0Ah	2-3	R/W	
Application ID Register	0Ch	0Ch	R/W	0Ch	0	R/W	
HOST-to-EC Mailbox Register	-	-	-	100h	0	R/WC	Note 7-3
EC-to-Host Mailbox Register	-	-	-	101h	1	R/W	Note 7-4
Memory Base Address 0 Register	-	-	-	104h	0-3	R/W	
Memory Read Limit 0 Register	-	-	-	108h	0-1	R/W	
Memory Write Limit 0 Register	-	-	-	10Ah	2-3	R/W	
Memory Base Address 1 Register	-	-	-	10Ch	0-3	R/W	
Memory Read Limit 1 Register	-	-	-	110h	0-1	R/W	
Memory Write Limit 1 Register	-	-	-	112h	2-3	R/W	
Interrupt Set Register	-	-	-	114h	0-1	R/W	
Host Clear Enable Register	-	-	-	116h	2-3	R/W	

**Note 7-3** Interrupt is cleared when read by the EC.

**Note 7-4** Interrupt is cleared when read by the host.

# MEC1618/MEC1618i

## 7.5.1 EMBEDDED MEMORY INTERFACE CONTROL REGISTERS

Mailbox Register, HOST-to-EC, and Mailbox Register, EC-to-HOST, are specifically designed to pass commands between the host and the EC (FIGURE 7-1: on page 153). If enabled, these registers can generate interrupts.

When the host performs a write of the HOST-to-EC mailbox register, an interrupt will be generated and seen by the EC if unmasked. When the EC writes the HOST-to-EC mailbox register using the EC-only offset address 100h, it can reset the register to 00h, providing a simple means for the EC to inform the host that an operation has been completed.

When the EC writes the EC-to-HOST mailbox register, an SIRQ event or an event such as SMI or PME may be generated and seen by the host if unmasked. The Host CPU can reset the EC-to-HOST mailbox register to 00h, providing a simple means for the host to inform that EC that an operation has been completed.

**PROGRAMMER'S NOTE:** The protocol used to pass commands back and forth through the Mailbox Registers Interface is left to the System designer. Microchip can provide an application example of working code in which the host uses the Mailbox registers to gain access to all of the EC registers.

## 7.6 Registers

### 7.6.1 HOST-TO-EC MAILBOX REGISTER

**TABLE 7-3: HOST-TO-EC MAILBOX REGISTER**

HOST OFFSET	00h				8-Bit			HOST SIZE	
EC OFFSET	00h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	HOST_EC_MBOX[7:0]								



**TABLE 7-4: HOST-TO-EC MAILBOX REGISTER**

HOST OFFSET	-				-			HOST SIZE	
EC OFFSET	100h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	
BIT NAME	HOST_EC_MBOX[7:0]								

## HOST\_EC\_MBOX[7:0]

If enabled, an interrupt to the EC marked by the [EM\\_MBX](#) bit in the [GIRQ15 Source Register](#) will be generated whenever the Host writes this register. The Host and the EC can read and write this register at offset 000h. The EC can also read this register at offset 100h.

Writes of a 1 to any bit in this register by the EC to this register at offset 100h will cause the bit to be cleared. Writes of a 0 to any bit have no effect.

## 7.6.2 EC-TO-HOST MAILBOX REGISTER

**TABLE 7-5: EC-TO-HOST MAILBOX REGISTER**

HOST OFFSET	01h				8-Bit			HOST SIZE	
EC OFFSET	01h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	
BIT NAME	EC_HOST_MBOX[7:0]								

# MEC1618/MEC1618i

TABLE 7-6: EC-TO-HOST MAILBOX REGISTER

HOST OFFSET	-				-			HOST SIZE	
EC OFFSET	101h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	EC_HOST_MBOX[7:0]								

## EC\_HOST\_MBOX[7:0]

An EC write to this register will set bit [EC\\_WR](#) in the [Interrupt Source Register](#) to '1b'. If enabled, setting bit [EC\\_WR](#) to '1b' generates a Host SIRQ event as well as the external nEM\_INT event. The EC can also read and write this register at offset 101h.

Writes of a 1 to any bit in this register at offset 01h, by the Host or by the EC, will cause the bit to be cleared. Writes of a 0 to any bit have no effect.

## 7.6.3 EC ADDRESS REGISTER

**TABLE 7-7: EC ADDRESS REGISTER**

HOST OFFSET	Byte 0: 02h Byte 1: 03h			8-bit t			HOST SIZE	
EC OFFSET	02h			16-bit			EC SIZE	
POWER	VTR			0000h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Region	EC_Address[14:8]						
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	EC_Address[7:2]						Access_Type	

### ACCESS\_TYPE

This field defines the type of access that occurs when the [EC Data Register](#) is read or written.

- 00:** 8-bit access. Any byte read of Byte 0 through Byte 3 in the [EC Data Register](#) causes the corresponding byte within the 32-bit double word addressed by EC\_Address to be loaded into the byte of [EC Data Register](#) and returned by the read. Any byte write to Byte 0 through Byte 3 in the [EC Data Register](#) writes the corresponding byte within the 32-bit double word addressed by EC\_Address, as well as the byte of the [EC Data Register](#).
- 01:** 16-bit access. A read of Byte 0 in the [EC Data Register](#) causes the 16 bits in the DCCM at an offset of EC\_Address to be loaded into Byte 0 and Byte 1 of the [EC Data Register](#). The read then returns the contents of Byte 0. A read of Byte 2 in the [EC Data Register](#) causes the 16 bits in the DCCM at an offset of EC\_Address+2 to be loaded into Byte 2 and Byte 3 of the [EC Data Register](#). The read then returns the contents of Byte 2. A read of Byte 1 or Byte 3 in the [EC Data Register](#) return the contents of the register, without any update from the DCCM.

# MEC1618/MEC1618i

---

A write of Byte 1 in the [EC Data Register](#) causes Bytes 1 and 0 of the [EC Data Register](#) to be written into the 16 bits in the DCCM at an offset of EC\_Address. A write of Byte 3 in the [EC Data Register](#) causes Bytes 3 and 2 of the [EC Data Register](#) to be written into the 16 bits in the DCCM at an offset of EC\_Address+2. A write of Byte 0 or Byte 2 in the [EC Data Register](#) updates the contents of the register, without any change to the DCCM.

- 10:** 32-bit access. A read of Byte 0 in the [EC Data Register](#) causes the 32 bits in the DCCM at an offset of EC\_Address to be loaded into the entire [EC Data Register](#). The read then returns the contents of Byte 0. A read of Byte 1, Byte 2 or Byte 3 in the [EC Data Register](#) returns the contents of the register, without any update from the DCCM.

A write of Byte 3 in the [EC Data Register](#) causes the [EC Data Register](#) to be written into the 32 bits in the DCCM at an offset of EC\_Address. A write of Byte 0, Byte 1 or Byte 2 in the [EC Data Register](#) updates the contents of the register, without any change to the DCCM.

- 11:** Auto-increment 32-bit access. This defines a 32-bit access, as in the 10 case. In addition, any read or write of Byte 3 in the [EC Data Register](#) causes the [EC Address Register](#) to be incremented by 1. That is, the EC\_Address field will point to the next 32-bit double word in the DCCM.

## **EC\_Address[14:2]**

This field defines the location in memory that can be read and/or written with the [EC Data Register](#). The address is an offset from the base of the Host-accessible region in the EC DCCM SRAM. The base of the Host-accessible region.

## **Region**

When this bit is 0, the address defined by [EC\\_Address\[14:2\]](#) is relative to the base address specified by the [Memory Base Address 0 Register](#). When this bit is 1, the address defined by [EC\\_Address\[14:2\]](#) is relative to the base address specified by the [Memory Base Address 1 Register](#).

## 7.6.4 EC DATA REGISTER

TABLE 7-8: EC DATA REGISTER

HOST OFFSET	Byte 0: 04h Byte 1: 05h Byte 2: 06h Byte 3: 07h				8-bit			HOST SIZE	
EC OFFSET	04h				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Data3[7:0]								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Data2[7:0]								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Data1[7:0]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Data0[7:0]								

## DATA

This is a 32-bit register which returns data to the Host from the EC DCCM at the address specified by [EC\\_Address\[14:2\]](#). The description of bits [Access\\_Type](#) in the [EC Address Register](#) defines which reads and writes from the Host trigger transfers of data between this register and the DCCM.

A write to the [EC Data Register](#) when the [EC Address Register](#) is in a read-only or a no-access region, as defined by the Memory Base and Limit registers, will update the [EC Data Register](#) but memory will not be modified. A read to the [EC Data Register](#) when the [EC Address Register](#) is in a no-access region, as defined by the Memory Base and Limit

# MEC1618/MEC1618i

registers, will not trigger a memory read and will not modify the [EC Data Register](#). In auto-increment mode ([Access\\_Type=11b](#)), reads of Byte 3 of the [EC Data Register](#) will still trigger increments of the [EC Address Register](#) when the address is out of bounds, while writes of Byte 3 will not.

## 7.6.5 INTERRUPT SOURCE REGISTER

TABLE 7-9: INTERRUPT SOURCE REGISTER

HOST OFFSET	Byte 0: 08h Byte 1: 09h				8-Bit		HOST SIZE	
EC OFFSET	08h				16-Bit		EC SIZE	
POWER	VTR				0000h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	EC_SWI[14:7]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R
BIT NAME	EC_SWI[6:0]							EC_WR

### EC\_WR

This bit is set autonomously when the [EC-to-Host Mailbox Register](#) has been written by the EC at offset 101h. An SIRQ event or an external nEM\_INT event to the Host is generated when any bit in this register ([EC\\_WR](#) or any bit in [EC\\_SWI\[14:0\]](#)) is '1b' and the corresponding bit in the [Interrupt Mask Register](#) register is '1b'.

This bit is automatically cleared by a read of the [EC-to-Host Mailbox Register](#) at offset 01h.

### EC\_SWI[14:0]

Each bit in this field is cleared when written with a '1b'. The ability to clear the bit can be disabled if the corresponding bit in the [Host Clear Enable Register](#) is set to '0b'.

The EC can generate an interrupt to the Host by setting any bit in this field to '1b'. The EC can set bits to '1b' by writing the corresponding bit in the [Interrupt Set Register](#) to '1b'.

## 7.6.6 INTERRUPT MASK REGISTER

**TABLE 7-10: INTERRUPT MASK REGISTER**

HOST OFFSET	Byte 0: 0Ah Byte 1: 0Bh				8-Bit		HOST SIZE	
EC OFFSET	0Ah				16-Bit		EC SIZE	
POWER	VTR				0000h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	EC_SWI_EN[14:7]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	EC_SWI_EN[6:0]							EC_WR_EN

### EC\_WR\_EN

If this bit is '1b', the interrupt generated by bit [EC\\_WR](#) in the [Interrupt Source Register](#) is enabled.

### EC\_SWI\_EN[14:0]

Each bit that is set to '1b' in this field enables the generation of an interrupt by the corresponding bit in the [EC\\_SWI\[14:0\]](#) field in the [Interrupt Source Register](#).

# MEC1618/MEC1618i

## 7.6.7 APPLICATION ID REGISTER

TABLE 7-11: APPLICATION ID REGISTER

HOST OFFSET	0Ch				8-Bit			HOST SIZE	
EC OFFSET	0Ch				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Application_ID[7:0]								

### APPLICATION\_ID

When this field is 00h it can be written with any value. When set to a non-zero value, writing that value will clear this register to 00h. When set to a non-zero value, writing any value other than the current contents will have no effect.



## 7.6.8 MEMORY BASE ADDRESS 0 REGISTER

This register is accessible to the EC only.

**TABLE 7-12: MEMORY BASE ADDRESS 0 REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	104h				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Memory_Base_Address_0[23:16]								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Memory_Base_Address_0[15:8]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
BIT NAME	Memory_Base_Address_0[7:2]						Reserved		

### MEMORY\_BASE\_ADDRESS\_0[23:2]

This register defines the beginning of a region in the Embedded Controller's Data Closely Coupled Memory that is shared between the Host and the EC. The region defined by this base register, [Region 0](#), is used when bit 15 of the [EC Address Register](#) is 0. The access will be to a memory location at an offset defined by the contents of the [EC Address Register](#), relative to the beginning of the region defined by this register. Therefore, a read or write to the memory that is triggered by the [EC Data Register](#) will occur at  $DCCM\_Base\_Address + Memory\_Base\_Address\_0[23:2] + EC\_Address[14:2]$ .

For example, if [Region](#) = 0, the [Memory\\_Base\\_Address\\_0\[23:2\]](#) = 1000h and the [EC\\_Address\[14:2\]](#) = 20h, then the AHB address of the access will be 80\_1020h.

# MEC1618/MEC1618i

## 7.6.9 MEMORY READ LIMIT 0 REGISTER

This register is accessible to the EC only.

**TABLE 7-13: MEMORY READ LIMIT 0 REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	108h				16-bit			EC SIZE	
POWER	VTR				0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Reserved	Memory_Read_Limit_0[14:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
BIT NAME	Memory_Read_Limit_0[7:2]						Reserved		

### MEMORY\_READ\_LIMIT\_0[14:2]

Whenever a read of any byte in [EC Data Register](#) is attempted, and bit 15 of [EC\\_Address](#) is 0, the field [EC\\_Address\[14:2\]](#) in the [EC Address Register](#) is compared to this field. As long as [EC\\_Address\[14:2\]](#) is less than [Memory\\_Read\\_Limit\\_0\[14:2\]](#) the [EC Data Register](#) will be loaded from the DCCM.

## 7.6.10 MEMORY WRITE LIMIT 0 REGISTER

This register is accessible to the EC only.

**TABLE 7-14: MEMORY WRITE LIMIT 0 REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	10Ah				16-bit			EC SIZE	
POWER	VTR				0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Reserved	Memory_Write_Limit_0[14:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
BIT NAME	Memory_Write_Limit_0[7:2]						Reserved		

### MEMORY\_WRITE\_LIMIT\_0[14:2]

Whenever a write of any byte in [EC Data Register](#) is attempted and bit 15 of [EC\\_Address](#) is 0, the field [EC\\_Address\[14:2\]](#) in the [EC Address Register](#) is compared to this field. As long as [EC\\_Address\[14:2\]](#) is less than [Memory\\_Write\\_Limit\\_0\[14:2\]](#) the addressed bytes in the [EC Data Register](#) will be written into the DCCM. If [EC\\_Address\[14:2\]](#) is greater than or equal to [Memory\\_Write\\_Limit\\_0\[14:2\]](#) no writes will take place.

# MEC1618/MEC1618i

## 7.6.11 MEMORY BASE ADDRESS 1 REGISTER

This register is accessible to the EC only.

**TABLE 7-15: MEMORY BASE ADDRESS 1 REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	10Ch				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Memory_Base_Address_1[23:16]								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Memory_Base_Address_1[15:8]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
BIT NAME	Memory_Base_Address_1[7:2]						Reserved		

### MEMORY\_BASE\_ADDRESS\_1[23:2]

This register defines the beginning of a region in the Embedded Controller's Data Closely Coupled Memory that is shared between the Host and the EC. The region defined by this base register, [Region 1](#), is used when bit 15 of the [EC Address Register](#) is 1. The access will be to a memory location at an offset defined by the contents of the [EC Address Register](#), relative to the beginning of the region defined by this register. Therefore, a read or write to the memory that is triggered by the [EC Data Register](#) will occur at  $DCCM\_Base\_Address + Memory\_Base\_Address\_1[23:2] + EC\_Address[14:2]$ .

For example, if [Region](#) = 1, the [Memory\\_Base\\_Address\\_1\[23:2\]](#) = 1000h and the [EC\\_Address\[14:2\]](#) = 20h, then the AHB address of the access will be 80\_1020h.

## 7.6.12 MEMORY READ LIMIT 1 REGISTER

This register is accessible to the EC only.

**TABLE 7-16: MEMORY READ LIMIT 1 REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	110h				16-bit			EC SIZE	
POWER	VTR				0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Reserved	Memory_Read_Limit_1[14:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
BIT NAME	Memory_Read_Limit_1[7:2]						Reserved		

### MEMORY\_READ\_LIMIT\_1[14:2]

Whenever a read of any byte in [EC Data Register](#) is attempted, and bit 15 of [EC\\_Address](#) is 1, the field [EC\\_Address\[14:2\]](#) in the [EC Address Register](#) is compared to this field. As long as [EC\\_Address\[14:2\]](#) is less than [Memory\\_Read\\_Limit\\_1\[14:2\]](#) the [EC Data Register](#) will be loaded from the DCCM.

# MEC1618/MEC1618i

## 7.6.13 MEMORY WRITE LIMIT 1 REGISTER

This register is accessible to the EC only.

TABLE 7-17: MEMORY WRITE LIMIT 1 REGISTER

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	112h				16-bit			EC SIZE	
POWER	VTR				0000h			VTR POR DEFAULT	
BUS	LPC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Reserved	Memory_Write_Limit_1[14:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
BIT NAME	Memory_Write_Limit_1[7:2]						Reserved		

### MEMORY\_WRITE\_LIMIT\_1[14:2]

Whenever a write of any byte in [EC Data Register](#) is attempted and bit 15 of [EC\\_Address](#) is 1, the field [EC\\_Address\[14:2\]](#) in the [EC Address Register](#) is compared to this field. As long as [EC\\_Address\[14:2\]](#) is less than [Memory\\_Write\\_Limit\\_1\[14:2\]](#) the addressed bytes in the [EC Data Register](#) will be written into the DCCM. If [EC\\_Address\[14:2\]](#) is greater than or equal to [Memory\\_Write\\_Limit\\_1\[14:2\]](#) no writes will take place.

## 7.6.14 INTERRUPT SET REGISTER

This register is accessible to the EC only.

**TABLE 7-18: INTERRUPT SET REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC OFFSET	114h			16-Bit			EC SIZE	
POWER	VTR			0000h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WS	R/WS	R/WS	R/WS	R/WS	R/WS	R/WS	R/WS
BIT NAME	EC_SWI_Set[14:7]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WS	R/WS	R/WS	R/WS	R/WS	R/WS	R/WS	R
BIT NAME	EC_SWI_Set[6:0]							Reserved

### EC\_SWI\_SET[14:0]

This register provides the EC with a means of updating the [Interrupt Source Register](#). Writing a bit in this field with a '1b' sets the corresponding bit in the [Interrupt Source Register](#) to '1b'. Writing a bit in this field with a '0b' has no effect. Reading this field returns the current contents of the [Interrupt Source Register](#).

# MEC1618/MEC1618i

## 7.6.15 HOST CLEAR ENABLE REGISTER

This register is accessible to the EC only.

TABLE 7-19: HOST CLEAR ENABLE REGISTER

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	116h				16-Bit			EC SIZE	
POWER	VTR				0000h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Host_Clr_Enable[14:7]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	
BIT NAME	Host_Clr_Enable[6:0]							Reserved	

### HOST\_CLR\_ENABLE[14:0]

When a bit in this field is '0b', the corresponding bit in the [Interrupt Source Register](#) cannot be cleared by writes to the [Interrupt Source Register](#). When a bit in this field is '1b', the corresponding bit in the [Interrupt Source Register](#) can be cleared when that register bit is written with a '1b'.

These bits allow the EC to control whether the status bits in the [Interrupt Source Register](#) are based on an edge or level event.



## 8.0 ACPI EMBEDDED CONTROLLER INTERFACE (ACPI-ECI)

### 8.1 Introduction

The [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) is a Host/EC Message Interface. The ACPI specification defines the standard hardware and software communications interface between the OS and an embedded controller. This interface allows the OS to support a standard driver that can directly communicate with the embedded controller, allowing other drivers within the system to communicate with and use the EC resources; for example, Smart Battery and AML code.

The [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) provides a four byte full duplex data interface which is a superset of the standard [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) one byte data interface. The [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) defaults to the standard one byte interface.

**Note 8-1** The EC host in [Table 8-8](#) & [Table 8-10](#) corresponds to the EC in the ACPI specification. This interface is referred to elsewhere in this chapter as [ACPI\\_EC](#).

**Note 8-2** The LPC host in [Table 8-8](#) & [Table 8-10](#) corresponds to the “System Host Interface to OS” in the ACPI specification. This interface is referred to elsewhere in this chapter as [ACPI\\_OS](#).

### 8.2 References

1. Advanced Configuration and Power Interface Specification, Revision 4.0 June 16, 2009, Hewlett-Packard Corporation Intel Corporation Microsoft Corporation Phoenix Technologies Ltd. Toshiba Corporation

### 8.3 Terminology

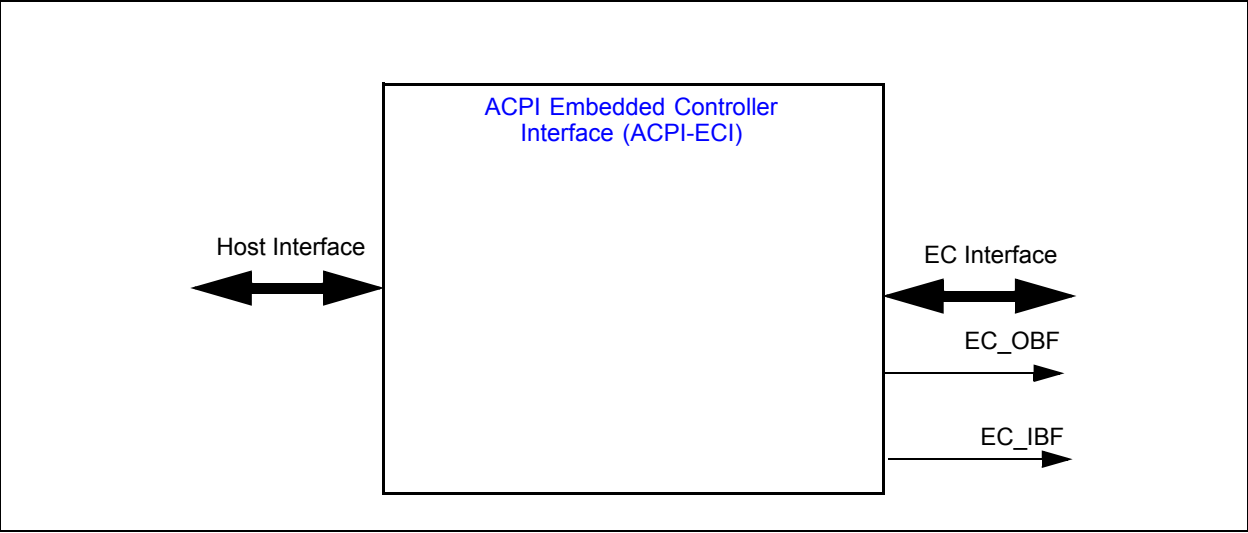
**TABLE 8-1: TERMINOLOGY**

Term	Definition
<b>ACPI_EC</b>	The EC host corresponding to the ACPI specification interface to the EC.
<b>ACPI_OS</b>	The LPC host corresponding to the ACPI specification interface to the “System Host Interface to OS”. <a href="#">ACPI_OS</a> terminology is not meant to distinguish the ACPI System Management from Operating System but merely the hardware path upstream towards the CPU.

## 8.4 Interface

This block is designed to be accessed externally and internally via a register interface.

FIGURE 8-1: ACPI EMBEDDED CONTROLLER INTERFACE (ACPI-ECI) I/O DIAGRAM



### 8.4.1 SIGNAL DESCRIPTION

TABLE 8-2: SIGNAL DESCRIPTION TABLE

Name	Direction	Description
EC_IBF	Output	Output follows IBF in STATUS OS-Register on page 188
EC_OBF	Output	Output follows inversion of OBF in STATUS OS-Register on page 188

### 8.4.2 HOST INTERFACE

The registers defined for the [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) are accessible by the System Host and the Embedded Controller as indicated in [Section 8.11, "EC-Only Registers"](#) and [Section 8.10, "Runtime Registers"](#).

## 8.5 Power, Clocks and Reset

This section defines the Power, Clock, and Reset parameters of the block.

### 8.5.1 POWER DOMAINS

TABLE 8-3: POWER SOURCES

Name	Description
VTR	The logic and registers implemented in this block reside on this single power well.

### 8.5.2 CLOCK INPUTS

This block only requires the Host interface clocks to synchronize registers access.

### 8.5.3 RESETS

**TABLE 8-4: RESET SIGNALS**

Name	Description
nSYS_RST	nSYS_RST resets all the logic and registers in <a href="#">ACPI Embedded Controller Interface (ACPI-ECI)</a> .

## 8.6 Interrupts

This section defines the Interrupt Sources generated from this block.

**TABLE 8-5: INTERRUPTS**

Source	Description
EC_OBF	<a href="#">EC_OBF</a> interrupt is asserted when the <a href="#">OBF</a> in the <a href="#">STATUS EC-Register</a> is cleared to '0'.
EC_IBF	<a href="#">EC_IBF</a> interrupt is asserted when the <a href="#">IBF</a> in the <a href="#">STATUS EC-Register</a> is set to '1'.

**Note:** The usage model from the ACPI specification requires both SMI's and SCI's. The [ACPI\\_OS](#) SMI & SCI interrupts are not implemented in the [ACPI Embedded Controller Interface \(ACPI-ECI\)](#). The [SMI\\_EVT](#) and [SCI\\_EVT](#) bits in the [Section 8.10.6, "STATUS OS-Register," on page 187](#) are software flags and this block does not initiate SMI or SCI events.

## 8.7 Low Power Modes

The ACPI Embedded Controller Interface (ACPI-ECI) automatically enters low power mode when no transaction targets it.

## 8.8 Description

The [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) provides an APCI-EC interface that adheres to the ACPI specification. The ACPI Embedded Controller Interface (ACPI-ECI) includes two modes of operation: [Legacy Mode](#) and [Four-byte Mode](#).

The ACPI Embedded Controller Interface (ACPI-ECI) defaults to [Legacy Mode](#) which provides single byte Full Duplex operation. [Legacy Mode](#) corresponds to the ACPI specification functionality as illustrated in [FIGURE 8-2: on page 181](#). The EC interrupts in [FIGURE 8-2: on page 181](#) are implemented as [EC\\_OBF](#) & [EC\\_IBF](#). See [Section 8.6, "Interrupts," on page 179](#).

In [Four-byte Mode](#), the ACPI Embedded Controller Interface (ACPI-ECI) provides four byte Full Duplex operation. [Four-byte Mode](#) is a superset of the ACPI specification functionality as illustrated in [FIGURE 8-2: on page 181](#).

Both [Legacy Mode](#) & [Four-byte Mode](#) provide Full Duplex Communications which allows data/command transfers in one direction while maintaining data from the other direction; communications can flow both ways simultaneously.

# MEC1618/MEC1618i

---

In Legacy Mode, ACPI Embedded Controller Interface (ACPI-ECI) contains three registers: [ACPI OS COMMAND Register](#), [STATUS OS-Register](#), and [OS2EC Data EC-Register Byte 0](#). The standard [ACPI Embedded Controller Interface \(ACPI-ECI\)](#) registers occupy two addresses in the [ACPI\\_OS](#) space ([Table 8-9](#)).

The [OS2EC Data EC-Register Byte 0](#) and [ACPI OS COMMAND Register](#) registers appear as a single 8-bit data register in the [ACPI\\_EC](#). The [CMD](#) bit in the [STATUS OS-Register](#) is used by the [ACPI\\_EC](#) to discriminate commands from data written by the [ACPI\\_OS](#) to the [ACPI\\_EC](#). [CMD](#) bit is controlled by hardware: [ACPI\\_OS](#) writes to the [OS2EC Data EC-Register Byte 0](#) register clear the [CMD](#) bit; [ACPI\\_OS](#) writes to the [ACPI OS COMMAND Register](#) set the [CMD](#) bit.

FIGURE 8-2: BLOCK DIAGRAM CORRESPONDING TO THE ACPI SPECIFICATION

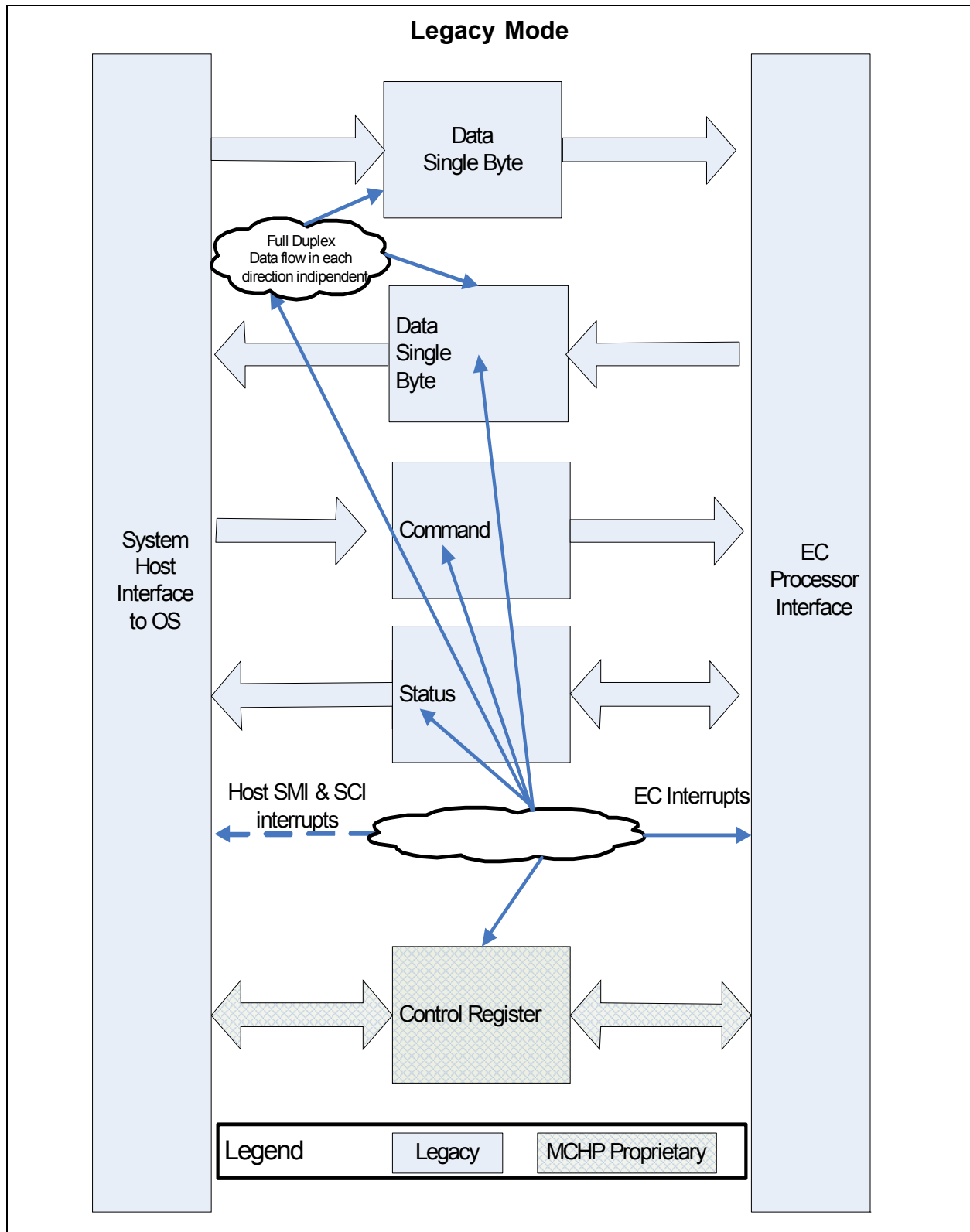
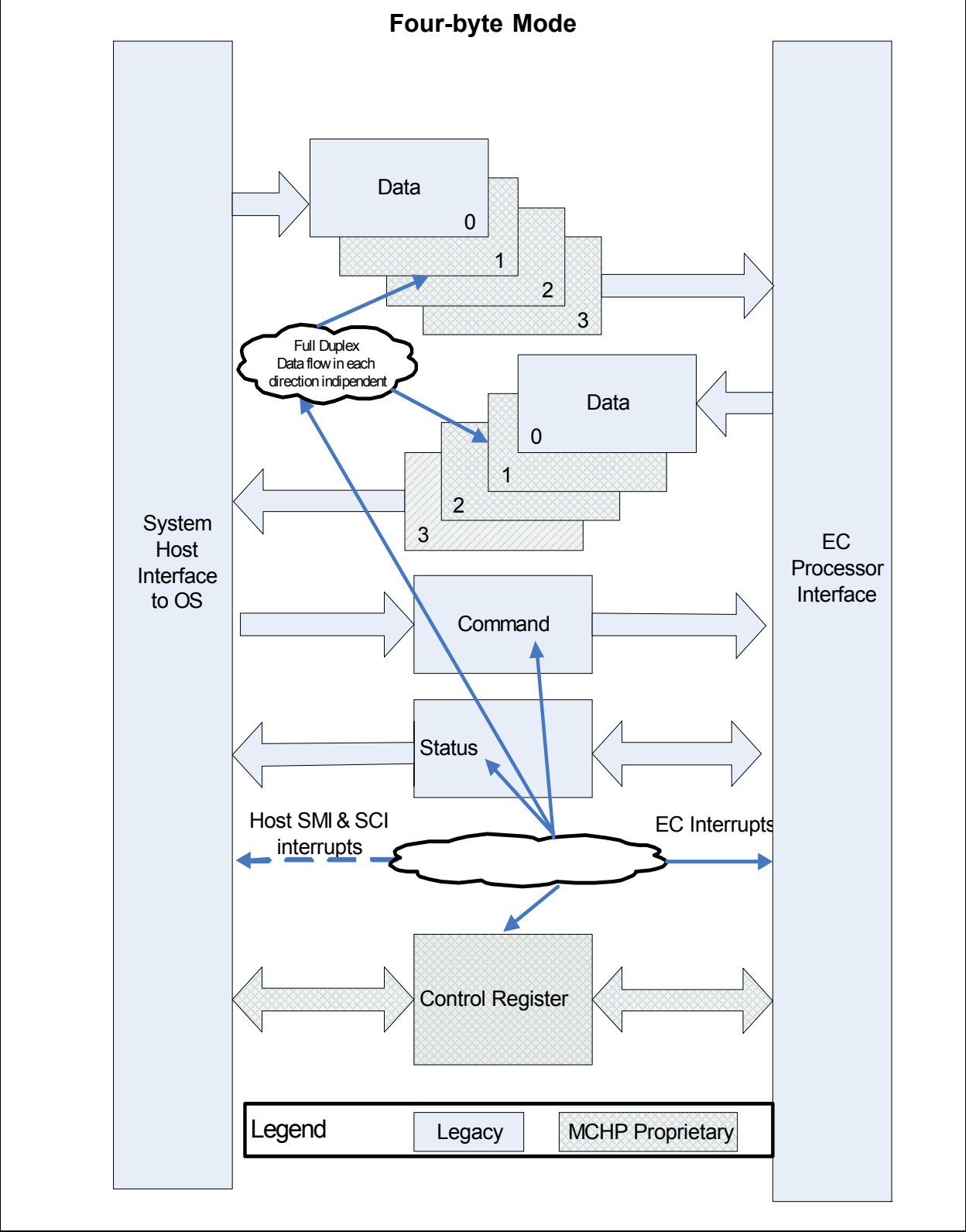


FIGURE 8-2: BLOCK DIAGRAM CORRESPONDING TO THE ACPI SPECIFICATION



## 8.9 Register Aliasing between Runtime and EC-Only Registers

Table 8-6, "Runtime Registers Aliasing into EC-Only Registers" indicates the aliasing from Runtime Registers to EC-Only Registers. The "Aliased Access" column distinguishes the aliasing based on access type. See individual register descriptions for more details.

**TABLE 8-6: Runtime Registers ALIASING INTO EC-ONLY REGISTERS**

Runtime Registers				Map of Aliasing into EC-Only Registers		
LPC Offset	EC Offset	Runtime Registers Register Name (Mnemonic)	Aliased Access	EC Offset	EC-Only Registers Register Name (Mnemonic)	Aliased Access
00h	00h	ACPI OS Data Register Byte 0	W	108h	OS2EC Data EC-Register Byte 0	R
00h	00h	ACPI OS Data Register Byte 0	R	100h	EC2OS Data EC-Register Byte 0	W
01h	01h	ACPI OS Data Register Byte 1	W	109h	OS2EC Data EC-Register Byte 1	R
01h	01h	ACPI OS Data Register Byte 1	R	101h	EC2OS Data EC-Register Byte 1	W
02h	02h	ACPI OS Data Register Byte 2	W	10Ah	OS2EC Data EC-Register Byte 2	R
02h	02h	ACPI OS Data Register Byte 2	R	102h	EC2OS Data EC-Register Byte 2	W
03h	03h	ACPI OS Data Register Byte 3	W	10Bh	OS2EC Data EC-Register Byte 3	R
03h	03h	ACPI OS Data Register Byte 3	R	103h	EC2OS Data EC-Register Byte 3	W
04h	04h	ACPI OS COMMAND Register	W	108h	OS2EC Data EC-Register Byte 0	R
04h	04h	STATUS OS-Register	R	104h	STATUS EC-Register	W
05h	05h	Byte Control OS-Register	R	105h	Byte Control EC-Register	R/W
06h	06h	Reserved		106h	Reserved	
07h	07h	Reserved		107h	Reserved	

Table 8-7, "EC-Only Registers Summary" indicates the aliasing from EC-Only Registers to Runtime Registers. The "Aliased Access" column distinguishes the aliasing based on access type. See individual register descriptions for more details.

# MEC1618/MEC1618i

**TABLE 8-7: EC-Only Registers SUMMARY**

EC-Only Registers			Map of Aliasing into Runtime Registers			
EC Offset	EC-Only Registers Register Name (Mnemonic)	Aliased Access	LPC Offset	EC Offset	Runtime Registers Register Name (Mnemonic)	Aliased Access
108h	OS2EC Data EC-Register Byte 0	R	00h	00h	ACPI OS Data Register Byte 0	W
108h	OS2EC Data EC-Register Byte 0	R	04h	04h	ACPI OS COMMAND Register	W
109h	OS2EC Data EC-Register Byte 1	R	01h	01h	ACPI OS Data Register Byte 1	W
10Ah	OS2EC Data EC-Register Byte 2	R	02h	02h	ACPI OS Data Register Byte 2	W
10Bh	OS2EC Data EC-Register Byte 3	R	03h	03h	ACPI OS Data Register Byte 3	W
104h	STATUS EC-Register	W	04h	04h	STATUS OS-Register	W
105h	Byte Control EC-Register	R/W	05h	05h	Byte Control OS-Register	R
106h	Reserved	R			Reserved	R
107h	Reserved	R			Reserved	R
100h	EC2OS Data EC-Register Byte 0	W	00h	00h	ACPI OS Data Register Byte 0	R
101h	EC2OS Data EC-Register Byte 1	W	01h	01h	ACPI OS Data Register Byte 1	R
102h	EC2OS Data EC-Register Byte 2	W	02h	02h	ACPI OS Data Register Byte 2	R
103h	EC2OS Data EC-Register Byte 3	W	03h	03h	ACPI OS Data Register Byte 3	R



## 8.10 Runtime Registers

The registers listed in the [Runtime Registers Register Summary](#) table are for a single instance of the [ACPI Embedded Controller Interface \(ACPI-ECI\)](#). The addresses of each register listed in this table are defined as a relative offset to the host “Base Address” defined in [Runtime Registers Address Range Table](#).

**TABLE 8-8: Runtime Registers ADDRESS RANGE TABLE**

Block Instance	Instance Number	Host	Address Space	Base Address
ACPI-EC	0	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">FF_0800h</a> +100h
ACPI-EC	0	LPC <a href="#">Note 8-2</a>	I/O	Programmed BAR
ACPI-EC	1	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">FF_0C00h</a> +100h
ACPI-EC	1	LPC <a href="#">Note 8-2</a>	I/O	Programmed BAR
ACPI-EC	2	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">FF_1000h</a> +100h
ACPI-EC	2	LPC <a href="#">Note 8-2</a>	I/O	Programmed BAR
ACPI-EC	3	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">FF_1400h</a> +100h
ACPI-EC	3	LPC <a href="#">Note 8-2</a>	I/O	Programmed BAR

**Note 8-3** The Base Address indicates where the first register can be accessed in a particular address space for a block instance.

**TABLE 8-9: Runtime Registers REGISTER SUMMARY**

Offset	Register Name (Mnemonic)
00h	<a href="#">ACPI OS Data Register Byte 0</a>
01h	<a href="#">ACPI OS Data Register Byte 1</a>
02h	<a href="#">ACPI OS Data Register Byte 2</a>
03h	<a href="#">ACPI OS Data Register Byte 3</a>
04h	<a href="#">ACPI OS COMMAND Register</a>
04h	<a href="#">STATUS OS-Register</a>
05h	<a href="#">Byte Control OS-Register</a>
06h	Reserved
07h	Reserved

# MEC1618/MEC1618i

## 8.10.1 ACPI OS DATA REGISTER BYTE 0

OS\_DATA\_B0  
; ALIAS=(OS2EC\_DATA\_ECREG\_B0)  
; ALIAS=EC2OS\_DATA\_ECREG\_B0

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	00h			
Bits	Description	Type	Default	Reset Event
7:0	ACPI-OS DATA BYTE 0 <b>OS_DAT_B0; ALIASED</b> This is byte 0 of the 32-bit <a href="#">ACPI-OS DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

### ACPI-OS DATA BYTES[3:0]

Writes by the [ACPI\\_OS](#) to the [ACPI-OS DATA BYTES\[3:0\]](#) are aliased to the [OS2EC DATA BYTES\[3:0\]](#). Reads by the [ACPI\\_OS](#) from the [ACPI-OS DATA BYTES\[3:0\]](#) are aliased to the [EC2OS DATA BYTES\[3:0\]](#).

All access to the [ACPI-OS DATA BYTES\[3:0\]](#) registers should be orderly: Least Significant Byte to Most Significant Byte when byte access is used.

Writes to any of the four [ACPI-OS DATA BYTES\[3:0\]](#) registers clears the [CMD](#) bit in the [STATUS OS-Register](#) (the state of the [Four Byte Access](#) bit in the [Byte Control OS-Register](#) has no impact.)

When the [Four Byte Access](#) bit in the [Byte Control OS-Register](#) is cleared to '0', the following access rules apply:

- Writes to the [ACPI OS Data Register Byte 0](#) sets the [IBF](#) bit in the [STATUS OS-Register](#).
- Reads from the [ACPI OS Data Register Byte 0](#) clears the [OBF](#) bit in the [STATUS OS-Register](#).
- All writes to [ACPI-OS DATA BYTES\[3:1\]](#) complete without error but the data are not registered.
- All reads from [ACPI-OS DATA BYTES\[3:1\]](#) return 00h without error.
- Access to [ACPI-OS DATA BYTES\[3:1\]](#) has no effect on the [IBF](#) & [OBF](#) bits in the [STATUS OS-Register](#).

When the [Four Byte Access](#) bit in the [Byte Control OS-Register](#) is set to '1', the following access rules apply:

- Writes to the [ACPI OS Data Register Byte 3](#) sets the [IBF](#) bit in the [STATUS OS-Register](#).
- Reads from the [ACPI OS Data Register Byte 3](#) clears the [OBF](#) bit in the [STATUS OS-Register](#).

## 8.10.2 ACPI OS DATA REGISTER BYTE 1

OS\_DATA\_B1  
; ALIAS=OS2EC\_DATA\_ECREG\_B1  
; ALIAS=EC2OS\_DATA\_ECREG\_B1

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	01h			
Bits	Description	Type	Default	Reset Event
7:0	ACPI-OS DATA BYTE 1 <b>OS_DAT_B1; ALIASED</b> This is byte 1 of the 32-bit <a href="#">ACPI-OS DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

## 8.10.3 ACPI OS DATA REGISTER BYTE 2

OS\_DATA\_B2  
; ALIAS=OS2EC\_DATA\_ECREG\_B2  
; ALIAS=EC2OS\_DATA\_ECREG\_B2

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	02h			
Bits	Description	Type	Default	Reset Event
7:0	ACPI-OS DATA BYTE 2 (OS_DAT_B2; ALIASED)  This is byte 2 of the 32-bit <a href="#">ACPI-OS DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

## 8.10.4 ACPI OS DATA REGISTER BYTE 3

OS\_DATA\_B3  
; ALIAS=OS2EC\_DATA\_ECREG\_B3  
; ALIAS=EC2OS\_DATA\_ECREG\_B3

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	03h			
Bits	Description	Type	Default	Reset Event
7:0	ACPI-OS DATA BYTE 3 (OS_DAT_B3; ALIASED)  This is byte 3 of the 32-bit <a href="#">ACPI-OS DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

# MEC1618/MEC1618i

## 8.10.5 ACPI OS COMMAND REGISTER

COMMAND\_OSREG  
; ALIAS=(OS2EC\_DATA\_ECREG\_B0)

Offset	04h			
Bits	Description	Type	Default	Reset Event
7:0	ACPI-OS COMMAND (OS_CMD; ALIASED) Writes to the this register are aliased in the OS2EC Data EC-Register Byte 0. Writes to the this register also set the CMD and IBF bits in the STATUS OS-Register	W	0h	nSYS_RST

## 8.10.6 STATUS OS-REGISTER

STATUS\_OSREG  
; ALIAS=(STATUS\_ECREG)

This read-only register is aliased to the STATUS EC-Register on page 197. the STATUS EC-Register on page 197 has read write access.

Offset	04h			
Bits	Description	Type	Default	Reset Event
7	User Defined (UD-Bit7_OS_OS; ALIASED)	R	0b	nSYS_RST
6	SMI_EVT (SMI_EVT_OS; ALIASED) This bit is set when an SMI event is pending; i.e., the ACPI_EC is requesting an SMI query; This bit is cleared when no SMI events are pending. This bit is an ACPI_EC-maintained software flag that is set when the ACPI_EC has detected an internal event that requires system management interrupt handler attention. The ACPI_EC sets this bit before generating an SMI. <b>Note:</b> The usage model from the ACPI specification requires both SMI's and SCI's. The ACPI_OS SMI & SCI interrupts are not implemented in the ACPI Embedded Controller Interface (ACPI-ECI). The SMI_EVT and SCI_EVT bits in the Section 8.10.6, "STATUS OS-Register," on page 187 are software flags and this block does not initiate SMI or SCI events.	R	0b	nSYS_RST

Offset	04h			
Bits	Description	Type	Default	Reset Event
5	<p>SCI_EVT (SCI_EVT_OS; ALIASED)</p> <p>This bit is set by software when an SCI event is pending; i.e., the <a href="#">ACPI_EC</a> is requesting an SCI query; SCI Event flag is clear when no SCI events are pending. This bit is an <a href="#">ACPI_EC</a>-maintained software flag that is set when the embedded controller has detected an internal event that requires operating system attention. The <a href="#">ACPI_EC</a> sets this bit before generating an SCI to the OS.</p> <p><b>Note:</b> The usage model from the ACPI specification requires both SMI's and SCI's. The <a href="#">ACPI_OS</a> SMI &amp; SCI interrupts are not implemented in the <a href="#">ACPI Embedded Controller Interface (ACPI-ECI)</a>. The <a href="#">SMI_EVT</a> and <a href="#">SCI_EVT</a> bits in the <a href="#">Section 8.10.6, "STATUS OS-Register," on page 187</a> are software flags and this block does not initiate SMI or SCI events.</p>	R	0b	nSYS_RST
4	<p>BURST (BURST_OS; ALIASED)</p> <p>The BURST bit is set when the <a href="#">ACPI_EC</a> is in Burst Mode for polled command processing; the BURST bit is cleared when the <a href="#">ACPI_EC</a> is in Normal mode for interrupt-driven command processing. The BURST bit is an <a href="#">ACPI_EC</a>-maintained software flag that indicates the embedded controller has received the Burst Enable command from the host, has halted normal processing, and is waiting for a series of commands to be sent from the host. Burst Mode allows the OS or system management handler to quickly read and write several bytes of data at a time without the overhead of SCIs between commands. The BURST bit is maintained by <a href="#">ACPI_EC</a> software, only.</p>	R	0b	nSYS_RST
3	<p>CMD (CMD_OS; ALIASED)</p> <p>This bit is set when the <a href="#">OS2EC Data EC-Register Byte 0</a> contains a command byte written into <a href="#">ACPI OS COMMAND Register</a>; this bit is cleared when the <a href="#">OS2EC DATA BYTES[3:0]</a> contains a data byte written into the <a href="#">ACPI-OS DATA BYTES[3:0]</a>.</p> <p>This bit is hardware controlled:</p> <ul style="list-style-type: none"> <li>• <a href="#">ACPI_OS</a> writes to any of the four <a href="#">ACPI-OS DATA BYTES[3:0]</a> bytes clears this bit</li> <li>• <a href="#">ACPI_OS</a> writes to the <a href="#">ACPI OS COMMAND Register</a> sets this bit.</li> </ul> <p><b>Note:</b> This bit allows the embedded controller to differentiate the start of a command sequence from a data byte write operation.</p>	R	0b	nSYS_RST
2	<p>User Defined (UD-Bit2_OS; ALIASED)</p>	R	0b	nSYS_RST

# MEC1618/MEC1618i

Offset	04h			
Bits	Description	Type	Default	Reset Event
1	<p><b>IBF</b> (IBF_OS; ALIASED)</p> <p>The Input Buffer Full bit is set to indicate that a the <b>ACPI_OS</b> has written a command or data to the <b>ACPI_EC</b> and that data is ready. This bit is automatically cleared when data has been read by the <b>ACPI_EC</b>.</p> <p><b>Note:</b> The setting and clearing of this <b>IBF</b> varies depending on the setting of the following bits: <b>CMD</b> bit in this register &amp; <b>Four Byte Access</b> bit in the <b>Byte Control OS-Register</b>. Three scenarios follow:</p> <ol style="list-style-type: none"> <li>1. The <b>IBF</b> is set when the <b>ACPI_OS</b> writes to the <b>ACPI OS COMMAND Register</b>. This same write autonomously sets the <b>CMD</b> bit in this register. The <b>IBF</b> is cleared if the <b>CMD</b> bit in this register is set and the <b>ACPI_EC</b> reads from the <b>OS2EC Data EC-Register Byte 0</b>.</li> <li>2. A write by the to the <b>ACPI_OS</b> to the <b>ACPI OS Data Register Byte 0</b> sets the <b>IBF</b> bit if the <b>Four Byte Access</b> bit in the <b>Byte Control OS-Register</b> is in the cleared to '0' state prior to this write. This same write autonomously clears the <b>CMD</b> bit in this register. A read of the <b>OS2EC Data EC-Register Byte 0</b> clears the <b>IBF</b> bit if the <b>Four Byte Access</b> bit in the <b>Byte Control OS-Register</b> is in the cleared to '0' state prior to this read.</li> <li>3. A write by the to the <b>ACPI_OS</b> to the <b>ACPI OS Data Register Byte 3</b> sets the <b>IBF</b> bit if the <b>Four Byte Access</b> bit in the <b>Byte Control OS-Register</b> is in the set to '1' state prior to this write. This same write autonomously clears the <b>CMD</b> bit in this register. A read of the <b>OS2EC Data EC-Register Byte 3</b> clears the <b>IBF</b> bit if the <b>Four Byte Access</b> bit in the <b>Byte Control OS-Register</b> is in the set to '1' state prior to this read.</li> </ol> <p>An <b>EC_IBF</b> interrupt signals the <b>ACPI_EC</b> that there is data available. The ACPI Specification usage model is as follows:</p> <ol style="list-style-type: none"> <li>1. The <b>ACPI_EC</b> reads the <b>STATUS EC-Register</b> and sees the <b>IBF</b> flag set,</li> <li>2. The <b>ACPI_EC</b> reads all the data available in the <b>OS2EC DATA BYTES[3:0]</b>. This causes the <b>IBF</b> bit to be automatically cleared by hardware.</li> <li>3. The <b>ACPI_EC</b> must then generate a software interrupt (See <b>Note: on page 179</b>) to alert the <b>ACPI_OS</b> that the data has been read and that the host is free to write more data to the <b>ACPI_EC</b> as needed.</li> </ol>	R	0h	nSYS_RST

Offset	04h			
Bits	Description	Type	Default	Reset Event
0	<p><b>OBF</b> (OBF_OS; ALIASED)</p> <p>The Output Buffer Full bit is set to indicate that a the <a href="#">ACPI_EC</a> has written a data to the <a href="#">ACPI_OS</a> and that data is ready. This bit is automatically cleared when all the data has been read by the <a href="#">ACPI_OS</a>.</p> <p><b>Note:</b> The setting and clearing of this <b>OBF</b> varies depending on the setting <a href="#">Four Byte Access</a> bit in the <a href="#">Byte Control OS-Register</a>. Two scenarios follow:</p> <ol style="list-style-type: none"> <li>1. The <b>OBF</b> bit is set if the <a href="#">Four Byte Access</a> bit in the <a href="#">Byte Control OS-Register</a> is '0' when the <a href="#">ACPI_EC</a> writes to the <a href="#">EC2OS Data EC-Register Byte 0</a>. The <b>OBF</b> is cleared if the <a href="#">Four Byte Access</a> bit in the <a href="#">Byte Control OS-Register</a> is cleared to '0' when the <a href="#">ACPI_OS</a> reads from the <a href="#">ACPI OS Data Register Byte 0</a>.</li> <li>2. The <b>OBF</b> is set if the <a href="#">Four Byte Access</a> bit in the <a href="#">Byte Control OS-Register</a> is set to '1' when the <a href="#">ACPI_EC</a> writes to the <a href="#">EC2OS Data EC-Register Byte 3</a>. The <b>OBF</b> is cleared if the <a href="#">Four Byte Access</a> bit in the <a href="#">Byte Control OS-Register</a> is set to '1' when the <a href="#">ACPI_OS</a> reads from the <a href="#">ACPI OS Data Register Byte 3</a>.</li> </ol> <p>The ACPI Specification usage model is as follows:</p> <ol style="list-style-type: none"> <li>1. The <a href="#">ACPI_EC</a> must generate a software interrupt (See <a href="#">Note: on page 179</a>) to alert the <a href="#">ACPI_OS</a> that the data is available.</li> <li>2. The <a href="#">ACPI_OS</a> reads the <a href="#">STATUS OS-Register</a> and sees the <b>OBF</b> flag set, the <a href="#">ACPI_OS</a> reads all the data available in the <a href="#">ACPI-OS DATA BYTES[3:0]</a>.</li> <li>3. The <a href="#">ACPI_OS</a> reads all the data available in the <a href="#">ACPI-OS DATA BYTES[3:0]</a>. This causes the <b>OBF</b> bit to be automatically cleared by hardware and the associated <a href="#">EC_OBF</a> interrupt to be asserted.</li> </ol>	R	0h	nSYS_RST

# MEC1618/MEC1618i

## 8.10.7 BYTE CONTROL OS-REGISTER

BYTE\_CNL\_OSREG  
; ALIAS=BYTE\_CNL\_ECREG

This register is aliased to the [Byte Control EC-Register on page 198](#). No behavioral differences occur due to address aliasing.

Offset	05			
Bits	Description	Type	Default	Reset Event
7:1	RESERVED	RES	-	-
0	<p>Four Byte Access (4BYTE_ACCESS_EN_OS;ALIASED)</p> <p>When this bit is set to '1', the ACPI Embedded Controller Interface (ACPI-ECI) accesses four bytes through the <a href="#">ACPI-OS DATA BYTES[3:0]</a>. When this bit is cleared to '0', the ACPI Embedded Controller Interface (ACPI-ECI) accesses one byte through the <a href="#">ACPI OS Data Register Byte 0</a>. The corresponds to <a href="#">Legacy Mode</a> described in <a href="#">Section 8.8, "Description," on page 179</a>.</p> <p><b>Note:</b> This bit effects the behavior of the <a href="#">IBF &amp; OBF</a> bits in the <a href="#">STATUS OS-Register</a>.</p> <p><b>Note:</b> See <a href="#">ACPI-OS DATA BYTES[3:0] on page 186</a>, <a href="#">OS2EC DATA BYTES[3:0] on page 193</a>, and <a href="#">EC2OS DATA BYTES[3:0] on page 195</a> for detailed description of access rules.</p>	R	0b	nSYS_RST

**APPLICATION NOTE:** The [ACPI\\_OS](#) access Base Address Register (BAR) should be configured to match the access width selected by the [Four Byte Access](#) bit in the [Byte Control OS-Register](#). This BAR is not described in this chapter.

## 8.11 EC-Only Registers

The registers listed in the [EC-Only Registers Register Summary](#) table are for a single instance of the [ACPI Embedded Controller Interface \(ACPI-ECI\)](#). The addresses of each register listed in this table are defined as a relative offset to the host "Base Address" defined in [EC-Only Registers Address Range Table](#).

**TABLE 8-10: EC-Only Registers ADDRESS RANGE TABLE**

Block Instance	Instance Number	Host	Address Space	Base Address
ACPI-EC	0	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">ACPI EC Channel 0 LDN 2h</a> , Base address <a href="#">FF_0800h</a>
ACPI-EC	1	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">ACPI EC Channel 1 LDN 3h</a> , Base address <a href="#">FF_0C00h</a>
ACPI-EC	2	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">ACPI EC Channel 2 LDN 4h</a> , Base address <a href="#">FF_1000h</a>
ACPI-EC	3	EC <a href="#">Note 8-1</a>	24-bit internal address space	<a href="#">ACPI EC Channel 3 LDN 5h</a> , Base address <a href="#">FF_1400h</a>



**TABLE 8-11: EC-Only Registers REGISTER SUMMARY**

Offset	Register Name (Mnemonic)
100h	EC2OS Data EC-Register Byte 0
101h	EC2OS Data EC-Register Byte 1
102h	EC2OS Data EC-Register Byte 2
103h	EC2OS Data EC-Register Byte 3
104h	STATUS EC-Register
105h	Byte Control EC-Register
106h	Reserved
107h	Reserved
108h	OS2EC Data EC-Register Byte 0
109h	OS2EC Data EC-Register Byte 1
10Ah	OS2EC Data EC-Register Byte 2
10Bh	OS2EC Data EC-Register Byte 3

**8.11.1 OS2EC DATA EC-REGISTER BYTE 0**

(OS2EC\_DATA\_ECREG\_B0)  
; ALIAS=OS\_DATA\_B0  
; ALIAS=COMMAND\_OSREG

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	108h			
Bits	Description	Type	Default	Reset Event
7:0	OS TO EC DATA BYTE 0 (OS2EC_DAT_B0) This is byte 0 of the 32-bit <a href="#">OS2EC DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

**OS2EC DATA BYTES[3:0]**

When the **CMD** bit in the **STATUS OS-Register** is cleared to '0', reads by the **ACPI\_EC** from the **OS2EC DATA BYTES[3:0]** are aliased to the **ACPI-OS DATA BYTES[3:0]**.

All access to the **OS2EC DATA BYTES[3:0]** registers should be orderly: Least Significant Byte to Most Significant Byte when byte access is used.

When the **Four Byte Access** bit in the **Byte Control OS-Register** is cleared to '0', the following access rules apply:

- Writes to the **OS2EC DATA BYTES[3:0]** have no effect on the **OBF** bit in the **STATUS OS-Register**.
- Reads from the **OS2EC Data EC-Register Byte 0** clears the **IBF** bit in the **STATUS OS-Register**.
- All reads from **OS2EC DATA BYTES[3:1]** return 00h without error.
- Access to **OS2EC DATA BYTES[3:1]** has no effect on the **IBF** & **OBF** bits in the **STATUS OS-Register**.

# MEC1618/MEC1618i

When the [Four Byte Access](#) bit in the [Byte Control OS-Register](#) is set to '1', the following access rules apply:

- Writes to the [OS2EC DATA BYTES\[3:0\]](#) have no effect on the [OBF](#) bit in the [STATUS OS-Register](#).
- Reads from the [OS2EC Data EC-Register Byte 3](#) clears the [IBF](#) bit in the [STATUS OS-Register](#).

## 8.11.2 OS2EC DATA EC-REGISTER BYTE 1

OS2EC\_DATA\_ECREG\_B1  
; ALIAS=[OS\\_DATA\\_B1](#)

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	109h			
Bits	Description	Type	Default	Reset Event
7:0	OS2EC Data BYTE 1 ( <a href="#">OS2EC_DAT_B1</a> )  This is byte 1 of the 32-bit <a href="#">OS2EC DATA BYTES[3:0]</a> .	R/W	0h	<a href="#">nSYS_RST</a>

## 8.11.3 OS2EC DATA EC-REGISTER BYTE 2

OS2EC\_DATA\_ECREG\_B2  
; ALIAS=[OS\\_DATA\\_B2](#)

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	10Ah			
Bits	Description	Type	Default	Reset Event
7:0	OS2EC Data BYTE 2 ( <a href="#">OS2EC_DAT_B2</a> )  This is byte 2 of the 32-bit <a href="#">OS2EC DATA BYTES[3:0]</a> .	R/W	0h	<a href="#">nSYS_RST</a>

## 8.11.4 OS2EC DATA EC-REGISTER BYTE 3

OS2EC\_DATA\_ECREG\_B3  
; ALIAS=[OS\\_DATA\\_B3](#)

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	10Bh			
Bits	Description	Type	Default	Reset Event
7:0	OS2EC Data BYTE 3 ( <a href="#">OS2EC_DAT_B3</a> )  This is byte 3 of the 32-bit <a href="#">OS2EC DATA BYTES[3:0]</a> .	R/W	0h	<a href="#">nSYS_RST</a>

## 8.11.5 EC2OS DATA EC-REGISTER BYTE 0

EC2OS\_DATA\_ECREG\_B0  
; ALIAS=OS\_DATA\_B0

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	100h			
Bits	Description	Type	Default	Reset Event
7:0	EC2OS DATA BYTE 0 (EC2OS_DAT_B0)  This is byte 0 of the 32-bit <a href="#">EC2OS DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

### EC2OS DATA BYTES[3:0]

Writes by the [ACPI\\_EC](#) to the [EC2OS DATA BYTES\[3:0\]](#) are aliased to the [ACPI-OS DATA BYTES\[3:0\]](#)

All access to the [EC2OS DATA BYTES\[3:0\]](#) registers should be orderly: Least Significant Byte to Most Significant Byte when byte access is used.

When the [Four Byte Access](#) bit in the [Byte Control OS-Register](#) is cleared to '0', the following access rules apply:

- Writes to the [EC2OS Data EC-Register Byte 0](#) set the [OBF](#) bit in the [STATUS OS-Register](#).
- Reads from the [EC2OS DATA BYTES\[3:0\]](#) have no effect on the [IBF](#) bit in the [STATUS OS-Register](#).
- All reads from [EC2OS DATA BYTES\[3:1\]](#) return 00h without error.
- All writes to [EC2OS DATA BYTES\[3:1\]](#) complete without error but the data are not registered.
- Access to [EC2OS DATA BYTES\[3:1\]](#) have no effect on the [IBF](#) & [OBF](#) bits in the [STATUS OS-Register](#).

When the [Four Byte Access](#) bit in the [Byte Control OS-Register](#) is set to '1', the following access rules apply:

- Writes to the [EC2OS Data EC-Register Byte 3](#) set the [OBF](#) bit in the [STATUS OS-Register](#).
- Reads from the [EC2OS DATA BYTES\[3:0\]](#) have no effect on the [IBF](#) bit in the [STATUS OS-Register](#).

## 8.11.6 EC2OS DATA EC-REGISTER BYTE 1

EC2OS\_DATA\_ECREG\_B1  
; ALIAS=OS\_DATA\_B1

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	101h			
Bits	Description	Type	Default	Reset Event
7:0	EC2OS DATA BYTE 1 (EC2OS_DAT_B1)  This is byte 1 of the 32-bit <a href="#">EC2OS DATA BYTES[3:0]</a> .	R/W	0h	nSYS_RST

# MEC1618/MEC1618i

## 8.11.7 EC2OS DATA EC-REGISTER BYTE 2

EC2OS\_DATA\_ECREG\_B2  
; ALIAS=[OS\\_DATA\\_B2](#)

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	102h			
Bits	Description	Type	Default	Reset Event
7:0	EC2OS DATA BYTE 2 ( <a href="#">EC2OS_DAT_B2</a> )  This is byte 2 of the 32-bit <a href="#">EC2OS DATA BYTES[3:0]</a> .	R/W	0h	<a href="#">nSYS_RST</a>

## 8.11.8 EC2OS DATA EC-REGISTER BYTE 3

EC2OS\_DATA\_ECREG\_B3  
; ALIAS=[OS\\_DATA\\_B3](#)

This register is aliased; see [ACPI-OS DATA BYTES\[3:0\] on page 186](#), [OS2EC DATA BYTES\[3:0\] on page 193](#), and [EC2OS DATA BYTES\[3:0\] on page 195](#) for detailed description of access rules.

Offset	103h			
Bits	Description	Type	Default	Reset Event
7:0	EC2OS DATA BYTE 3 ( <a href="#">EC2OS_DAT_B0</a> )  This is byte 3 of the 32-bit <a href="#">EC2OS DATA BYTES[3:0]</a> .	R/W	0h	<a href="#">nSYS_RST</a>

## 8.11.9 STATUS EC-REGISTER

(STATUS\_ECREG)  
; ALIAS=STATUS\_OSREG

This register is aliased to the [STATUS OS-Register on page 188](#). The [STATUS OS-Register](#) is a read only version of this register.

Offset	104h			
Bits	Description	Type	Default	Reset Event
7	User Defined (UD-Bit7_EC; ALIASED)	R/W	0b	nSYS_RST
6	SMI_EVT (SMI_EVT_EC; ALIASED) See <a href="#">SMI_EVT</a> bit in <a href="#">STATUS OS-Register on page 188</a> for bit description.	R/W	0b	nSYS_RST
5	SCI_EVT (SCI_EVT_EC; ALIASED) See <a href="#">SCI_EVT</a> bit in <a href="#">STATUS OS-Register on page 188</a> for bit description.	R/W	0b	nSYS_RST
4	BURST (BURST_EC; ALIASED) See <a href="#">BURST</a> bit in <a href="#">STATUS OS-Register on page 188</a> for bit description.	R/W	0b	nSYS_RST
3	CMD (CMD_EC; ALIASED) See <a href="#">CMD</a> bit in <a href="#">STATUS OS-Register on page 188</a> for bit description.	R	0b	nSYS_RST
2	User Defined (UD-Bit2_EC; ALIASED)	R/W	0b	nSYS_RST
1	IBF (IBF_EC; ALIASED) See <a href="#">IBF</a> bit in <a href="#">STATUS OS-Register on page 188</a> for bit description.	R	0h	nSYS_RST
0	OBF (OBF_EC; ALIASED) See <a href="#">OBF</a> bit in <a href="#">STATUS OS-Register on page 188</a> for bit description.	R	0h	nSYS_RST

**APPLICATION NOTE:** The [IBF](#) and [OBF](#) bits are not cleared ('0') by hardware when [VCC\\_PWRGD](#) is asserted or when the LPC interface powers down; for example, following system state changes S3->S0, S5->S0, G3-> S0. For further information on how these bits are cleared, refer to [IBF](#) and [OBF](#) bit descriptions in [STATUS OS-Register on page 188](#).

# MEC1618/MEC1618i

---

## 8.11.10 BYTE CONTROL EC-REGISTER

BYTE\_CNL\_ECREG  
; ALIAS=BYTE\_CNL\_OSREG

This register is aliased to the [Byte Control OS-Register on page 192](#). The [Byte Control OS-Register](#) is a read only version of this register.

Offset	105h			
Bits	Description	Type	Default	Reset Event
7:1	RESERVED	RES	-	-
0	Four Byte Access (4BYTE_ACCESS_EN_OS;ALIASED) See <a href="#">Four Byte Access</a> bit in <a href="#">Byte Control OS-Register on page 192</a> for bit description.	R/W	0b	nSYS_RST

## 9.0 8042 EMULATED KEYBOARD CONTROLLER

### 9.1 General Description

The MEC1618/MEC1618i keyboard controller uses the EC to produce a superset of the features provided by the industry-standard 8042 keyboard controller. The [8042 Emulated Keyboard Controller](#) is a Host/EC Message Interface with hardware assists to emulate 8042 behavior and provide Legacy GATEA20 support.

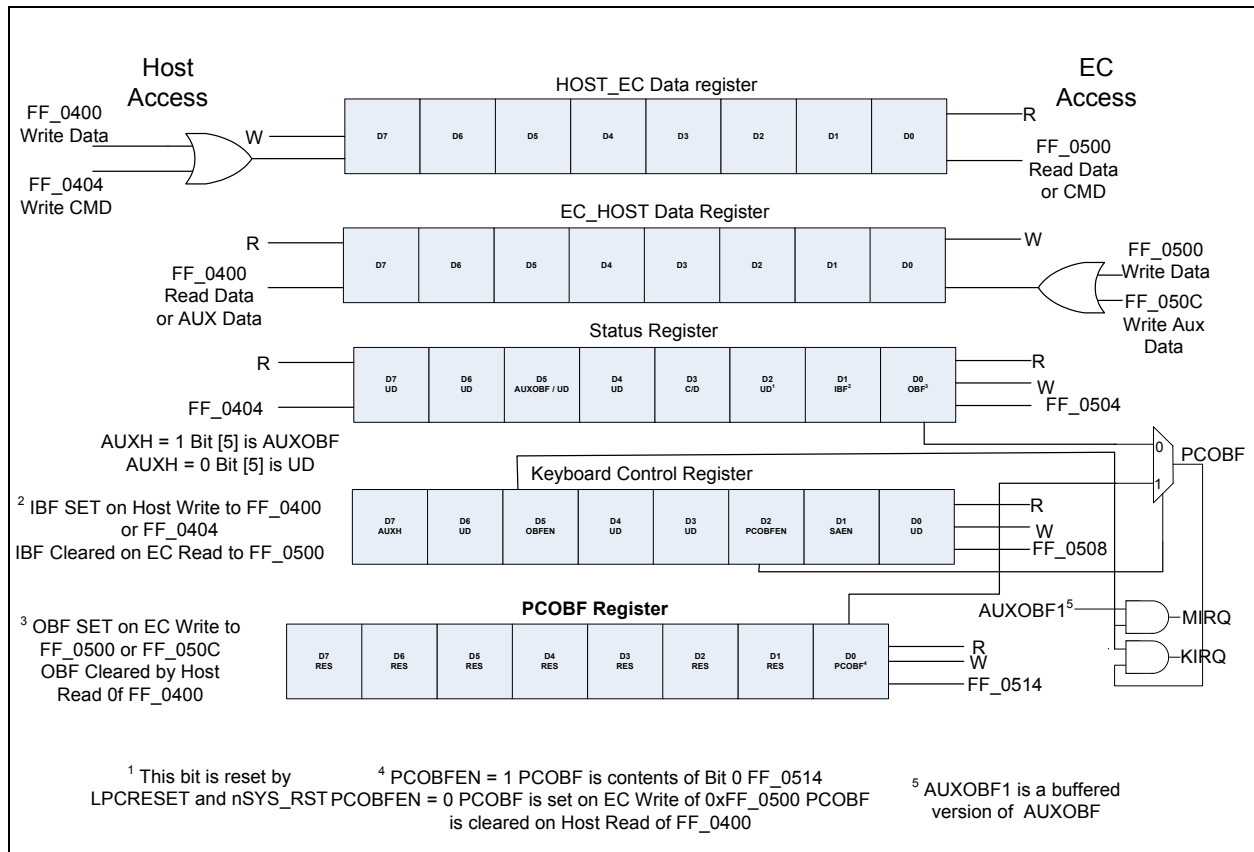
**Note:** There is no VCC emulation in hardware for this interface.

#### 9.1.1 FEATURES

- Legacy Keyboard Support
- Emulated 8042 Operation
- Port 92 Legacy A20M Support

#### 9.1.2 BLOCK DIAGRAM

**FIGURE 9-1: BLOCK DIAGRAM OF 8042 Emulated Keyboard Controller**



# MEC1618/MEC1618i

## 9.1.3 BLOCK DIAGRAM SIGNAL LIST

**TABLE 9-1:** 8042 Emulated Keyboard Controller SIGNAL LIST

Signal Name	Direction	Description
EC_IBF	Output	Interrupt generated by the host writing either data or command to the data register
EC_OBF	Output	Interrupt generated by the host reading either data or aux data from the data register
KIRQ	Output	Routed to the Host SIRQ
MIRQ	Output	Routed to the Host SIRQ
KBRST	Output	Routed to Pin Function

## 9.2 Power, Clocks and Reset

### 9.2.1 POWER DOMAIN

This block is powered by VTR.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

### 9.2.2 CLOCKS

This block has four clock inputs: [LPC Bus Clock](#), 1MHZ ([MCLK\\_DIV20\\_EN\\_HST](#)) clock source. [LPC Bus Clock](#) is used to the accessible and clock the registers in this block. The 1MHz [Host Clock Domain](#) is used to clock the counter in the CPU\_RESET circuitry.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

## 9.3 Power On Reset

This block is reset on a [nSYS\\_RST](#) and [nSIO\\_RESET](#). On a reset, all Register are reset to 00h and the state machines are set to idle.

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 9.4 Interrupts

The [8042 Emulated Keyboard Controller](#) can generate a [KBD\\_OBF](#) interrupt when the [OBF](#) bit in the [Keyboard Status Read Register](#) is cleared (falling edge sensitive) and a [KBD\\_IBF](#) interrupt when the [IBF](#) bit in the [Keyboard Status Read Register](#) is set (active high, level sensitive). These interrupt sources are routed to the [GIRQ19 Source Register](#).

### 9.4.1 8042 EMULATED KEYBOARD CONTROLLER (LDN 1H) SIRQ ROUTING

The [8042 Emulated Keyboard Controller](#) can generate two SIRQ events for the EC-to-HOST EC events: [KIRQ](#) & [MIRQ](#). For the [KIRQ](#) interrupt the Interrupt Configuration Register, [SELECT](#) on [page 66](#) is cleared to '0' and for [MIRQ](#) the Interrupt Configuration Register, [SELECT](#) is set to '1'.

See [Section 4.0, "Logical Device Configuration", Section 4.8, "SERIRQ Interrupts," on page 65](#).

## 9.5 Instance Description

There are two blocks defined in this chapter: [8042 MSG Interface](#) and the [Port92-Legacy](#). The MEC1618/MEC1618i has one instance of each block.



## 9.6 Registers

The [8042 Emulated Keyboard Controller](#) has a [8042 MSG Interface](#) block which has its own Logical Device Number, and Base Address as indicated in [Table 9-2](#). The Host LPC I/O addresses for the [8042 MSG Interface](#) is selected via Base Address Register (see [Section 4.6.2, "Base Address Registers," on page 60](#)). LPC access to configuration registers is through Host Access Configuration Port (see [Section 4.5.1, "Host Access Port," on page 58](#).)

The [8042 Emulated Keyboard Controller](#) also has a [Port92-Legacy](#) block which has a separate Logical Device Number and Base Address Register as indicated in [Table 9-2](#). The Base Address Register for the [Port92-Legacy](#) has only one writable bit, the Valid Bit, since the only I/O accessible Register has a fixed address.

[Table 9-3](#) is a register summary for the [8042 MSG Interface](#) block and [Table 9-14 on page 209](#) is a register summary for the [Port92-Legacy](#) block.

**TABLE 9-2:      [8042 Emulated Keyboard Controller](#) BASE ADDRESS TABLE**

<a href="#">8042 Emulated Keyboard Controller</a> Blocks	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
<a href="#">8042 MSG Interface</a>	1h	<a href="#">FF_0400h</a>
<a href="#">Port92-Legacy</a>	8h	<a href="#">FF_2000h</a>

**Note 9-1**      The Host LPC I/O addresses for this instance is selected via a Base Address Register (see [Section 4.6.2, "Base Address Registers," on page 60](#)). LPC access to configuration registers is through the Host Access Configuration Port (see [Section 4.5.1, "Host Access Port," on page 58](#).)

[Table 9-3](#) is a register summary for the [8042 MSG Interface](#) block. The LPC I/O address for each Run-Time Register is described below as an offset from its Base Address Register. Each EC address is indicated as an SPB Offset from its AHB base address. Each Configuration register access through the [Host Access Port](#) is via its LDN indicated in [Table 9-2 on page 201](#) and its [Host Access Port](#) index which is described as "Host Config Index" in the tables below.

# MEC1618/MEC1618i

**TABLE 9-3:**    8042 MSG Interface REGISTER SUMMARY

Register Name	Host I/O Access			CMD Note 9-2	EC Interface			Notes
	Host I/O Index	SPB Offset	Host Type		SPB Offset	Byte Lane	EC Type	
HOST_EC Data/CMD Register	00h	00h	W	0	100h	0	R	
	04h	04h		1		0	R	
EC_HOST Data/AUX Data Register	00h	00h	R	0	100h	0	W	
				0	10Ch	0		
Keyboard Status Read Register	04h	04h	R	0	104h	3	R/W	
PCOBF Register	-	-	-	0	114h	0	R/W	
Keyboard Control Register	-	-	-	0	108h	0	R/W	
	Host Access			N/A	EC Interface			
Register Name	Host Config. Index	SPB Offset	Host Type		EC Offset	Byte Lane	EC Type	
Activate Register	30h	330h	R/W		330h	0	R/W	

**Note 9-2**    CMD is bit D3 in the [Keyboard Status Read Register](#).

**Note 9-3**    All Registers listed in [Table 9-3](#) are powered by VTR and reset by [nSYS\\_RST](#).

## 9.7 8042 MSG Interface Configuration Registers

### 9.7.1 ACTIVATE REGISTER

**TABLE 9-4:** 8042 MSG Interface ACTIVATE REGISTER

HOST CONFIG INDEX	330h				8-bit			HOST SIZE	
EC OFFSET	NA				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R							R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	Reserved							Activate	

#### ACTIVATE

0=1MHz Clock to the CPU\_REST block is disabled, ONLY if the PGEN is not active

1=1MHz Clock to the CPU\_REST block is enabled

# MEC1618/MEC1618i

## 9.8 8042 MSG Interface Runtime Registers

### 9.8.1 HOST\_EC DATA / CMD REGISTER

This is an 8-bit HOST write-only register. When written with Data, the **C/D** status bit of the status register is cleared to zero and the **IBF** bit is set.

When written with a command, the **C/D** status bit of the status register is set to one and the **IBF** bit is set to a 1.

TABLE 9-5: HOST\_EC DATA/CMD REGISTER

HOST I/O INDEX	00h Data 04h Command			8-bit			HOST SIZE	
EC OFFSET	100h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	W	W	W	W	W	W	W	W
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Data							

### 9.8.2 EC\_HOST DATA / AUX DATA REGISTER

This is an 8-bit read-only register. When read, by the HOST, the **PCOBF** and/or **AUXOBF** interrupts are cleared and the **OBF** flag in the status register is cleared.

TABLE 9-6: EC\_HOST DATA/AUX DATA REGISTER

HOST I/O INDEX	00h			8-bit			HOST SIZE	
EC OFFSET	100h Data or 10Ch Aux Data			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	W	W	W	W	W	W	W	W
BIT NAME	Data							

## 9.8.3 KEYBOARD STATUS READ

**TABLE 9-7: KEYBOARD STATUS READ REGISTER**

HOST I/O INDEX	4h			8-bit			HOST SIZE	
EC OFFSET	104h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R/W	R/W	R/W	R/W	R	R/W	R	R
BIT NAME	UD	UD	AUXOB F	UD	C/D	UD <sup>1</sup>	IBF	OBF

This register is read-only for the Host and read/write by the EC. The EC cannot write to bits 0, 1, or 3 of the Status register.

**APPLICATION NOTE:** The **IBF** and **OBF** bits are not cleared ('0') by hardware when **VCC\_PWRGD** is asserted or when the LPC interface powers down; for example, following system changes state S3->S0, S5->S0, G3-> S0. To clear the **IBF** bit in firmware, read offset 100h; to clear the **OBF** bit in firmware, read offset 000h.

### UD<sup>1</sup>

Read/Write by EC. These bits are user-definable. This bit is reset to '0' when LRESET# pin signal function is asserted. See [Table 3.7, "Host Interface," on page 57](#).

### UD

Read/Write by EC. These bits are user-definable.

### C/D

Command Data - This bit specifies whether the input data register contains data or a command ("0" = data, "1" = command). During a host command write operation, this bit is set to "1", during a host data write operation, this bit is set to "0".

### IBF

Input Buffer Full - This flag is set to "1" whenever the host system writes data or a command into the [HOST\\_EC Data/CMD Register](#). Setting this flag activates the EC's [EC\\_IBF](#) interrupt if enabled. When the EC reads the [HOST\\_EC Data / CMD Register](#), this bit is automatically reset and the interrupt is cleared.

### OBF

The Output Buffer Full (OBF) bit is set when the EC writes a byte of Data or AUX Data into the [EC\\_HOST Data / AUX Data Register](#). When the host reads the data, the **OBF** bit is automatically cleared by hardware and a [EC\\_OBF](#) interrupt is generated.

### AUXOBF

Auxiliary Output Buffer Full - This flag is set to "1" whenever the EC writes AUX Data into [EC\\_HOST Data / AUX Data Register](#). This flag is reset to "0" whenever the EC writes into the Data into [EC\\_HOST Data / AUX Data Register](#).

# MEC1618/MEC1618i

TABLE 9-8: PCOBF REGISTER

HOST OFFSET	NA				8-bit		HOST SIZE	
EC OFFSET	114h				8-bit		EC SIZE	
POWER	VTR				00h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R							R/W
BIT NAME	Reserved							PCOBF

**Note 9-4** Refer to the [PCOBF Description](#) for information on this register. This is a “1” bit register (bits 1-7=0 on read).

## 9.9 EC-to-Host Keyboard Communication

The EC can write to the [EC\\_HOST Data/AUX Data Register](#) via EC OFFSET 100h or EC OFFSET 10Ch (Aux Host Data) respectively. A write to either of these addresses automatically sets bit 0 (OBF) in the Status register. A write to EC OFFSET 100h also sets [PCOBF](#). A write to EC OFFSET 10Ch also sets [AUXOBF](#). See [Table 9-9](#).

TABLE 9-9: HOST-INTERFACE FLAGS

EC Address	Flag
EC OFFSET 100h (R/W)	<a href="#">PCOBF</a> (KIRQ) output signal goes high
EC OFFSET 10Ch (W)	<a href="#">AUXOBF</a> (MIRQ) output signal goes high

The [HOST\\_EC Data / CMD Register](#) and [EC\\_HOST Data / AUX Data Register](#) are each 8-bits wide. A write to this 8-bit register by the EC loads the [EC\\_HOST Data / AUX Data Register](#), sets the [OBF](#) flag and sets the [PCOBF](#) output if enabled.

**Note 9-5** Refer to the [PCOBF](#) and [Keyboard Status Read Register](#) descriptions for more information.

### 9.9.1 PCOBF DESCRIPTION

(The following description assumes that OBFEN = 1 in [Keyboard Control Register on page 208](#)); PCOBF is gated onto KIRQ. The KIRQ signal is a system interrupt which signifies that the EC has written to the KBD Data Read register via EC OFFSET 100h. On power-up, PCOBF is reset to 0. PCOBF will normally reflect the status of writes to EC OFFSET 100h, if PCOBFEN (bit 2 of Configuration register “0”) = “0”. (KIRQ is normally selected as IRQ1 for keyboard support). PCOBF is cleared by hardware on a HOST read of the [EC\\_HOST Data / AUX Data Register](#)

Additional flexibility has been added which allows firmware to directly control the PCOBF output signal, independent of data transfers to the host-interface data output register. This feature allows the MEC1618/MEC1618i to be operated via the host “polled” mode. This firmware control is active when PCOBFEN = 1 and firmware can then bring PCOBF high by writing a “1” to the LSB of the 1-bit data register, PCOBF, at EC OFFSET 114h. The firmware must also clear this bit by writing a “0” to the LSB of the 1-bit data register at EC OFFSET 114h.

The PCOBF register is also readable; bits 1-7 will return a “0” on the read back. The value read back on bit 0 of the register always reflects the present value of the PCOBF output. If PCOBFEN = 1, then this value reflects the output of the firmware latch at EC OFFSET 114h. If PCOBFEN = 0, then the value read back reflects the in-process status of write cycles to EC OFFSET 100h (i.e., if the value read back is high, the host interface output data register has just been written to). If OBFEN=0, then KIRQ is driven inactive (low).

## 9.9.2 AUXOBF1 DESCRIPTION

(The following description assumes that OBFEN = 1 in [Keyboard Control Register on page 208](#)); This bit is multiplexed onto MIRQ. The AUXOBF1/MIRQ signal is a system interrupt which signifies that the EC has written to the [EC\\_HOST Data / AUX Data Register](#).

On power-up, after [nSYS\\_RST](#), AUXOBF1 is reset to 0. AUXOBF1 will normally reflects the status of writes to EC OFFSET 10Ch. (MIRQ is normally selected as IRQ12 for mouse support). AUXOBF1 is cleared by hardware on a read of the Host Data Register. If OBFEN=0, then KIRQ is driven inactive (low).

**TABLE 9-10: STATUS AND INTERRUPT BEHAVIOR OF WRITING TO OUTPUT DATA REGISTER**

Write to Register	Host I/F Status Register Bits		OBFEN=0	OBFEN=1
	AUXOBF (D5)	OBF (D0)		
EC OFFSET 100h	0	1	KIRQ=0	KIRQ=1
EC OFFSET 10Ch	1	1	MIRQ=0	MIRQ=1

**TABLE 9-11: OBFEN AND PCOBFEN EFFECTS ON KIRQ**

OBFEN	PCOBFEN	
0	X	KIRQ is inactive and driven low
1	0	KIRQ = PCOBF@EC OFFSET 100h
1	1	KIRQ = PCOBF@EC OFFSET 114h

**TABLE 9-12: OBFEN AND AUXH EFFECTS ON MIRQ**

OBFEN	AUXH	
0	X	MIRQ is inactive and driven low
1	0	MIRQ = PCOBF@EC OFFSET 10Ch; Status Register D5 = User Defined
1	1	MIRQ = PCOBF@EC OFFSET 10Ch; Status Register D5 = Hardware Controlled

# MEC1618/MEC1618i

## 9.9.2.1 Keyboard Control

TABLE 9-13: KEYBOARD CONTROL REGISTER

HOST OFFSET	NA			8-bit			HOST SIZE	
EC OFFSET	108h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	AUXH	UD	OBFEN	UD	UD	PCOB-FEN	SAEN	UD

### UD

user defined bit

### AUXH

Aux in Hardware; when high, [AUXOBF](#) of the status register is set in hardware by a write to EC OFFSET 10Ch. When low, [AUXOBF](#) of the status register is a user defined bit (UD) and R/W.

### OBFEN

When set, [PCOBF](#) is gated onto KIRQ and [AUXOBF1](#) is gated onto MIRQ. When low, KIRQ and MIRQ are driven low. Software should not change this bit when [OBF](#) of the status register is equal to 1.

### PCOBFEN

When high, [PCOBF](#) reflects whatever value was written to the [PCOBF](#) firmware latch assigned to 114h. When low, [PCOBF](#) reflects the status of writes to EC OFFSET 100h (the output data register).

### SAEN

Software-assist enable. When set to “1,” [SAEN](#) allows control of the [GATEA20](#) signal via firmware. If [SAEN](#) is reset to ‘0’, [GATEA20](#) corresponds to either the last host-initiated control of [GATEA20](#) or the firmware write to EC OFFSETs 108h or 10Ch.

## 9.10 Legacy Support

### 9.11 Port92-Legacy Registers

The [Table 9-14](#) is a register summary for the [Port92-Legacy](#) block. The LPC I/O address of the [PORT92 Register](#) is fixed. Each EC address is indicated as an SPB Offset from its AHB base address listed for the [Port92-Legacy](#) block in [Table 9-2, “8042 Emulated Keyboard Controller Base Address Table,” on page 201](#). Each Configuration register access through the [Host Access Port](#) address via its LDN indicated in [Table 9-2 on page 201](#) and its [Host Access Port](#) index which is described as “Host Config Index” in the tables below.



**TABLE 9-14: Port92-Legacy SUPPORT REGISTER SUMMARY**

	Host I/O Access			EC Interface			Notes
Register Name	Host I/O Address	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
<a href="#">GATEA20 Control Register</a>	-	-	-	100h	0	R/W	
<a href="#">SETGA20L Register</a>	-	-	-	108h	0	W	
<a href="#">RSTGA20L Register</a>	-	-	-	10Ch	0	W	
<a href="#">PORT92 Register</a>	92h	000h	R/W	000h	0	R/W	
	Host Access			EC Interface			
Register Name	Host Config. Index	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
<a href="#">PORT92 Enable Register</a>	30h	3F0h	R/W	330h	0	R/W	

**Note 9-6** CMD is bit D3 in the [Keyboard Status Read Register](#).

All Registers listed in [Table 9-14](#) are powered by VTR and reset by [nSYS\\_RST](#) except the [PORT92 Register](#) which is powered by VTR and reset by [nSIO\\_RESET](#) (See [Section , "iRESET OUT," on page 114](#)).

## 9.12 Configuration Registers

### 9.12.1 PORT 92 ENABLE

The MEC1618/MEC1618i supports LPC I/O writes to port 92h as a quick alternate mechanism for generating a CPU\_RESET pulse or controlling the state of [GATEA20](#).

**TABLE 9-15: PORT92 ENABLE REGISTER**

HOST CONFIG INDEX	30h				8-bit			HOST SIZE	
EC OFFSET	330h				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R							R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	Reserved							P92_EN	

# MEC1618/MEC1618i

## 9.13 Runtime Registers

### 9.13.1 PORT 92

The MEC1618/MEC1618i supports LPC I/O writes to port HOST I/O address 92h as a quick alternate mechanism for generating a CPU\_RESET pulse or controlling the state of [GATEA20](#).

The [PORT92 Register](#) resides at HOST I/O address 92h and is used to support the alternate reset (ALT\_RST#) and alternate [GATEA20](#) (ALT\_A20) functions. This register defaults to 00h on assertion of [nSIO\\_RESET](#) (See [Section , "iRESET OUT," on page 114](#)).

Setting the Port92 Enable bit ([PORT92 Enable Register](#)) enables the Port92h Register. When Port92 is disabled, by clearing the Port92 Enable bit, then access to this register is completely disabled (I/O writes to host 92h are ignored and I/O reads float the system data bus SD[7:0]).

**TABLE 9-16: PORT92 REGISTER**

HOST I/O ADDRESS	0092h				8-bit			HOST SIZE	
EC OFFSET	00h				8-bit			EC SIZE	
POWER	VTR				00h			nSIO_RESET DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R						R/W	R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	Reserved						AL-T_GATEA20	AL-T_CPU_RESET	

#### ALT\_CPU\_RESET

This bit provides an alternate means to generate a CPU\_RESET pulse. The CPU\_RESET output provides a means to reset the system CPU to effect a mode switch from Protected Virtual Address Mode to the Real Address Mode. This provides a faster means of reset than is provided through the EC keyboard controller. Writing a "1" to this bit will cause the ALT\_RST# internal signal to pulse (active low) for a minimum of 6μs after a delay of 14μs. Before another ALT\_RST# pulse can be generated, this bit must be written back to "0".

#### ALT\_GATEA20

This bit provides an alternate means for system control of the MEC1618/MEC1618i GATEA20 pin.

0=ALT\_A20 is driven low

1=ALT\_A20 is driven high

When Port 92 is enabled, writing a 0 to bit 1 of the [PORT92 Register](#) forces ALT\_A20 low. ALT\_A20 low drives GATEA20 low, if A20 from the keyboard controller is also low. When Port 92 is enabled, writing a 1 to bit 1 of the [PORT92 Register](#) forces ALT\_A20 high. ALT\_A20 high drives GATEA20 high regardless of the state of A20 from the keyboard controller.

### 9.13.2 GATE A20

The MEC1618/MEC1618i contains on-chip logic support for the [GATEA20](#) hardware speed-up feature. [GATEA20](#) is part of the control required to mask address line A20 to emulate 8086 addressing.

In addition to the ability for the host to control the [GATEA20](#) output signal directly, a configuration bit called “[SAEN](#)” (Software Assist Enable, bit 1 of [Keyboard Control Register](#) is provided; when set, SAEN allows firmware to control the [GATEA20](#) output.

When SAEN is set, a 1-bit register ([GATEA20 Control Register](#)) controls the [GATEA20](#) output. The register bit allocation is shown in [Table 9-13](#).

**Note 9-7** Refer to the [GATEA20 Control](#) description for information on this register. This is a one bit register (Bits 1-7=0 on read).

## 9.13.3 GATEA20 CONTROL

**TABLE 9-17: GATEA20 CONTROL REGISTER**

HOST OFFSET	NA				8-bit		HOST SIZE	
EC OFFSET	100h				8-bit		EC SIZE	
POWER	VTR				00h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R							R/W
BIT NAME	Reserved							GATEA20

### GATEA20

Writing a “0” into bid [GATEA20](#) causes the [GATEA20](#) output to go low, and vice versa.

Host control and firmware control of [GATEA20](#) affect two separate register elements. Read back of [GATEA20](#) through the use of EC OFFSET 100h reflects the present state of the [GATEA20](#) output signal: if [SAEN](#) is set, the value read back corresponds to the last firmware-initiated control of [GATEA20](#); if [SAEN](#) is reset, the value read back corresponds to the last host-initiated control of [GATEA20](#).

Host control of the [GATEA20](#) output is provided by the hardware interpretation of the “[GATEA20](#) sequence” (see [Table 9-18](#)). The foregoing description assumes that the [SAEN](#) configuration bit is reset.

When the MEC1618/MEC1618i receives a “D1” command followed by data (via the host interface), the on-chip hardware copies the value of data bit 1 in the received data field to the [GATEA20](#) host latch. At no time during this host-interface transaction will [PCOBF](#) or the [IBF](#) flag (bit 1) in the [Keyboard Status Read Register](#) be activated; for example, this host control of [GATEA20](#) is transparent to firmware, with no consequent degradation of overall system performance. [Table 9-18](#) details the possible [GATEA20](#) sequences and the MEC1618/MEC1618i responses.

On [VCC\\_POR](#), [GATEA20](#) will be set.

An additional level of control flexibility is offered via a memory-mapped synchronous set and reset capability. Any data written to EC OFFSET 108h causes the [GATEA20](#) host latch to be set; any data written to EC OFFSET 10Ch causes it to be reset. This control mechanism should be used with caution. It was added to augment the “normal” control flow as described above, not to replace it. Since the host and the firmware have asynchronous control capability of the host latch via this mechanism, a potential conflict could arise. Therefore, after using the EC OFFSET 108h and EC OFFSET 10Ch addresses, firmware should read back the [GATEA20](#) status via EC OFFSET 100h (with [SAEN](#) = 0) to confirm the actual [GATEA20](#) response.

# MEC1618/MEC1618i

**TABLE 9-18: GATEA20 COMMAND/DATA SEQUENCE EXAMPLES**

Data Byte	R/W	D[0:7]	IBF FLAG	GATEA20	Comments
1 0 1	W W W	D1 DF FF	0 0 0	Q 1 1	GATEA20 Turn-on Sequence
1 0 1	W W W	D1 DD FF	0 0 0	Q 0 0	GATEA20 Turn-off Sequence
1 1 0 1	W W W W	D1 D1 DF FF	0 0 0 0	Q Q 1 1	GATEA20 Turn-on Sequence(*)
1 1 0 1	W W W W	D1 D1 DD FF	0 0 0 0	Q Q 0 0	GATEA20 Turn-off Sequence(*)
1 1 1	W W W	D1 XX** FF	0 1 1	Q Q Q	Invalid Sequence

**Note 9-8**

- All examples assume that the SAEN configuration bit is 0.
- “Q” indicates the bit remains set at the previous state.
- \*Not a standard sequence.
- \*\*XX = Anything except D1.
- If multiple data bytes, set IBF and wait at state 0. Let the software know something unusual happened.
- For data bytes, only D[1] is used; all other bits are don't care.

**TABLE 9-19: SETGA20L REGISTER**

HOST OFFSET	NA				8-bit			HOST SIZE	
EC OFFSET	108h				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	W								
BIT NAME									

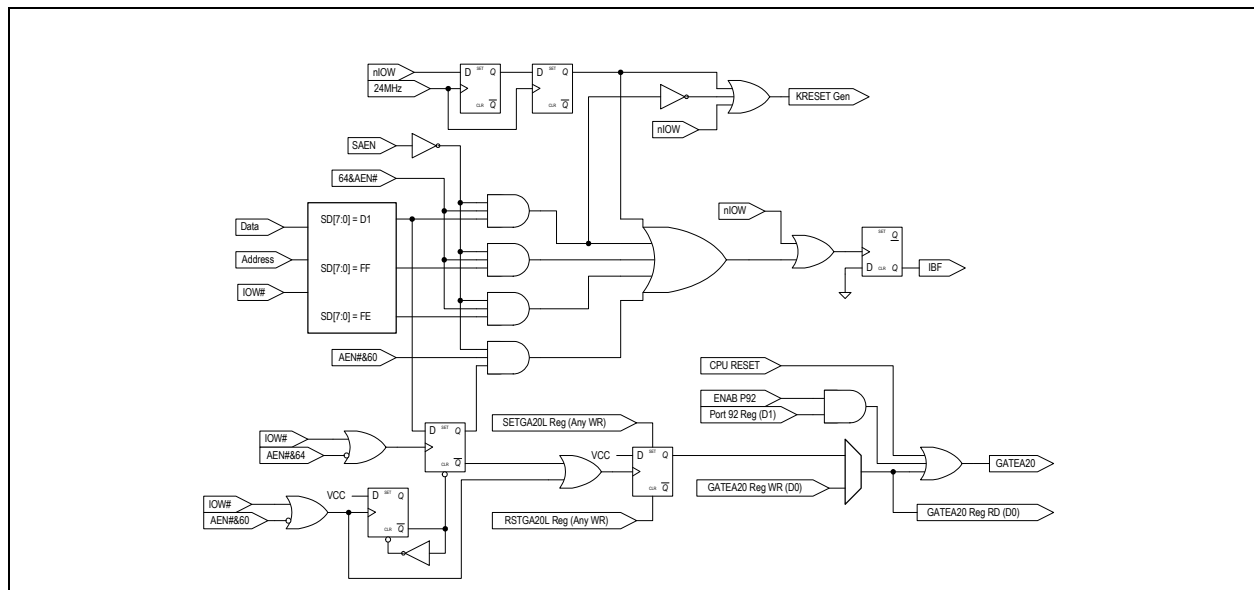
**Note 9-9** Refer to the [GATEA20](#) Hardware Speed-up description for information on this register. A write to this register sets [GATEA20](#).

**TABLE 9-20: RSTGA20L REGISTER**

HOST OFFSET	NA				8-bit			HOST SIZE	
EC OFFSET	10Ch				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	W								
BIT NAME									

**Note 9-10** Refer to the [GATEA20](#) Hardware Speed-up description for information on this register. A write to this register re-sets [GATEA20](#).

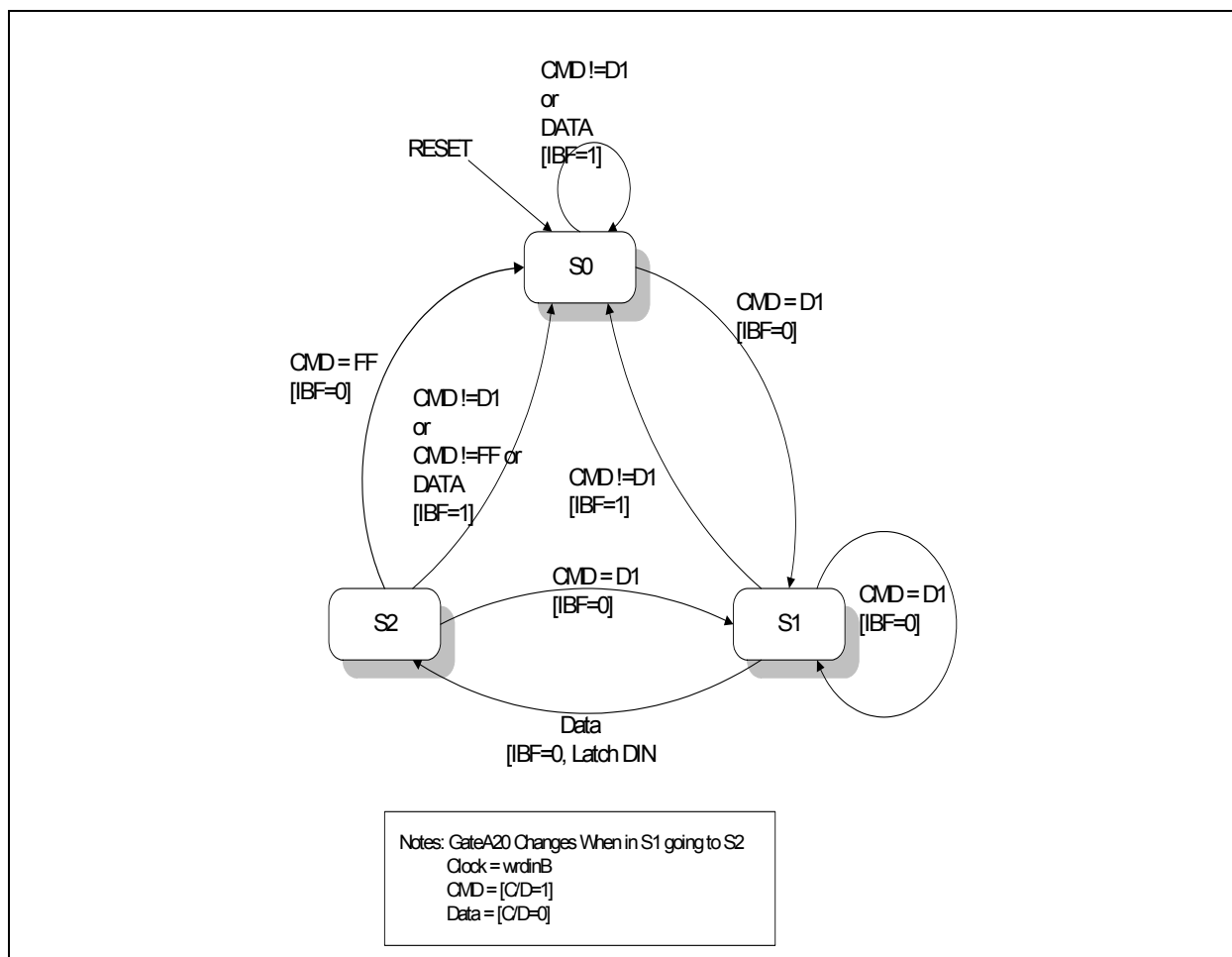
**FIGURE 9-2: [GATEA20](#) IMPLEMENTATION DIAGRAM**



**Note 9-11** Host Commands (FF, FE, & D1) do not cause IBF. The method of blocking IBF in [Figure 9-2](#) is the nIOW not being asserted when FF, FE, & D1 Host commands are written”.

The hardware [GATEA20](#) state machine returns to state S1 from state S2 when CMD = D1 ([Figure 9-3](#)).

FIGURE 9-3: GATEA20 STATE MACHINE



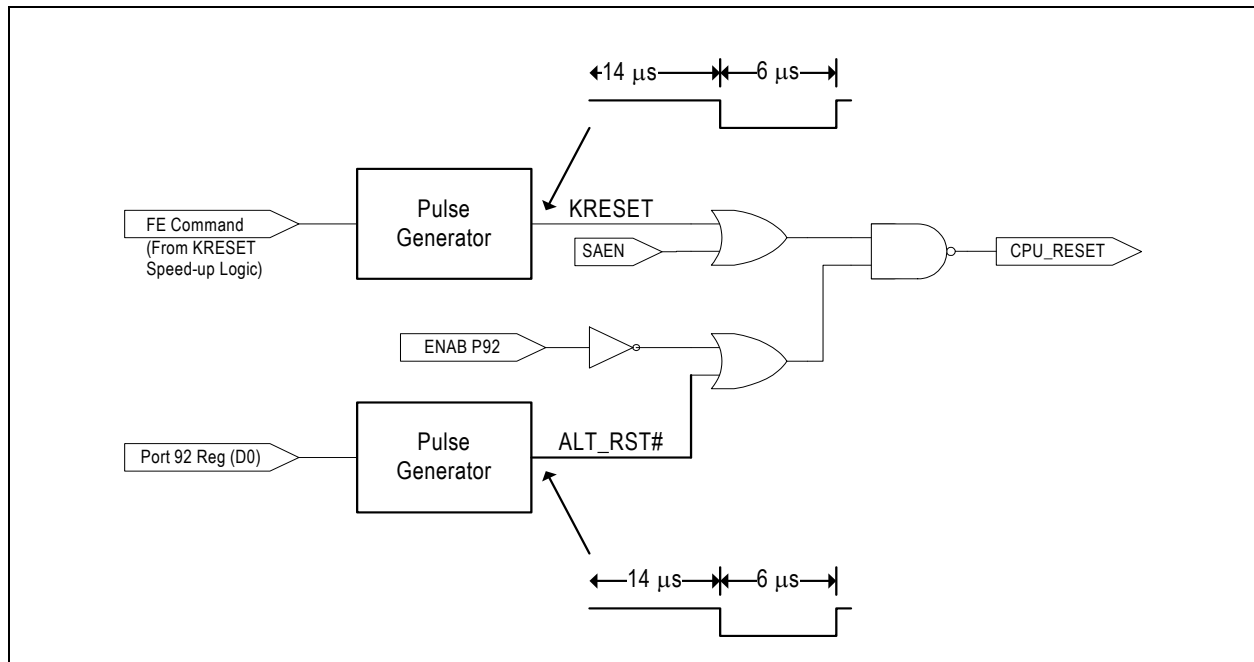
## 9.14 CPU\_RESET Hardware Speed-Up

The [ALT\\_CPU\\_RESET](#) bit generates, under program control, the ALT\_RST# signal, which provides an alternate, means to drive the MEC1618/MEC1618i CPU\_RESET pin which in turn is used to reset the Host CPU. The ALT\_RST# signal is internally Nanded together with the KBDRESET# pulse from the KRESET Speed up logic to provide an alternate software means of resetting the host CPU.

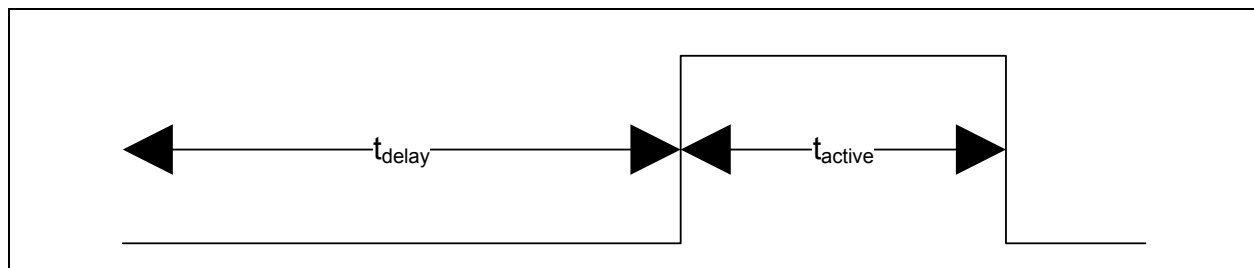
**Note 9-12** Before another ALT\_RST# pulse can be generated, [ALT\\_CPU\\_RESET](#) must be cleared to "0" either by an [nSIO\\_RESET](#) (See [Section , "iRESET OUT," on page 114](#)) or by a write to the [PORT92 Register](#) with bit 0 = "0". A ALT\_RST# pulse is not generated in the event that the [ALT\\_CPU\\_RESET](#) bit is cleared and set before the prior ALT\_RESET# pulse has completed.

**Note 9-13** This function is qualified by the SLP\_EN signal or the [Activate](#) bit. If either of these signals goes to '0', then the 1MHz clocks source is disabled if the PGEN is not currently active. See [EC Blocks Sleep Enables/Clock Required Registers Bit Names on page 122](#) & [Section 5.4.8.5.4, "Block Sleep Enables," on page 97](#).

**FIGURE 9-4: CPU\_RESET IMPLEMENTATION DIAGRAM**



**FIGURE 9-5: CPU\_RESET TIMING**



**TABLE 9-21: CPU\_RESET TIMING PARAMETERS**

Name	Description	Min	Typ	Max	Units
$t_{\text{delay}}$	Delay prior to active pulse	14	15	15.5	$\mu\text{s}$
$t_{\text{active}}$	Active pulse width	6	8	8.5	$\mu\text{s}$

**Note 9-14** Figure 9-5 & Figure 9-21 refers to Figure 9-4 in which CPU\_RESET is the inverse of ALT\_RST# & KRESET.

**Note 9-15** The KBRST pin function is the output of CPU\_RESET described in Section 9.14, "CPU\_RESET Hardware Speed-Up," on page 214.

## 10.0 ACPI PM1 BLOCK INTERFACE

### 10.1 General Description

The MEC1618/MEC1618i supports ACPI as described in this section. These features comply with the ACPI Specification, Revision 1.0, through a combination of hardware and EC software.

The MEC1618/MEC1618i implements the ACPI fixed registers but includes only those bits that apply to the power button sleep button and RTC alarm events. The ACPI [WAK\\_SLP\\_TYPx](#), and [SLP\\_](#) bits are also supported.

### 10.2 Power, Clocks and Reset

#### 10.2.1 POWER DOMAIN

This block is powered by VTR

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

#### 10.2.2 CLOCKS

This block has one clock input, the [LPC Bus Clock](#).

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

#### 10.2.3 POWER ON RESET

This block is reset on a [nSYS\\_RST](#). After [nSYS\\_RST](#) is asserted, all registers are set to '0'.

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

### 10.3 Interrupts

#### 10.3.1 SCI INTERRUPTS TO THE HOST

The functions described in the following sub-sections can generate a SCI event on the EC\_SCI# pin. In the MEC1618/MEC1618i, an SCI event is considered the same as an ACPI wakeup or runtime event. The EC can also generate a SCI on the EC\_SCI# pin by setting the [EC\\_SCI\\_](#) bit in the [EC\\_SCI# Pin Interface on page 224](#).

#### 10.3.2 INTERRUPTS TO THE EC

An Interrupt is generated to the EC on [PM1\\_CTL2](#) bit of [GIRQ13 Source Register](#) by the Host writing to [Power Management 1 Control Register 2 \(PM1\\_CNTRL 2\)](#)

An Interrupt is generated to the EC on [PM1\\_EN2](#) bit of [GIRQ13 Source Register](#) by the Host writing to [Power Management 1 Enable Register 2 \(PM1\\_EN 2\)](#)

An Interrupt is generated to the EC on [PM1\\_STS2](#) bit of [GIRQ13 Source Register](#) by the Host writing to [Power Management 1 Status Register 2 \(PM1\\_STS 2\)](#)

#### 10.3.3 ACPI PM1 BLOCK SCI EVENT-GENERATING FUNCTIONS

##### 10.3.3.1 Power Button with Override

The power button has a status and an enable bit in the [PM1\\_BLK](#) of registers to provide an SCI upon the button press. The status bit is software Read/Writable by the EC; the enable bit is Read-only by the EC. It also has a status and enable bit in the [PM1\\_BLK](#) of registers to indicate and control the power button override (fail-safe) event. These bits are not required by ACPI. The power button override event status bit is software Read/Writable by the EC; the enable bit is software read-only by the EC. The enable bit for the override event is located at bit 1 in the [Power Management 1 Control Register 2 \(PM1\\_CNTRL 2\)](#).

The [PWRBTN\\_](#) bit is set by the Host to enable the generation of an SCI due to the power button event. The status bit is set by the EC when it generates a power button event and is cleared by the Host writing a '1' to this bit (writing a '0' has no effect); it can also be cleared by the EC. If the enable bit is set, the EC will generate an SCI power management event.



### 10.3.3.2 Sleep Button

The sleep button has a status and an enable bit in the PM1\_BLK of registers to provide an SCI upon the button press. The status bit is software Read/Writable by the EC; the enable bit is Read-only by the EC.

The [SLPBTN](#) bit is set by the Host to enable the generation of an SCI due to the sleep button event. The status bit is set by the EC when it generates a sleep button event and is cleared by the Host writing a '1' to this bit (writing a '0' has no effect); it can also be cleared by the EC. If the enable bit is set, the EC will generate an SCI power management event.

### 10.3.4 RTC ALARM

The ACPI specification requires that the RTC alarm generate a hardware wake-up event from the sleeping state. The RTC alarm can be enabled as an SCI event and its status can be determined through bits in the PM1\_BLK of registers. The status bit is software Read/Writable by the EC; the enable bit is Read-only by the EC.

The [RTC](#) bit is set by the Host to enable the generation of an SCI due to the RTC alarm event. The status bit is set by the EC when the RTC generates an alarm event and is cleared by the Host writing a '1' to this bit (writing a '0' has no effect); it can also be cleared by the EC. If the enable bit is set, the EC will generate an SCI power management event.

## 10.4 Registers

Each instance of the [ACPI PM1 Block Interface](#) has its own Logical Device Number, and Base Address as indicated in [Table 10-1](#).

**TABLE 10-1: [ACPI PM1 Block Interface](#) BASE ADDRESS TABLE**

<a href="#">ACPI PM1 Block Interface</a> Instance	<a href="#">Table 3-1 on page 51</a> LDN	AHB Base Address
<a href="#">ACPI PM1 Block Interface</a>	6h	<a href="#">FF_1800h</a>

The ACPI register model consists of a number of fixed register blocks that perform designated functions. A register block consists of a number of registers that perform Status, Enable and Control functions. The ACPI specification deals with events (which have an associated interrupt status and enable bits, and sometimes an associated control function) and control features. The status registers illustrate what defined function is requesting ACPI interrupt services (SCI). Any status bit in the ACPI specification has the following attributes:

Status bits are only set through some defined hardware or EC event.

Unless otherwise noted, status bits are cleared by the system writing a "1" to that bit position, and upon [nSYS\\_RST](#). Writing a '0' has no effect.

Status bits only generate interrupts while their associated bit in the enable register is set.

Function bit positions in the status register have the same bit position in the enable register (there are exceptions to this rule, special status bits have no enables).

Note that this implies that if the respective enable bit is reset and the hardware event occurs, the respective status bit is set; however no interrupt is generated until the enable bit is set. This allows software to test the state of the event (by examining the status bit) without necessarily generating an interrupt. There are a special class of status bits that have no respective enable bit, these are called out specifically, and the respective enable bit in the enable register is marked as reserved for these special cases.

The enable registers allow the setting of the status bit to generate an interrupt (under EC control). As a general rule, there is an enable bit in the enable register for every status bit in the status register. The control register provides special controls for the associated event, or special control features that are not associated with an interrupt event. The order of a register block is the status registers, followed by enable registers, followed by control registers.

The registers in the MEC1618/MEC1618i [ACPI PM1 Block Interface](#) occupy eight addresses in the host I/O space and are specified as offsets from the ACPI PM1 Block base address ([Table 10-1](#)).

# MEC1618/MEC1618i

**TABLE 10-2: ACPI PM1 Block Interface REGISTER SUMMARY**

Register Name	Host Access			EC Access			Notes
	Host I/O Index	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
Power Management 1 Status Register 1 (PM1_STS) 1	0h	00h	R	100h	0	R	
Power Management 1 Status Register 2 (PM1_STS) 2	1h	01h	R/WC	101h	1	R/W	Table 10-4 Note 10-4
Power Management 1 Enable Register 1 (PM1_EN) 1	2h	02h	R	102h	2	R	
Power Management 1 Enable Register 2 (PM1_EN) 2	3h	03h	R/W	103h	3	R	Table 10-6 Note 10-5
Power Management 1 Control Register 1 (PM1_CNTRL) 1	4h	04h	R	104h	0	R	
Power Management 1 Control Register 2 (PM1_CNTRL) 2	5h	05h	R/W	105h	1	R	Table 10-8 Note 10-6
Power Management 2 Control Register 1 (PM2_CNTRL) 1	6h	06h	R	106h	2	R	Note 10-2
Power Management 2 Control Register 2 (PM2_CNTRL) 2	7h	07h	R	107h	3	R	Note 10-2
EC_PM_STS Register	-	-	-	110h	0	R/W	

**Note 10-1** Byte 0 of this register is reserved.

**Note 10-2** These registers return '0' when read, writes have no effect.

## 10.4.1 POWER MANAGEMENT 1 STATUS 1 (PM1\_STS 1)

**TABLE 10-3: POWER MANAGEMENT 1 STATUS REGISTER 1 (PM1\_STS 1)**

HOST OFFSET	0h				8-bit			HOST SIZE	
EC OFFSET	100h							EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								

### RESERVED

Reserved bits return '0' when read.

## 10.4.2 POWER MANAGEMENT 1 STATUS 2 (PM1\_STS 2)

**TABLE 10-4: POWER MANAGEMENT 1 STATUS REGISTER 2 (PM1\_STS 2)**

HOST OFFSET	1h				8-bit			HOST SIZE	
EC OFFSET	101h							EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/WC	R	R	R	R/WC	R/WC	R/WC	R/WC	
EC TYPE	R/W	R	R	R	R/W	R/W	R/W	R/W	
BIT NAME	WAK_STSTS	Reserved			PWRBNTNOR_STSTS	RTC_STSTS	SLPBNTN_STSTS	PWRBNTN_STSTS	

**Note 10-3** These bits are set/cleared by the EC directly i.e., writing '1' sets the bit and writing '0' clears it. These bits can also be cleared by the Host software writing a one to this bit position and by nSYS\_RST. Writing a 0 by the Host has no effect.

**Note 10-4** An interrupt (PM1\_CTL2) is generated to the EC when the Host writes to this register.

### PWRBTN\_STS

This bit can be set or cleared by the EC to simulate a Power button status if the power is controlled by the EC. The Host writing a one to this bit can also clear this bit. The EC must generate the associated SCI interrupt under software control.

# MEC1618/MEC1618i

## SLPBTN\_STS

This bit can be set or cleared by the EC to simulate a Sleep button status if the sleep state is controlled by the EC. The Host writing a one to this bit can also clear this bit. The EC must generate the associated SCI interrupt under software control.

## RTC\_STS

This bit can be set or cleared by the EC to simulate a RTC status. The Host writing a one to this bit can also clear this bit. The EC must generate the associated SCI interrupt under software control.

## PWRBTNOR\_STS

This bit can be set or cleared by the EC to simulate a Power button override event status if the power is controlled by the EC. The Host writing a one to this bit can also clear this bit. The EC must generate the associated hardware event under software control.

## WAK\_STS

This bit can be set or cleared by the EC. The Host writing a one to this bit can also clear this bit.

### 10.4.3 POWER MANAGEMENT 1 ENABLE 1 (PM1\_EN 1)

**TABLE 10-5: POWER MANAGEMENT 1 ENABLE REGISTER 1 (PM1\_EN 1)**

HOST OFFSET	02h			8-bit			HOST SIZE	
EC OFFSET	102h						EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							

## RESERVED

Reserved bits return '0' when read.

## 10.4.4 POWER MANAGEMENT 1 ENABLE 2 (PM1\_EN 2)

**TABLE 10-6: POWER MANAGEMENT 1 ENABLE REGISTER 2 (PM1\_EN 2)**

HOST OFFSET	03h			8-bit			HOST SIZE	
EC OFFSET	103h						EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R/W	R/W	R/W
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved					RTC_EN	SLPBTN_EN	PWRBTN_EN

**Note 10-5** An interrupt (PM1\_EN2) is generated to the EC when the Host writes to this register.

### PWRBTN\_EN

This bit can be read or written by the Host. It can be read by the EC.

### SLPBTN\_EN

This bit can be read or written by the Host. It can be read by the EC.

### RTC\_EN

This bit can be read or written by the Host. It can be read by the EC.

## 10.4.5 POWER MANAGEMENT 1 CONTROL 1 (PM1\_CNTRL 1)

**TABLE 10-7: POWER MANAGEMENT 1 CONTROL REGISTER 1 (PM1\_CNTRL 1)**

HOST OFFSET	04h				8-bit			HOST SIZE	
EC OFFSET	104h							EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								

### RESERVED

Reserved bits return '0' when read.

# MEC1618/MEC1618i

## 10.4.6 POWER MANAGEMENT 1 CONTROL 2 (PM1\_CNTRL 2)

**TABLE 10-8: POWER MANAGEMENT 1 CONTROL REGISTER 2 (PM1\_CNTRL 2)**

HOST OFFSET	5h			8-bit			HOST SIZE	
EC OFFSET	105h						EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	W	R/W			R/W	R/W
EC TYPE	R	R	R/WC	R			R	R
BIT NAME	Reserved		SLP_EN	SLP_TYPx			PWRBTNOR_EN	Reserved

**Note 10-6** An interrupt ([PM1\\_CTL2](#)) is generated to the EC when the Host writes to this register.

### **PWRBTNOR\_EN**

This bit can be set or cleared by the Host, read by the EC.

### **SLP\_TYPx**

These bits can be set or cleared by the Host, read by the EC.

### **SLP\_EN**

Refer to [Table 10-9, "SLP\\_Definition"](#).

**TABLE 10-9: SLP\_DEFINITION**

Host / EC	R/W	Description
Host	Read	Always reads 0
	Write	Writing a 0 has no effect, Writing a 1 sets this bit
EC	Read	Reads the value of the bit
	Write	Writing a 0 has no effect, Writing a 1 clears this bit

## 10.4.7 POWER MANAGEMENT 2 CONTROL 1 (PM2\_CNTRL 1)

**TABLE 10-10: POWER MANAGEMENT 2 CONTROL REGISTER 1 (PM2\_CNTRL 1)**

HOST OFFSET	06h				8-bit			HOST SIZE	
EC OFFSET	106h							EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								

### RESERVED

Reserved bits return '0' when read.

## 10.4.8 POWER MANAGEMENT 2 CONTROL 2 (PM2\_CNTRL 2)

**TABLE 10-11: POWER MANAGEMENT 2 CONTROL REGISTER 2 (PM2\_CNTRL 2)**

HOST OFFSET	07h			8-bit			HOST SIZE	
EC OFFSET	107h						EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							

### RESERVED

Reserved bits return '0' when read.

# MEC1618/MEC1618i

## 10.5 EC\_SCI# Pin Interface

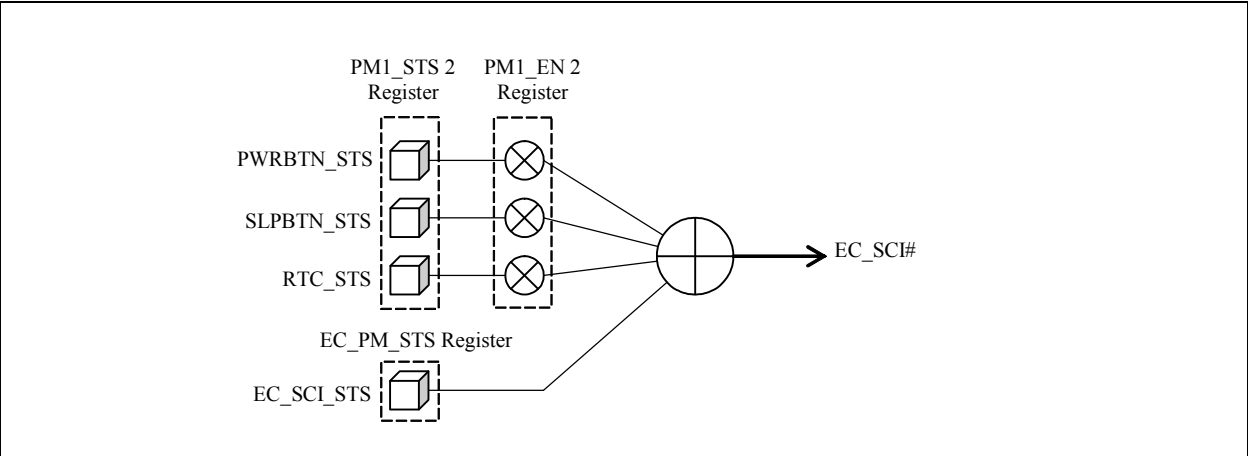
The EC\_SCI# pin logic hardware is shown below in [Figure 10-1](#).

Any or all of the [PWRBTN\\_STS](#), [SLPBTN\\_STS](#), and [RTC\\_](#) bits in the [Power Management 1 Status Register 2 \(PM1\\_STS 2\)](#) can assert the EC\_SCI# pin if enabled by the [PWRBTN\\_](#), [SLPBTN\\_](#), and [RTC\\_](#) bits in the [PM1\\_EN 2](#) register.

The [EC\\_SCI\\_](#) bit can assert the EC\_SCI# pin at any time, without being enabled. The [EC\\_SCI\\_](#) bit is located in the [EC\\_PM\\_STS Register](#).

The [EC\\_SCI\\_](#) bit is in the MEC1618/MEC1618i and is read/write by the EC. If the [EC\\_SCI\\_](#) bit is “1”, an interrupt is generated on the EC\_SCI# pin.

**FIGURE 10-1: HARDWARE EC\_SCI# INTERFACE**



### 10.5.1 EC\_PM\_STS REGISTER

**TABLE 10-12: EC\_PM\_STS REGISTER**

HOST OFFSET								HOST SIZE
EC OFFSET	110h							8-bit
POWER	VTR							00h
BUS	<a href="#">LPC SPB</a>							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	UD[6:0]							<a href="#">EC_SCI_</a> STS

#### EC\_SCI\_STS

If the [EC\\_SCI\\_](#) bit is “1”, an interrupt is generated on the EC\_SCI# pin.

#### UD[6:0]

User-defined bits. This bits do not generate an interrupt.



## 11.0 MAILBOX REGISTER INTERFACE

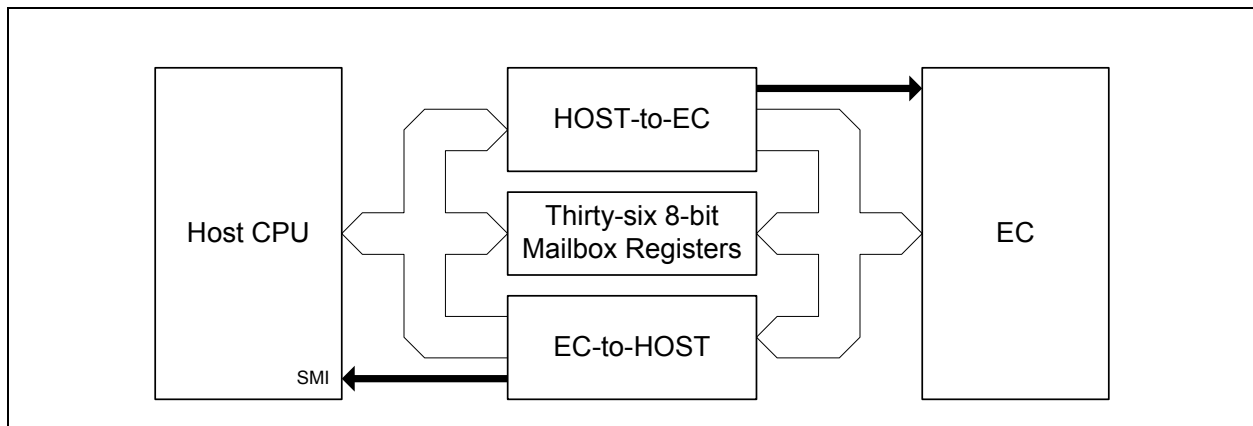
### 11.1 General Description

The [MailBox Register Interface](#) provides a standard run-time mechanism for the host to communicate with the Embedded Controller (EC) and other logical components in the MEC1618/MEC1618i ([Figure 11-1](#)). The Mailbox Registers Interface includes a total of 36 index-addressable 8-bit registers and a [Mailbox Registers Interface Host Access Port](#). Thirty-two of 36 index-addressable 8-bit registers are EC Mailbox registers. The [Mailbox Registers Interface Host Access Port](#) consists of two 8-bit run-time registers that occupy two addresses in the HOST I/O space. The [Mailbox Registers Interface Host Access Port](#) is used by the host to access the 36 index-addressable 8-bit registers.

**Note:** In this specification, host access to registers in the Mailbox Registers Interface through the host access port are identified by the prefix MBX in front of a hexadecimal index address.

#### 11.1.1 BLOCK DIAGRAM

**FIGURE 11-1: MAILBOX BLOCK DIAGRAM**



### 11.2 Power, Clocks and Reset

#### 11.2.1 POWER DOMAIN

This block is powered by the VTR power supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

#### 11.2.2 CLOCKS

This block has one clock input, the [LPC Bus Clock](#).

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

#### 11.2.3 RESET

This block is reset when [nSYS\\_RST](#) is asserted.

In addition the [MBX\\_INDEX Register](#) & [MBX\\_DATA Register](#) are reset when [VCC\\_PWRGD](#) Signal Pin function is de-asserted.

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 11.3 Interrupts

The [MailBox Register Interface](#) can generate an interrupt event for the HOST-to-EC events. See [HOST-to-EC Mailbox Register on page 231](#). The interrupt source is routed onto the [MBX](#) bit in the [GIRQ15 Source Register](#) and is a level, active high signal.

### 11.3.1 MAILBOX REGISTER INTERFACE (LDN 0H) SIRQ ROUTING

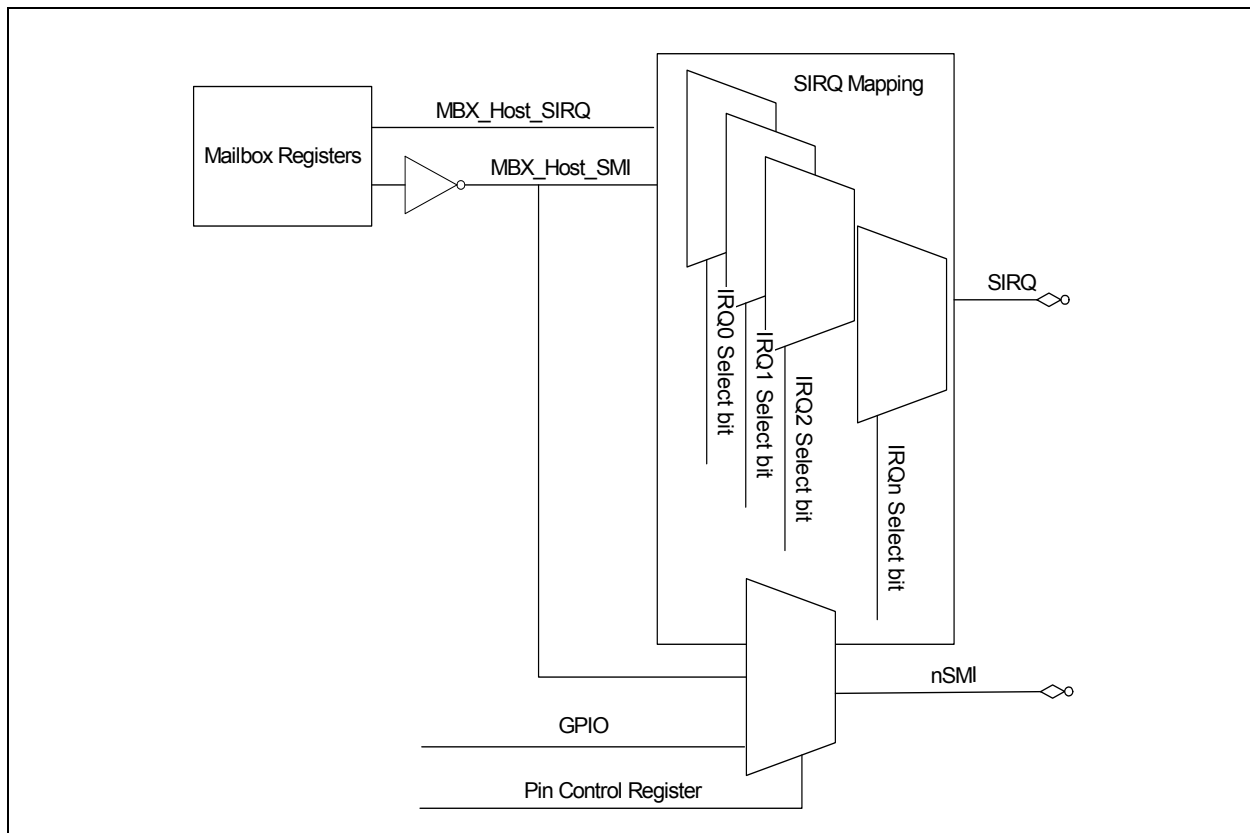
The [MailBox Register Interface](#) can generate a SIRQ event for the EC-to-HOST EC events. See [HOST-to-EC Mailbox Register on page 231](#). This interrupt is routed to the SIRQ block (see [Section 4.8.1, "SERIRQ Configuration Registers," on page 65](#)). For this interrupt, the [SELECT on page 66](#) is cleared to '0' in the Interrupt Configuration Register for the selected SIRQ frame.

The [MailBox Register Interface](#) can generate a SMI event from the [SMI Interrupt Source Register on page 232](#). The SMI event can be routed to any frame in the SIRQ stream and to the nSMI pin. To enable SMI routing to the SIRQ stream, the bit [SELECT on page 66](#) is set to '1' in the Interrupt Configuration Register for the selected SIRQ frame. The SMI event can be routed to nSMI pin by selecting the nSMI signal function in the associated [Pin Control Register on page 406](#).

The SMI event produces a standard active low on the serial IRQ stream and active low on the open drain nSMI pin. See [FIGURE 11-2: Mailbox SIRQ and SMI Routing on page 226](#).

See [Section 4.8.1, "SERIRQ Configuration Registers," on page 65](#).

**FIGURE 11-2: MAILBOX SIRQ AND SMI ROUTING**



## 11.4 Registers Summary

The [MailBox Register Interface](#) has its own Logical Device Number, and Base Address as indicated in [Table 11-1](#). The Host LPC I/O addresses for the [MailBox Register Interface](#) are selected via a Base Address Register (see [Section 4.6.2, "Base Address Registers," on page 60](#)). LPC access to configuration registers is through Host Access Configuration Port (see [Section 4.5.1, "Host Access Port," on page 58](#).)

[Table 11-2](#) is a register summary for the [MailBox Register Interface](#) block.

**TABLE 11-1:   MailBox Register Interface BASE ADDRESS TABLE**

MailBox Register Interface Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
Mailbox Interface	0h	FF_0000h

**Note:** The Host LPC I/O addresses for this instance is selected via a Base Address Register (see [Section 4.6.2, "Base Address Registers," on page 60](#)). LPC access to configuration registers is through the Host Access Configuration Port (see [Section 4.5.1, "Host Access Port," on page 58](#).)

The [Table 11-2](#) is a register summary for one instance of the [MailBox Register Interface](#). The LPC I/O address for each Run-Time Register is described below as an offset from its Base Address Register. Each EC address is indicated as an SPB Offset from its AHB base address. Each Configuration register is accessed through the [Host Access Port](#) is via its LDN indicated in [Table 11-1 on page 227](#) and its [Host Access Port](#) index which is described as "Host Config Index" in the tables following.

# MEC1618/MEC1618i

**TABLE 11-2: MailBox Register Interface REGISTER SUMMARY**

		Host I/O Access			EC Interface			Notes
	Register Name	Host I/O Index	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
	<a href="#">MBX_INDEX Register</a>	00h	00h	R/W	-	-	-	
	<a href="#">MBX_DATA Register</a>	01h	01h	R/W	-	-	-	
		<b>MAILBOX INDEX</b>						
1.	<a href="#">HOST-to-EC Mailbox Register</a>	MBX 00h	-	R/W	100h	0	R/WC	<a href="#">Note 11-1</a>
2.	<a href="#">EC-to-Host Mailbox Register</a>	MBX 01h	-	R/WC	104h	0	R/W	<a href="#">Note 11-2</a>
3.	<a href="#">SMI Interrupt Source Register</a>	MBX 02h	-	<a href="#">Table 11-7</a>	108h	0	<a href="#">Table 11-7</a>	
4.	<a href="#">SMI Interrupt Mask Register</a>	MBX 03H	-	R/W	10Ch	0	R/W	
5.	<b>Mailbox register [0]</b>	MBX10h	-	R/W	110h	0	R/W	
6.	<b>Mailbox register [1]</b>	MBX11h	-			1		
7.	<b>Mailbox register [2]</b>	MBX12h	-			2		
8.	<b>Mailbox register [3]</b>	MBX13h	-			3		
9.	<b>Mailbox register [4]</b>	MBX14h	-	R/W	114h	0	R/W	
10.	<b>Mailbox register [5]</b>	MBX15h	-			1		
11.	<b>Mailbox register [6]</b>	MBX16h	-			2		
12.	<b>Mailbox register [7]</b>	MBX17h	-			3		
13.	<b>Mailbox register [8]</b>	MBX18h	-	R/W	118h	0	R/W	
14.	<b>Mailbox register [9]</b>	MBX19h	-			1		
15.	<b>Mailbox register [A]</b>	MBX1Ah	-			2		
16.	<b>Mailbox register [B]</b>	MBX1Bh	-			3		
17.	<b>Mailbox register [C]</b>	MBX1Ch	-	R/W	11Ch	0	R/W	
18.	<b>Mailbox register [D]</b>	MBX1Dh	-			1		
19.	<b>Mailbox register [E]</b>	MBX1Eh	-			2		
20.	<b>Mailbox register [F]</b>	MBX1Fh	-			3		

**TABLE 11-2: MailBox Register Interface REGISTER SUMMARY (CONTINUED)**

		Host I/O Access			EC Interface			Notes
	Register Name	Host I/O Index	SPB Offset	Host Type	SPB Offset	Byte Lane	EC Type	
21.	Mailbox register [10]	MBX20h	-	R/W	120h	0	R/W	
22.	Mailbox register [11]	MBX21h	-			1		
23.	Mailbox register [12]	MBX22h	-			2		
24.	Mailbox register [13]	MBX23h	-			3		
25.	Mailbox register [14]	MBX24h	-	R/W	124h	0	R/W	
26.	Mailbox register [15]	MBX25h	-			1		
27.	Mailbox register [16]	MBX26h	-			2		
28.	Mailbox register [17]	MBX27h	-			3		
29.	Mailbox register [18]	MBX28h	-	R/W	128h	0	R/W	
30.	Mailbox register [19]	MBX29h	-			1		
31.	Mailbox register [1A]	MBX2Ah	-			2		
32.	Mailbox register [1B]	MBX2Bh	-			3		
33.	Mailbox register [1C]	MBX2Ch	-	R/W	12Ch	0	R/W	
34.	Mailbox register [1D]	MBX2Dh	-			1		
35.	Mailbox register [1E]	MBX2Eh	-			2		
36.	Mailbox register [1F]	MBX2Fh	-			3		

**Note 11-1** Interrupt is cleared when read by the EC.

**Note 11-2** Interrupt is cleared when read by the host.

## 11.4.1 MAILBOX REGISTERS INTERFACE HOST ACCESS PORT

The Mailbox registers access port is two runtime registers that occupy two addresses in the Host I/O space: [MBX\\_INDEX Register](#) & [MBX\\_DATA Register](#).

To access a Mailbox register after the Mailbox Registers Interface Base Address has been initialized, write the Mailbox register index address to the MBX Index port and read or write the Mailbox register data from the MBX data port.

See [Table 11-2, "MailBox Register Interface Register Summary," on page 228](#).

## 11.4.2 MAILBOX CONTROL REGISTERS

Mailbox Register, HOST-to-EC, and Mailbox Register, EC-to-HOST, are specifically designed to pass commands between the host and the EC ([FIGURE 11-1: on page 225](#)). If enabled, these registers can generate interrupts.

Mailbox Register and Mailbox Register are not dual-ported, so the HOST BIOS and Keyboard BIOS must be designed to properly share these registers. When the host performs a write of the HOST-to-EC mailbox register, an interrupt will be generated and seen by the EC if unmasked. When the EC writes FF to the HOST-to-EC mailbox register, resets the register to 00h, providing a simple means for the EC to inform the host that an operation has been completed.

When the EC writes the EC-to-HOST mailbox register, an SMI may be generated and seen by the host if unmasked. When the Host CPU writes FFh to the EC-to-HOST mailbox register, the EC-to-HOST register resets to 00h, providing a simple means for the host to inform that EC that an operation has been completed.

# MEC1618/MEC1618i

**PROGRAMMER'S NOTE:** The protocol used to pass commands back and forth through the Mailbox Registers Interface is left to the System designer. Microchip can provide an application example of working code in which the host uses the Mailbox registers to gain access to all of the EC registers.

## 11.5 Register Details

### 11.5.1 MAILBOX INDEX REGISTER

**TABLE 11-3: MBX\_INDEX REGISTER**

HOST OFFSET	00h				8-Bit		HOST SIZE		
EC OFFSET	NA						EC SIZE		
POWER	VTR				00h 00h		nSYS_RST DEFAULT & VCC_PWRGD DE-ASSERTION		
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	INDEX[7:0]								

### 11.5.2 MAILBOX DATA REGISTER

**TABLE 11-4: MBX\_DATA REGISTER**

HOST OFFSET	01				8-Bit		HOST SIZE		
EC OFFSET	NA						EC SIZE		
POWER	VTR				00h 00h		nSYS_RST DEFAULT & VCC_PWRGD DE-ASSERTION		
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	DATA[7:0]								

## 11.5.3 HOST-TO-EC MAILBOX REGISTER

**TABLE 11-5: HOST-TO-EC MAILBOX REGISTER**

HOST OFFSET	MBX_00h				8-Bit			HOST SIZE	
EC OFFSET	100h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	
BIT NAME	HOST_EC_MBOX[7:0]								

### HOST\_EC\_MBOX[7:0]

If enabled, an interrupt to the EC marked by the **MBX** bit in the **GIRQ15 Source Register** will be generated whenever the Host writes this register.

This register is cleared when written with FFh.

## 11.5.4 EC-TO-HOST MAILBOX REGISTER

**TABLE 11-6: EC-TO-HOST MAILBOX REGISTER**

HOST OFFSET	MBX_01h				8-Bit			HOST SIZE	
EC OFFSET	104h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	EC_HOST_MBOX[7:0]								

### EC\_HOST\_MBOX[7:0]

An EC write to this register will set bit **EC\_WR** in the **SMI Interrupt Source Register** to '1b'. If enabled, setting bit **EC\_WR** to '1b' generates a Host SMI.

This register is cleared when written with FFh.

# MEC1618/MEC1618i

## 11.5.5 SMI INTERRUPT SOURCE REGISTER

TABLE 11-7: SMI INTERRUPT SOURCE REGISTER

HOST OFFSET	MBX_02h				8-Bit			HOST SIZE	
EC OFFSET	108h				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-	
BIT NAME	EC_SWI[6:0]							EC_WR	

### EC\_WR

This bit is set autonomously when the [EC-to-Host Mailbox Register](#) has been written. An SMI to the Host is generated when any bit in this register ([EC\\_WR](#) or any bit in [EC\\_SWI\[6:0\]](#)) is '1b' and the corresponding bit in the [SMI Interrupt Mask Register](#) register is '1b'.

This bit is automatically cleared by a read of the [EC-to-Host Mailbox Register](#). The bit is also cleared when written with a '1b', by either the Host or the EC.

### EC\_SWI[6:0]

The EC can generate an SMI to the Host by writing any non-zero value to this field.

Each bit in this field is cleared when written with a '1b'.

## 11.5.6 SMI INTERRUPT MASK REGISTER

TABLE 11-8: SMI INTERRUPT MASK REGISTER

HOST OFFSET	MBX_03h				8-Bit			HOST SIZE	
EC OFFSET	10Ch				8-Bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	EC_SWI_EN[6:0]							EC_WR_EN	



## **EC\_WR\_EN**

If this bit is '1b', bit [EC\\_WR](#) in the [SMI Interrupt Source Register](#) is enabled.

## **EC\_SWI\_EN[6:0]**

Each bit that is set to '1b' in this field enables the corresponding bit in the [EC\\_SWI\[6:0\]](#) field in the [SMI Interrupt Source Register](#).

## 12.0 TWO PIN SERIAL PORT (UART)

### 12.1 General Description

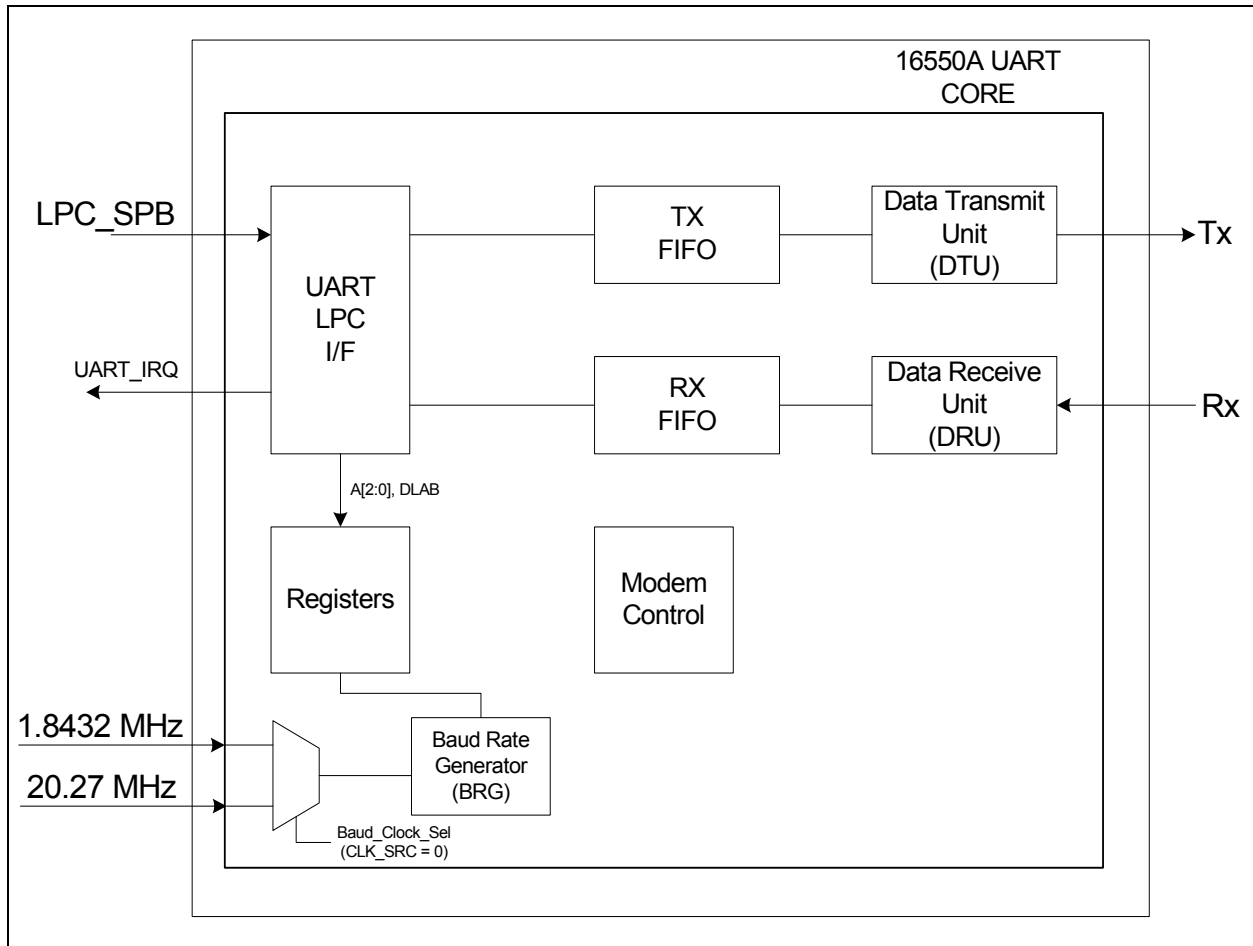
The MEC1618/MEC1618i incorporates one full function UART. The UART is compatible with the 16450, the 16450 ACE registers and the 16C550A. The UART performs serial-to-parallel conversion on received characters and parallel-to-serial conversion on transmit characters. Two sets of baud rates are provided. When the 1.8432 MHz source clock is selected, standard baud rates from 50 to 115.2K are available. When the source clock is 20.27 MHz, baud rates from 79.18 K to 1.267 K are available. The character options are programmable for 1 start; 1, 1.5 or 2 stop bits; even, odd, sticky or no parity; and prioritized interrupts. The UART contains a programmable baud rate generator that is capable of dividing the input clock or crystal by a number from 1 to 65535. The UART is also capable of supporting the MIDI data rate. Refer to the Configuration Registers for information on disabling, powerdown and changing the base address of the UART. The interrupt from a UART is enabled by programming OUT2 of the UART to a logic "1". OUT2 being a logic "0" disables that UART's interrupt. The UART is accessible by both the Host and the EC.

#### 12.1.1 FEATURES

- Programmable word length, stop bits and parity
- Programmable baud rate generator
- Interrupt generator
- Loop-back mode
- Interface registers
- 16-byte Transmit FIFO
- 16-byte Receive FIFO
- Multiple clock sources
- VTR & VCC operation
- Pin Polarity control
- Low power sleep mode

## 12.1.2 BLOCK DIAGRAM

**FIGURE 12-1: SERIAL PORT (UART) BLOCK DIAGRAM**



## 12.1.3 BLOCK DIAGRAM SIGNAL LIST

**TABLE 12-1: SERIAL PORT (UART) REGISTER INTERFACE PORT LIST**

Signal Name	Direction	Description
UART_INT	Output	Host Interrupt routed to SERIRQ
EC IF	I/O Bus	Bus used for register access
MCLK	Input	Block operating clock
UART_RX	Input	UART Receive data pin
UART_TX	Output	UART Transmit data pin
UART_CLK	Input	UART Alternate clock pin (1.8432MHz)
nSYS_RST	input	VTR POR reset
nSIO_RESET	input	VCC POR reset

# MEC1618/MEC1618i

## 12.2 Power, Clocks and Reset

### 12.2.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

### 12.2.2 CLOCKS

[Registers](#) in this block are clocked at the [LPC Bus Clock](#) rate which is derived by the [MCLK](#). Baud rates are derived from 1.8432MHz. The 1.8432MHz. is itself derived from either [MCLK](#) or sourced from from UART\_CLK Signal Pin Function.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

In order to maintain communicating with acceptable error, an accurate baud clock is required.

**Note 12-1** The [FREQ\\_LOCK](#) bit in the [PCR Status and Control Register on page 114](#) must be set in order to insure an accurate baud clock when the [CLK\\_SRC](#) bit is '0' in the [Configuration Select Register on page 254](#), the baud clock is internally sourced.

**Note 12-2** When the [CLK\\_SRC](#) bit is '1' in the [Configuration Select Register on page 254](#), the baud clock is externally sourced from the UART\_CLK pin. The UART\_CLK requires a frequency of 1.8432 MHz  $\pm$  2%.

### 12.2.3 RESET

[Table 12-2](#) details the effect of [nSYS\\_RST](#) or [nSIO\\_RESET](#) on each of the runtime registers of the Serial Port.

**TABLE 12-2: RESET FUNCTION TABLE**

Register Signal	Reset Control	Reset State
Interrupt Enable Register	RESET	All bits low
Interrupt Identification Reg.		Bit 0 is high; Bits 1 - 7 low
FIFO Control		All bits low
Line Control Reg.		
MODEM Control Reg.		
Line Status Reg.		All bits low except 5, 6 high
MODEM Status Reg.		Bits 0 - 3 low; Bits 4 - 7 input
TXD1, TXD2		High
INTRPT (RCVR errs)	RESET/Read LSR	Low
INTRPT (RCVR Data Ready)	RESET/Read RBR	
INTRPT (THRE)	RESET/Read IIR/Write THR	
OUT2B	RESET	High
RTSB		
DTRB		
OUT1B		

**TABLE 12-2: RESET FUNCTION TABLE (CONTINUED)**

Register Signal	Reset Control	Reset State
RCVR FIFO	RESET/ FCR1*FCR0/_FCR0	All Bits Low
XMIT FIFO	RESET/ FCR1*FCR0/_FCR0	

The Runtime register can be configured to be reset on either [nSYS\\_RST](#) or [nSIO\\_RESET](#). The **POWER** bit in the [Configuration Select Register](#) controls which reset effects the runtime registers. The Refer to [Table 12-2](#) for effected registers and [Section 5.0, "Power, Clocks, and Resets"](#) for definitions of [nSYS\\_RST](#) on [page 108](#) or [nSIO\\_RESET](#) on [page 76](#).

See [Section 5.6, "Reset Interface,"](#) on [page 104](#) for details on reset.

## 12.3 Interrupts

### 12.3.1 EC INTERRUPT

The [Two Pin Serial Port \(UART\)](#) can generate an EC interrupt event. The interrupt source is routed onto the [UART\\_RX](#) bit in the [GIRQ15 Source Register](#), and is a level sensitive, active high signal.

### 12.3.2 HOST INTERRUPT

The [Two Pin Serial Port \(UART\)](#) can generate a SIRQ event to the Host. See the [Interrupt Enable Register \(IER\)](#) on [page 241](#) and the [Interrupt Identification Register \(IIR\)](#) on [page 243](#). This interrupt is routed to the SIRQ block (see [SERIRQ Configuration Registers](#) on [page 65](#)).

## 12.4 Registers

The [Two Pin Serial Port \(UART\)](#) registers are located on the Host SPB.

Each instance of the [Two Pin Serial Port \(UART\)](#) has its own Logical Device Number, and Base Address as indicated in [Table 12-3](#).

**TABLE 12-3: Two Pin Serial Port (UART) BASE ADDRESS TABLE**

Two Pin Serial Port (UART) Instance	LDN from (Table 3-2 on page 51)	AHB Base Address
UART	7h	FF_1C00h

**Note 12-3** The Host LPC I/O addresses for each instance is selected via a Base Address Register (see [Section 4.6.2, "Base Address Registers,"](#) on [page 60](#)). LPC access to configuration registers is through the Host Access Configuration Port (see [Section 4.5.1, "Host Access Port,"](#) on [page 58](#).)

[Table 12-4](#) is a register summary for one instance of the [Two Pin Serial Port \(UART\)](#). The LPC I/O address for each Run-Time Register is described below as an offset from its Base Address Register. Each EC address is indicated as an SPB Offset from its AHB base address. Each Configuration register access through the [Host Access Port](#) is via its LDN indicated in [Table 12-3](#) on [page 237](#) and its [Host Access Port](#) index which is described as "Host Config Index" in the tables below.

The following table summarizes the registers allocated for the Controller. The offset field in the following table is the offset from the Embedded Controller's (EC) Base Address.

# MEC1618/MEC1618i

**TABLE 12-4: Two Pin Serial Port (UART) REGISTER SUMMARY**

Register Name	Host I/O Access			DLAB (Note 12-4)	EC Interface			Notes
	Host I/O Index	SPB Off-set	Host Type		SPB Off-set	Byte Lane	EC Type	
Receive Buffer Register (RB)	00h	00h	R	0	00h	0	R	Note 12-5
Transmit Buffer Register (TB)	00h	00h	W	0	00h	0	W	Note 12-5
Programmable Baud Rate Generator (and Divisor) (LSByte)	00h	00h	R/W	1	00h	0	R/W	Note 12-5
Programmable Baud Rate Generator (and Divisor) (MSByte)	01h	01h	R/W	1	01h	1	R/W	Note 12-5
Interrupt Enable Register (IER)	01h	01h	R/W	0	01h	1	R/W	Note 12-5
FIFO Control Register (FCR)	02h	02h	W	X	02h	2	W	Note 12-5
Interrupt Identification Register (IIR)	02h	02h	R	X	02h	2	R	Note 12-5
Line Control Register (LCR)	03h	03h	R/W	X	03h	3	R/W	Note 12-5
Modem Control Register (MCR)	04h	04h	R/W	X	04h	0	R/W	Note 12-5
Line Status Register (LSR)	05h	05h	R	X	05h	1	R	Note 12-5
Modem Status Register (MSR)	06h	06h	R	X	06h	2	R	Note 12-5
Scratchpad Register (SCR)	07h	07h	R/W	X	07h	3	R/W	Note 12-5
Register Name	Host Access			N/A	EC Interface			
	Host Config. Index	SPB Off-set	Host Type		EC Off-set	Byte Lane	EC Type	
Activate	30h	330h	R/W		330h	0	R/W	Note 12-5
Configuration Select Register	F0h	3F0h	R/W		3F0h	0	R/W	Note 12-5

**Note 12-4** DLAB is Bit 7 of the Line Control Register

**Note 12-5** Access to this register should be limited to 8-bit loads and stores. 16-bit or 32-bit stores will be blocked and 16-bit or 32-bit loads have unexpected results. JTAG Debugger access should indirect using peek\_poke\_arc macros described in [IEEE Std 1149.1](#).

## 12.5 Register Summary

**TABLE 12-5: REGISTER SUMMARY**

Address (Note 12-6)	R/W	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADDR = 0 DLAB = 0	R	Receive Buffer r	Data Bit 7	Data Bit 6	Data Bit 5	Data Bit 4	Data Bit 3	Data Bit 2	Data Bit 1	Data Bit 0 (Note 12-7)
ADDR = 0 DLAB = 0	W	Transmitter Holding r	Data Bit 7	Data Bit 6	Data Bit 5	Data Bit 4	Data Bit 3	Data Bit 2	Data Bit 1	Data Bit 0
ADDR = 1 DLAB = 0	R/W	Interrupt Enable r	Reserved				Enable Modem Status Interrupt (EMSI)	Enable Receiver Line Status Interrupt (ELSI)	Enable Transmitter Holding Register Empty Interrupt (ETHREI)	Enable Received Data Available Interrupt (ERDAI)
ADDR = 2	R	Interrupt Ident. r	FIFOs Enabled (Note 12-11)	FIFOs Enabled (Note 12-11)	Reserved		Interrupt ID Bit (Note 12-11)	Interrupt ID Bit	Interrupt ID Bit	"0" if Interrupt Pending
ADDR = 2	W	FIFO Control r	RCVR Trigger MSB	RCVR Trigger LSB	Reserved		DMA Mode Select (Note 12-12)	XMIT FIFO Reset	RCVR FIFO Reset	FIFO Enable
ADDR = 3	R/W	Line Control r	Divisor Latch Access Bit (DLAB)	Set Break	Stick Parity	Even Parity Select (EPS)	Parity Enable (PEN)	Number of Stop Bits (STB)	Word Length Select Bit 1 (WLS1)	Word Length Select Bit 0 (WLS0)
ADDR = 4	R/W	MODEM Control r	Reserved			Loop	OUT2 (Note 12-9)	OUT1 (Note 12-9)	Request to Send (RTS)	Data Terminal Ready (DTR)
ADDR = 5	R/W	Line Status r	Error in RCVR FIFO (Note 12-11)	Transmitter Empty (TEMT) (Note 12-8)	Transmitter Holding Register (THRE)	Break Interrupt (BI)	Framing Error (FE)	Parity Error (PE)	Over-run Error (OE)	Data Ready (DR)
ADDR = 6	R/W	MODEM Status r	Data Carrier Detect (DCD)	Ring Indicator (RI)	Data Set Ready (DSR)	Clear to Send (CTS)	Delta Data Carrier Detect (DDCD)	Trailing Edge Ring Indicator (TERI)	Delta Data Set Ready (DDSR)	Delta Clear to Send (DCTS)
ADDR = 7	R/W	Scratch r (Note 12-10)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADDR = 0 DLAB = 1	R/W	Divisor Latch (LS)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADDR = 1 DLAB = 1	R/W	Divisor Latch (MS)	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

### UART REGISTER SUMMARY NOTES:

**Note 12-6** DLAB is Bit 7 of the Line Control Register (ADDR = 3).

**Note 12-7** Bit 0 is the least significant bit. It is the first bit serially transmitted or received.

**Note 12-8** When operating in the XT mode, this bit will be set any time that the transmitter shift register is empty.

**Note 12-9** This bit no longer has a pin associated with it.

# MEC1618/MEC1618i

**Note 12-10** When operating in the XT mode, this register is not available.

**Note 12-11** These bits are always zero in the non-FIFO mode.

**Note 12-12** Writing a one to this bit has no effect. DMA modes are not supported in this chip.

## 12.6 Detailed Description of Accessible Runtime Registers

### 12.6.1 RECEIVE BUFFER REGISTER (RB)

**TABLE 12-6: RECEIVE BUFFER (RB)**

HOST OFFSET	0h (DLAB=0)			8-bit			HOST SIZE	
EC OFFSET	0h (DLAB=0)			8-bit			EC SIZE	
POWER	VCC or VTR			00h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Received Data byte [7:0]							

#### RECEIVED DATA BYTE

This register holds the received incoming data byte. Bit 0 is the least significant bit, which is transmitted and received first. Received data is double buffered; this uses an additional shift register to receive the serial data stream and converts it to a parallel 8-bit word that is transferred to the Receive Buffer register. The shift register is not accessible.



## 12.6.2 TRANSMIT BUFFER REGISTER (TB)

**TABLE 12-7: TRANSMIT BUFFER (TB)**

HOST OFFSET	0h (DLAB=0)				8-bit			HOST SIZE	
EC OFFSET	0h (DLAB=0)				8-bit			EC SIZE	
POWER	VCC or VTR				00h			nSYS_RST or VCC Power Good DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	W	W	W	W	W	/W	W	W	
EC TYPE		W	W	W	W	W	W	W	
BIT NAME	Transmit data byte [7:0]								

### TRANSMIT DATA BYTE

This register contains the data byte to be transmitted. The transmit buffer is double buffered, utilizing an additional shift register (not accessible) to convert the 8-bit data word to a serial format. This shift register is loaded from the Transmit Buffer when the transmission of the previous byte is complete.

## 12.6.3 INTERRUPT ENABLE REGISTER (IER)

**TABLE 12-8: INTERRUPT ENABLE (IER)**

HOST OFFSET	1h (DLAB=0)				8-bit			HOST SIZE	
EC OFFSET	1h (DLAB=0)				8-bit			EC SIZE	
POWER	VCC or VTR				00h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Reserved				EMSI	ELSI	ETHREI	ERDAI	

The lower four bits of this register control the enables of the five interrupt sources of the Serial Port interrupt. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of this register. Similarly, setting the appropriate bits of this register to a high, selected interrupts can be enabled. Disabling the interrupt system inhibits the Interrupt Identi-

# MEC1618/MEC1618i

fication Register and disables any Serial Port interrupt out of the MEC1618/MEC1618i. All other system functions operate in their normal manner, including the Line Status and MODEM Status Registers. The contents of the Interrupt Enable Register are described below.

## ERDAI

This bit enables the Received Data Available Interrupt (and timeout interrupts in the FIFO mode) when set to logic "1".

## ETHREI

This bit enables the Transmitter Holding Register Empty Interrupt when set to logic "1".

## ELSI

This bit enables the Received Line Status Interrupt when set to logic "1". The error sources causing the interrupt are Overrun, Parity, Framing and Break. The Line Status Register must be read to determine the source.

## EMSI

This bit enables the MODEM Status Interrupt when set to logic "1". This is caused when one of the Modem Status Register bits changes state.

## 12.6.4 FIFO CONTROL REGISTER (FCR)

**TABLE 12-9: FIFO CONTROL (FCR)**

HOST OFFSET	02h			8-bit			HOST SIZE	
EC OFFSET	02h			8-bit			EC SIZE	
POWER	VCC or VTR			00h			nSYS_RST or VCC Power Good DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	W	W	W	W	W	W	W	W
EC TYPE	W	W	W	W	W	W	W	W
BIT NAME	RECV FIFO Trigger Level		Reserved			Clear XMIT FIFO	Clear RECV FIFO	EXRF

This is a write only register at the same location as the IIR.

**Note 12-13** DMA is not supported.

## EXRF

Enable XMIT and RECV FIFO. Setting this bit to a logic "1" enables both the XMIT and RCVR FIFOs. Clearing this bit to a logic "0" disables both the XMIT and RCVR FIFOs and clears all bytes from both FIFOs. When changing from FIFO Mode to non-FIFO (16450) mode, data is automatically cleared from the FIFOs. This bit must be a 1 when other bits in this register are written to or they will not be properly programmed.

## CLEAR RECV FIFO

Setting this bit to a logic "1" clears all bytes in the RCVR FIFO and resets its counter logic to "0". The shift register is not cleared. This bit is self-clearing.

## CLEAR XMIT FIFO

Setting this bit to a logic "1" clears all bytes in the XMIT FIFO and resets its counter logic to "0". The shift register is not cleared. This bit is self-clearing.

## RCV FIFO TRIGGER LEVEL

These bits are used to set the trigger level for the RCVR FIFO interrupt.

**TABLE 12-10: RCV FIFO TRIGGER LEVEL**

Bit 7	Bit 6	RCV FIFO Trigger Level (Bytes)
0	0	1
	1	4
1	0	8
	1	14

## 12.6.5 INTERRUPT IDENTIFICATION REGISTER (IIR)

**TABLE 12-11: INTERRUPT IDENTIFICATION (IIR)**

HOST OFFSET	02h				8-bit			HOST SIZE	
EC OFFSET	02h				8-bit			EC SIZE	
POWER	VCC or VTR				01h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	FIFO_En		Reserved		IntID			IPEND	

By accessing this register, the host CPU can determine the highest priority interrupt and its source. Four levels of priority interrupt exist. They are in descending order of priority:

1. Receiver Line Status (highest priority)
2. Received Data Ready
3. Transmitter Holding Register Empty
4. MODEM Status (lowest priority)

Information indicating that a prioritized interrupt is pending and the source of that interrupt is stored in the Interrupt Identification Register (refer to [Table 12-12](#)). When the CPU accesses the IIR, the Serial Port freezes all interrupts and indicates the highest priority pending interrupt to the CPU. During this CPU access, even if the Serial Port records new interrupts, the current indication does not change until access is completed. The contents of the IIR are described below.

### IPEND

This bit can be used in either a hardwired prioritized or polled environment to indicate whether an interrupt is pending. When bit 0 is a logic "0", an interrupt is pending and the contents of the IIR may be used as a pointer to the appropriate internal service routine. When bit 0 is a logic "1", no interrupt is pending.

# MEC1618/MEC1618i

## INTID

These three bits of the IIR are used to identify the highest priority interrupt pending as indicated by [Table 12-12](#). In non-FIFO mode, Bit[3] is a logic "0". In FIFO mode Bit[3] is set along with Bit[2] when a timeout interrupt is pending.

**TABLE 12-12: INTERRUPT CONTROL TABLE**

FIFO Mode Only	Interrupt Identification Register			Interrupt Set and Reset Functions			
Bit 3	Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	0	1	-	None	None	-
	1	1	0	Highest	Receiver Line Status	Overrun Error, Parity Error, Framing Error or Break Interrupt	Reading the Line Status Register
		0		Second	Received Data Available	Receiver Data Available	Read Receiver Buffer or the FIFO drops below the trigger level.
Character Timeout Indication					No Characters Have Been Removed From or Input to the RCVR FIFO during the last 4 Char times and there is at least 1 char in it during this time	Reading the Receiver Buffer Register	
1	0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if Source of Interrupt) or Writing the Transmitter Holding Register
0		0		Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

## FIFO\_EN

These two bits are set when the FIFO CONTROL Register bit 0 equals 1.

## 12.6.6 LINE CONTROL REGISTER (LCR)

**TABLE 12-13: LINE CONTROL (LCR)**

HOST OFFSET	03h			8-bit			HOST SIZE	
EC OFFSET	03h			8-bit			EC SIZE	
POWER	VCC or VTR			00h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	DLAB	Break Control	Stick Parity	Parity Select	Enable Parity	Stop Bits	Word Length	

This register contains the format information of the serial line. The bit definitions are:

### WORD LENGTH

These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

### STOP BITS

This bit specifies the number of stop bits in each transmitted or received serial character. [Table 12-14](#) summarizes the information.

**TABLE 12-14: STOP BITS**

Bit 2	Word Length	Number of Stop BitsS
0	--	1
1	5 bits	1.5
	6 bits	2
	7 bits	
	8 bits	

**Note 12-14** The receiver ignores all stop bits beyond the first, regardless of the number used in transmitting.

**TABLE 12-15: SERIAL CHARACTER**

Bit 1	Bit 0	Word Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

# MEC1618/MEC1618i

The Start, Stop and Parity bits are not included in the word length.

## ENABLE PARITY

Parity Enable bit. When bit 3 is a logic "1", a parity bit is generated (transmit data) or checked (receive data) between the last data word bit and the first stop bit of the serial data. (The parity bit is used to generate an even or odd number of 1s when the data word bits and the parity bit are summed).

## PARITY SELECT

Even Parity Select bit. When bit 3 is a logic "1" and bit 4 is a logic "0", an odd number of logic "1"s is transmitted or checked in the data word bits and the parity bit. When bit 3 is a logic "1" and bit 4 is a logic "1" an even number of bits is transmitted and checked.

## STICK PARITY

Stick Parity bit. When parity is enabled it is used in conjunction with bit 4 to select Mark or Space Parity. When LCR bits 3, 4 and 5 are 1 the Parity bit is transmitted and checked as a 0 (Space Parity). If bits 3 and 5 are 1 and bit 4 is a 0, then the Parity bit is transmitted and checked as 1 (Mark Parity). If bit 5 is 0 Stick Parity is disabled.

Bit 3 is a logic "1" and bit 5 is a logic "1", the parity bit is transmitted and then detected by the receiver in the opposite state indicated by bit 4.

## BREAK CONTROL

Set Break Control bit. When bit 6 is a logic "1", the transmit data output (TXD) is forced to the Spacing or logic "0" state and remains there (until reset by a low level bit 6) regardless of other transmitter activity. This feature enables the Serial Port to alert a terminal in a communications system.

## DLAB

Divisor Latch Access Bit (DLAB). It must be set high (logic "1") to access the Divisor Latches of the Baud Rate Generator during read or write operations. It must be set low (logic "0") to access the Receiver Buffer Register, the Transmitter Holding Register, or the Interrupt Enable Register.

## 12.6.7 MODEM CONTROL REGISTER (MCR)

**TABLE 12-16: MODEM CONTROL (MCR)**

HOST OFFSET	04h			8-bit			HOST SIZE	
EC OFFSET	04h			8-bit			EC SIZE	
POWER	VCC or VTR			00h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved			LOOP-BACK	OUT2	OUT1	RTS	DTR

This 8-bit register controls the interface with the MODEM or data set (or device emulating a MODEM). The contents of the MODEM control register are described below.

## DTR

This bit controls the Data Terminal Ready (nDTR) output. When bit 0 is set to a logic "1", the nDTR output is forced to a logic "0". When bit 0 is a logic "0", the nDTR output is forced to a logic "1".

## RTS

This bit controls the Request To Send (nRTS) output. Bit 1 affects the nRTS output in a manner identical to that described above for bit 0.

## OUT1

This bit controls the Output 1 (OUT1) bit. This bit does not have an output pin and can only be read or written by the CPU.

## OUT2

Output 2 (OUT2). This bit is used to enable an UART interrupt. When OUT2 is a logic "0", the serial port interrupt output is forced to a high impedance state - disabled. When OUT2 is a logic "1", the serial port interrupt outputs are enabled.

## LOOPBACK

This bit provides the loopback feature for diagnostic testing of the Serial Port. When bit 4 is set to logic "1", the following occur:

1. The TXD is set to the Marking State (logic "1").
2. The receiver Serial Input (RXD) is disconnected.
3. The output of the Transmitter Shift Register is "looped back" into the Receiver Shift Register input.
4. All MODEM Control inputs (nCTS, nDSR, nRI and nDCD) are disconnected.
5. The four MODEM Control outputs (nDTR, nRTS, OUT1 and OUT2) are internally connected to the four MODEM Control inputs (nDSR, nCTS, RI, DCD).
6. The Modem Control output pins are forced inactive high.
7. Data that is transmitted is immediately received.

This feature allows the processor to verify the transmit and receive data paths of the Serial Port. In the diagnostic mode, the receiver and the transmitter interrupts are fully operational. The MODEM Control Interrupts are also operational but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

## 12.6.8 LINE STATUS REGISTER (LSR)

**TABLE 12-17: LINE STATUS (LSR)**

HOST OFFSET	05h				8-bit			HOST SIZE	
EC OFFSET	05h				8-bit			EC SIZE	
POWER	VCC or VTR				60h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	FIFO Error	Transmit Error	Transmit Empty	Break Interrupt	Frame Error	Parity Error	Overrun Error	Data Ready	

## DATA READY

Data Ready (DR). It is set to a logic "1" whenever a complete incoming character has been received and transferred into the Receiver Buffer Register or the FIFO. Bit 0 is reset to a logic "0" by reading all of the data in the Receive Buffer Register or the FIFO.

# MEC1618/MEC1618i

---

## OVERRUN ERROR

Overrun Error (OE). Bit 1 indicates that data in the Receiver Buffer Register was not read before the next character was transferred into the register, thereby destroying the previous character. In FIFO mode, an overrun error will occur only when the FIFO is full and the next character has been completely received in the shift register, the character in the shift register is overwritten but not transferred to the FIFO. The OE indicator is set to a logic "1" immediately upon detection of an overrun condition, and reset whenever the Line Status Register is read.

## PARITY ERROR

Parity Error (PE). Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even parity select bit. The PE is set to a logic "1" upon detection of a parity error and is reset to a logic "0" whenever the Line Status Register is read. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is indicated when the associated character is at the top of the FIFO.

## FRAME ERROR

Framing Error (FE). Bit 3 indicates that the received character did not have a valid stop bit. Bit 3 is set to a logic "1" whenever the stop bit following the last data bit or parity bit is detected as a zero bit (Spacing level). The FE is reset to a logic "0" whenever the Line Status Register is read. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is indicated when the associated character is at the top of the FIFO. The Serial Port will try to resynchronize after a framing error. To do this, it assumes that the framing error was due to the next start bit, so it samples this 'start' bit twice and then takes in the 'data'.

## BREAK INTERRUPT

Break Interrupt (BI). Bit 4 is set to a logic "1" whenever the received data input is held in the Spacing state (logic "0") for longer than a full word transmission time (that is, the total time of the start bit + data bits + parity bits + stop bits). The BI is reset after the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is indicated when the associated character is at the top of the FIFO. When break occurs only one zero character is loaded into the FIFO. Restarting after a break is received, requires the serial data (RXD) to be logic "1" for at least 1/2 bit time.

Bits 1 through 4 are the error conditions that produce a Receiver Line Status Interrupt BIT 3.

**Note 12-15** Whenever any of the corresponding conditions are detected and the interrupt is enabled.

## TRANSMIT EMPTY

Transmitter Holding Register Empty (THRE). Bit 5 indicates that the Serial Port is ready to accept a new character for transmission. In addition, this bit causes the Serial Port to issue an interrupt when the Transmitter Holding Register interrupt enable is set high. The THRE bit is set to a logic "1" when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic "0" whenever the CPU loads the Transmitter Holding Register. In the FIFO mode this bit is set when the XMIT FIFO is empty, it is cleared when at least 1 byte is written to the XMIT FIFO. Bit 5 is a read only bit.

## TRANSMIT ERROR

Transmitter Empty (TEMT). Bit 6 is set to a logic "1" whenever the Transmitter Holding Register (THR) and Transmitter Shift Register (TSR) are both empty. It is reset to logic "0" whenever either the THR or TSR contains a data character. Bit 6 is a read only bit. In the FIFO mode this bit is set whenever the THR and TSR are both empty,

## FIFO ERROR

This bit is permanently set to logic "0" in the 450 mode. In the FIFO mode, this bit is set to a logic "1" when there is at least one parity error, framing error or break indication in the FIFO. This bit is cleared when the LSR is read if there are no subsequent errors in the FIFO.



## 12.6.9 MODEM STATUS REGISTER (MSR)

**TABLE 12-18: MODEM STATUS (MSR)**

HOST ADDRESS	06h			8-bit			HOST SIZE	
EC OFFSET	06h			8-bit			EC SIZE	
POWER	VCC or VTR			xxxx0000b			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	DCD#	RI#	DSR	CTS	DCD	RI	DSR	CTS

This 8-bit register provides the current state of the control lines from the MODEM (or peripheral device). In addition to this current state information, four bits of the MODEM Status Register (MSR) provide change information.

These bits are set to logic "1" whenever a control input from the MODEM changes state. They are reset to logic "0" whenever the MODEM Status Register is read.

### CTS

Delta Clear To Send (DCTS). Bit 0 indicates that the nCTS input to the chip has changed state since the last time the MSR was read.

### DSR

Delta Data Set Ready (DDSR). Bit 1 indicates that the nDSR input has changed state since the last time the MSR was read.

### RI

Trailing Edge of Ring Indicator (TERI). Bit 2 indicates that the nRI input has changed from logic "0" to logic "1".

### DCD

Delta Data Carrier Detect (DDCD). Bit 3 indicates that the nDCD input to the chip has changed state.

**Note 12-16** Whenever bit 0, 1, 2, or 3 is set to a logic "1", a MODEM Status Interrupt is generated.

### CTS

This bit is the complement of the Clear To Send (nCTS) input. If bit 4 of the MCR is set to logic "1", this bit is equivalent to nRTS in the MCR.

### DSR

This bit is the complement of the Data Set Ready (nDSR) input. If bit 4 of the MCR is set to logic "1", this bit is equivalent to DTR in the MCR.

### RI#

This bit is the complement of the Ring Indicator (nRI) input. If bit 4 of the MCR is set to logic "1", this bit is equivalent to OUT1 in the MCR.

### DCD

This bit is the complement of the Data Carrier Detect (nDCD) input. If bit 4 of the MCR is set to logic "1", this bit is equivalent to OUT2 in the MCR.

# MEC1618/MEC1618i

**APPLICATION NOTE:** The Modem Status Register (MSR) only provides the current state of the UART MODEM control lines in Loopback Mode. The MEC1618/MEC1618i does not support external connections for the MODEM Control inputs (nCTS, nDSR, nRI and nDCD) or for the four MODEM Control outputs (nDTR, nRTS, OUT1 and OUT2).

## 12.6.10 SCRATCHPAD REGISTER (SCR)

**TABLE 12-19: SCRATCH PAD (SCR)**

HOST OFFSET	07h			8-bit			HOST SIZE	
EC OFFSET	07h			8-bit			EC SIZE	
POWER	VCC or VTR			00h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Scratch							

### SCRATCH

This 8-bit read/write register has no effect on the operation of the Serial Port. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.

## 12.6.11 PROGRAMMABLE BAUD RATE GENERATOR (AND DIVISOR)

**TABLE 12-20: PROGRAMMABLE BAUD RATE GENERATOR (AND DIVISOR)**

HOST OFFSET	BYTE1: 01h (DLAB = 1) BYTE0: 00h (DLAB = 1)				8-bit			HOST SIZE	
EC OFFSET	BYTE1: 01h (DLAB = 1) BYTE0: 00h (DLAB = 1)				8-bit			EC SIZE	
POWER	VCC or VTR				0000h			nSYS_RST or nSIO_RESET DEFAULT	
BUS	LPC SPB								
BYTE1 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Baud_Clock_Sel	Baud_Rate_Divisor							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Baud_Rate_Divisor[7:0]								

### BAUD\_CLOCK\_SEL

If the CLK\_SRC bit is '0' and the [Baud\\_Clock\\_Sel](#) bit is '0,' the 1.8432MHz clock is used to generate the baud clock. [Table 12-21](#) shows some baud rates that can be generated with this clock. The CLK\_SRC bit is D0 in the UART Logical Device configuration register offset 0xF0.

If the CLK\_SRC bit is '0' and the [Baud\\_Clock\\_Sel](#) bit is '1,' [MCLK](#) is used to generate the baud clock. [Table 12-22](#) shows some baud rates that can be generated with this clock.

If the CLK\_SRC bit is '1,' the [Baud\\_Clock\\_Sel](#) bit as no effect.

### BAUD\_RATE\_DIVISOR

The Serial Port contains a programmable Baud Rate Generator that is capable of dividing the internal clock source by any divisor from 1 to 65535. The clock source is either a 1.8432MHz clock derived from [MCLK](#) or [MCLK](#) directly. The output frequency of the Baud Rate Generator is 16x the Baud rate. Two eight bit latches store the divisor in 16 bit binary format. These Divisor Latches must be loaded during initialization in order to insure desired operation of the Baud Rate Generator. Upon loading either of the Divisor Latches, a 16 bit Baud counter is immediately loaded. This prevents long counts on initial load. If a 0 is loaded into the BRG registers, the output divides the clock by the number 3. If a 1 is loaded, the output is the inverse of the input oscillator. If a two is loaded, the output is a divide by 2 signal with a 50% duty cycle. If a 3 or greater is loaded, the output is low for 2 bits and high for the remainder of the count.

[Table 12-21](#) and [Table 12-22](#) shows the baud rates possible.

# MEC1618/MEC1618i

---

**TABLE 12-21: UART BAUD RATES (1.8432MHZ SOURCE)**

Desired Baud Rate	Divisor Used to Generate 16X Clock
50	2304
75	1536
110	1047
134.5	857
150	768
300	384
600	192
1200	96
1800	64
2000	58
2400	48
3600	32
4800	24
7200	16
9600	12
19200	6
38400	3
57600	2
115200	1

**TABLE 12-22: UART BAUD RATES (MCLK SOURCE)**

Desired Baud Rate	BAUD_CLOCK_SEL	Divisor Used to Generate 16X Clock
79180	1	16
105573	1	12
115170	1	11
126688	1	10
140764	1	9
158359	1	8
180982	1	7
211146	1	6

**TABLE 12-22: UART BAUD RATES (MCLK SOURCE) (CONTINUED)**

Desired Baud Rate	BAUD_CLOCK_SEL	Divisor Used to Generate 16X Clock
253375	1	5
316719	1	4
422292	1	3
633438	1	2
1266875	1	1

## 12.7 Detailed Description of Configuration Registers

### 12.7.1 ACTIVATE

**TABLE 12-23: ACTIVATE REGISTER**

HOST OFFSET	30h				8-bit			HOST SIZE	
EC OFFSET	330h				32-bit			EC SIZE	
POWER	VTR				00b			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R/W	
EC TYPE	R	R	R	R	R	R	R	R/W	
BIT NAME	Reserved							Activate	

### ACTIVATE

When this bit is 1, the UART logical device is powered and functional. When this bit is 0, the UART logical device is powered down and inactive.

# MEC1618/MEC1618i

## 12.7.2 CONFIGURATION

TABLE 12-24: CONFIGURATION SELECT REGISTER

HOST OFFSET	F0h				8-bit			HOST SIZE	
EC OFFSET	3F0h				8-bit			EC SIZE	
POWER	VTR				00b			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R	R	R	R	R	R	R	R/W	
EC TYPE	R	R	R	R	R	R/W	R/W	R/W	
BIT NAME	Reserved					Polarity	Power	CLK_SRC	

### CLK\_SRC

When this bit is 0, the UART clock is derived from the internal [20 MHz Oscillator](#). When this bit is 1, the UART clock is derived from an external clock source

### POWER

When this bit is 1, the UART Runtime Registers (the registers at offsets 0h through 7h from the base of the UART Logical Device) are controlled by VCC. They are set to their POR defaults on a [nSIO\\_RESET](#). In addition, pins associated with the UART are powered down and place in a High-Z state on [nSIO\\_RESET](#).

When this bit is 0, the UART Runtime Registers are controlled by VTR. They are set to their POR defaults on an [nSYS\\_RST](#). In addition, the state of the UART pins is controlled by VTR.

### POLARITY

When the Polarity bit is asserted ('1'), the UART\_TX and UART\_RX pins functions are inverted. When the Polarity bit is not asserted (default), the UART\_TX and UART\_RX pins functions are not inverted.

## 12.8 Sleep Enable/ Clock Request Power State Controls

**TABLE 12-25: UART BLOCK CLOCK GATING BEHAVIOR**

Activate	External Sleep Input	Block Idle Status (Note 12-17)	Clock Required Status Output	State	Description
0	X	X	0	DISABLED	<a href="#">Two Pin Serial Port (UART)</a> is disabled by firmware and the core clock is not needed. Note: it is up to the host to guarantee that the block is not in use before the Activate bit is de-asserted.
1	0	NOT IDLE	1	NORMAL OPERATION	The block is neither disabled by firmware nor commanded to sleep.
		IDLE	0		
	1	NOT IDLE	1	PREPARING TO SLEEP	A sleep command has been asserted but the core clock is still required because the block is not idle.
		IDLE	0	SLEEPING	A sleep command has been asserted, the block is idle and the core clock can be stopped.

**Note 12-17** The [Two Pin Serial Port \(UART\)](#) 'idle' status is defined in [Table 12-26](#).

**TABLE 12-26: UART IDLE STATUS**

Transmitter Active?	Receiver Active?	Character Time-out Active	Status
NO	NO	NO	Idle
YES	X	X	Not Idle
X	YES	X	
X	X	YES	

\_\_\_\_\_

The MFC1618/MFC1618i Embedded Flash Subsystem includes a 102KB embedded Flash memory. The memory





## 13.3 Port List

**TABLE 13-1: EMBEDDED FLASH PORT LIST**

SIGNAL NAME	DIRECTION	DESCRIPTION
ARC_CLK_DISABLE.	INPUT	Indicates the ARC is sleeping
FLASH_SLP_EN	INPUT	Indicates that the system is trying to shut down the ring oscillator
FLASH_CLK_REQ	OUTPUT	Indicates when the flash is ready to sleep (have clocks removed)

## 13.4 Power, Clocks and Reset

### 13.4.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.0, "Power, Clocks, and Resets," on page 74](#) for details on power domains.

### 13.4.2 CLOCKS

This block uses the 20.27 MHz [MCLK](#). All Flash signal timing is derived from the [MCLK](#).

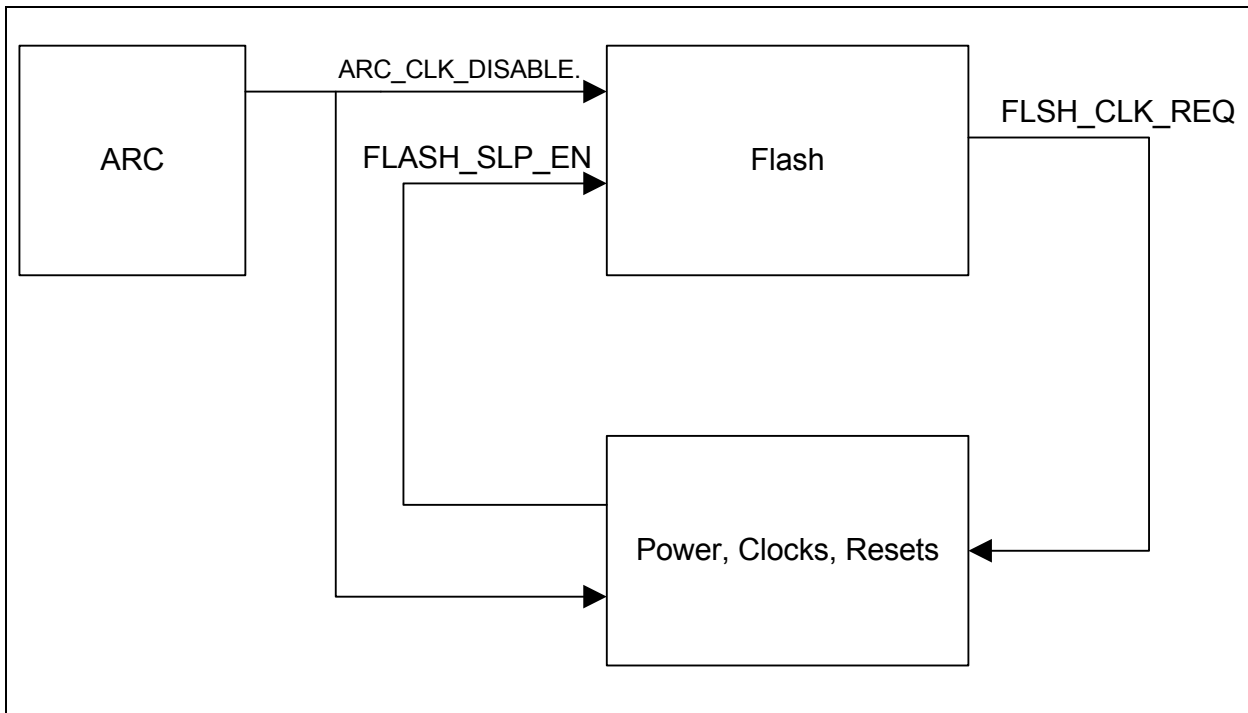
See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

#### 13.4.2.1 Clock Idle

The Embedded Flash controller will keep the internal oscillator operating as long as the controller is not in the Standby state. This permits the controller to complete any program or erase operation even though the Embedded Controller may be in its sleep state.

The ARC, [Section 5.0, "Power, Clocks, and Resets," on page 74](#), and the [Embedded Flash Subsystem](#) interact to determine when the MEC1618/MEC1618i can stop the oscillator.

**FIGURE 13-2: FLASH\_CLK\_REQ INTERFACE**



**TABLE 13-2: Embedded Flash Subsystem POWER MANAGEMENT**

ARC Sleeping <a href="#">ARC_CLK_DISABLE.</a>	Flash Mode <a href="#">Section 13.9</a>	<a href="#">FLASH_SLP_EN</a>	<a href="#">FLASH_CLK_REQ</a>	Power	Description
0	X	0	1	High power state	ARC not sleeping clocks required and block is not commanded to sleep.
X	Not Standby Mode	X	1	High power state	Force clocks for Host or JTAG access (i.e. while ARC is sleeping)
1		1	1	High power state	ARC is sleeping but the Flash clock is still required because the block is not idle.
1	Standby Mode	1	0	Low power state	A sleep command has been asserted, the block is idle and the core clocks are stopped.

## 13.4.3 RESET

This block is reset by [nSYS\\_RST](#). Following a reset, all registers are set to their default values, and the internal state machines are reset to the standby state.

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

The JTAG interface registers referenced in this chapter have an asynchronous reset. See [Note 13-1 on page 260](#).

## 13.4.4 TRACKING FLASH PROGRAM OR ERASE ACTIVITY

When the [Embedded Flash Command Register](#) is placed in [Program Mode](#) or [Erase Mode](#), the [FLASH](#) bit is asserted in the [Power-Fail and Reset Status Register](#).

**APPLICATION NOTE:** The purpose of the [FLASH](#) bit in the [Power-Fail and Reset Status Register](#) is to provide the EC with status. After this bit is asserted by hardware, EC firmware should clear the [FLASH](#) bit as soon as possible after every [Program Mode](#) or [Erase Mode](#) operation. EC software can detect unexpected events which may indicate flash corruption. For example, VTR POR, VCC POR, LRESET#, and VCC\_PWRGD transitions, as well as, completion of host flash erase or programming are example events when firmware should examine & clear the [FLASH](#) bit in the [Power-Fail and Reset Status Register](#) can be useful.

If a reset occurs, the software reset handler can examine the [FLASH](#) bit to determine if the Flash memory might be corrupted. Corruption can occur if Flash programming or erasure is interrupted by a reset.

If VCC power is removed, or if LRESET# is asserted, the LPC interface becomes inactive and the Embedded Flash controller must ensure that the flash subsystem is in a consistent state. When the LPC bus is inoperative the EC should have full access to the Embedded Flash. [Table 13-3, "VCC PWRGD and LRESET# Behavior"](#) describes the different possible states and the response.

**TABLE 13-3: VCC PWRGD AND LRESET# BEHAVIOR**

<a href="#">Embedded Flash Configuration Register</a>	<a href="#">Embedded Flash Command Register</a>	<b>Action on LRESET# Asserted or VCC PWRGD De-Asserted</b>
<a href="#">Host_Ctl</a>	<a href="#">Flash_Mode</a>	
0	X	No action (see <a href="#">Note 13-1</a> )
1	<a href="#">Standby Mode</a>	<ul style="list-style-type: none"> <li>Reg_Ctl_En, Reg_Ctl and Host_Ctl are set to '0b'.</li> <li>EC_Int in <a href="#">Embedded Flash Command Register</a> is set to '1b'.</li> <li>The Embedded Flash is set to the <a href="#">Instruction Memory Interface</a> and a wakeup interrupt is sent to the EC.</li> </ul>
1	<a href="#">Read Mode</a>	<ul style="list-style-type: none"> <li>Any read of the Embedded Flash array in progress is completed.</li> <li>All strobes to the Embedded Flash array are set to '0b'.</li> <li>Reg_Ctl_En, Reg_Ctl and Host_Ctl are set to '0b'. The Program Mode is set to Standby.</li> <li>EC_Int in <a href="#">Embedded Flash Command Register</a> is set to '1b'.</li> <li>The Embedded Flash is set to the <a href="#">Instruction Memory Interface</a> and a wakeup interrupt is sent to the EC.</li> </ul>
1	<a href="#">Program Mode</a>	<ul style="list-style-type: none"> <li>Any word currently being programmed to the Embedded Flash array in progress is completed.</li> <li>The Program Mode epilogue sequence is issued.</li> <li>All strobes to the Embedded Flash array are set to '0b'.</li> <li>Reg_Ctl_En, Reg_Ctl and Host_Ctl are set to '0b'. The Program Mode is set to Standby.</li> <li>EC_Int in <a href="#">Embedded Flash Command Register</a> is set to '1b'.</li> <li>The Embedded Flash is set to the <a href="#">Instruction Memory Interface</a> and a wakeup interrupt is sent to the EC.</li> </ul>

# MEC1618/MEC1618i

**TABLE 13-3: VCC PWRGD AND LRESET# BEHAVIOR (CONTINUED)**

Embedded Flash Configuration Register	Embedded Flash Command Register	Action on LRESET# Asserted or VCC PWRGD De-Asserted
Host_Ctl	Flash_Mode	
1	Erase Mode	<ul style="list-style-type: none"><li>• The erase sequence is completed.</li><li>• All strobes to the Embedded Flash array are set to '0b'.</li><li>• Reg_Ctl_En, Reg_Ctl and Host_Ctl are set to '0b'.</li><li>• EC_Int in <a href="#">Embedded Flash Command Register</a> is set to '1b'. The Program Mode is set to Standby.</li><li>• The Embedded Flash is set to the <a href="#">Instruction Memory Interface</a> and a wakeup interrupt is sent to the EC.</li></ul>

**Note 13-1** The assertion of LRESET# or the de-assertion of VCC PWRGD has no effect in this state; the EC can set or clear Reg\_Ctl\_En, Reg\_Ctl and Host\_Ctl and EC\_Int is not set.

**APPLICATION NOTE:** Firmware should examine [FLASH](#) bit in the [Power-Fail and Reset Status Register](#) on [page 131](#) after either VCC POR or LRESET# assertion.

## 13.5 Interrupts

The [Embedded Flash Subsystem](#) can generate interrupts to the EC for four events, three of which are reported in the three error bits in the [Embedded Flash Status Register](#): [Protect\\_Err](#), [CMD\\_Err](#), & [Busy\\_Err](#). The error bits are routed onto the [FLASH\\_CMD\\_ERR](#), [FLASH\\_PROTECT\\_ERR](#), [FLASH\\_BUSY\\_ERR](#) bits of the [GIRQ15 Source Register](#) on [page 327](#).

In addition, asserting [EC\\_Int](#) in the [Embedded Flash Command Register](#) can be used to generate an interrupt to the EC. This bit are routed onto the [FLASH\\_EC\\_INT](#), bit of the [GIRQ15 Source Register](#) on [page 327](#).

EC interrupts generated by the LRESET# and VCC PWRGD pin signals are utilized as part of algorithms described in [Section 13.11, "Programming the Embedded Flash Array," on page 271](#). The LRESET# pin signal sets the [LRESET#](#) bit in the [GIRQ15 Source Register](#) on [page 327](#). The VCC PWRGD pin signal sets the [VCC\\_PWRGD\\_INT](#) bit in the [GIRQ14 Source Register](#) on [page 324](#).

## 13.6 Flash Memory Array

The [Flash Memory Array](#) is composed of a Main Memory array, and an [EEPROM Emulation](#) array as defined in [Table 13-4](#).

An erase operation in the [Flash Memory Array](#) sets the affected memory array bits to one, while program operations write zeros. To reprogram any '0' bit in a page to '1,' the page must be erased.

The Flash Memory Array erases and programs with a 3.3V power supply; its IO interface operates at 1.8V. To modify the contents of the [Flash Memory Array](#), VTR must be >3V before program or erase operations may begin. A summary of the MEC1618/MEC1618i [Flash Memory Array](#) features is shown below in [Table 13-4](#).

**TABLE 13-4: MEC1618/MEC1618I 192K FLASH FEATURE SUMMARY**

Feature	Description
PROG/ERASE VOLTAGE	3.3V ± 10% (T <sub>J</sub> = 0°C to 125°C)
READ VOLTAGE	1.8V ± 10% (T <sub>J</sub> = 0°C to 125°C)
BUS WIDTH	32-bit
READ ACCESS/CYCLE TIME	30 ns max. (main array) 40 ns max ( <a href="#">EEPROM Emulation</a> )
MAIN MEMORY BLOCK	48k x 32 in 96 pages of 2048 bytes

**TABLE 13-4: MEC1618/MEC1618i 192K FLASH FEATURE SUMMARY (CONTINUED)**

Feature		Description
EEPROM Emulation MEMORY BLOCK		256 x 32 in 128 pages of 8 bytes
BOOT BLOCK	SIZE	4096 bytes
	LOCATION	Bottom
ERASE	TYPES	Page/Mass (2048 bytes/page)
	Program/ Erase Cycles	10,000 cycles (main array) 250,000 cycles (EEPROM Emulation)
PROGRAMMING		Per 32-bit word
INTERFACE		All Program and Erase Operations are Enabled via a Command Sequence Interface using the Embedded Flash <a href="#">Register Interface</a> .

## 13.6.1 FLASH ADDRESS MAPPING

The Main Block of the [Flash Memory Array](#) is located at 00\_0000h in the EC address space. Address greater than 192K will wrap around. All locations can be used for both program and data by the EC.

The control registers used for programming the Flash are part of the Embedded Flash Logical Device. They are located in AHB address space starting at location [FF\\_3800h](#). The contents of the Main Block can also be read using the control registers.

## 13.6.2 EEPROM EMULATION

### 13.6.2.1 Overview

The MEC1618/MEC1618i includes [EEPROM Emulation](#) as defined in [Table 13-4, "MEC1618/MEC1618i 192K Flash Feature Summary," on page 260](#). Support for [EEPROM Emulation](#) include the [EEPROM\\_ACCESS](#), [EEPROM\\_PROTECT](#), and [EEPROM\\_FORCE\\_BLOCK](#) bits in the [Embedded Flash Configuration Register](#), and the [EEPROM\\_BLOCK](#) bit in the [Embedded Flash Status Register](#). See also the [Internal Register](#) and the [EEPROM Unlock Register](#).

### 13.6.2.2 Internal Register

The [Embedded Flash Subsystem](#) includes a 31-bit [Internal Register](#) that cannot be directly read or written over SPB. Use of the [Internal Register](#) is described in the next section, [Section 13.6.2.3, "EEPROM Initialization"](#).

### 13.6.2.3 EEPROM Initialization

The [Embedded Flash Subsystem](#) controller reads offset 3FFh from the [EEPROM Emulation](#) Memory Block and loads the least significant 31 bits of the data into the [Internal Register](#). If the low 31 bits are not 7FFF\_FFFFh, and bit 31 is a '1,' the [EEPROM\\_BLOCK](#) bit in the [Embedded Flash Status Register](#) is set to '1' and the [EEPROM Emulation](#) Memory Block becomes inaccessible.

### 13.6.2.4 Functional Description

In order to permit [EEPROM Emulation](#) access from within the Boot Block of the Flash main body, EC accesses to the on-chip ROM should not trigger the Boot Block lock function.

When [EEPROM\\_ACCESS](#) is '1,' the Sector Select pins of the Flash IP use Flash\_Address[17:8] for sector selection while accessing the Flash main memory block, and Flash\_Address[9:3] while accessing the [EEPROM Emulation](#) memory block.

When [EEPROM\\_ACCESS](#) is '1,' the Flash access time rises from 30ns to 40ns. This may or may not affect the Read mode register access. Program time and erase time are the same for the [EEPROM Emulation](#) region and the rest of the [Flash Memory Array](#).

Note that the [EEPROM Emulation](#) memory block is not erased with the main memory block when the Erase function is chosen by the [Embedded Flash Command Register](#). When Mass Erase is initiated by JTAG, both the [EEPROM Emulation](#) memory block and the main memory block are erased.

# MEC1618/MEC1618i

## 13.6.2.4.1 Password Use

Offset 3FCh is the last 32-bit word in the [EEPROM Emulation](#) memory block. If bit 31 is a '0' in that word, then the location can be treated as [EEPROM Emulation](#) data that stores only positive integers in the range 0 to 2 billion (or so). If bit 31 is a '1' and the rest of the word is not all 1's (that is, if the word is not in its initial, erased, state), then the word contains a password that can be used to unlock the [EEPROM Emulation](#) memory block.

To keep the [EEPROM Emulation](#) memory block secure, and to insure that the EC firmware is authorized to read its contents, the EC firmware can use this password. The following code can be executed as part of the 4KB Boot Block portion of the main flash array:

```
int password;
#define SECRET_PASSWORD 0xFFFFFFFF
#define Unlock_register 0xFF3920 // address of key register

password = eeprom_read(0x3FC);
if( password == 0xFFFFFFFF )
{
    // EEPROM is unlocked and the key not yet installed
    // establish key and force a reboot
    eeprom_write(0x3FF, SECRET_PASSWORD);
    force_reset();
} else {
    // unlock the EEPROM
    *Unlock_register = SECRET_PASSWORD;
}
```

The subroutines [eeprom\\_write\(\)](#) and [eeprom\\_read\(\)](#) in the above example are part of the [EEPROM API](#) that is included in the [Boot ROM](#); [force\\_reset\(\)](#) forces a system reset, using, for example, the watchdog timer. If the EC firmware knows the right password, it can unblock the [EEPROM Emulation](#) memory block; if it does not, then the block will remain inaccessible. The code runs in the Boot Block so that the SECRET\_PASSWORD (a constant in the code) cannot be read via JTAG or over the LPC bus, and thus can only be known to a valid EC firmware block.

## 13.7 Instruction Memory Interface

When [Reg\\_Ctl](#) of the [Embedded Flash Command Register](#) is '0b', access to the Embedded Flash memory is through the Instruction Closely Coupled Memory interface of the EC. The registers in the [Register Interface](#) may be read or written, but no operations will be initiated on the Embedded Array. The Flash memory can be read with no wait states at the peak EC clock rate.

The minimum flash read access time through the EC's Instruction Closely Coupled Memory interface is shown in [Table 13-5](#).

**TABLE 13-5: FLASH READ ACCESS TIME**

	Parameter	Symbol	Value			Units	Notes
			Min	Typ	Max		
1.	Flash Read Access Time	$t_{\text{Flash\_Rd}}$	2	–	–	MCLK	

## 13.8 Register Interface

The [Embedded Flash Subsystem](#) has its own Logical Device Number and Base Address as indicated in [Table 13-6](#).

**TABLE 13-6: Embedded Flash Subsystem BASE ADDRESS TABLE**

<a href="#">Embedded Flash Interface Instance</a>	LDN	AHB Base Address
<a href="#">Embedded Flash Interface</a>	Eh	FF_3800h

[Table 13-7, "Embedded Flash Subsystem Register Summary," on page 264](#) summarizes the registers allocated for the Embedded Flash Subsystem.

The [Table 13-7](#) is a register summary for one instance of the [Embedded Flash Subsystem](#). The LPC I/O address for each Run-Time Register is described below as an offset from its Base Address Register. Each EC address is indicated as an SPB Offset from its AHB base address.

Both the Host and the Embedded Controller can communicate with the [Embedded Flash Subsystem](#) through a set of registers located on the LPC SPB bus. The Host uses the two byte [Flash Mailbox Interface](#), while the EC can access the register interface described in [Section 13.8.2, "EC Register Interface"](#).

## 13.8.1 FLASH MAILBOX INTERFACE

The [Flash Mailbox Interface](#) provides the Host with a Message Interface similar to the [MailBox Register Interface](#) (see [Section 11.0](#)). The [Flash Mailbox Interface](#) port occupies two addresses in the Host I/O space: [FL\\_MBX\\_INDEX Register](#) & [FL\\_MBX\\_DATA Register](#).

To access a Flash Mailbox register after the Flash Mailbox BAR has been initialized, write the Host Offset of the desired 8-bit register into the Flash Mailbox INDX register. The register can then be accessed by reading or writing the Flash Mailbox DATA register.

It consists of 16 index-addressable 8-bit registers. These registers correspond to the four 32-bit registers that are shared between the Host and the EC.

To access a Flash Mailbox register after the Flash Mailbox BAR has been initialized, write the Host Offset of the desired 8-bit register into the Flash Mailbox INDX register. The register can then be accessed by reading or writing the Flash Mailbox DATA register.

**Note 13-2** In this specification, Host access to registers in the Flash Mailbox through the Flash Mailbox Access Port are identified by the prefix INDX in OFFSET fields in the register tables in [Section 13.12, "Detailed Description of Accessible Registers"](#).

# MEC1618/MEC1618i

## 13.8.2 EC REGISTER INTERFACE

The following table summarizes the registers allocated for the [Embedded Flash Subsystem](#). The offset field in the following table is the offset from the EC Base Address.

**TABLE 13-7: EMBEDDED FLASH SUBSYSTEM REGISTER SUMMARY**

	Host I/O Access			EC Interface		
Flash Mailbox Interface Register Name	Host I/O Address	SPB Offset	Host Type	SPB Offset	EC Type	nSYS_RST
FL_MBX_INDEX Register	00h	00h	R/W	00h	R/W	00h
FL_MBX_DATA Register	04h	04h	R/W	04h	R/W	00h
Register Name	Flash Mailbox Index		Host Type	SPB Offset	EC Type	VTR POR
Embedded Flash Data Register	INDX 00h, 01h, 02h, 03h	-	R/W	100h	R/W	0000_0000h
Embedded Flash Address Register	INDX 04h, 05h, 06h,	-	R/W	104h	R/W	0000_0000h
Embedded Flash Command Register	INDX 08h, 09h	-	R/W	108h	R/W	0000_0000h
Embedded Flash Status Register	INDX 0Ch, 0Dh	-	R/WC	10Ch	R/WC	0000_0000h
Embedded Flash Configuration Register	N/A	-	-	110h	R/W	0000_0000h
Embedded Flash Initialization Register	N/A	-	-	114h	R/W	0000_0000h
EEPROM Unlock Register	N/A	-	-	120h	W	-

## 13.8.3 FLASH ADDRESS AND DATA REGISTERS

The [Embedded Flash Data Register](#) and the [Embedded Flash Address Register](#) are each implemented as a two entry FIFO, as illustrated in Figure 13-3, "Embedded Flash Controller Address and Data Registers". The figure is suggestive and does not represent the precise implementation. Writes are always directed to the Tail register of the FIFO and reads are always sourced from the Head register. Each of the two registers has an associated Valid bit (**V** in the figure). The Tail Valid bit is set when the full Tail register is written and cleared when the Tail register is copied into the Head register. The Head Valid bit of the [Embedded Flash Data Register](#) is cleared whenever the full Head register is read by the Host or EC, or when the Embedded Flash Controller completes a Program transaction. The Head Valid bit of the [Embedded Flash Address Register](#) is cleared whenever the Embedded Flash controller completes a Read, Program or Erase function. Reading the [Embedded Flash Address Register](#) by a Host or EC register read does not affect the Head Valid bit. If the Head Valid bit is cleared and the Tail Valid bit is set, the Tail register is copied into the Head register, the Tail Valid bit is cleared and the Head Valid bit is set. All Valid bits are cleared when [Flash\\_Mode](#) of the [Embedded Flash Command Register](#) is set to [Standby Mode](#).

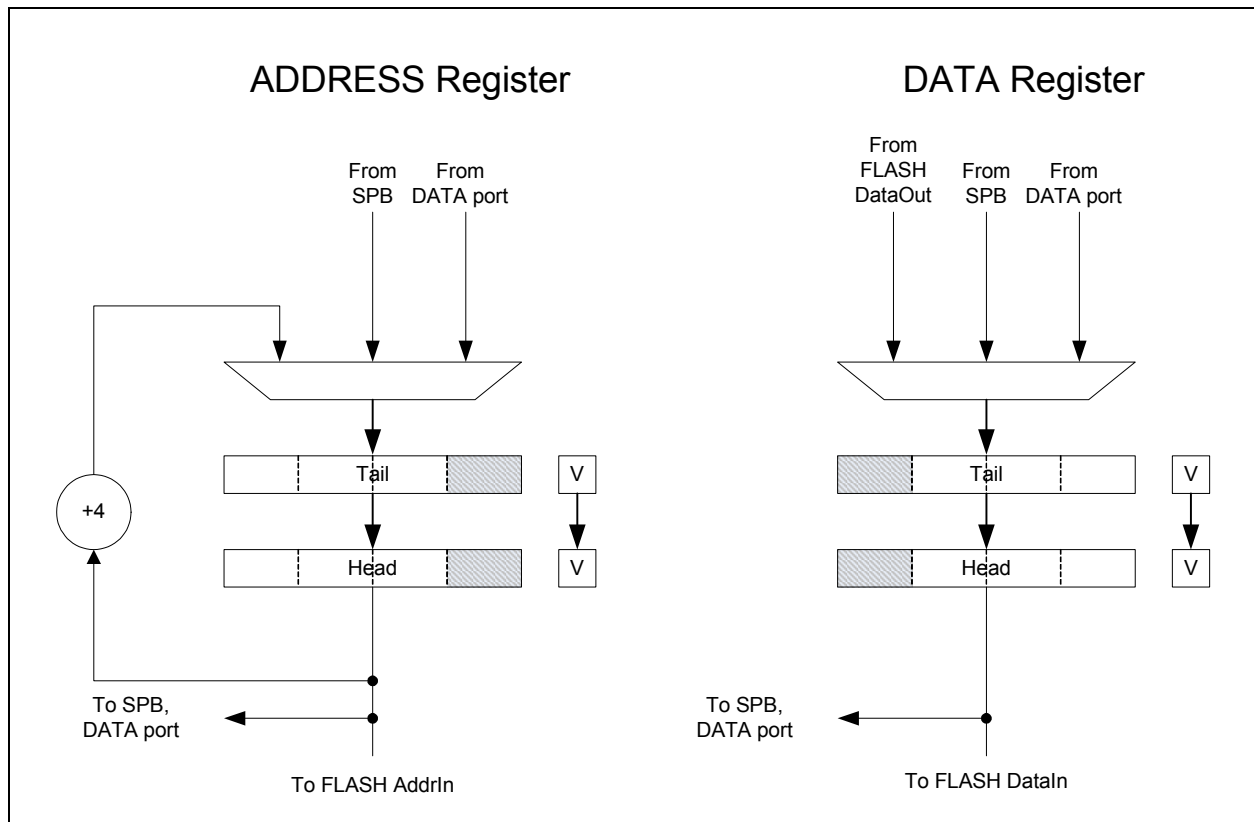
The Embedded Flash controller uses byte 0 of the [Embedded Flash Address Register](#) and byte 3 of the [Embedded Flash Data Register](#) to determine when the full register has been read or written. Writing byte 0 of the [Embedded Flash Address Register](#) sets the Valid bit on the Tail of the Address FIFO, while writing byte 3 of the [Embedded Flash Data Register](#) sets the Valid bit for the Tail of the Data FIFO. Reading byte 3 of the [Embedded Flash Data Register](#) clears the Valid bit for the Head of the Data FIFO. As stated above, reading any byte of the [Embedded Flash Address Register](#) does not cause a change in the Head Valid bit.



When the Valid bits of both the Head and Tail of the [Embedded Flash Data Register](#) are set, [Data\\_Full](#) in the [Embedded Flash Status Register](#) is set. If either bit is cleared, [Data\\_Full](#) is cleared. When the Valid bits of both the Head and Tail of the [Embedded Flash Address Register](#) are set, [Address\\_Full](#) in the [Embedded Flash Status Register](#) is set. If either bit is cleared, [Address\\_Full](#) is cleared.

The requesting Master will be stalled while attempting to read the [Embedded Flash Data Register](#) if the Embedded Flash controller is Busy and in the process of reading data into the [Embedded Flash Data Register](#). This will result in at most one EC wait state. Because the Host cannot read the Flash registers faster than the controller can read from the Flash array, a Host Read access over the LPC bus will not have any added Wait SYNC cycles because the Flash controller is busy. In other cases, a Host or EC read of either the [Embedded Flash Data Register](#) or the [Embedded Flash Address Register](#) always returns the value in the Head register, whether the Valid bit is set or not.

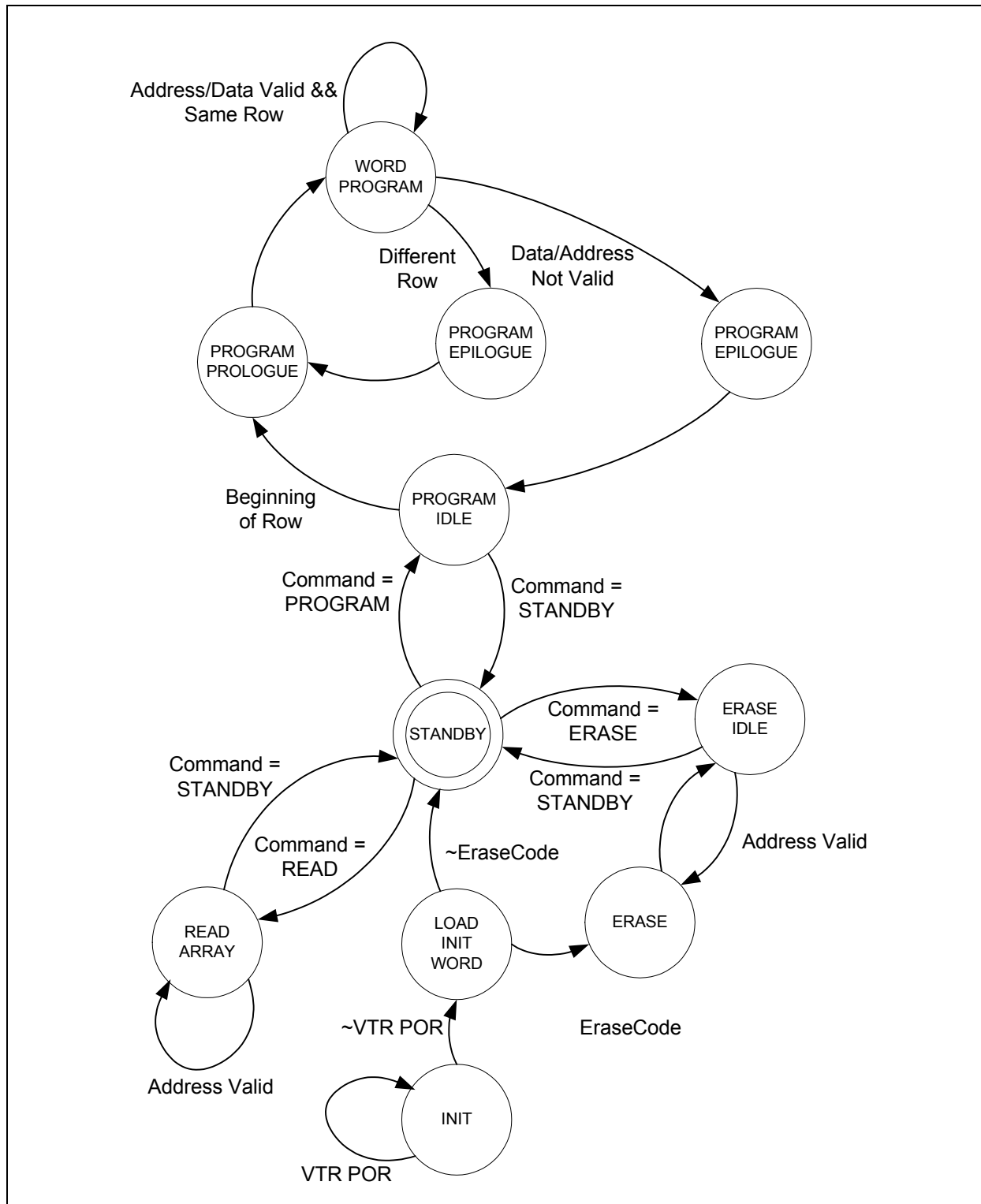
**FIGURE 13-3: EMBEDDED FLASH CONTROLLER ADDRESS AND DATA REGISTERS**



## 13.9 Embedded Flash Controller State Sequencing

The Embedded Flash controller proper timing on all strobe signals of the Embedded Flash array. Figure 13-4, "Embedded Flash Controller State Diagram" illustrates the primary state transitions for the controller.

**FIGURE 13-4: EMBEDDED FLASH CONTROLLER STATE DIAGRAM**



During VTR Reset, the Embedded Flash controller is kept in an initial state (INIT). In normal operation, when VTR POR (`nSYS_RST`) is de-asserted, the Embedded Flash controller reads the initialization value into the [Embedded Flash Initialization Register](#), as described in [Section 13.10.5, "Flash Data Initialization," on page 271](#), after which it remains in [Standby Mode](#). If the `ME` (Mass Erase) field in the JTAG TEST REGISTER 4/Reset Register (`Dh`) is '1b' when VTR POR (`nSYS_RST`) is de-asserted the Embedded Flash controller, after transitioning to the [Flash Data Initialization](#) state will transition to [Erase Mode](#). See [Section 13.10.6, "Emergency Mass Erase," on page 271](#) for detailed sequence and description.

After initialization during normal operation, the Embedded Flash controller is placed into one of four modes ([Standby Mode](#), [Read Mode](#), [Program Mode](#) or [Erase Mode](#)) by setting `Flash_Mode` of the [Embedded Flash Command Register](#) to the appropriate values (0, 1, 2 or 3, respectively). Any write to the [Embedded Flash Command Register](#) while the controller is busy (when `Busy` of the [Embedded Flash Status Register](#) is '1b') will not modify the state and will set `Busy_Err` in the [Embedded Flash Status Register](#).

## 13.9.1 STANDBY MODE

When in this mode all strobes to the Flash Memory Array are de-asserted and the Flash Memory Array is placed in its lowest power state. Both the [Embedded Flash Data Register](#) FIFO and the [Embedded Flash Address Register](#) FIFO are flushed. The registers can be read and written by software without error. Data returned on a read of a flushed FIFO is undefined. On a transition from [Standby Mode](#) to [Program Mode](#), [Read Mode](#) or [Erase Mode](#) the FIFOs will be invalid and no action dependent on valid data in a FIFO will take place until new data is written into the registers.

The Embedded Flash controller must return to [Standby Mode](#) after any other state. An attempt to set the controller into a state other than [Standby Mode](#) when the controller is in [Program Mode](#), [Read Mode](#) or [Erase Mode](#) will put the controller in [Standby Mode](#) and set `CMD_Err` of the [Embedded Flash Status Register](#). `Flash_Mode` of the [Embedded Flash Command Register](#) will also be left in [Standby Mode](#). If the controller is busy, writes to the [Embedded Flash Command Register](#) are ignored and `Busy_Err` in the [Embedded Flash Status Register](#) is set.

## 13.9.2 READ MODE

When the Embedded Flash controller is in Read Mode the addressing strobes are kept de-asserted as long as no Read is in progress. When the strobes are de-asserted, the Embedded Flash array is maintained in its lowest power state (equivalent to [Standby Mode](#)). When the Valid bit associated with the Head register in the [Embedded Flash Address Register](#) is '1b' and the Data FIFO is not full, the data in the Embedded Flash array that corresponds to the [Embedded Flash Address Register](#) is loaded into the Tail register of the [Embedded Flash Data Register](#). The Data FIFO can hold two read results.

The Address FIFO is advanced when the Embedded Flash controller completes a read from the Embedded Flash array and stores the result in the Data FIFO. When byte 3 (the most significant byte) of the [Embedded Flash Data Register](#) is read, the Data FIFO is advanced. A read of the will return the value of the Head register but will not advance the FIFO. A read of the [Embedded Flash Data Register](#) when the FIFO is empty but the controller is busy reading from the array will stall the read until the controller has completed the lookup and the data can be returned to the requestor.

### 13.9.2.1 Read Mode Timing Parameters

The minimum flash read access time through the [EC Register Interface](#) is one [MCLKs](#).

### 13.9.2.2 Burst Read Mode

When `Burst` in the [Embedded Flash Command Register](#) is set to '1b', the Read function automatically increments the [Embedded Flash Address Register](#) in order to minimize the time necessary to read a block of memory from the Embedded Flash. As in [Read Mode](#), a Flash read is initiated when the Head of the Address FIFO is Valid and the Data FIFO is empty. Whenever the Tail register in the Address FIFO is not Valid and the Head register in the FIFO is Valid, the Head register is incremented by 4 and written into the Tail. The Address FIFO will thus contain a sequence of consecutive word addresses as long as the Embedded Flash controller remains in [Burst Read Mode](#).

Reading the [Embedded Flash Data Register](#) from either the EC or the JTAG interface always reads 32 bits at a time, so the Data Register FIFO is always advanced when the EC or JTAG does a data read. In [Burst Read Mode](#) the entire Flash memory can be read by repeatedly reading the [Embedded Flash Data Register](#) without the need for reading or writing any other register in the Embedded Flash Subsystem.

The [Flash Mailbox Interface](#) has an additional mechanism. When in [Burst Read Mode](#), the `INDX` portal has special behavior when it has the value 0h through 3h (that is, when `INDX` is set to point to the [Embedded Flash Data Register](#)). Every time the DATA portal is read by the Host, `INDX` will be automatically incremented by 1 when its value is 0h through 2h. If DATA is read by the Host when `INDX` is 3h, `INDX` is set to 0h and the Valid bit of the Head register of the Data

# MEC1618/MEC1618i

FIFO is cleared. If there is a valid address in the Address FIFO, the sequence to read the next data value, described above, will be initiated. When in [Burst Read Mode](#), the entire contents of the Flash memory can be read by the Host with a sequence of reads to DATA, without any intervening writes to INDX.

**APPLICATION NOTE:** In [Burst Read Mode](#) the [Embedded Flash Data Register](#) will always contain the data from the next two locations after the last data location that is read. Software should ensure that the last word read does not cause an unintended read into a protected region. For example, a Burst Read that is intended to read the last word before the Protected Data Region will attempt to read into the protected region, which may cause an unintended protection error. To read the last data before the protected region, software should turn off [Burst Read Mode](#) and return to [Standby Mode](#) two words before the end of the page before the protected region. It should then re-enter [Read Mode](#) and read the last two words by explicitly writing the addresses into the [Embedded Flash Address Register](#).

## 13.9.3 PROGRAM MODE

When the Embedded Flash controller is in [Standby Mode](#), setting the [Embedded Flash Command Register](#) to [Program Mode](#) will set up the Embedded Flash array for programming. The [Embedded Flash Data Register](#) and the [Embedded Flash Address Register](#) FIFOs are used for [Program Mode](#) in a manner similar to their use in [Read Mode](#). When the Head registers in both the Address and Data FIFO are Valid, the Flash strobes are sequenced in order to write the contents of the Data register into the Flash Memory at the address specified in the Address register. At the end of the Word Program sequence the two FIFOs are advanced.

**Note:** In [Program Mode](#) there should always be an erase before writing the same DWORD twice.

### 13.9.3.1 Program Mode Timing Parameters

The following three sequences are used in programming the Embedded Flash: Program Prologue, Word Program, Program Epilogue.

During all three sequences, the controller asserts Busy while the sequence is in progress. Flash programming time is shown in [Table 13-8](#).

**TABLE 13-8: FLASH PROGRAMMING TIME**

	Parameter	Symbol	Value			Units	Notes
			Min	Typ	Max		
1.	Single DWORD Programming Time	–	19	–	–	μs	
2.	Burst DWORD Programming Time (single row)	–	13.5	–	–	μs	

The Program Prologue sequence is issued before the first time any word in a row can be programmed. The Program Epilogue sequence is issued after the last time any word of a row is programmed. The Embedded Flash controller automatically issues the Program Prologue and Program Epilogue sequences as required. After the controller is placed in [Program Mode](#), the Program Prologue is issued as soon as the Head registers in both the [Embedded Flash Data Register](#) and the [Embedded Flash Address Register](#) are Valid. After the Program Prologue is completed, the controller will immediately issue the Word Program sequence.

As long as the Head registers in both the [Embedded Flash Data Register](#) and the [Embedded Flash Address Register](#) are Valid the controller will continue to issue Word Program sequences. If either FIFO is invalid at the end of the Word Program sequence, the controller will issue the Program Epilogue sequence and the controller will be left in the initial [Program Mode](#) state. If new data arrives in the FIFOs, then the Program Prologue followed by the Word Program sequence will be issued.

### 13.9.3.2 Burst Program Mode

**Burst Program Mode** is enabled whenever **Burst** in the **Embedded Flash Command Register** is set and the controller is in **Program Mode**. The behavior is similar to **Burst Read Mode**. When **Burst** is enabled, the **Embedded Flash Address Register** FIFO is always kept filled automatically with incrementing addresses. Whenever the Tail register in the Address FIFO is not Valid and the Head register is valid, the Head register is incremented by 4 and stored in the Tail. The controller otherwise behaves as in **Program Mode**. When both the Head register of the Address FIFO and the Head register of the Data FIFO are Valid, a Word Program sequence is initiated (with the possible addition of Program Prologue and Program Epilogue, as described above). When the programming sequence is complete, the Head registers of both FIFOs are marked not Valid. With this mechanism the entire Flash array can be programmed with a sequence of writes to the **Embedded Flash Data Register** without any writes to the **Embedded Flash Address Register** or **Embedded Flash Command Register**.

The **Flash Mailbox Interface** has an additional mechanism, as in **Burst Read Mode**. The **INDX** portal has special behavior when it has the value 0h through 3h (that is, when **INDX** is set to point to the **Embedded Flash Data Register**). Every time the **DATA** portal is written by the Host, **INDX** will be automatically incremented by 1 when its value is 0h through 2h. If **DATA** is written by the Host when **INDX** is 3h, **INDX** is set to 0h and the Tail register of the Data FIFO is advanced to the Head and the Tail is marked not Valid. If there is a valid address in the Head of the Address FIFO, the Word Program sequence will be initiated. In **Burst Program Mode**, the entire contents of the Flash memory can be programmed by the Host with a sequence of writes to **DATA**, without any intervening writes to **INDX**.

### 13.9.4 ERASE MODE

When the **Embedded Flash Command Register** is set to **Erase Mode**, the state machine will wait until a valid address is written into the **Embedded Flash Address Register**. After the register is valid, **Busy** in the **Embedded Flash Status Register** is set to '1b' and the Embedded Flash controller sequences the Embedded Flash array strobes to erase part or all of the Embedded Flash. All bits in the erased area are set to '1b'. **Busy** remains set during the operation. When the erase operation is complete **Busy** is set to '0b'. The controller remains in **Erase Mode** and can accept additional addresses in the **Embedded Flash Address Register**.

The **EEPROM\_ACCESS** field in the **Embedded Flash Configuration Register** determines which array is selected to be erased: Main Memory array, or **EEPROM Emulation** array.

Bits[23:19] of the **Embedded Flash Address Register** selects between page and mass erase of the entire array selected. Some of the Bits[17:0] of the **Embedded Flash Address Register** select which page is erased and the other bits are ignored. The **Embedded Flash Address Register** should be configured with the proper page address before setting the **Embedded Flash Command Register** to **Erase Mode**.

Table 13-9, "Erase Mode Functions" provides the required decoding of the memory to be erased.

**TABLE 13-9: ERASE MODE FUNCTIONS**

<b>EEPROM_ACCESS</b> <b>Embedded Flash</b> <b>Configuration Register</b>	<b>Address</b> <b>Register</b> <b>Bits[23:19]</b>	<b>Area Erased</b>
0	xxxx0b	2KB Page in Embedded Flash Main Array specified by <b>Embedded Flash Address Register</b> bits[17:11] & bits[10:0] are ignored
0	11111b	Mass erase of entire Embedded Flash Main Array
1	xxxx0b	8 byte Page in <b>EEPROM Emulation</b> MEMORY BLOCK specified by <b>Embedded Flash Address Register</b> bits[9:3] & bits[2:0] are ignored
1	11111b	Mass erase of entire <b>EEPROM Emulation</b> MEMORY BLOCK
<b>Note:</b> All other combinations are MCHP reserved and will have unintended results.		

## 13.10 Flash Lock Controls

### 13.10.1 FLASH WRITE PROTECT

When the [Boot\\_Lock](#) in the [Embedded Flash Configuration Register](#) is asserted, the bottom 4K bytes (0000h – 0FFFh) of the Flash Main Memory Array (the Boot Block) are write protected and cannot be changed by any programming method including [Program Mode](#) or [Erase Mode](#). The Mass Erase function is therefore disabled, since it would erase the Boot Block. The rest of the Embedded Flash array, is not affected. If [Boot\\_Lock](#) is not asserted, all programming and erase functions, including [Erase Mode](#), are enabled and the bottom 4K bytes are not protected.

### 13.10.2 FLASH BOOT PROTECT

The Embedded Flash Boot Block can be protected from all reads and writes, from either the Host or the EC. If [Boot\\_Protect\\_En](#) in the [Embedded Flash Configuration Register](#) is '1b' the first time there is a address reference to the Embedded Flash Subsystem that is outside the Boot Block (that is, the first time the EC does either a program fetch to an address that is larger than 00\_0FFFh or a data reference to an address that is larger than 00\_0FFFh and less than 80\_0000h), [Boot\\_Block](#) and [Boot\\_Lock](#) in the [Embedded Flash Status Register](#) are set. [Boot\\_Lock](#) prevents any programming or erase operations on the Boot Block, as described in [Section 13.10.1, "Flash Write Protect"](#). Mass Erase is also disabled. In addition, any attempt to read the Boot Block, from either the [Instruction Memory Interface](#) or the [Register Interface](#) will return FFFF\_FFFFh. After being set, only a VTR POR ([nSYS\\_RST](#)) can clear [Boot\\_Block](#).

Typically, Flash Boot Protect will be the last function performed as part of boot. The EC boot code should configure the Interrupt Vector Table to be outside the Boot Block. It should then set the [Boot\\_Protect\\_En](#) bit, then jump to a location outside of the Boot Block. From that point on the Boot Block will be protected for both read and write access from either the Host or the EC.

#### 13.10.2.1 JTAG Disable to protect the flash Boot Block

If [Boot\\_JTAG\\_Block](#) in the [Embedded Flash Initialization Register](#) is set to '0b', the [JTAG Debug Data Registers](#) interface will not be accessible to the MEC1618/MEC1618i JTAG pins as long as [Boot\\_Block](#) in the [Embedded Flash Status Register](#) is '0b'. This insures that an external device cannot read or write the Boot Block while the EC is executing within it. If [Boot\\_JTAG\\_Block](#) is in its default state of '1b', an external debugger could potentially halt the EC and then read or reprogram any word in the Boot Block.

As described in [Section 13.10.5, "Flash Data Initialization"](#), data in the [Embedded Flash Initialization Register](#) are configured by programming the initialization vector in the Embedded Flash array.

### 13.10.3 FLASH DATA PROTECT

The Flash Data Protect Region is the last 4KB in the Embedded Flash main array. If [Data\\_Protect](#) in the [Embedded Flash Configuration Register](#) is '1b' the Flash Data Protect Region is protected from any read or write accesses by the Host through the Mailbox interface. The EC can access the Data Protect Region through both the [Instruction Memory Interface](#) and the [Register Interface](#), as long as the JTAG port is not enabled. After JTAG is enabled, by de-asserting the JTAG Reset pin ([JTAG\\_RST#](#)), [Data\\_Block](#) in the [Embedded Flash Status Register](#) is set to '1b' and the entire Data Protect Region is blocked from all accesses, by the Host, by the EC or by JTAG. Any reads return FFFF\_FFFFh, and any attempt to program data in the region will be blocked. Any page erase operation in the Data Protect Region is blocked and Mass Erase is also disabled. After the [Data\\_Block](#) is set, it can only be cleared by a VTR POR ([nSYS\\_RST](#)). Clearing [Data\\_Protect](#) in the [Embedded Flash Configuration Register](#) will not change [Data\\_Block](#), so the only way to access the protected region after JTAG is enabled is to power cycle the device.

### 13.10.4 FLASH EEPROM PROTECT

The [Flash Data Protect](#) feature protects the [EEPROM Emulation](#) Memory. If [EEPROM\\_PROTECT](#) in the [Embedded Flash Configuration Register](#) is '1b' the [EEPROM Emulation](#) Memory is protected. The EC can access the [EEPROM Emulation](#) Memory through the [Register Interface](#), as long as the JTAG port is not enabled. After JTAG is enabled, by de-asserting the JTAG Reset pin ([JTAG\\_RST#](#)), [EEPROM\\_BLOCK](#) in the [Embedded Flash Status Register](#) is set to '1b' and all accesses (read, program, & erase) by the EC or by JTAG to the [EEPROM Emulation](#) Memory are blocked. Any read of the [EEPROM Emulation](#) Memory returns FFFF\_FFFFh.

After [EEPROM\\_PROTECT](#) is set, it can only be cleared by a VTR POR ([nSYS\\_RST](#)). Clearing [EEPROM\\_PROTECT](#) in the [Embedded Flash Configuration Register](#) will not change the [EEPROM\\_BLOCK](#) bit, so the only way to access the [EEPROM Emulation](#) Memory after JTAG is enabled is to power cycle the device.

## 13.10.5 FLASH DATA INITIALIZATION

Before the EC come out of reset, the Embedded Flash Controller

- reads one 32-bit word from offset FFCh of the Flash main memory block and writing it to the [Embedded Flash Initialization Register](#) on page 285.

The read-only [Embedded Flash Initialization Register](#) can be used for configuration and initialization data. After being written, this data will stay constant until the upper page of the Boot Block is erased and the VTR power is cycled.

[Boot\\_JTAG\\_Block](#) bit in the [Embedded Flash Initialization Register](#) is used to control JTAG access during the boot period. If this bit is programmed to '0b', JTAG cannot access any locations within the MEC1618/MEC1618i, other than the JTAG test registers, while the Boot Block is readable. If [Boot\\_Block](#) in the [Embedded Flash Status Register](#) is set, JTAG accesses are enabled, although any access to the Boot Block is blocked by [Boot\\_Block](#). If [Boot\\_JTAG\\_Block](#) is left in the default '1b' state, a JTAG master could halt the EC just after VTR POR ([nSYS\\_RST](#)) and manipulate all locations in the MEC1618/MEC1618i, including any data in the Boot Block.

## 13.10.6 EMERGENCY MASS ERASE

If [Boot\\_JTAG\\_Block](#) of the [Embedded Flash Initialization Register](#) is asserted but the EC boot code is faulty, it is possible that the EC could become stuck in the boot sequence without ever enabling JTAG access or LPC access. In that state, there would be no way to erase the Embedded Flash and reprogram the EC, since both the JTAG port and LPC access would be inaccessible. For this reason, a fail-safe mechanism is included in the Embedded Flash controller

If the [ME](#) (Mass Erase) field is '1b' when VTR POR ([nSYS\\_RST](#)) is de-asserted the Embedded Flash controller, after transitioning to the [Flash Data Initialization](#) state will transition instead to [Erase Mode](#). After the Mass Erase is completed the Embedded Flash is held in an idle state and the Embedded Controller will remain idle. JTAG and LPC will be blocked as well.

An additional power cycle of the MEC1618/MEC1618i will be required to reset the [Embedded Flash Initialization Register](#) and re-enable the Embedded Flash and the Embedded Controller. After the additional power cycle JTAG can be used to reprogram the Embedded Flash. [Emergency Mass Erase](#) affects both the Main Memory array, and the [EEPROM Emulation](#).

In order to trigger an Emergency Mass Erase, the JTAG interface should be used to perform the following sequence:

1. Place the JTAG\_RST# pin in the asserted (low) state and apply VTR Power.
2. De-assert the JTAG\_RST# pin (high) to take the JTAG interface out of the reset state (see [Section 43.11, "JTAG Interface Timing,"](#) on page 653).
3. The [POR EN](#) field in the JTAG [RESET TEST Register \(2h\)](#) should be set to '1b' in order to enable JTAG control of the VTR POR circuitry.
4. The [VTR POR](#) field in the JTAG [RESET TEST Register \(2h\)](#) should be set to '1b' in order to generate a VTR POR in the MEC1618/MEC1618i.
5. The [ME](#) field in the JTAG [TEST REGISTER 4/Reset Register \(Dh\)](#) should be set to '1b'
6. The [VTR POR](#) field in the JTAG [RESET TEST Register \(2h\)](#) should be set to '0b' which terminates the VTR POR.
7. Delay long enough to ensure a Mass Erase completes and the VTR POR ([nSYS\\_RST](#) timing) should be set to '1b' in order to generate a VTR POR in the MEC1618/MEC1618i
8. The [VTR POR](#) field in the JTAG [RESET TEST Register \(2h\)](#) should be set to '0b' which terminates the VTR POR.

## 13.11 Programming the Embedded Flash Array

In normal operation, the EC fetches all of its instructions from the Embedded Flash array. While it is executing out of Flash, the program interface using the registers described in the next section cannot be used. When the [Register Interface](#) is used, the EC cannot fetch either instructions or data out of the Embedded Flash. The EC can either sleep, or run entirely out of the [EC Data Memory](#). When running out of [EC Data Memory](#), EC firmware should insure that all interrupts are disabled, unless the interrupt vector table is relocated to the [EC Data Memory](#) along with any interrupt handler that is enabled.

There are three methods by which the Embedded Flash array can be programmed:

- The Flash may be programmed through the JTAG interface. Through the JTAG interface, the EC can be halted or directed to run entirely out of the [EC Data Memory](#). After the EC no longer requires the Flash, the registers in the next section can be used to program the array, by first setting [Reg\\_Ctl\\_En](#) in the [Embedded Flash Configuration Register](#) (enabling register control of the Flash), then using the other registers to program the array. After programming is completed, [Reg\\_Ctl\\_En](#) in the [Embedded Flash Configuration Register](#) is cleared (enabling EC



instruction access) and then setting the EC into the RUN state. See [Section 41.0, "JTAG and XNOR," on page 604](#) for a description of JTAG operation.

- The Flash may be programmed by the Host. After co-ordination between the Host and the EC, the EC configures the [Embedded Flash Configuration Register](#) to enable the register programming mode and to give the Host write access to the registers (by setting both [Reg\\_Ctl\\_En](#) and [Host\\_Ctl](#) to '1b'). The EC would then disable all interrupts except the EC\_Int Flash Mailbox interrupt from the Host. The EC Interrupts for LRESET# and VCC PWRGD pin signals should also be enabled (see [Section 13.5, "Interrupts," on page 260](#)), in order to restore control to the EC in the event the Host loses power and does not return control to the EC properly. The EC then signals the Host that it can proceed. At this point the EC should verify that the Host can take control, by checking the state of LRESET# and VCC PWRGD. If LRESET# is asserted or VCC PWRGD is de-asserted, the Host will not respond to the EC signal to take control, and the EC should therefore cancel the hand off to the Host, clear [Reg\\_Ctl\\_En](#) and [Host\\_Ctl](#), re-enable interrupts and continue. If the Host is capable of responding to the EC signal, the EC finally either puts itself to sleep or runs entirely out of the. When the Host finishes programming the Embedded Flash array, it sets [EC\\_Int](#) in the [Embedded Flash Command Register](#) to '1b', which wakes the EC.

The Host may also program the Flash through the use of the [MailBox Register Interface](#). The Host communicates the address to be modified and the data to program through a pre-arranged set of mailbox registers, then sends an interrupt to the EC. The EC would then program the Flash while running out of the on-chip SRAM, as described in the following bullet item.

- The EC can itself program the Flash array by loading a program to do so into the on-chip 4KB SRAM. After the EC is running out of SRAM, it can disable all interrupts and configure the Flash array to be programmable through the registers (by setting [Reg\\_Ctl\\_En](#) in the [Embedded Flash Configuration Register](#)). After programming is completed, the EC can reset the [Reg\\_Ctl](#) bit to make the Flash accessible to instruction fetch.

In all cases, the [Embedded Flash Data Register](#) can be updated to set up the next write while the Flash is busy programming the current write.

## 13.11.1 TRANSFERRING CONTROL TO THE HOST

In order to transfer control of the Embedded Flash to the Host, the EC has to enable the Host access to the Register interface, inform the Host that it has access, then either sleep or execute code entirely out of the [EC Data Memory](#). The EC must sleep or execute in RAM because it will no longer be able to use the ICCM instruction memory interface. The following sequence is an example of how this transfer can be accomplished:

```
// This code is located in the Flash
// Command_Address is 0xFF3908
// Config_Address is 0xFF3910
//
*Config_Address |= REG_CTL_EN | HOST_CTL; // Permit the Host to control the Flash
registers
Disable_all_interrupts();
Enable_Interrupt(EC_Int);                // Enable transition from Host back to EC
Enable_Interrupt(LRESET#);
Enable_Interrupt(VCC_PWRGD);
Host_Message(FLASH_READY);              // Tell Host it can take control of the Flash
if( _asserted(LRESET) || deasserted(VCCPWRGD) ) {
    cancel_flash_transfer();             // Host is not powered: annul transfer
                                        // restore Flash configuration/command registers
    restore_interrupts();
    return();                           // return to regular EC code
} else {
    sleep();                             // Wait for EC_Int
}
```

The next code sequence would be executed by the Host after it receives the message from the EC that it may take control of the Flash:

```
// INDX is the address of INDX in the Host I/O space
// DATA is the address of DATA in the Host I/O space
//
INDX = 9; // index of Command register byte where Reg_Ctl bit is located
DATA = REG_CTL; Take control of the Flash Address and Data registers
```



As an alternative to the Flash-based EC code described above, the transfer function can run partially in Flash and partially in SRAM. In this mode it is not necessary to enable LRESET# or VCC PWRGD interrupts, since SRAM-based EC code can check the state of those signals through polling: Normal termination of Flash programming occurs as before when the Host sets the EC\_Int bit in the Command register.

```
// This code is located in the Flash
// Command Address is 0xFF3908
// Config Address is 0xFF3910
//
*Config_Address |= REG_CTL_EN;    // Permit control of Flash via register interface
Disable_all_interrupts();
Enable_Interrupt(EC_Int);         // Enable transition back to EC
goto SRAM_Flash_Code;
...
SRAM_Flash_Code:                  // start of code in SRAM
*Command_Address |= REG_CTL;
*Config_Address |= HOST_CTL;
Host_Message(FLASH_READY);
while(true) {                     // commence polling on LRESET/VCC PWRGD
    if( asserted(LRESET) || deasserted(VCCPWRGD) ) {
        cancel_flash_transfer(); // Host is not powered: annul transfer
                                // restore Flash configuration/command registers
        restore_interrupts();
        return();               // return to EC code in Flash
    }
}
```

## 13.11.2 READING THE EMBEDDED FLASH

The Embedded Flash memory, in both the main array can most conveniently be read with Burst mode set. An initial Flash address should be configured, and then the rest of the Flash memory can be read with just a sequence of reads to the data register. The following pseudocode illustrates how reading the Flash from the Host might proceed:

```
// Byte_array[] is a data structure to receive the data. It is a sequence of bytes
// Flashbase is the first address in the flash memory to be read
// Limit is the total number of bytes to read
// INDX is the address of INDX in the Host I/O space
// DATA is the address of DATA in the Host I/O space
//
// this code works as long as the read does not wrap around the end of the Flash array
//
INDX = 8; // index of Command register
DATA = READ | BURST;
INDX = 6; // bits 24:16 of Flash Address register
DATA = Flashbase.byte2;
INDX = 5; // bits 15:8 of Flash Address register
DATA = Flashbase.byte1;
INDX = 4; // bits 7:0 of Flash Address register
DATA = Flashbase.byte0; // the Flash address advances in the Address FIFO
                        // Read from the Flash is started
INDX = 0; // bits 7:0 of Flash Data register
for( i = 0; i < Limit; i++)
{
    Byte_array[i] = DATA;
}
INDX = 8; // index of Command register
DATA = STANDBY; // Exit READ mode and flush Address/Data FIFOs
```

Reading from the EC is similar, except that data can be read four bytes at a time:

```
// Word_array[] is a data structure to receive the data.
// It is a sequence of 32-bit words
// Flashbase is the first address in the flash memory to be read
// Limit is the total number of 32-bit words to read
// Flash Address is 0xFF3904
// Data Address is 0xFF3900
// Command Address is 0xFF3908
```

```
//
// this code works as long as the read does not wrap around the end of the Flash array
//
*Command_Address = READ | BURST;
*Flash_Address = Flashbase;          // the Flash address advances in the Address FIFO
for( i = 0; i < Limit; i++)
{
    Word_array[i] = *Flash_Data;
}
*Command_Address = STANDBY; // Exit READ mode and flush Address/Data FIFOs
```

## 13.11.3 WRITING THE EMBEDDED FLASH

The Embedded Flash memory can most conveniently be programmed with Burst mode set. An initial Flash address should be configured, and then the rest of the Flash memory can be written with just a sequence of writes to the data register. The following pseudocode illustrates how reading the Flash from the Host might proceed:

```
// Byte_array[] is a data structure that sources the data. It is a sequence of bytes
// Flashbase is the first address in the flash memory to be programmed
// Limit is the total number of bytes to write. It is assumed to be a multiple of
// the 256-byte row size for simplicity
// INDX is the address of INDX in the Host I/O space
// DATA is the address of DATA in the Host I/O space
//
// this code works as long as the write block does not wrap around
// the end of the Flash array.
//
INDX = 8; // index of Command register
DATA = PROGRAM | BURST;
INDX = 6; // bits 24:16 of Flash Address register
DATA = Flashbase.byte2;
INDX = 5; // bits 15:8 of Flash Address register
DATA = Flashbase.byte1;
INDX = 4; // bits 7:0 of Flash Address register
DATA = Flashbase.byte0;          // the Flash address advances in the Address FIFO
for( i = 0; i < Limit; )
{
    INDX = 0xC; // index of Status register, to check space in Data FIFO
    while( DATA & DATA_FULL ); // stall while Data FIFO has no space
    INDX = 0; // bits 7:0 of Flash Data register
    // Write one word (four bytes) into the Flash array
    DATA = Byte_array[i++];
    DATA = Byte_array[i++];
    DATA = Byte_array[i++];
    DATA = Byte_array[i++];
}
INDX = 8; // index of Command register
DATA = STANDBY; // Exit PROGRAM mode and flush Address/Data FIFOs
```

Writing from the EC is similar, except that data can be written four bytes at a time

```
// Word_array[] is a data structure to receive the data.
// It is a sequence of 32-bit words
// Flashbase is the first address in the flash memory to be written
// Limit is the total number of 32-bit words to write
// Flash_Address is 0xFF3904
// Data_Address is 0xFF3900
// Command_Address is 0xFF3908
// Status_Address is 0xFF390C
//
// this code works as long as the program block does not wrap around
// the end of the Flash array
//
*Command_Address = PROGRAM | BURST;
*Flash_Address = Flashbase;          // the Flash address advances in the Address FIFO
for( i = 0; i < Limit; i++)
{
```

```
while( *Status_Address & DATA_FULL ); // stall while Data FIFO has no space
*Flash_Data = Word_array[i];
}
*Command_Address = STANDBY; // Exit PROGRAM mode and flush Address/Data FIFOs
```

## 13.12 Detailed Description of Accessible Registers

See [Section 13.8, "Register Interface," on page 262](#)

### 13.12.1 EMBEDDED FLASH DATA REGISTER

The [Embedded Flash Data Register](#) can only be written when the Embedded Flash Controller is in [Program Mode](#). In [Standby Mode](#), [Read Mode](#) and [Program Mode](#) the register is read-only. Writes will complete but have no effect.

# MEC1618/MEC1618i

**TABLE 13-10: EMBEDDED FLASH DATA REGISTER**

HOST INDEX	BYTE3: INDX 03h BYTE2: INDX 02h BYTE1: INDX 01h BYTE0: INDX 00h				8-bit			HOST SIZE	
EC OFFSET	100h				32/16/8-bit			EC SIZE	
POWER	VTR				0000_0000h			nSYS_RST DEFAULT	
BUS	LPC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Flash_Data[31:24]								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Flash_Data[23:16]								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Flash_Data[15:8]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Flash_Data[7:0]								

## FLASH\_DATA[31:0]

This 32-bit register holds the data to be written into the Flash memory array during a program cycle, as well as the data returned from a Flash memory read during a Read From SPB cycle. It should be set up before the [Embedded Flash Address Register](#) is configured.

## 13.12.2 EMBEDDED FLASH ADDRESS REGISTER

**TABLE 13-11: EMBEDDED FLASH ADDRESS REGISTER**

HOST INDEX	BYTE3: INDX 07h BYTE2: INDX 06h BYTE1: INDX 05h BYTE0: INDX 04h						8-bit	HOST SIZE
EC OFFSET	104h				32/16/8-bit			EC SIZE
POWER	VTR				0000_0000h			nSYS_RST DEFAULT
BUS	LPC SPB							
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Flash_Address[23:16]							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Flash_Address[15:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R	R
BIT NAME	Flash_Address[7:2]						Reserved	

### FLASH\_ADDRESS[23:0]

This register represents a byte address for the Embedded Flash array. Since all read and write operations are to 32-bit quantities, the low-order to bits (Flash\_Address[1:0] must always be 0, so these bits are reserved. If the Flash state machine is in [Read Mode](#), writing Byte0 of the [Embedded Flash Address Register](#) initiates the Read sequence. If the Flash state machine is in [Program Mode](#), the Program sequence is initiated when there is both a valid Data value in the [Embedded Flash Data Register](#) and the [Embedded Flash Address Register](#) has been updated by writing Byte0. Bit18 is not used in the MEC1618/MEC1618i.

# MEC1618/MEC1618i

---

Bits[23:19] are only examined in [Erase Mode](#). See [Section 13.9.4, "Erase Mode," on page 269](#) for required decoding during [Erase Mode](#).

The following provides the required decoding during [Read Mode](#) and [Program Mode](#):

The [EEPROM\\_ACCESS Embedded Flash Configuration Register](#) and the determines which array is accessed: Main Memory array, or [EEPROM Emulation](#) array.

[Table 13-12, "Read and Program Mode Decoding"](#) provides the required decoding during [Read Mode](#) and [Program Mode](#).

**TABLE 13-12: READ AND PROGRAM MODE DECODING**

<a href="#">EEPROM_ACCESS</a> Embedded Flash Configuration Register	Area Erased
0	Embedded Flash Main Array specified by <a href="#">Embedded Flash Address Register</a> bits[17:2].
1	<a href="#">EEPROM Emulation</a> MEMORY BLOCK specified by <a href="#">Embedded Flash Address Register</a> bits[9:2].

## 13.12.3 EMBEDDED FLASH COMMAND REGISTER

This register is Read-Only while **Busy** in the **Embedded Flash Status Register** is '1b'. An attempt to write this register while **Busy** is asserted will not modify the register and will set **Busy\_Err** in the **Embedded Flash Status Register**.

A write to this register causes the Embedded Flash controller to transition to the selected state.

**TABLE 13-13: EMBEDDED FLASH COMMAND REGISTER**

HOST INDEX	BYTE0: INDX 8h BYTE1: INDX 9h				8-bit		HOST SIZE	
EC OFFSET	108h				32/16/8-bit		EC SIZE	
POWER	VTR				00h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	R	R	R	R	R	R	R	R/W
EC TYPE	R	R	R	R	R	R	R	R/W
BIT NAME	Reserved							Reg_Ctl
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R/W	R/W	R/W	R/W	R/W
EC TYPE	R	R	R	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved				EC_Int	Burst	Flash_Mode[1:0]	

### FLASH\_MODE

The field determines which Master may write the registers in the Embedded Flash controller:

- 0 The Embedded Flash controller is placed in the Standby mode.
- 1 The Embedded Flash controller is placed in Read mode.
- 2 The Embedded Flash controller is placed in Program mode.
- 3 The Embedded Flash controller is placed in Erase mode.

### BURST

If the Embedded Flash controller is in **Read Mode** or **Program Mode** and this bit is '1b', the contents of the Head register in the **Embedded Flash Address Register** FIFO will be incremented by 4 and written into the Tail register whenever the Head register is Valid. When this bit is '0b', the **Embedded Flash Address Register** and the **INDEX** register are not incremented automatically.

# MEC1618/MEC1618i

---

See [Section 13.9.2.2, "Burst Read Mode," on page 267](#) and [Section 13.9.3.2, "Burst Program Mode," on page 269](#) for information about the use of **Burst**.

## EC\_INT

Setting this bit to '1b' generates an EC interrupt and simultaneously sets [Reg\\_Ctl\\_En](#) and [Host\\_Ctl](#) in the [Embedded Flash Configuration Register](#) as well as [Reg\\_Ctl\\_En](#) in this register to '0b'. By setting these bits to '0b' the Embedded Flash array is configured for EC instruction access, so that when the EC awakes and responds to the interrupt request triggered by [EC\\_Int](#), it can fetch the interrupt vector from its usual location in the Embedded Flash array. See [Section 13.5, "Interrupts," on page 260](#).

The response to setting this bit is the same as the response to VCC PWRGD or PCIRESET signals, as detailed in [Table 13-3, "VCC PWRGD and LRESET# Behavior"](#). If the Embedded Flash Controller is in either [Erase Mode](#) or [Program Mode](#) the programming sequence is terminated, the Embedded Flash controller is set to [Standby Mode](#), [Reg\\_Ctl](#), [Reg\\_Ctl\\_En](#) and [Host\\_Ctl](#) are set to '0b' and the wakeup interrupt is sent to the EC.

## REG\_CTL

When this bit is set, the address input and the control strobes of the Flash Memory Array are sourced from the registers in the [Embedded Flash Subsystem](#). Software on either the Host or the EC can read and write the Flash Memory Array by reading and writing the registers in the subsystem. The EC cannot execute instructions out of the Flash Memory Array.

When this bit is cleared (which is the default), the address input and the control strobes of the Flash Memory Array are sourced from the Instruction Closely Coupled Memory interface. The EC can execute instructions directly from the Flash Memory Array. The other registers in the [Embedded Flash Subsystem](#) can be read or written, but do not affect the Flash Memory Array.

<b>Note:</b> A 32-bit write from either the EC or JTAG with <a href="#">Reg_Ctl_En</a> set to '1b' can simultaneously write the Byte 0 control bits in the <a href="#">Embedded Flash Command Register</a> .
--

This bit can only be set if [Reg\\_Ctl\\_En](#) in the [Embedded Flash Configuration Register](#) is '1b'.



## 13.12.4 EMBEDDED FLASH STATUS REGISTER

**TABLE 13-14: EMBEDDED FLASH STATUS REGISTER**

HOST INDEX	BYTE 1: INDX Dh BYTE 0: INDX Ch				8-bit		HOST SIZE	
EC OFFSET	10Ch				32/16/8-bit		EC SIZE	
POWER	VTR				0000_0000h		nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	R	R	R	R	R	R/WC	R/WC	R/WC
EC TYPE	R	R	R	R	R	R/WC	R/WC	R/WC
BIT NAME	Reserved					Pro- tect_Err	CMD_Err	Busy- _Err
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	EEPROM_ BLOCK	Data_ Block	Boot_ Block	Reserve d	Boot_ Lock	Address_ Full	Data_ Full	Busy

### BUSY

This bit reflects the state of the Embedded Flash controller. This bit is set while the controller is processing a flash control sequence. While this bit is set the [Embedded Flash Command Register](#) can not be written. If a write is attempted on this register, the write fails and [Busy\\_Err](#) in this register is set.

This bit is read-only.

### DATA\_FULL

This bit reflects the value of the Valid bit of the Tail register of the [Embedded Flash Data Register](#). When it is set the FIFO is full and any additional writes will set [Busy\\_Err](#).

This bit is read-only.

### ADDRESS\_FULL

This bit reflects the value of the Valid bit of the Tail register of the [Embedded Flash Address Register](#). When it is set the FIFO is full and any additional writes will set [Busy\\_Err](#).

This bit is read-only.

## BOOT\_LOCK

This bit is set whenever the Boot Block is write-protected. The Boot Block will be write-protected when [Boot\\_Lock](#) in the [Embedded Flash Configuration Register](#) is set. This bit is also set when [Boot\\_Block](#) is set, so that whenever the Boot Block is read-protected it is also write-protected.

A copy is maintained in this register so that the Host can access it, since the Host has no access to the [Embedded Flash Configuration Register](#). See [Section 13.10.1, "Flash Write Protect," on page 270](#) for a description of write-protecting the boot block.

This bit is read-only.

## BOOT\_BLOCK

When this bit is '1b', the Boot Block is protected from all access. See [Section 13.10.2, "Flash Boot Protect," on page 270](#). After being set, it will only be cleared by a VTR Power On Reset.

This bit is read-only.

## DATA\_BLOCK

When this bit is '1b', the Protected Data Block is protected from all access. See [Section 13.10.3, "Flash Data Protect," on page 270](#). After being set, it will only be cleared by a VTR Power On Reset.

This bit is read-only.

## EEPROM\_BLOCK

The [EEPROM\\_BLOCK](#) bit is similar to the [Data\\_Block](#). When [EEPROM\\_BLOCK](#) is '1,' the [EEPROM Emulation](#) is protected from all access. After being set, [EEPROM\\_BLOCK](#) will only be cleared by a VTR POR.

The [EEPROM\\_BLOCK](#) bit is read only.

## BUSY\_ERR

This bit is set if

- A write to the [Embedded Flash Command Register](#) occurs while [Busy](#) is set.
- A write to the [Embedded Flash Address Register](#) occurs while [Address\\_Full](#) is set.
- A write to the [Embedded Flash Data Register](#) occurs while [Data\\_Full](#) is set

This bit is sticky: after being set, it remains set until cleared by a write with the value '1b' to this bit.

## CMD\_ERR

If the Embedded Flash controller is in [Read Mode](#), [Program Mode](#) or [Erase Mode](#), this bit is set if the [Embedded Flash Command Register](#) is set to any value other than [Standby Mode](#).

This bit is sticky: after being set, it remains set until cleared by a write with the value '1b' to this bit.

## PROTECT\_ERR

If [Boot\\_Lock](#) or [Boot\\_Block](#) in the [Embedded Flash Status Register](#) is set and a program operation to an address within the range 0000h - 0FFFh, or a page erase operation to page 0, or a mass erase operation to the main Embedded Flash array occurs, this bit is set. In addition, if [Boot\\_Block](#) is set and any read access to the Embedded Flash in the range 0000h through 0FFFh is attempted, either through the [Instruction Memory Interface](#) or the [Register Interface](#), this bit will be set.

This bit is sticky: after being set, it remains set until cleared by a write with the value '1b' to this bit.

## 13.12.5 EMBEDDED FLASH CONFIGURATION REGISTER

**TABLE 13-15: EMBEDDED FLASH CONFIGURATION REGISTER**

HOST INDEX	N/A			N/A			HOST SIZE	
EC OFFSET	110h			32/16/8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	W	R/W	R/W
BIT NAME	Reserved					EEPROM_FOR CE_BLOCK	EEPROM_PROTECT	EEPROM_ACCESS
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved		INHIBIT_JTAG	Data_Protect	Boot_Protect_En	Boot_Lock	Host_Ctl	Reg_Ctl_En

**Note:** This register is accessible only by the EC.

### REG\_CTL\_EN

When this bit is set, [Reg\\_Ctl](#) of the [Embedded Flash Command Register](#) can be set to 1. When this bit is clear (the default), [Reg\\_Ctl](#) is forced to 0 and cannot be set.

Because this bit overrides any writes to [Reg\\_Ctl](#), EC firmware can clear this bit to prevent the Host from getting access to the Embedded Flash register interface.

### HOST\_CTL

When this bit is set, the [Embedded Flash Address Register](#), the [Embedded Flash Data Register](#) and the [Embedded Flash Command Register](#) can be read and written by the Host via the Flash Memory Mailbox registers in the [Flash Mailbox Interface](#). The EC is inhibited from reading or writing these registers. Writes have no effect and all reads return all 0's.

# MEC1618/MEC1618i

---

When this bit is clear (the default), these registers can be read and written by the EC. The Host is inhibited from reading or writing these registers. Writes have no effect and all reads return all 0's.

The [Embedded Flash Status Register](#) is always readable by both the Host and the EC, independent of the state of Host\_Ctl.

## BOOT\_LOCK

The Boot\_Lock bit permits the EC to lock the Flash boot block. When Boot\_Lock is '1b', the Flash boot block is locked. Any attempt to write data in the address range 00000h through 00FFFh in the Embedded Flash address space will fail and set [Busy](#) in the [Embedded Flash Status Register](#). When Boot\_Lock is '0b' (the default), the Flash boot block is unlocked.

## BOOT\_PROTECT\_EN

When this bit is set, the Boot Protect function is enabled. The first time the EC does either a program fetch to an address that is larger than 00\_0FFFh or a data reference to an address that is larger than 00\_0FFFh and less than 80\_0000h causes [Boot\\_Block](#) in the [Embedded Flash Status Register](#) to be set and all further access to the Boot Block will be prohibited. See [Section 13.10.2, "Flash Boot Protect," on page 270](#).

## DATA\_PROTECT

If this bit is '1b', the top two 2KB pages in the Embedded Flash array are not readable or writable through the Host access mailbox. The two pages are accessible by the EC through both the [Instruction Memory Interface](#) and the [Register Interface](#), as long as the JTAG port is not enabled. After JTAG is enabled on the pins, [Data\\_Block](#) in the [Embedded Flash Status Register](#) is set and the two pages become inaccessible to all interfaces.

If this bit is '0b' (the default), the top two pages can be read and written normally.

## INHIBIT\_JTAG

When this bit is '1b', the JTAG interface is blocked from any access to the EC or the internal buses in the MEC1618/MEC1618i. Only the registers in the JTAG interface are accessible. If this bit is '0b', accesses by the JTAG interface is blocked if is Boot\_JTAG\_Block in the [Embedded Flash Initialization Register](#) is '0' and Boot\_Block in the [Embedded Flash Status Register](#) is '0'.

## EEPROM\_ACCESS

When the [EEPROM\\_ACCESS](#) bit is '1,' Flash accesses through the [Embedded Flash Subsystem](#) register interface access the [EEPROM Emulation](#) Flash array instead of the main flash array. When the [EEPROM\\_ACCESS](#) bit is '0' (default), Flash accesses through the register interface access the main flash body. When the [EEPROM\\_ACCESS](#) bit is set, the [Reg\\_Ctl](#) bit in the Flash Command register is forced to '1,' enabling register access to the Flash IP block. In addition, the [Host\\_Ctl](#) bit is forced to '0,' preventing any Host access while in [EEPROM Emulation](#) mode. Although the EC could set [Host\\_Ctl](#) to '0' in the same write used to set the [EEPROM\\_ACCESS](#) bit, the hardware interlock with [Host\\_Ctl](#) is a fail-safe mechanism to insure that the Host cannot access the [EEPROM Emulation](#).

## EEPROM\_PROTECT

The [EEPROM\\_PROTECT](#) bit functions like [Data\\_Protect](#). If [EEPROM\\_PROTECT](#) is '1,' the [EEPROM Emulation](#) is accessible through the register interface as long as JTAG is not enabled. After JTAG is enabled on the pins, the [EEPROM\\_BLOCK](#) bit in the [Embedded Flash Status Register](#) is set to '1' and the entire [EEPROM Emulation](#) becomes both read and write protected.

## EEPROM\_FORCE\_BLOCK

Writing the [EEPROM\\_FORCE\\_BLOCK](#) bit to '1' sets the [EEPROM\\_BLOCK](#) in the [Embedded Flash Status Register](#) to '1' locking down the [EEPROM Emulation](#). The [EEPROM\\_FORCE\\_BLOCK](#) is write-only.

## 13.12.6 EMBEDDED FLASH INITIALIZATION REGISTER

The Embedded Flash Initialization Register is a read-only register that is loaded from address 0\_0FFCh in the Embedded Flash Array when VTR Power On Reset is de-asserted but while the EC is still held in Reset. The address is the last 32-bit word in the Boot Block.

This register is reset by hardware to be all 0's. The [Boot\\_JTAG\\_Block](#) bit will be 0, which blocks JTAG access to the ARC address space. If the read of the Flash is successful, the value of this register will be FFFF\_FFFFh when the Flash is fully erased. In this state JTAG is not blocked.

**TABLE 13-16: EMBEDDED FLASH INITIALIZATION REGISTER**

HOST INDEX	N/A			N/A			HOST SIZE	
EC OFFSET	114h			32/16/8-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Initial_Data[31:24]							
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Initial_Data[23:16]							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Initial_Data[15:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Initial_Data[7:1]							Boot_JTAG_Block

**Note:** This register is accessible only by the EC.

# MEC1618/MEC1618i

## BOOT\_JTAG\_BLOCK

If this bit is '0b' the JTAG interface is blocked from any access to the MEC1618/MEC1618i as long as [Boot\\_Block](#) in the [Embedded Flash Status Register](#) is '0b'. This means that as long as the Boot Block is accessible by the EC an external JTAG function cannot read or write any address inside the device, including the Boot Block. After boot code renders the Boot Block inaccessible by causing [Boot\\_Block](#) to be set, JTAG functionality is enabled. JTAG cannot read or write any data in the Embedded Flash Boot Block, including the data used to load the [Embedded Flash Initialization Register](#).

If this bit is '1b' (the value that will be loaded if the Boot Block is erased), JTAG access will be enabled as soon as VTR POR is de-asserted.

## BITS[31:1] INITIAL\_DATA

The data in this field are loaded after VTR POR is de-asserted, along with [Boot\\_JTAG\\_Block](#). The data can be used for device identification or configuration.

## 13.12.7 FLASH MAILBOX INDEX REGISTER

TABLE 13-17: FL\_MBX\_INDEX REGISTER

HOST OFFSET	00h						8-Bit	HOST SIZE	
EC OFFSET	00h			8-Bit					EC SIZE
POWER	VTR			00h					VCC POR DEFAULT
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	INDEX7	INDEX6	INDEX5	INDEX4	INDEX3	INDEX2	INDEX1	INDEX0	

## 13.12.8 FLASH MAILBOX DATA REGISTER

TABLE 13-18: FL\_MBX\_DATA REGISTER

HOST OFFSET	04h						8-Bit	HOST SIZE	
EC OFFSET	04h			8-bit					EC SIZE
POWER	VTR			00h					VCC POR DEFAULT
BUS	LPC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
EC TYPE	-	-	-	-	-	-	-	-	
BIT NAME	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0	

## 13.12.9 EEPROM UNLOCK REGISTER

**TABLE 13-19: EEPROM UNLOCK REGISTER**

HOST INDEX	N/A			N/A			HOST SIZE	
EC OFFSET	120h			32/16/8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	LPC SPB							
BYTE[3:1] BIT	D31	D30	D29	...		D10	D9	D0
HOST TYPE	–	–	–	–	–	–	–	–
EC TYPE	W	W	W	W	W	W	W	W
BIT NAME	EEPROM_UNLOCK							

### EEPROM\_UNLOCK

When the 31-bit [EEPROM\\_UNLOCK](#) register is written, the least significant 31 bits of the write are compared to the [Internal Register](#). If all bits match, the [EEPROM\\_BLOCK](#) bit in the [Embedded Flash Status Register](#) cleared, and the [EEPROM Emulation](#) array can be read or written.

The [EEPROM\\_UNLOCK](#) register is write only.

# MEC1618/MEC1618i

---

## 14.0 ARC 625D EMBEDDED CONTROLLER

### 14.1 General Description

This chapter contains a description of the Embedded Controller used in the MEC1618/MEC1618i.

The Embedded Controller on the MEC1618/MEC1618i is an ARC 625D Processor, Revision 4.6, by ARC International. The ARC625D is a full-featured 32-bit embedded processor. Its features include:

- 5-stage instruction pipeline with single-cycle instruction execution
- Static branch prediction
- 32-bit data, instruction and address buses
- 16- and 32-bit instructions, with no overhead for switching between 16- and 32-bits
- 32 32-bit general purpose registers
- Scoreboarded data memory pipeline to reduce data stalls
- Debug features
  - Debug host can access all registers and CPU memory, with a JTAG interface to host tools
  - Multiple action points for real-time instruction and data breakpoints
- Industry standard AHB system interface
- Power saving features
  - Sleep mode via software instruction
  - Clock gating
- Two highly configurable action points for debugging
- SYNC instruction (added in ARC 625D Processor, Revision 4.6)
- [Delay Register](#), a Microchip extension register to the ARC

The ARC625D is highly configurable. The configuration used in the MEC1618/MEC1618i incorporates:

- 4K [Boot ROM](#) to download an (in-factory environment) SRAM-based EC application
- 192KB single cycle Embedded Flash Closely Coupled instruction memory
- 16-KB Single Cycle 32-bit wide dual-ported SRAM, accessible as both Closely Coupled Data Memory and Instruction Memory
- Interrupt controller with 32 interrupts
- Normalize instruction, which can find leading ones and zeros in a word
- Multiply instruction, which completes a 32x32 multiply in 3 cycles
- Divide Assist instruction
- Two full-featured [Actionpoints \(Dedicated Breakpoint Blocks\)](#), which can trigger breakpoints on both instruction accesses and data access

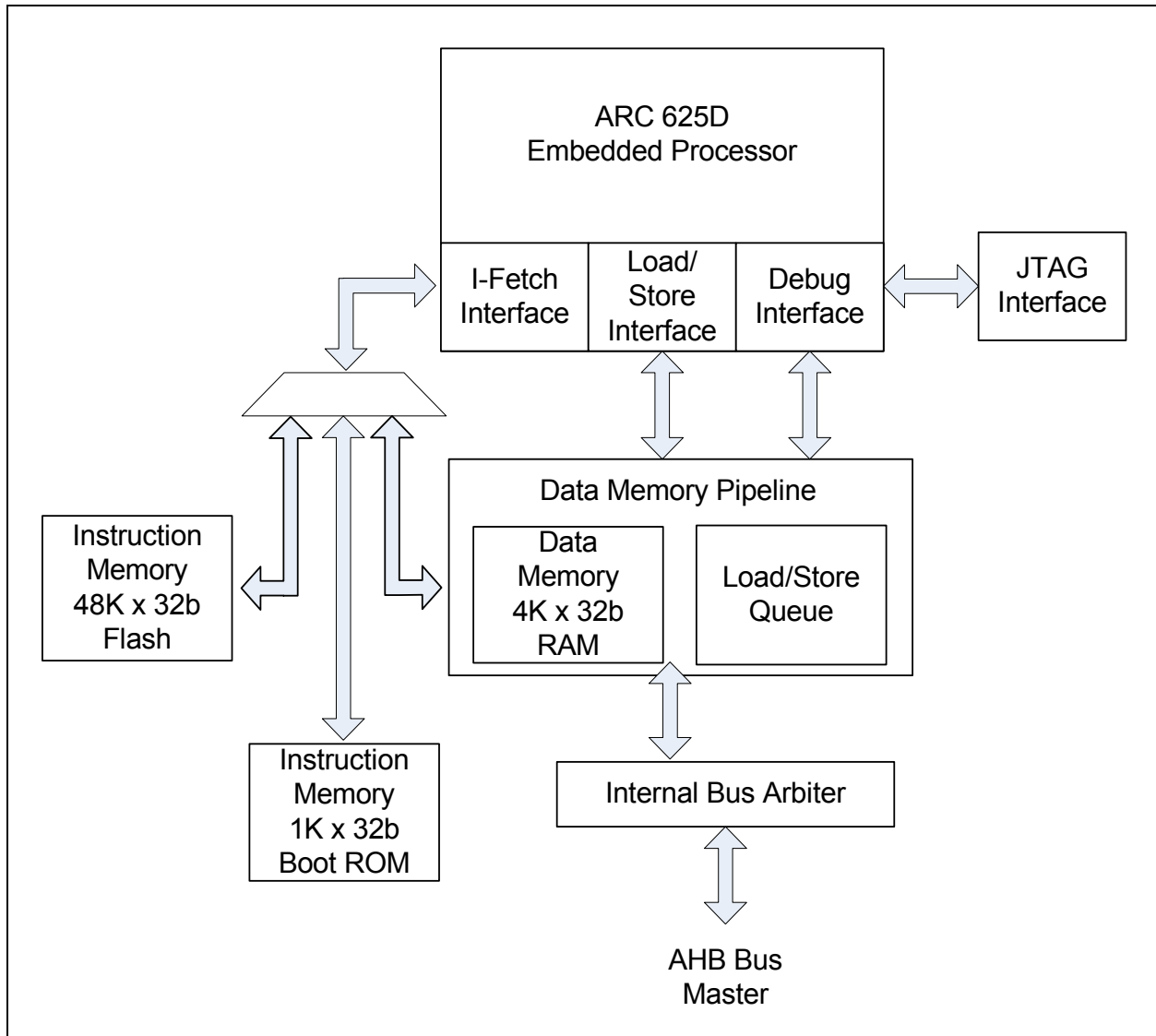
For details on the architecture of the ARC625D processor, see [ARCompact™ Instruction Set Architecture Programmer's Reference](#), ARC International, April 2009.

<b>Note:</b> Metaware version 8.7 or newer must be used.
--



## 14.2 Block Diagram

FIGURE 14-1: ARC BLOCK DIAGRAM



### 14.2.1 POWER ON RESET

This block is reset on a [nEC\\_RST](#). See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

The default location of the interrupt vector table for the ARC processor is now 010\_0000h to access the [Boot ROM](#). The [Boot ROM](#) downloads an SRAM-based EC application using the MEC1618/MEC1618i built-in serial UART. If no communication is established within a configurable timeout period, the [Delay Register](#) restores the UART to its default state and starts loading code from the first physical address of the embedded Flash.

### 14.3 EC Clocking

The [ARC 625D Embedded Controller](#) can be configured to run at various clock rates as described in [Section 5.7.3, "Clock Control Register," on page 115](#) (see [Section 5.0, "Power, Clocks, and Resets," on page 74](#)).

## 14.4 EC Memory Map

The ARC processor executes code out of the [EC Instruction Memory](#). This instruction memory is an ARC ICCM (Instruction Closely-Coupled Memory) and each 32-bit word can be accessed in one processor cycle. Data references can come from either the [EC Data Memory](#) or from addresses located in the [AHB Address Space](#). The [EC Data Memory](#) is an ARC DCCM (Data Closely-Coupled Memory), so each 32-bit word can read or written in one cycle.

See [ARC Address Space on page 49](#) for further details on the ARC address space.

### 14.4.1 EC DATA MEMORY

The EC has a 16KB Closely Coupled Data memory, implemented with static RAM and organized 4K x 32 bits. Loads and stores to this memory are completed in one cycle. The base address of the memory is 80\_0000h in the EC address space and extends to location 80\_3FFFh. The EC cannot execute instructions from this address range.

The 16KB Data memory also appears in the instruction space in the address range 6\_0000h through 6\_3FFF as described in [Section 14.4.2, "EC Instruction Memory"](#).

Any data load/store to non-existent memory will return FFFF\_FFFFh.

### 14.4.2 EC INSTRUCTION MEMORY

The primary instruction memory for the EC is a 48K x 32 bit Embedded Flash memory, located at locations 00\_0000h through 02\_FFFFh in the EC address space. The address range 10\_0000h through 10\_0FFFh is mapped to the [Boot ROM](#). (See [Section 14.2.1, "Power On Reset," on page 289](#))

Instruction fetches to these two blocks complete in one cycle. The ARC can access locations in the Embedded Flash Memory or the [Boot ROM](#) through load and store instructions.

The 16KB Data memory also appears in the instruction space in the address range 6\_0000h through 6\_3FFF. The memory is dual-ported, so instruction fetches from this space can occur in parallel with data loads and stores, without wait states for either instruction fetch or data reference. If the Embedded Flash memory is configured to be only accessible via the register interface (see [Section 14.9, "EC Registers," on page 293](#)), the EC can execute instructions out of the SRAM. For example, the EC could run code that programs the Flash while the Embedded Flash memory is set for the register interface.

[RAM\\_Select](#) in the [AHB SRAM Configuration Register](#) can be used to disable instruction access to the SRAM. If instruction access to the SRAM is not needed, disabling it saves power.

Instruction fetches in the range of 00\_0000h through 7F\_FFFFh do not incur bus errors. Any instruction fetch to non-existent memory will return FFFF\_FFFFh.

## 14.5 ARC Pipelining

The ARC625D processor is pipelined with five pipe stages. Loads and stores are further pipelined through the Load/Store Queue as shown in [Figure 17.1, "ARC Block Diagram"](#), so loads and stores will take additional cycles to complete. The AHB bus is also pipelined. Because of the different pipelines, it is difficult to determine exactly how long a load or store to a register will take if the register is located on either the LPC SPB bus or the EC SPB bus.

Because the ARC processor issues all instructions in order and resolves data hazards within the pipeline, software will typically not have to consider pipeline effects. Stores will complete in the order issued and no load instruction will return data until all stores issued previously have completed. However, there may be some situations in which it is necessary to guarantee that pipelines have flushed and all stores have completed before further code execution. The following three instruction sequence provides this guarantee:

1. `STORE` to a memory location
2. `LOAD` from the same memory location to a processor register
3. Issue any instruction that uses the register in step 2) as one of its sources

The following assembly code is an example of the sequence:

```
; R0 = a value to be written to an AHB memory location
; R1 = the AHB address of the location to be written
;
ST R0, [R1,0]          ;; store
LD R0, [R1,0]          ; load from same location
ADD R0, R0, 0 ; dummy instruction dependent on R0
```

## 14.6 Microchip ARC Extensions

Microchip provides the following extension registers to the [ARC 625D Embedded Controller](#):

### 14.6.1 DELAY REGISTER

The [Delay Register](#) is an ARC Extension register in the Auxiliary registers Address Space. Writing a [DELAY\\_VALUE](#) to the [Delay Register](#) halts the ARC pipeline for 1  $\mu$ s to 32  $\mu$ s, depending on the [DELAY\\_VALUE](#) which is the number of microseconds to delay minus one. Valid [DELAY\\_VALUE](#) arguments are 0 - 31. Only the least significant five bits of the written [Delay Register](#) value are examined; all bits above bit 4 are ignored.

[DELAY\\_VALUE](#) arguments are written to the [Delay Register](#) with the ARC Auxiliary Register store instruction (sr). Loads of the [Delay Register](#), with the ARC Auxiliary Register load instruction (lr) will return the last stored [DELAY\\_VALUE](#). There is no delay associated with [Delay Register](#) loads.

While the instruction is holding up the pipeline, no new instructions can enter (for example, any interrupt processing is held for the duration of the auxiliary store instruction). Writes to the [Delay Register](#) produce no other side effects.

Address	FFFF_FFF9h in the Auxiliary Registers Address Space			
Bits	Description	Type ( <a href="#">Note 14-1</a> )	Default	Reset Event
31:5	RESERVED	RES	0	—
4:0	DELAY_VALUE The delay time in microseconds is the <a href="#">DELAY_VALUE</a> + 1. Valid <a href="#">DELAY_VALUE</a> arguments are 0 - 31.	R/W	0	nEC_RST

**Note 14-1** Reads of the [Delay Register](#) return the last written [DELAY\\_VALUE](#).

### 14.6.2 EXAMPLE

An example subroutine that can delay an arbitrary number of microseconds is shown below in [Figure 14-2](#).

**FIGURE 14-2: DELAY REGISTER CODE EXAMPLE**

```
#define DELAY_REGISTER 0xFFFFFFFF9

void forced_delay( unsigned delay_in_microseconds )
{
    while( delay_in_microseconds > 32 )
    {
        _sr(31, DELAY_REGISTER);
        delay_in_microseconds -= 32;
    }
    _sr(--delay_in_microseconds, DELAY_REGISTER);
}
```

## 14.7 EC AHB Bus Interface

The ARC Embedded Controller has a single AHB Bus Master interface; see [Section 3.3.2, "AHB Address Space," on page 51](#). The ARC can have at most one access pending on the AHB at one time. The ARC can perform 8-bit, 16-bit and 32-bit loads and stores on the AHB. Instruction fetches over the AHB can take the form of a 32-bit word load.

Possible AHB bus errors are described in [Section 3.4.3, "AHB Bus Errors," on page 55](#). The ARC processor responds to a bus error with Memory Error exception. The first address that caused a memory error is recorded in the [AHB Error Address Register](#). Because ARC exceptions are imprecise, and since several bus errors can occur between the time a bus error address is recorded and the time the ARC processes the exception, it is not always possible to determine which instruction caused the bus error.

## 14.8 Actionpoints (Dedicated Breakpoint Blocks)

Actionpoints are defined in the ARC 600 Ancillary Components Manual, Chapter 4. They are dedicated hardware blocks that provide an alternative source of breakpoints when the debugger cannot write to memory (e.g., the code being debugged is in ROM). They also provide the ability to break on memory or Aux register accesses.

The primary justification for including Actionpoints in the design is to provide breakpointing for code in ROM, while code is running at full speed (as opposed to being single-stepped). The debugger by default prefers to write Breakpoint instructions (BRK\_S: 7FFFh) into memory in order to perform breakpoints at specified PC values. It will instead use actionpoints if:

- The memory area is declared as ROM,
  - or -
- The flag "-off=prefer\_soft\_bp" is given to the debugger.

Actionpoints are controlled by a dedicated set of Aux Registers, in the range 220h - 237h, organized as 3 registers per actionpoint. These are:

- AMV: A 32-bit value (Address or sometimes Data). This register supplies the initial trigger value, as masked by the AMM register. Upon triggering, it is over-written by hardware with the exact value seen.
- AMM: A 32-bit mask applied to the AMV register, making any desired bits don't-cares.
- AC: Control register, selecting modes

The status of all actionpoints is visible in the Aux Register DEBUG, at 5h.

The MEC1618/MEC1618i incorporates two full-featured ARC Actionpoints, Actionpoint 0 and Actionpoint 1.

### 14.8.1 ACTIONPOINT CONFIGURATIONS

Actionpoints may be configured (at processor HW build) to be one of two configurations, Minimal or Full.

Minimal actionpoints may be configured to do the following:

- Trigger on an access by address and access type (Instruction, Bus access, or Aux Register)
- Instruction breakpoints trigger on execution at the address, not at the fetch itself
- Act by either Halt (debugger acts) or SW Interrupt (target SW acts)
- Qualify between Reads and Writes (or both)
- Qualify by masking bits of the address
- Invert Condition (Trigger if No Match)
- Gang actionpoints in pairs or quads: both/all must match

Full actionpoints add the following capabilities:

- Match on opcode for instruction fetches
- Match on data value for data read/write in Aux registers or memory
- Two 34-bit inputs from arbitrary sources (per actionpoint)

### 14.8.2 SIGNIFICANT LIMITATIONS

Address ranges may degrade performance. Because address matching is bit-masked, it may take multiple actionpoints to refine an address range. Even then, the final range is liable to be too big. The debugger allows a range to be too big, and continues from the breakpoint if the resulting trigger was not in the desired range. Note that this means that the

program was being halted at undesired / unexpected times, and so is not running at full speed. A reliable way to avoid this is to specify a range only as a power of 2 in size, aligned on a boundary that is also a power of 2, the same or larger than the range.

There is no way to trigger on both the value and the address of a bus read (Memory, I/O), because the data and the address are not present simultaneously. An Aux register access, however, can trigger on data when either read or written. Do not try to enable Read and Write in the same actionpoint, because that will select only the Write data bus to monitor.

## 14.8.3 DEBUGGER SUPPORT

As of version 8.7 of the Metaware toolset, the debugger supports:

1. Break on Instruction fetch by address  
Actionpoint is used if ROM detected or “-off=prefer\_soft\_bp” argument is specified
2. Break on Memory Space data accesses  
Read/Write or Both  
Address and Mask  
Range, if size is power of two and target aligned to a power of two (requires 2 Minimal actionpoints, paired)  
Value and Mask (requires 2 Minimal actionpoints, paired)  
Value and Range (requires 4 Full actionpoints, quadded)
3. Aux Register accesses  
Read/Write or Both

## 14.9 EC Registers

**TABLE 14-1: AHB SRAM CONFIGURATION REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC ADDRESS	F0_FC00h				32-bit			EC SIZE	
POWER	VTR				0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE[3-1] BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R/W	
BIT NAME	Reserved							RAM_Select	

# MEC1618/MEC1618i

## RAM\_SELECT

When this bit is clear (the default case), the 16KB on-chip SRAM that is part of the EC can only be accessed by loads and stores starting at address 80\_0000h. The EC can read and write data in the SRAM at addresses starting at 80\_0000h but cannot directly execute instructions.

When this bit is set, the 16KB SRAM is configured to be simultaneously accessible in the address range address 6\_0000h through 6\_3FFFh. The EC can execute directly out of the SRAM. The EC can still read and write data in the SRAM, with no time penalty per load or store.

**TABLE 14-2: AHB ERROR ADDRESS REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	F0_FC04h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	EC_Addr[23:16]							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	EC_Addr[15:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	EC_Addr[7:0]							

## EC\_ADDR[23:0]

If an AHB bus error occurs as the result of an EC AHB bus access, the address that caused the error is held. After an address is held, additional bus errors are ignored, so this register records the first AHB address that caused an AHB bus error. Any write to this register re-enables capturing AHB bus addresses.

## 15.0 EC INTERRUPT AGGREGATOR

### 15.1 General Description

The [EC Interrupt Aggregator](#) works in conjunction with the ARC625D processor's interrupt interface to handle hardware interrupts and exceptions.

Exceptions are synchronous to instructions, are not maskable, and have higher priority than interrupts. All three exceptions - reset, memory error, and instruction error - are hardwired directly to the processor.

Interrupts are typically asynchronous and are maskable. As shown in [Figure 15-1](#), certain interrupts are connected to the processor, but the majority are connected to the [EC Interrupt Aggregator](#). The latter latches, arbitrates, and forwards the highest-level active interrupt to the processor's IRQ3 interrupt input. It also generates a jump vector associated with the selected interrupt. This vector is made available in one of the processor core's registers and is used to address a location in the Interrupt Vector Table (in memory) that contains the address of the interrupt handler.

The aggregator provides four priority levels for incoming interrupts. The processor provides three: mid- and low priorities for interrupts and high priority for exceptions.

Interrupts classified as wake events can be recognized without a running clock, e.g., while the MEC1618/MEC1618i is in sleep state.

The [EC Interrupt Aggregator](#) can also operate in legacy mode to maintain compatibility with previous generation. In this mode it forwards up to 16 output interrupts to the processor's IRQ[8:23] but does not generate jump vectors.

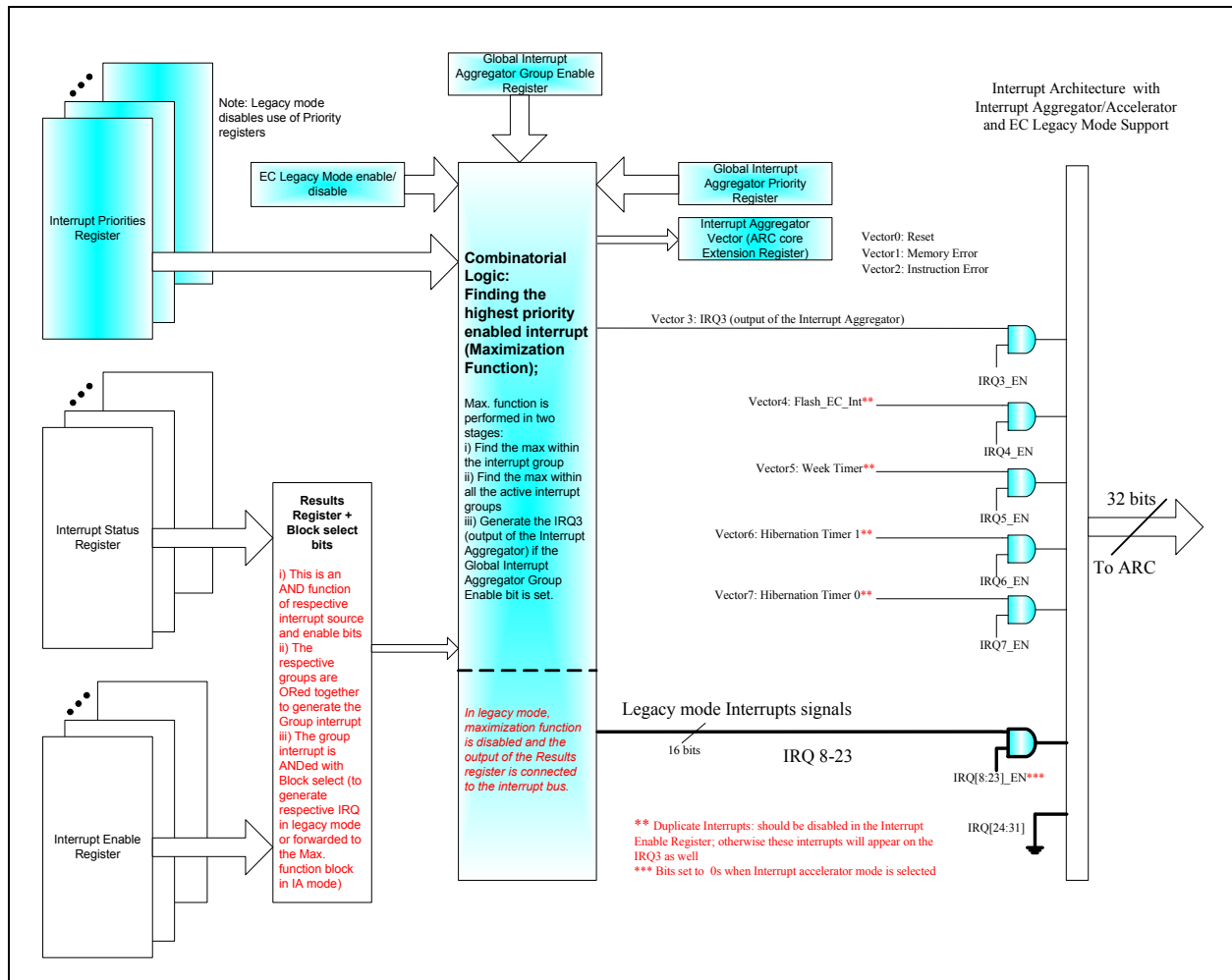
This chapter focuses on the [EC Interrupt Aggregator](#). Please refer to ARC International's *ARCompact™ Instruction Set Architecture Programmer's Reference*, March 2005 for more information on interrupt and exception handling by the ARC625D core.

Features of the [EC Interrupt Aggregator](#):

- Edge-triggered inputs
- 4 priority levels
- 16 x 31 input interrupts (31 interrupts per group; 16 groups)
- Wake interrupts recognized while clock is stopped
- Programmable base address of the Vector Table
- Assist fast interrupt handling by software
  - Provides interrupt's jump vector in processor's extension register for fast access to Interrupt Vector Table
  - Support for NORM instruction to quickly locate the active highest-level interrupt



**FIGURE 15-1: ARCHITECTURAL VIEW OF INTERRUPT AGGREGATOR**



## 15.2 Interrupt Summary

Table 15-1, "EC Interrupt Structure" summarizes the ARC interrupts, priorities and vector locations.

- Link registers (ILINK1 / ILINK2) are the processor's registers that hold the value of the next PC when an interrupt occurs.
- Inside the processor, exceptions have HIGH priority and interrupts have MID or LOW priority. Within a priority level, a higher numbered interrupt has higher priority. For example, the Flash\_EC\_Int Interrupt with relative priority L26 has higher priority than the Week Timer's Interrupt with a relative priority L25. An exception is interrupt #7, which always has highest priority within its level.
- Byte offset: The ARC processor implements a table of jumps rather than interrupt vectors. When an interrupt occurs, the processor jumps to fixed addresses in memory, which contain a jump instruction to the interrupt handler. Byte offsets are vector offsets to the jump table.

Details on processor handling of interrupts can be found in the [ARCompact™ Instruction Set Architecture Programmer's Reference](#), ARC International, April 2009.

# MEC1618/MEC1618i

**TABLE 15-1: EC INTERRUPT STRUCTURE**

Vector	Name	Link Register	Priority (Default)	Relative Priority	Byte Offset
0	Reset	-	High	H1	0x00
1	Memory Error	ILINK2	High	H2	0x08
2	Instruction Error	ILINK2	High	H3	0x10
3	Interrupt Aggregator	ILINK1	level 1 (low)	L27	0x18
4	Flash_EC_Int	ILINK1	level 1 (low)	L26	0x20
5	Week Timer	ILINK1	level 1 (low)	L25	0x28
6	Hibernation Timer 1	ILINK2	level 2 (mid)	M2	0x30
7	Hibernation Timer 0	ILINK2	level 2 (mid)	M1	0x38
8	IRQ8	ILINK1	level 1 (low)	L24	0x40
9	IRQ9	ILINK1	level 1 (low)	L23	0x48
10	IRQ10	ILINK1	level 1 (low)	L22	0x50
11	IRQ11	ILINK1	level 1 (low)	L21	0x58
12	IRQ12	ILINK1	level 1 (low)	L20	0x60
13	IRQ13	ILINK1	level 1 (low)	L19	0x68
14	IRQ14	ILINK1	level 1 (low)	L18	0x70
15	IRQ15	ILINK1	level 1 (low)	L17	0x78
16	IRQ16	ILINK1	level 1 (low)	L16	0x80
17	IRQ17	ILINK1	level 1 (low)	L15	0x88
18	IRQ18	ILINK1	level 1 (low)	L14	0x90
19	IRQ19	ILINK1	level 1 (low)	L13	0x98
20	IRQ20	ILINK1	level 1 (low)	L12	0xA0
21	IRQ21	ILINK1	level 1 (low)	L11	0xA8
22	IRQ22	ILINK1	level 1 (low)	L10	0xB0
23	IRQ23	ILINK1	level 1 (low)	L9	0xB8
24	IRQ24 (tied low)	ILINK1	level 1 (low)	L8	0xC0
25	IRQ25 (tied low)	ILINK1	level 1 (low)	L7	0xC8
26	IRQ26 (tied low)	ILINK1	level 1 (low)	L6	0xD0
27	IRQ27 (tied low)	ILINK1	level 1 (low)	L5	0xD8
28	IRQ28 (tied low)	ILINK1	level 1 (low)	L4	0xE0
29	IRQ29 (tied low)	ILINK1	level 1 (low)	L3	0xE8

TABLE 15-1: EC INTERRUPT STRUCTURE (CONTINUED)

Vector	Name	Link Register	Priority (Default)	Relative Priority	Byte Offset
30	IRQ30 (tied low)	ILINK1	level 1 (low)	L2	0xF0
31	IRQ31 (tied low)	ILINK1	level 1 (low)	L1	0xF8

## 15.3 Operation

### 15.3.1 REGISTER CONTROL OF INPUT INTERRUPTS

Associated with each interrupt are

- a Source bit that is set to indicate when an interrupt is active
- an Interrupt Enable bit to allow interrupt generation
- a Result bit to indicate when an enabled interrupt is active
- a priority level determined by its 2 Priority Level bits

Input interrupts are organized into groups; each group comprises 31 interrupts. Associated with each group is a set of 32-bit registers (Source, Enable, Result, Priority) described above. In addition, incoming interrupts can also be controlled on group basis, i.e., all interrupts in a group can be enabled / disabled by a bit in the Group Select register. This is summarized in [Table 15-2](#).

TABLE 15-2: INTERRUPT AGGREGATOR GROUP REGISTERS

Register Name	Width (Bits)	Purpose
Interrupt Source	31	Latches asynchronous signals from on-chip devices
Interrupt Enable	31	Enables each interrupt
Interrupt Priority	62	A 2 bit priority for each of the 31 possible interrupt sources
Interrupt Result	31	Each bit is 1 if the corresponding Interrupt Source bit is 1, the interrupt is enabled, and the priority of the interrupt is equal to or greater than the current priority. The content of this register changes continuously, i.e., combinatorially, based on outputs from the other three registers.

The 31<sup>st</sup> bit (i.e. the most significant bit) of the Result register is to control the use of the NORM instruction. should set to 0. Setting this bit to '0' enables the use of the ARC NORM (normalize) instruction as a Find-First-One instruction (that is, NORM will return the bit number of the highest numbered bit that is a 1).

# MEC1618/MEC1618i

## 15.3.2 REGISTER CONTROL OF OUTPUT INTERRUPTS AND GLOBAL REGISTERS

Output interrupts to the processor, IRQ[23:03], are individually enabled. The lowest three LSBs, [2:0], are not used due to the three exceptions (reset, memory error, instruction error) being directly connected to the processor.

Two global registers, IA Priority and IA Vector, are implemented as core extension registers. This means they can be used in any ARC instruction that can reference the full 6-bit core register number.

**TABLE 15-3: INTERRUPT AGGREGATOR GLOBAL REGISTERS**

Register Name	Width (Bits)	Purpose
IA Priority	3	The current priority level for the interrupt. A setting of 4 or higher blocks all interrupts from the Interrupt Aggregator. Interrupts at this priority level or higher will be allowed to be propagated to IRQ3. This is implemented as Core Extension register R55.
IA Group Enable	16	Enables individual Interrupt Group within IA
IA Vector	24	The address of a vector in memory. The vector is the address of an interrupt handler, for the interrupt selected by the Interrupt Aggregator. This is an ARC Core Extension Register R56.
IA IRQ Enable	20	Enable individual IRQ lines going into ARC

## 15.3.3 GENERATION OF INTERRUPT OUTPUTS AND JUMP VECTOR

1. The Interrupt Status register and the Interrupt Enable register respective bits are ANDed together.
2. All the bits in the same group are ORed
3. The resultant ORed bit is ANDed with the respective Block Select Register bit to generate the IRQ<sub>i</sub> in **legacy mode**.
4. In the **accelerated mode**, the results of step 3 along with the priorities set in the GIRQ<sub>x</sub> Interrupt Priority Registers are fed into the Maximization Function block to generate the highest priority interrupt that is propagated to IRQ3. The address of the respective interrupt handler is subsequently loaded in the IA Vector register.
5. Prior to sending the interrupts to the ARC, IRQ<sub>i</sub>\_EN is ANDed with the respective IRQ<sub>i</sub>.

### 15.3.3.1 Priority Levels

The Interrupt Aggregator adds 4 levels of interrupt priority to the 2 levels the ARC provides. Each of the potential chip interrupts (16 interrupt source registers times up to 31 sources per register) can independently be assigned one of 4 priority levels. The Interrupt Priority Register in each group has a 2-bit priority field for each of the 31 possible interrupt sources. The 3-bit IA Priority Register sets a current priority level for all groups.

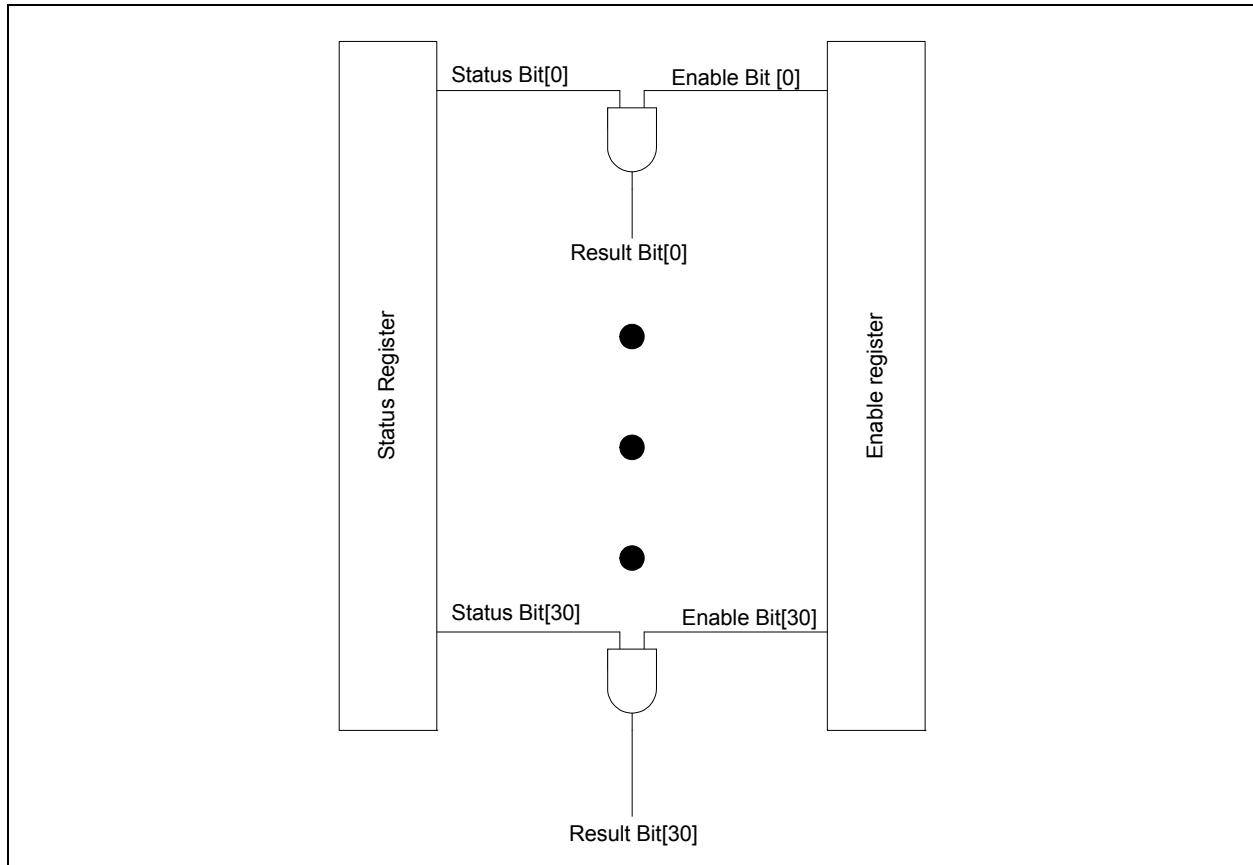
An individual interrupt *i* is enabled if bit *i* in the Interrupt Source register is 1 (asserted), bit *i* in the Interrupt Enable register is 1 (enabled), and the 2 bits for interrupt *i* in the Interrupt Priority Register represent a number that is greater than or equal to the IA Priority Register as well as greater or equal to the priority level of any other interrupt that is currently enabled.

The MEC1618/MEC1618i Interrupt Aggregator selects the interrupt with the highest priority among all active interrupts. Interrupts at priorities below the current are blocked. Within a group the interrupt with the higher bit number has priority over interrupts with lower bit number (assuming that these bits are at the same priority level).

### 15.3.3.2 Interrupt 'Result'

As shown in [Figure 15-1](#) the Source and Enable bits are latched, but the Result bits are not. The latter change combinatorially with inputs Source and Enable inputs.

**FIGURE 15-2: RESULTS REGISTER EXAMPLE**



### 15.3.3.3 Group Interrupt Request

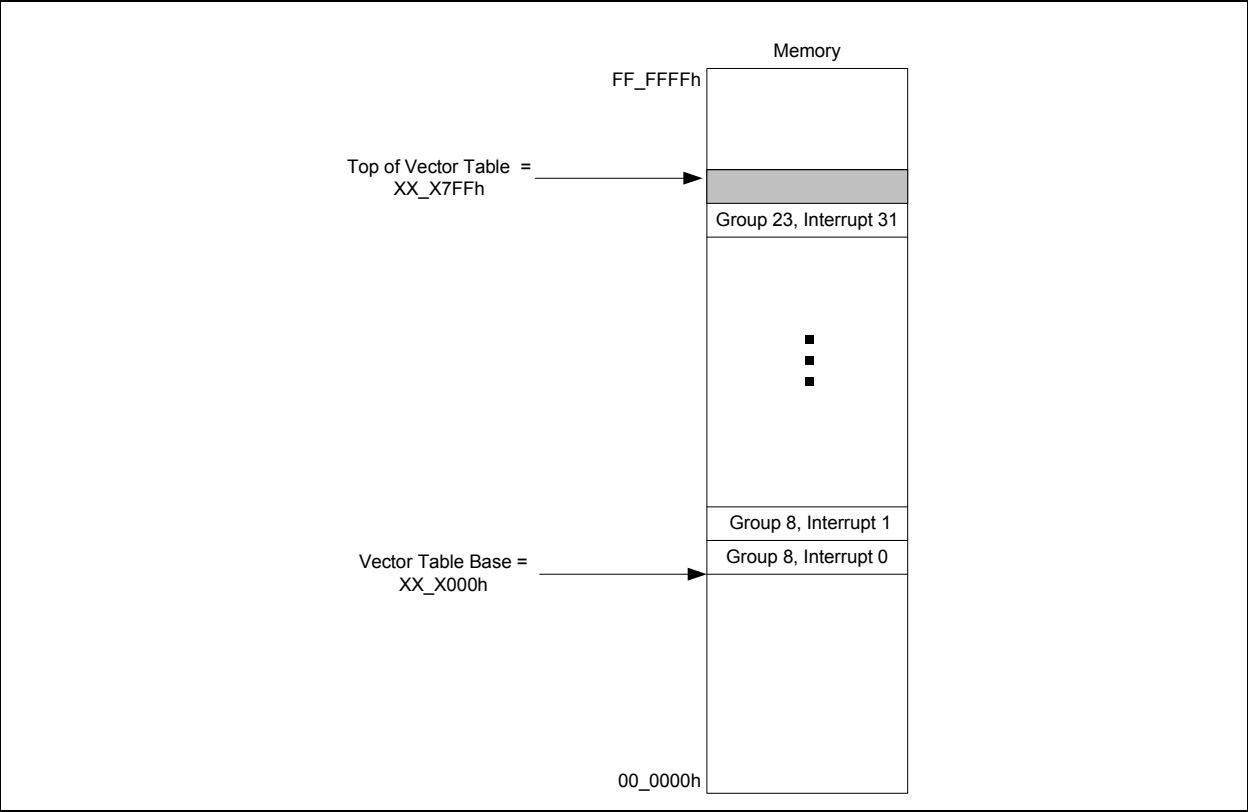
There are 16 Group Interrupt Request signals, one for each bank in the Interrupt Aggregator. An interrupt is propagated within the interrupt chain (that results in the IRQ3) if the priority assigned to the source bit in the Group Priority Register is greater than or equal to the contents of the IA Priority Register (Current Priority bits). For example, assigning a priority of 3 to a source bit means maximum priority for that source, which will always be enabled if the corresponding enable bit is set and the IA Priority Register is 3 or less. Setting the IA Priority Register to 4 or greater disables all interrupts in the Interrupt Aggregator. Software must maintain the IA Priority Register, and stack the value in memory if nested interrupts are required.

### 15.3.3.4 Interrupt Vector Generation

The Interrupt Aggregator continually selects from among all of its active inputs to generate one IRQ that is connected to the ARC interrupt controller on ARC input IRQ3. At the same time, the Aggregator generates an index into an Interrupt Vector table which addresses a pointer to the handler for the interrupt that is to be serviced.

The Interrupt Aggregator Interrupt Vector Table is a table of 4-byte addresses that is 2KB in length. will be used(16 Groups times 31 interrupts per group times 4 bytes per address). The [FIGURE 15-3: Interrupt Aggregator Vector Table on page 302](#) illustrates the Interrupt Vector Table:

FIGURE 15-3: INTERRUPT AGGREGATOR VECTOR TABLE

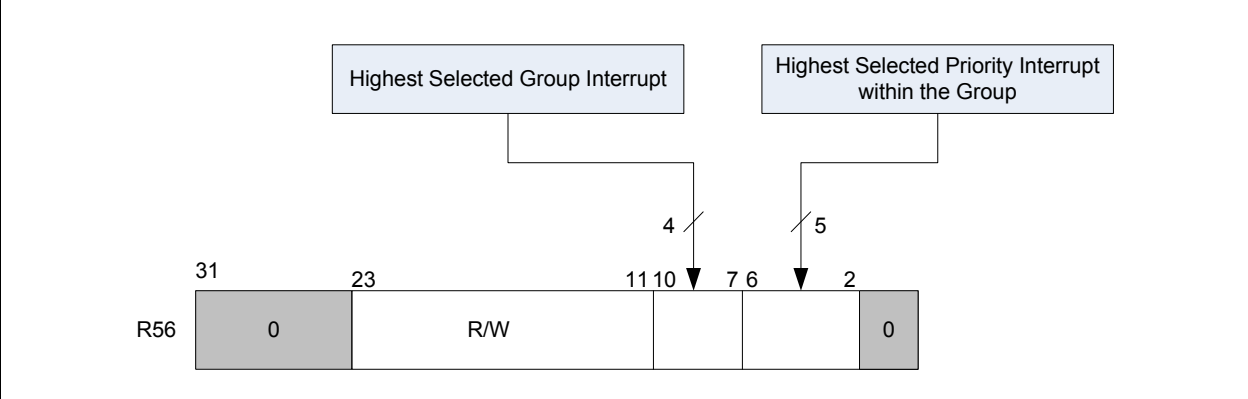


The table must begin on a 2KB address boundary. Since there are only 31 possible interrupt sources per Group Interrupt Request, the 32nd vector entry is always null. The table will typically reside in the program Flash memory.

A new extension register is added to the ARC register set to support interrupt vectoring. Register R56 is a pointer into the Interrupt Vector table and may be used in any ARC instruction that can reference one of the general purpose registers. Bits 23 through 11 of R56 are readable and writable; setting this range establishes the base address of the table. Bits 10 through 0 are read-only, which is the reason the vector table must start on a 2KB address boundary. Since ARC addresses are 16 bits, bits 31 through 24 of R56 are reserved and always read as 0.

The [FIGURE 15-4: Interrupt Vector Generation on page 302](#) illustrates the mechanism by which bits 10 through 2 of R56 are set:

FIGURE 15-4: INTERRUPT VECTOR GENERATION



A Highest Group Interrupt block selects the highest numbered GIRQ currently active and enabled. It should be noted that even though the group numbering ranges from GIRQ8-GIRQ22, there are four bits allocated to represent these numbers. For example, if GIRQ8 is selected then value 0x0 will be assigned to bits [10:7] of the R56. Likewise, if GIRQ22 is selected, value 0xE will be assigned to bits [10:7] of R56. [Table 15-4, "GIRQi Mapping to Bits \[10:7\] of R56"](#) shows the mapping of the GIRQi to the bits [10:7] bits of R56.

**TABLE 15-4: GIRQI MAPPING TO BITS [10:7] OF R56**

GIRQ <sub>i</sub>	Mapped Bit Values in the Group Select Field
GIRQ8	0000
GIRQ9	0001
GIRQ10	0010
GIRQ11	0011
GIRQ12	0100
GIRQ13	0101
GIRQ14	0110
GIRQ15	0111
GIRQ16	1000
GIRQ17	1001
GIRQ18	1010
GIRQ19	1011
GIRQ20	1100
GIRQ21	1101
GIRQ22	1110

The “Highest Priority Interrupt within the Group” block selects the highest numbered interrupt within the group that is active and enabled. The five bit result becomes bits 6 through 2 of R56. The low two bits of R56 are always 0, since the register always points to a word-aligned address.

A two instruction sequence is sufficient to vector to one of the Interrupt Aggregator interrupts. The following two-instruction sequence fits into the IRQ3 8-byte vector slot used by the ARC:

```

LDRx, [R56]           ;fetch the address for the active interrupt
J[Rx]                 ;Jump to handler

```

The register Rx should be reserved for exclusive use by the interrupt handler, using either Register R53 or R54, or using compiler option -Hreg\_reserve=x, so that the register is never live an interrupt is serviced.

# MEC1618/MEC1618i

---

An alternate ARC handler would be:

```
JCommon_Int_Handler      ;Jump to handler
```

where `Common_Int_Handler` is a routine that handles interrupt state, performs any interrupt stack maintenance, and sets up whatever is required for the actual interrupt handler. The jump instruction is 8 bytes long and can reach any address in the ARC address space. When the common code is finished, the same two instruction sequence listed above can be used to jump to the correct interrupt handler.

The minimal ARC vector would consist of one 8-byte jump instruction. The time penalty for adding the load from R56 is two cycles, one for the standard ARC load delay and an additional cycle the ARC requires when fetching data from instruction space.

## 15.3.4 NON MASKABLE INTERRUPTS

The ARC does not have a non maskable interrupt input. It is straightforward to assign one of the Hibernation or Week timers, which can generate ARC interrupts on ARC IRQ5 through IRQ7, to ARC priority level 2, and all other interrupts to priority level 1. Since no other interrupt handler should ever disable interrupts on Priority 2, the timer interrupt will always be enabled.

It may not be possible to use a non maskable interrupt when the flash memory is being programmed by the Host. The Flash interrupt is asserted when a Host has completed Flash programming and the flash array is again available to the ARC. While the flash is being programmed, the ARC has no program space (which is normally in the flash), and so must sleep. Since it has no vector table, it cannot respond to interrupts while the flash is busy. All interrupts except `Flash_EC_Int` must be disabled; the Flash interrupt can be enabled since it is only asserted when the flash is again available to the ARC. One simple way to ensure this is to assign the `Flash_EC_Int` interrupt Priority Level 2, assign all other ARC IRQs to Priority Level 1, and disable Priority Level 1 before putting the ARC to sleep. An alternative would be to remap the ARC Interrupt Vector Base Register to the SRAM and define an interrupt vector table, along with its handlers, in the SRAM.

## 15.3.5 WAKE CAPABLE INTERRUPTS

The [EC Interrupt Aggregator](#) routes logic from WAKE Event Sources to the [WAKE](#) input of the [Power, Clocks, and Resets Power Management Interface](#) to wake the system. This logic requires no clocks.

The interrupt sources AND'ed with the corresponding Enable bit will be OR'ed to produce a wake event

The wake up sources are identified in a "WAKE" column of the Bit definitions table for each IRQ's Source Register.

## 15.3.6 NON-WAKE CAPABLE INTERRUPTS

These interrupts require a running clock in their source block to be recognized and presented to the interrupt aggregator. Please consult the WAKE column of the Bit definitions table for each IRQ's Source Register.

## 15.3.7 INTERRUPTS DIRECTLY CONNECTED TO ARC PROCESSOR

Interrupts from the two Hibernation Timers, the Week Timer and the Embedded Flash interface are routed to the EC Interrupt lines IRQ7 through IRQ4, respectively.

**PROGRAMMER'S NOTE:** In non-legacy mode the Hibernation Timers, the Week Timer and the Embedded Flash interface are directly connected to the interrupt line and should be disabled in the Interrupt Enable Register. Otherwise these interrupts will be duplicated on the IRQ3 as well as their respective interrupt lines.

## 15.3.8 DISABLING INTERRUPTS

Because of pipeline latency, delay through the Load/Store queue and traffic on the AHB bus, writes to SPB registers can potentially take many processor cycles to complete. Because of this latency, the [IA Group Select Register](#) register and the [IRQ Enable Register](#) should not be used for disabling interrupts for software operations like critical sections. Several interrupts could potentially fire between the `STORE` instruction to the interrupt registers and the instruction after the `STORE`.



The ARC `FLAG` instruction is used to modify the E1 and E2 interrupt enable bits in the `STATUS32` register. If the `FLAG` instruction is used, software can insure that no unexpected interrupts will be processed in the middle of a critical section. The following example illustrates how the `FLAG` instruction might be used to implement a critical section:

```
FLAG; 0                ; disable all interrupts
SYNC                  ; Complete I/O ops
;
; <critical section code here>
;
FLAG; 6;              ; enable all interrupts
```

## 15.4 Power, Clocks and Reset

### 15.4.1 POWER DOMAIN

This block is powered by VTR for wake up capability.

See [Section 5.5, "Power Configuration," on page 101](#) for details on power domains.

### 15.4.2 CLOCKS

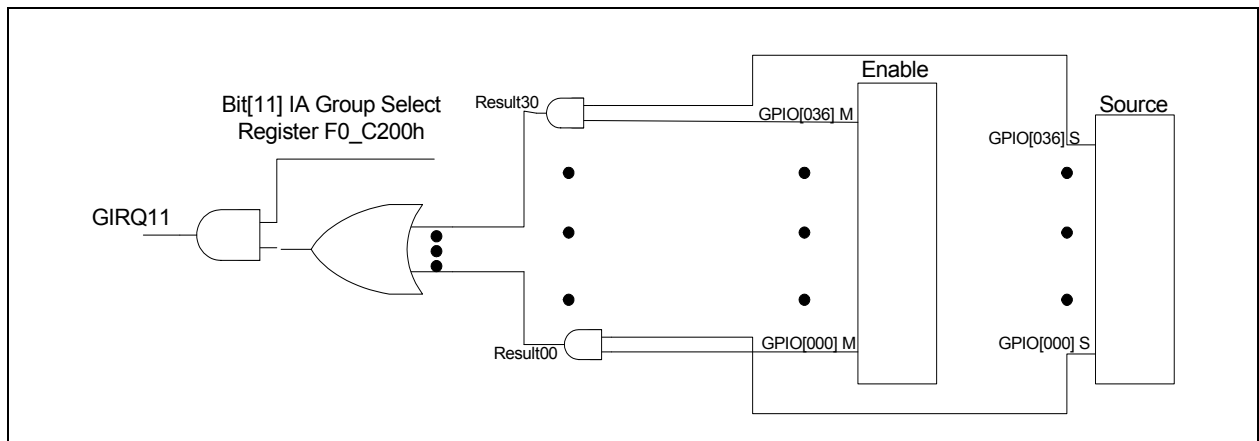
Use MCLK

### 15.4.3 RESET

This block is reset by **nSYS\_RST**. Following a reset, Interrupt Source, Enable and Result registers default to '0' and all interrupts are enabled.

#### 15.4.4 INTERRUPT ROUTING

**FIGURE 15-5: GPIO INTERRUPT STRUCTURE EXAMPLE**



## 15.5 Registers Overview

### 15.5.1 ADDRESSING

The **EC Interrupt Aggregator** has its own Logical Device Number, and Base Address as indicated in [Table 15-5](#).

**TABLE 15-5: EC Interrupt Aggregator BASE ADDRESS TABLE**

Block	LDN	AHB Base Address
EC Interrupt Aggregator	30h	F0_C000h

# MEC1618/MEC1618i

## 15.5.2 REGISTERS SUMMARY

Table 15-6 is a register summary for the [EC Interrupt Aggregator](#) block. Each EC address is indicated as an SPB Offset from its AHB base address.

**TABLE 15-6: [EC Interrupts](#) REGISTER SUMMARY**

Register Name	EC Interface			Notes	Logical Devices
	SPB Offset	Byte Lane	EC Type	Bit Definitions	
<a href="#">GIRQ8 Source Register</a>	00h	3-0	R/WC	<a href="#">Table 15-9</a>	GPIO140- GPIO176
<a href="#">GIRQ8 Enable Register</a>	04h	3-0	R/W		
<a href="#">GIRQ8 Result Register</a>	08h	3-0	R		
<a href="#">GRIQ8a Priority Register</a>	0Ch	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ8b Priority Register</a>	10h	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ9 Source Register</a>	14h	3-0	R/WC	<a href="#">Table 15-13</a>	GPIO100- GPIO136
<a href="#">GIRQ9 Enable Register</a>	18h	3-0	R/W		
<a href="#">GIRQ9 Result Register</a>	1Ch	3-0	R		
<a href="#">GRIQ9a Priority Register</a>	20h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ9b Priority Register</a>	24h	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ10 Source Register</a>	28h	3-0	R/WC	<a href="#">Table 15-17</a>	GPIO040- GPIO076
<a href="#">GIRQ10 Enable Register</a>	2Ch	3-0	R/W		
<a href="#">GIRQ10 Result Register</a>	30h	3-0	R		
<a href="#">GRIQ10a Priority Register</a>	34h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ10b Priority Register</a>	38h	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ11 Source Register</a>	3Ch	3-0	R/WC	<a href="#">Table 15-22</a>	GPIO000- GPIO036
<a href="#">GIRQ11 Enable Register</a>	40h	3-0	R/W		
<a href="#">GIRQ11 Result Register</a>	44h	3-0	R		
<a href="#">GRIQ11a Priority Register</a>	48h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ11b Priority Register</a>	4Ch	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ12 Source Register</a>	50h	3-0	R/WC	<a href="#">Table 15-26</a>	SMBus
<a href="#">GIRQ12 Enable Register</a>	54h	3-0	R/W		
<a href="#">GIRQ12 Result Register</a>	58h	3-0	R		
<a href="#">GRIQ12a Priority Register</a>	5Ch	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ12b Priority Register</a>	60h	3-0	R/W	<a href="#">Table 15-8</a>	

**TABLE 15-6: EC Interrupts REGISTER SUMMARY (CONTINUED)**

Register Name	EC Interface			Notes	Logical Devices
	SPB Offset	Byte Lane	EC Type	Bit Definitions	
GIRQ13 Source Register	64h	3-0	R/WC	Table 15-30	ACPI PM1
GIRQ13 Enable Register	68h	3-0	R/W		
GIRQ13 Result Register	6Ch	3-0	R		
GRIQ13a Priority Register	70h	3-0	R/W	Table 15-7	
GRIQ13b Priority Register	74h	3-0	R/W	Table 15-8	
GIRQ14 Source Register	78h	3-0	R/WC	Table 15-34	LPC Interface, EC GP-SPI, Embedded Flash Interface
GIRQ14 Enable Register	7Ch	3-0	R/W		
GIRQ14 Result Register	80h	3-0	R		
GRIQ14a Priority Register	84h	3-0	R/W	Table 15-7	
GRIQ14b Priority Register	88h	3-0	R/W	Table 15-8	
GIRQ15 Source Register	8Ch	3-0	R/WC	Table 15-38	Mailbox Interface, UART
GIRQ15 Enable Register	90h	3-0	R/W		
GIRQ15 Result Register	94h	3-0	R		
GRIQ15a Priority Register	98h	3-0	R/W	Table 15-7	
GRIQ15b Priority Register	9Ch	3-0	R/W	Table 15-8	
GIRQ16 Source Register	A0h	3-0	R/WC	Table 15-42	RC ID, ADC, PECI
GIRQ16 Enable Register	A4h	3-0	R/W		
GIRQ16 Result Register	A8h	3-0	R		
GRIQ16a Priority Register	ACh	3-0	R/W	Table 15-7	
GRIQ16b Priority Register	B0h	3-0	R/W	Table 15-8	
GIRQ17 Source Register	B4h	3-0	R/WC	Table 15-46	TACH
GIRQ17 Enable Register	B8h	3-0	R/W		
GIRQ17 Result Register	BCh	3-0	R		
GRIQ17a Priority Register	C0h	3-0	R/W	Table 15-7	
GRIQ17b Priority Register	C4h	3-0	R/W	Table 15-8	
GIRQ18 Source Register	C8h	3-0	R/WC	Table 15-50	BC Bus Master
GIRQ18 Enable Register	CCh	3-0	R/W		

# MEC1618/MEC1618i

**TABLE 15-6: EC Interrupts REGISTER SUMMARY (CONTINUED)**

Register Name	EC Interface			Notes	
	SPB Offset	Byte Lane	EC Type	Bit Definitions	Logical Devices
<a href="#">GIRQ18 Result Register</a>	D0h	3-0	R		
<a href="#">GRIQ18a Priority Register</a>	D4h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ18b Priority Register</a>	D8h	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ19 Source Register</a>	DCh	3-0	R/WC	<a href="#">Table 15-54</a>	Keyboard Controller (8042), PS/2
<a href="#">GIRQ19 Enable Register</a>	E0h	3-0	R/W		
<a href="#">GIRQ19 Result Register</a>	E4h	3-0	R		
<a href="#">GRIQ19a Priority Register</a>	E8h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ19b Priority Register</a>	ECh	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ20 Source Register</a>	F0h	3-0	R/WC	<a href="#">Table 15-58</a>	ACPI EC MSG
<a href="#">GIRQ20 Enable Register</a>	F4h	3-0	R/W		
<a href="#">GIRQ20 Result Register</a>	F8h	3-0	R		
<a href="#">GRIQ20a Priority Register</a>	FCh	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ20b Priority Register</a>	100h	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ21 Source Register</a>	104h	3-0	R/WC	<a href="#">Table 15-62</a>	Reserved
<a href="#">GIRQ21 Enable Register</a>	108h	3-0	R/W		
<a href="#">GIRQ21 Result Register</a>	10Ch	3-0	R		
<a href="#">GRIQ21a Priority Register</a>	110h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ21b Priority Register</a>	114h	3-0	R/W	<a href="#">Table 15-8</a>	

TABLE 15-6: EC Interrupts REGISTER SUMMARY (CONTINUED)

Register Name	EC Interface			Notes	Logical Devices
	SPB Offset	Byte Lane	EC Type	Bit Definitions	
<a href="#">GIRQ22 Source Register</a>	118h	3-0	R/WC	<a href="#">Table 15-66</a>	GPIO[217:200]
<a href="#">GIRQ22 Enable Register</a>	11Ch	3-0	R/W		
<a href="#">GIRQ22 Result Register</a>	120h	3-0	R		
<a href="#">GRIQ22a Priority Register</a>	124h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ22b Priority Register</a>	128h	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">GIRQ23 Source Register</a>	12Ch	3-0	R/WC	<a href="#">Table 15-70</a>	Week Alarm Timer, 16-bit Timer Hibernation Timer Input Capture and Compare Timer
<a href="#">GIRQ23 Enable Register</a>	130h	3-0	R/W		
<a href="#">GIRQ23 Result Register</a>	134h	3-0	R		
<a href="#">GRIQ23a Priority Register</a>	138h	3-0	R/W	<a href="#">Table 15-7</a>	
<a href="#">GRIQ23b Priority Register</a>	13Ch	3-0	R/W	<a href="#">Table 15-8</a>	
<a href="#">IA Group Select Register</a>	200h	3-0	R/W	<a href="#">Table 15-73</a>	
<a href="#">IRQ Enable Register</a>	204h	3-0	R/W	<a href="#">Table 15-74</a>	

## 15.6 Register Descriptions

Input interrupts are grouped into groups of up to 31 interrupts each. Associated with each group is a set of Source, Enable, Priority, and Result registers. Registers are 32-bit. There are two Priority registers for each group since an interrupt's priority level is encoded by 2 bits.

The aforementioned four register 'types' are first described in generic terms. Subsequent sections describe register bits in terms of specific interrupts they are associated with.

### 15.6.1 GIRQX SOURCE REGISTERS

There are 16 Group Source Enable registers, one per Interrupt Group.

#### Int\_Source[30:0]:

A bit in this field is set when the corresponding interrupt input is active. The Interrupt Aggregator recognizes level-triggered, active-high inputs. Other input types are to be captured and relayed to the Aggregator. For example, the GPIO interface can register external edge-triggered interrupts and forward them to the Aggregator as active-high, level interrupts.

### 15.6.2 GIRQX ENABLE REGISTERS

There are 16 Group Interrupt Enable registers, one per Interrupt Group.

#### Int\_Enable[30:0]:

Each bit in this field enables an interrupt from the like-numbered bit in the associated Interrupt Source register. Interrupt *i* is disabled if Int\_Enable[*i*] is 0 and enabled if Int\_Enable[*i*] is 1 and the Int\_Priority[*i*] is greater than or equal to the current priority level. See [Table 15-6, "EC Interrupts Register Summary"](#) for EC Offset addresses for the 16 GIRQX Enable registers.

# MEC1618/MEC1618i

## 15.6.3 GIRQX A PRIORITY REGISTERS

There are 16 Group Interrupt A Priority registers, one per Interrupt Group. Group Interrupt A Priority registers, combined with the Group Interrupt B Priority registers, determine the 2-bit priority level for all interrupts in an Interrupt Group. The format of all A Priority registers is the same, described in [Table 15-7, "GIRQx A Priority Register"](#): *It should be noted that at times not all corresponding source register bits are used, hence the respective Int\_priority bits should not be populated.*

**TABLE 15-7: GIRQX A PRIORITY REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	xxh				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority15		Int_Priority14		Int_Priority13		Int_Priority12		
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority11		Int_Priority10		Int_Priority9		Int_Priority8		
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority7		Int_Priority6		Int_Priority5		Int_Priority4		
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority3		Int_Priority2		Int_Priority1		Int_Priority0		

### INT\_PRIORITY[15:0]

Each of the 16 2-bit fields in this register sets the priority level for interrupts assigned to Interrupt Source register bit 15 through bit 0, in the same Interrupt Group. See [Table 15-6, "EC Interrupts Register Summary"](#) for EC Offset addresses for the sixteen GIRQx A Priority registers.

## 15.6.4 GIRQX B PRIORITY REGISTERS

There are 16 Group Interrupt B Priority registers, one per Interrupt Group. Group Interrupt B Priority registers, combined with the Group Interrupt A Priority registers, determine the 2-bit priority level for all interrupts in an Interrupt Group. The format of all B Priority registers is the same, described in [Table 15-8, "GIRQx B Priority Register"](#): *It should be noted that at times not all corresponding source register bits are used, hence the respective Int\_priority bits should not be populated.*

**TABLE 15-8: GIRQX B PRIORITY REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	xxh				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Reserved		Int_Priority30		Int_Priority29		Int_Priority28		
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority27		Int_Priority26		Int_Priority25		Int_Priority24		
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority23		Int_Priority22		Int_Priority21		Int_Priority20		
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Int_Priority19		Int_Priority18		Int_Priority17		Int_Priority16		

### INT\_PRIORITY[30:16]

Each of the 16 2-bit fields in this register sets the priority level for interrupts assigned to Interrupt Source register bit 30 through bit 16, in the same Interrupt Group. See [Table 15-6, "EC Interrupts Register Summary"](#) for EC Offset addresses for the sixteen GIRQx B Priority registers.

# MEC1618/MEC1618i

## 15.6.5 GIRQX RESULT REGISTERS

There are 16 Group Interrupt Result registers, one per Interrupt Group.

### INT\_RESULT[30:0]

Each bit in this field is 1 if an interrupt from the like-numbered bit in the associated Interrupt Source register is active. Interrupt *i* in each Interrupt Group is active if and only if Int\_Source[*i*] is 1, Int\_Enable[*i*] is 1 and Int\_Priority[*i*] is greater than or equal to IA Priority register as well as any enabled interrupt in all Interrupt Groups. The GIRQx Result Register is not latched but is a function of Int\_Source, Int\_Enable and Int\_Priority. See [Figure 15-2](#) for an explanation of how Result is generated.

## 15.6.6 GIRQ8 SOURCE REGISTER

**TABLE 15-9: GIRQ8 SOURCE REGISTER**

HOST ADDRESS	N/A								HOST SIZE
EC OFFSET	00h					32-bit			EC SIZE
POWER	VTR					0000_0000h			nSYS_RST DEFAULT
BUS	EC SPB								
BIT	D31	D30	D29	■	■	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-10</a> & <a href="#">Note 15-1 on page 312</a>							
BIT NAME	Reserved	GPIO[140:176]							

**TABLE 15-10: BIT DEFINITIONS GIRQ8 SOURCE REGISTER**



Bit	EC Type	Signal Name	Wake	Description
[7:0]	R/WC	GPIO[147:140]	Y	GPIO Interrupt Signal. This bit is set when a interrupt source is triggered. This bit is sticky; after being set, it remains set until cleared by a write with the value '1' to this bit. Write to this bit with a value of '0' have no effect.
[15:8]	R/WC	GPIO[157:150]	Y	
[20:16]	R/WC	GPIO[164:160]	Y	
[23:21]	R/WC	GPIO[167:165]	Y	
[30:24]	R/WC	GPIO[176:170]	Y	
31	R	Reserved	N	This bit is always reserved

**Note 15-1** Reserved Bits shown in the Source register are also reserved read only in the Corresponding Enable & Result register.





## 15.6.7 GIRQ8 ENABLE REGISTER

**TABLE 15-11: GIRQ8 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	04h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RW except for reserved bits See <a href="#">Table 15-10</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[140:176]						

## 15.6.8 GIRQ8 RESULT REGISTER

**TABLE 15-12: GIRQ8 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	08h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[140:176]						



### **NORM**

This bit is a read write bit used by the NORM instruction.

# MEC1618/MEC1618i

## 15.6.9 GIRQ9 SOURCE REGISTER

**TABLE 15-13: GIRQ9 SOURCE REGISTER**



HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	14h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-14</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[100:136]						

**TABLE 15-14: BIT DEFINITIONS GIRQ9 SOURCE REGISTER**

Bit	EC Type	Signal Name	Wake	Description
[7:0]	R/WC	GPIO[107:100]	Y	GPIO Interrupt Signal. This bit is set when a interrupt source is triggered. This bit is sticky; after being set, it remains set until cleared by a write with the value '1' to this bit. Write to this bit with a value of '0' have no effect.
[15:8]	R/WC	GPIO[117:110]	Y	
[23:16]	R/WC	GPIO[127:120]	Y	
[29:24]	R/WC	GPIO[135:130]	Y	
30	R	GPIO136	N	Reserved
31	R	Reserved	N	This bit is always reserved



## 15.6.10 GIRQ9 ENABLE REGISTER

**TABLE 15-15: GIRQ9 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	18h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RW except for reserved bits See <a href="#">Table 15-14</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[100:136]						

## 15.6.11 GIRQ9 RESULT REGISTER

**TABLE 15-16: GIRQ9 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	1Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[100:136]						



### **NORM**

This bit is a read write bit used by the NORM instruction.

# MEC1618/MEC1618i

## 15.6.12 GIRQ10 SOURCE REGISTER

**TABLE 15-17: GIRQ10 SOURCE REGISTER**



HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	28h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-18</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[040:076]						

**TABLE 15-18: BIT DEFINITIONS GIRQ10 SOURCE REGISTER**

Bit	EC Type	Signal Name	Wake	Description
[7:0]	R/WC	GPIO[047:040]	Y	GPIO Interrupt Signal. This bit is set when a interrupt source is triggered. This bit is sticky; after being set, it remains set until cleared by a write with the value '1' to this bit. Write to this bit with a value of '0' have no effect.
[15:8]	R/WC	GPIO[057:050]	Y	
[23:16]	R/WC	GPIO[067:060]	Y	
[30:24]	R	GPIO[076:070]	Y	
31	R	Reserved	N	This bit is always reserved

## 15.6.13 GIRQ10 ENABLE REGISTER

**TABLE 15-19: G IRQ10 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	2Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RW except for reserved bits See <a href="#">Table 15-18</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[040:076]						

## 15.6.14 GIRQ10 RESULT REGISTER

**TABLE 15-20: GIRQ10 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	30h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[040:076]						



### **NORM**

This bit is a read write bit used by the NORM instruction.

# MEC1618/MEC1618i

## 15.6.15 GIRQ11 SOURCE REGISTER

**TABLE 15-21: GIRQ11 SOURCE REGISTER**



HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	3Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-22</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[000:036]						

**TABLE 15-22: BIT DEFINITIONS GIRQ11 SOURCE REGISTER**

Bit	EC Type	Signal Name	Wake	Description
[7:0]	R/WC	GPIO[007:000]	Y	GPIO Interrupt Signal. This bit is set when a interrupt source is triggered. This bit is sticky; after being set, it remains set until cleared by a write with the value '1' to this bit. Write to this bit with a value of '0' have no effect.
[15:8]	R/WC	GPIO[017:010]	Y	
[23:16]	R/WC	GPIO[027:020]	Y	
[26:24]	R/WC	GPIO[032:030]	Y	
[30:27]	R/WC	GPIO[036:033]	Y	
31	R	Reserved	N	This bit is always reserved



## 15.6.16 GIRQ11 ENABLE REGISTER

**TABLE 15-23: GIRQ11 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	40h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RW except for reserved bits See <a href="#">Table 15-22</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	GPIO[000:036]						

## 15.6.17 GIRQ11 RESULT REGISTER

**TABLE 15-24: GIRQ11 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	44h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[000:036]						

### **NORM**

This bit is a read write bit used by the NORM instruction.

# MEC1618/MEC1618i

## 15.6.18 GIRQ12 SOURCE REGISTER

**TABLE 15-25: GIRQ12 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	50h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-26</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-26, “Bit Definitions GIRQ12 Source Register,” on page 320</a>						

**TABLE 15-26: BIT DEFINITIONS GIRQ12 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	SMB0	N	I2C/SMBus controller 0 interrupt. This interrupt is signaled when the I2C/SMBus controller 0 asserts its interrupt request
1	SMB1	N	I2C/SMBus controller 1 interrupt. This interrupt is signaled when the I2C/SMBus controller 1 asserts its interrupt request
2	SMB2	N	I2C/SMBus controller 2 interrupt. This interrupt is signaled when the I2C/SMBus controller 2 asserts its interrupt request
3	SMB00 WK	Y	I2C/SMBus Port 00 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB00.
4	SMB01 WK	Y	I2C/SMBus Port 01 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB01.
5	SMB02 WK	Y	I2C/SMBus Port 02 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB02.
6	SMB03 WK	Y	I2C/SMBus Port 03 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB03.
7	SMB04 WK	Y	I2C/SMBus Port 04 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB04.
8	SMB05 WK	Y	I2C/SMBus Port 05 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB05.
9	SMB06 WK	Y	I2C/SMBus Port 06 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB06.
10	SMB07 WK	Y	I2C/SMBus Port 07 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB07.



**TABLE 15-26: BIT DEFINITIONS GIRQ12 SOURCE REGISTER (CONTINUED)**

Bit	Signal Name	Wake	Description
11	SMB08 WK	Y	I2C/SMBus Port 10 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB10.
12	SMB09 WK	Y	I2C/SMBus Port 09 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB09.
13	SMB10 WK	Y	I2C/SMBus Port 10 Wake interrupt. This interrupt is asserted when an edge is detected on the SDAT pin of port SMB10.
14	Reserved	N	Reserved
15	Reserved	N	Reserved
16	Reserved	N	Reserved
17	SB_TSI	Y	SB-TSI (Port 11) wake interrupt.
30-18	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-2** Source bits have corresponding Enable and Result bits.

## 15.6.19 GIRQ12 ENABLE REGISTER

**TABLE 15-27: GIRQ12 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	54h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-26</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-26</a> , “Bit Definitions GIRQ12 Source Register,” on page 320						

# MEC1618/MEC1618i

## 15.6.20 GIRQ12 RESULT REGISTER

TABLE 15-28: GIRQ12 RESULT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	58h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-26, “Bit Definitions GIRQ12 Source Register,” on page 320</a>						

### NORM

This bit is a read write bit used by the NORM instruction.

## 15.6.21 GIRQ13 SOURCE REGISTER

TABLE 15-29: GIRQ13 SOURCE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	64h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-30</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-30, “Bit Definitions GIRQ13 Source Register,” on page 323</a>						

TABLE 15-30: BIT DEFINITIONS GIRQ13 SOURCE REGISTER

Bit	Signal Name	Wake	Description
0	PM1_CTL2	N	PM1_CTL2 written by Host
1	PM1_EN2	N	PM1_EN2 written by Host
2	PM1_STS2	N	PM1_STS2 written by Host
15:3	Reserved	N	Reserved
16	DMA_0	N	DMA Channel 0
17	DMA_1	N	DMA Channel 1
18	DMA_2	N	DMA Channel 2
19	DMA_3	N	DMA Channel 3
20	DMA_4	N	DMA Channel 4
21	DMA_5	N	DMA Channel 5
22	DMA_6	N	DMA Channel 6
23	DMA_7	N	DMA Channel 7
30-24	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-3** Source bits have corresponding Enable and Result bits.

#### 15.6.22 GIRQ13 ENABLE REGISTER

TABLE 15-31: GIRQ13 ENABLE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	68h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W except for reserved bits See <a href="#">Table 15-30</a> & <a href="#">Note 15-1</a> on page 312						
BIT NAME	Reserved	Refer To <a href="#">Table 15-30</a> , “Bit Definitions GIRQ13 Source Register,” on page 323						

# MEC1618/MEC1618i

## 15.6.23 GIRQ13 RESULT REGISTER

TABLE 15-32: GIRQ13 RESULT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	6Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-30, “Bit Definitions GIRQ13 Source Register,”</a> on page 323						

### NORM

This bit is a read write bit used by the NORM instruction.

## 15.6.24 GIRQ14 SOURCE REGISTER

TABLE 15-33: GIRQ14 SOURCE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	78h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-34</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-34, “Bit Definition GIRQ14 source Register,” on page 325</a>						

TABLE 15-34: BIT DEFINITION GIRQ14 SOURCE REGISTER

Bit	Signal Name	Wake	Description
0	LPCPD#	Y	LPC Power Down pin state
1	LRESET#	Y	LRESET Interrupt. This interrupt is signaled when LRESET is asserted.
2	LPC_AHB_ERR	N	either a LPC BAR conflict or an AHB bus error occurred as a result of an LPC access.
3	SPI_TXBE_GP	N	Tx buffer empty from the GP_SPI block on EC AHB
4	SPI_RXBF_GP	N	Rx buffer bfull from the GP_SPI block on EC AHB
5	FLASH_BUSY_ERR	N	Embedded Flash Busy Error
6	FLASH_CMD_ERR	N	Embedded Flash Command Error
7	FLASH_PROTECT_ERR	N	Embedded Flash Protect Error
8	FLASH_EC_INT	N	Host-to-EC Interrupt that transfers control of the Flash to the EC
9	VCC_PWRGD_INT	Y	VCC_PWRGD (from GPIO 57)
15:10	Reserved	-	Reserved
16	GP_SPI_TXBE	N	Tx buffer empty from the GP_SPI block on LPC AHB
17	GP_SPI_RXBF	N	Rx buffer bfull from the GP_SPI block on LPC AHB
18	ASIF_INT	N	Interrupt from the ASIF block's EC logical interface
30-19	Reserved	-	Reserved
31	Reserved	-	This bit is always reserved

**Note 15-4** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.25 GIRQ14 ENABLE REGISTER

**TABLE 15-35: GIRQ14 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	7Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-34</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-34</a> , “Bit Definition GIRQ14 source Register,” on page 325						

## 15.6.26 GIRQ14 RESULT REGISTER

**TABLE 15-36: GIRQ14 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	80h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-34</a> , “Bit Definition GIRQ14 source Register,” on page 325						

### **NORM**

This bit is a read write bit used by the NORM instruction.

## 15.6.27 GIRQ15 SOURCE REGISTER

**TABLE 15-37: GIRQ15 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	8Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-38</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-38</a> , “Bit Definition GIRQ15 Source Register,” on page 327						

**TABLE 15-38: BIT DEFINITION GIRQ15 SOURCE REGISTER**



Bit	Signal Name	Wake	Description
0	UART_RX	N	UART Interrupt
1	MBX	N	Mailbox Register Interface EC Interrupt
2	EM_MBX	N	<a href="#">Embedded Memory Interface</a> Host-to-EC Mailbox Interrupt
7-3	Reserved	N	Reserved
8	BDP0_INT	N	<a href="#">BIOS Debug Ports</a>
9	BDP1_INT	N	
10	LED0	N	<a href="#">Blinking/Breathing PWM</a>
11	LED1	N	
12	LED2	N	
30-13	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-5** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.28 GIRQ15 ENABLE REGISTER

**TABLE 15-39: GIRQ15 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	90h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-38</a> & <a href="#">Note 15-1</a> on page 312						
BIT NAME	Reserved	Refer To <a href="#">Table 15-38</a> , “Bit Definition GIRQ15 Source Register,” on page 327						

## 15.6.29 GIRQ15 RESULT REGISTER

**TABLE 15-40: GIRQ15 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	94h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-38</a> , “Bit Definition GIRQ15 Source Register,” on page 327						



### **NORM**

This bit is a read write bit used by the NORM instruction.



## 15.6.30 GIRQ16 SOURCE REGISTER

**TABLE 15-41: GIRQ16 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	A0h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-42</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-42</a> , “Bit Definition IRQ16 Source Register,” on page 329						

**TABLE 15-42: BIT DEFINITION IRQ16 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	RCID	N	0-to-1 transition of RC_ID done flag
1	ADC_ONESTAT	N	ADC's one-shot conversion completion interrupt
2	ADC_RTPSTAT	N	ADC's repeated conversion interrupt
3	PECI_INT	N	PECI interrupt
4	CEC_INT	N	<a href="#">HDMI-CEC Interface Controller</a>
7-5	Reserved	N	Reserved
8	Reserved	N	Reserved
9	Reserved	N	<b>Reserved</b>
30-10	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-6** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.31 GIRQ16 ENABLE REGISTER

TABLE 15-43: GIRQ16 ENABLE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	A4h			32-bit			EC SIZE	
POWER	VTR			0000_00000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-42</a> & <a href="#">Note 15-1</a> on page 312						
BIT NAME	Reserved	Refer To <a href="#">Table 15-42</a> , “Bit Definition IRQ16 Source Register,” on page 329						

## 15.6.32 GIRQ16 RESULT REGISTER

TABLE 15-44: GIRQ16 RESULT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	A8h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-42</a> , “Bit Definition IRQ16 Source Register,” on page 329						

### NORM

This bit is a read write bit used by the NORM instruction.

## 15.6.33 GIRQ17 SOURCE REGISTER

**TABLE 15-45: GIRQ17 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	B4h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-46</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-46</a> , “Bit Definition GIRQ17 Source Register,” on page 331						

**TABLE 15-46: BIT DEFINITION GIRQ17 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	TACH0	N	Fan TACH 0 Interrupt.
1	TACH1	N	Fan TACH 1 Interrupt.
2	TACH2	N	Fan TACH 2 Interrupt.
3	TACH3	N	Fan TACH 3 Interrupt.
4	TACH4	N	Fan TACH 4 Interrupt.
5	TACH5	N	Fan TACH 5 Interrupt.
30-6	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-7** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.34 GIRQ17 ENABLE REGISTER

**TABLE 15-47: GIRQ17 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	B8h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-46</a> & <a href="#">Note 15-1</a> on page 312						
BIT NAME	Reserved	Refer To <a href="#">Table 15-46</a> , “Bit Definition GIRQ17 Source Register,” on page 331						

## 15.6.35 GIRQ17 RESULT REGISTER

**TABLE 15-48: GIRQ17 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	98h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-46</a> , “Bit Definition GIRQ17 Source Register,” on page 331						

### **NORM**

This bit is a read write bit used by the NORM instruction.

## 15.6.36 GIRQ18 SOURCE REGISTER

**TABLE 15-49: GIRQ18 SOURCE REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	C8h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-50</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-50</a> , “Bit Definition IRQ18 Source Register,” on page 333						

**TABLE 15-50: BIT DEFINITION IRQ18 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	BCM_BUSY_CLR[A]	N	BC-LINK Busy Clear Flag Interrupt
1	BCM_ERR[A]	N	BC_LINK Error Flag Interrupt
2	BCM_INT#[A]	Y	Interrupt from the BC_LINK Companion
3	BCM_BUSY_CLR[B]	N	BC-LINK Busy Clear Flag Interrupt
4	BCM_ERR[B]	N	BC_LINK Error Flag Interrupt
5	BCM_INT#[B]	Y	Interrupt from the BC_LINK Companion
6	Reserved	N	Reserved
7	Reserved	N	Reserved
8	Reserved	N	Reserved
9	BCM_BUSY_CLR[D]	N	BC-LINK Busy Clear Flag Interrupt
10	BCM_ERR[D]	N	BC_LINK Error Flag Interrupt
11	BCM_INT#[D]	Y	Interrupt from the BC_LINK Companion
15-12	Reserved	N	Reserved
16	KEYSCAN	Y	Interrupt from keyscan block
30-17	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-8** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.37 GIRQ18 ENABLE REGISTER

**TABLE 15-51: GIRQ18 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	CCh			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-50</a> & <a href="#">Note 15-1</a> on page 312						
BIT NAME	Reserved	Refer To <a href="#">Table 15-50</a> , “Bit Definition IRQ18 Source Register,” on page 333						

## 15.6.38 GIRQ18 RESULT REGISTER

**TABLE 15-52: GIRQ18 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	D0h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-50</a> , “Bit Definition IRQ18 Source Register,” on page 333						

### **NORM**

This bit is a read write bit used by the NORM instruction.

## 15.6.39 GIRQ19 SOURCE REGISTER

**TABLE 15-53: GIRQ19 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	DCh			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-54</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-54, “Bit Definition GIRQ19 Source Register,” on page 335</a>						

**TABLE 15-54: BIT DEFINITION GIRQ19 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	KBD_OBF	N	Keyboard Controller OBF Interrupt. This interrupt is signaled when the OBF bit in the KBD status register has been clear.
1	KBD_IBF	N	Keyboard Controller IBF Interrupt. This interrupt is signaled when the host writes to the KBD Command or Data port.
12-2	Reserved	N	Reserved
13	PS2_ACT_0	N	PS2_0 Activity Interrupt form PS/2 Block
14	PS2_ACT_1	N	PS2_1 Activity Interrupt form PS/2 Block
15	PS2_ACT_2	N	PS2_2 Activity Interrupt form PS/2 Block
16	Reserved	N	Reserved
17	PS2_WK_0A	Y	PS2_0A Start Detectform pin signal
18	PS2_WK_0B	Y	PS2_0B Start Detectform pin signal
19	PS2_WK_1A	Y	PS2_1A Start Detectform pin signal
20	PS2_WK_1B	Y	PS2_1B Start Detectform pin signal
21	PS2_WK_2	Y	PS2_2 Start Detectform pin signal
30-22	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-9** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.40 GIRQ19 ENABLE REGISTER

**TABLE 15-55: GIRQ19 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	E0h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-54</a> & <a href="#">Note 15-1</a> on page 312						
BIT NAME	Reserved	Refer To <a href="#">Table 15-54</a> , “Bit Definition GIRQ19 Source Register,” on page 335						

## 15.6.41 GIRQ19 RESULT REGISTER

**TABLE 15-56: GIRQ19 RESULT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	E4h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-54</a> , “Bit Definition GIRQ19 Source Register,” on page 335						

### **NORM**

This bit is a read write bit used by the NORM instruction.



## 15.6.42 GIRQ20 SOURCE REGISTER

**TABLE 15-57: GIRQ20 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	F0h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-58</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-58, “Bit Definition IRQ20 Source Register,” on page 337</a>						

**TABLE 15-58: BIT DEFINITION IRQ20 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	EC2_OBF	N	Embedded Controller 2 OBF Interrupt. This interrupt is signaled when the OBF bit in the EC2 status register has been clear.
1	EC2_IBF	N	Embedded Controller 2 IBF Interrupt. This interrupt is signaled when the host writes to the EC2 Command or Data port.
2	EC1_OBF	N	Embedded Controller 1 OBF Interrupt. This interrupt is signaled when the OBF bit in the EC1 status register has been clear.
3	EC1_IBF	N	Embedded Controller 1 IBF Interrupt. This interrupt is signaled when the host writes to the EC1 Command or Data port.
4	EC0_OBF	N	Embedded Controller 0 OBF Interrupt. This interrupt is signaled when the OBF bit in the EC0 status register has been clear.
5	EC0_IBF	N	Embedded Controller 0 IBF Interrupt. This interrupt is signaled when the host writes to the EC0 Command or Data port.
6	EC3_OBF	N	Embedded Controller 3 OBF Interrupt. This interrupt is signaled when the OBF bit in the EC3 status register has been clear.
7	EC3_IBF	N	Embedded Controller 3 IBF Interrupt. This interrupt is signaled when the host writes to the EC3 Command or Data port.
30-8	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-10** Source bits have corresponding Enable and Result bits.

# MEC1618/MEC1618i

## 15.6.43 GIRQ20 ENABLE REGISTER

TABLE 15-59: GIRQ20 ENABLE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	F4h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved	Refer to <a href="#">Table 15-58, “Bit Definition IRQ20 Source Register,” on page 337</a>						

## 15.6.44 GIRQ20 RESULT REGISTER

TABLE 15-60: GIRQ20 RESULT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	C8h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer to <a href="#">Table 15-58, “Bit Definition IRQ20 Source Register,” on page 337</a>						

### NORM

This bit is a read write bit used by the NORM instruction.

## 15.6.45 GIRQ21 SOURCE REGISTER

**TABLE 15-61: GIRQ21 SOURCE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	104h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	RWC except for reserved bits See <a href="#">Table 15-62</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-62</a> , “Bit Definition IRQ21 Source Register,” on page 339						

**TABLE 15-62: BIT DEFINITION IRQ21 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
30:0	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved

**Note 15-11** Source bits have corresponding Enable and Result bits.

## 15.6.46 GIRQ21 ENABLE REGISTER

**TABLE 15-63: GIRQ21 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	108h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W except for reserved bits See <a href="#">Table 15-62</a> & <a href="#">Note 15-1 on page 312</a>						
BIT NAME	Reserved	Refer To <a href="#">Table 15-62</a> , “Bit Definition IRQ21 Source Register,” on page 339						

# MEC1618/MEC1618i

## 15.6.47 GIRQ21 RESULT REGISTER

TABLE 15-64: GIRQ21 RESULT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	10Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer To <a href="#">Table 15-62, “Bit Definition IRQ21 Source Register,” on page 339</a>						

### NORM

This bit is a read write bit used by the NORM instruction.

## 15.6.48 GIRQ22 SOURCE REGISTER

TABLE 15-65: GIRQ22 SOURCE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	118h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	Reserved	Refer To <a href="#">Table 15-66, “Bit Definitions GIRQ22 Source Register,” on page 341</a>						

**TABLE 15-66: BIT DEFINITIONS GIRQ22 SOURCE REGISTER**

Bit	EC Type	Signal Name	Wake	Description
[7:0]	R/WC	GPIO[207:200]	Y	GPIO Interrupt Signal. This bit is set when a interrupt source is triggered. This bit is sticky; after being set, it remains set until cleared by a write with the value '1' to this bit. Write to this bit with a value of '0' have no effect.
[15:8]	R/WC	GPIO[217:210]	Y	
[20:16]	R/WC	GPIO[227:220]	Y	
[29:24]	R/WC	GPIO[235:230]	Y	
30	R	GPIO236	N	Reserved
31	R	Reserved	N	This bit is always reserved

## 15.6.49 GIRQ22 ENABLE REGISTER

**TABLE 15-67: GIRQ22 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	11Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved	Refer To <a href="#">Table 15-66</a> , “Bit Definitions GIRQ22 Source Register,” on page 341						

# MEC1618/MEC1618i

## 15.6.50 GIRQ22 RESULT REGISTER

TABLE 15-68: GIRQ22 RESULT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	120h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	Refer To <a href="#">Table 15-66, “Bit Definitions GIRQ22 Source Register,” on page 341</a>						

### NORM

This bit is a read write bit used by the NORM instruction.

## 15.6.51 GIRQ23 SOURCE REGISTER

TABLE 15-69: GIRQ23 SOURCE REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	12Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	Reserved	Refer To <a href="#">Table 15-70, “Bit Definitions GIRQ23 Source Register,” on page 343</a>						

**TABLE 15-70: BIT DEFINITIONS GIRQ23 SOURCE REGISTER**

Bit	Signal Name	Wake	Description
0	TIMER0	N	16-bit Timer Interrupt
1	TIMER1	N	16-bit Timer Interrupt
2	TIMER2	N	16-bit Timer Interrupt
3	TIMER3	N	16-bit Timer Interrupt
4	Reserved	N	Reserved
5	PFR	N	Power Fail Register Interrupt
6	Reserved	N	Reserved
7	WEEK_ALR	Y	Week Alarm Interrupt. Week Timer has reached it's terminal count.
8	VCI_OVRD_IN	Y	Pin input of VCI_OVRD_IN
9	VCI_IN0	Y	input of VCI_IN0# pins
10	VCI_IN1	Y	input of VCI_IN1# pins
11	VCI_IN2	Y	input of VCI_IN2# pins
12	VCI_IN3	Y	input of VCI_IN3# pins
13	HTIMER0	Y	Hibernation Timer Interrupt. This interrupt is signaled when the hibernation timer has counter down to zero.
14	HTIMER1	Y	Hibernation Timer Interrupt. This interrupt is signaled when the hibernation timer has counter down to zero.
15	Reserved	N	Reserved
16	CAPTURE TIMER	N	The Free Running timer in the Capture/Compare Time transitioned from FFFF_FFFFh to 0000_0000h
17	CAPTURE 0	N	Capture register 0 in the Capture/Compare Timer unit has acquired a new value
18	CAPTURE 1	N	Capture register 1 in the Capture/Compare Timer unit has acquired a new value
19	CAPTURE 2	N	Capture register 2 in the Capture/Compare Timer unit has acquired a new value
20	CAPTURE 3	N	Capture register 3 in the Capture/Compare Timer unit has acquired a new value
21	CAPTURE 4	N	Capture register 4 in the Capture/Compare Timer unit has acquired a new value
22	CAPTURE 5	N	Capture register 5 in the Capture/Compare Timer unit has acquired a new value
23	COMPARE 0	N	Compare register 0 in the Capture/Compare Time unit tripped
24	COMPARE 1	N	Compare register 1 in the Capture/Compare Time unit tripped
25	VCI_IN4	Y	input of VCI_IN4# pin
26	VCI_IN5	Y	input of VCI_IN5# pin

# MEC1618/MEC1618i

**TABLE 15-70: BIT DEFINITIONS GIRQ23 SOURCE REGISTER (CONTINUED)**

Bit	Signal Name	Wake	Description
[30:27]	Reserved	N	Reserved
31	Reserved	N	This bit is always reserved



## 15.6.52 GIRQ23 ENABLE REGISTER

**TABLE 15-71: GIRQ23 ENABLE REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	130h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	■	■	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved	Refer To <a href="#">Table 15-70</a> , “Bit Definitions GIRQ23 Source Register,” on page 343						

## 15.6.53 GIRQ23 RESULT REGISTER

**TABLE 15-72: GIRQ23 RESULT REGISTER**

HOST ADDRESS							HOST SIZE	
EC OFFSET	134h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29			D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W-	R	R	R	R	R	R	R
BIT NAME	NORM	Refer To <a href="#">Table 15-70</a> , “Bit Definitions GIRQ23 Source Register,” on page 343						

### Bit 31 NORM

This bit is a read write bit used by the NORM instruction.



## 15.6.54 IA GROUP SELECT REGISTER

**TABLE 15-73: IA GROUP SELECT REGISTER**

HOST ADDRESS							HOST SIZE	
EC OFFSET	200h			32-bit			EC SIZE	
POWER	VTR			00FF_FF00h			VTR POR DEFAULT	
BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	GIRQ_Enable[23:16]							
BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	GIRQ_Enable[15:8]							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							

### GIRQ\_ENABLE[23:8]

Group IRQ Enable. Enable or disable all interrupts in a group.

0= All Interrupts in the associated GIRQ will be disabled.

1= All Interrupts in the associated GIRQ will be enabled.

# MEC1618/MEC1618i

## 15.6.55 IRQ ENABLE REGISTER

**TABLE 15-74: IRQ ENABLE REGISTER**

HOST ADDRESS							HOST SIZE	
EC OFFSET	204h			32-bit			EC SIZE	
POWER	VTR			00FF_FF00h			VTR POR DEFAULT	
BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	IRQ Enable[23:16]							
BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	IRQ Enable[15:8]							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R	R	R
BIT NAME	IRQ Enable[7:3]					Reserved		

### IRQ ENABLE[31:0]

IRQ Enable. Enable or disable all IRQs going to the ARC.

0= Respective IA IRQi will be disabled.

1= Respective IA IRQi will be enabled.

## 15.7 Extension Core Registers

### 15.7.1 EXTENSION CORE REGISTER R56

Register R56 is the Interrupt Aggregator Vector register. The Interrupt Aggregator automatically generates an address of a 4-byte location in the EC address space. The location contains a 4-byte address of an interrupt handler for the highest priority interrupt selected by the Interrupt Aggregator.

This register can be used in any ARC instruction with a 6-bit register address field.

**TABLE 15-75: IA VECTOR REGISTER R56**

BIT	D31	D30	D29	D28	D27	D26	D25	D24
TYPE	R	R	R	R	R	R	R	R
FIELD	0	0	0	0	0	0	0	0
BIT	D23	D22	D21	D20	D19	D18	D17	D16
TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
FIELD	VT_Base[12:5]							
BIT	D15	D14	D13	D12	D11	D10	D9	D8
TYPE	R/W	R/W	R/W	R/W	R/W	R	R	R
FIELD	VT_Base[4:0]					Group_Select[3:1]		
BIT	D7	D6	D5	D4	D3	D2	D1	D0
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	Group_Select[0]	Bit_Select[4:0]					0	0

#### D[6:2]: BIT\_SELECT[4:0]

This field is the index of the interrupt with the highest priority within the current group that is asserted, enabled, and set to a priority level that is equal to or greater than the current IA priority level. The interrupt with the highest bit number has the highest priority within a group.

This field is set by hardware and is read-only.

#### D[10:7]: GROUP\_SELECT[3:0]

This field is the index of the GIRQ with the highest priority in the Interrupt Aggregator among GIRQs in which at least one interrupt is asserted, enabled and set to a priority level that is equal to or greater than the current IA priority level. The GIRQ with the highest index has the highest priority among the GIRQs.

This field is set by hardware and is read-only.

#### VT\_BASE[11:0]

This field corresponds to bits 23:11 of the Interrupt Vector Table in the AHB address space. The IQ Vector Register is an address in the 24-bit AHB address space which contains the address of an interrupt handler. The AHB address space includes the [Boot Memory](#), the [Flash Memory](#) and the [Data/Instruction SRAM](#).

This field can be read and written by firmware. It should be initialized to the base address of the Interrupt Vector Table before interrupts are enabled.

# MEC1618/MEC1618i

## 15.7.2 EXTENSION CORE REGISTER R55

[Extension Core Register R55](#) is a 6-bit register that sets the current priority level for the Interrupt Aggregator. Bits 31:6 are reserved and always return 0 on reads. An interrupt priority level of 4 or higher disables all interrupts.

This register can be used in any ARC instruction with a 6-bit register address field.

**TABLE 15-76: IA PRIORITY REGISTER R55**

BIT	D31	D30	D29	D28	D27	D26	D25	D24
TYPE	R	R	R	R	R	R	R	R
FIELD	Reserved							
BIT	D23	D22	D21	D20	D19	D18	D17	D16
TYPE	R	R	R	R	R	R	R	R
FIELD	Reserved							
BIT	D15	D14	D13	D12	D11	D10	D9	D8
TYPE	R	R	R	R	R	R	R	R
FIELD	Reserved							
BIT	D7	D6	D5	D4	D3	D2	D1	D0
EC TYPE	R	R	R	R	R	R/W	R/W	R/W
BIT NAME	Reserved		Hi_Pri			Current_Priority		

### CURRENT\_PRIORITY[2:0]

This is a read/write field that determines the current priority level for the Interrupt Aggregator. Only interrupts that are configured to be at this priority level or higher will be used to generate the interrupt vector. A Current\_Priority value of 4 or higher will block all interrupts in the Interrupt Aggregator

### HI\_PRI

This three-bit field is set whenever [Extension Core Register R56](#) (IA Vector Register) is read. The value is the **priority level + 1** assigned to the interrupt that corresponds to the vector read in R56. It is the hardware that adds the 1 to the current priority level of the generated interrupt. For example, if the generated interrupt had a priority 2 then hardware will add 1 to it and write a value of 3 to the Hi\_Pri bits.

## 16.0 WATCHDOG TIMER INTERFACE

### 16.1 General Description

The function of the Watchdog Timer is to provide a mechanism to detect if the embedded controller has failed.

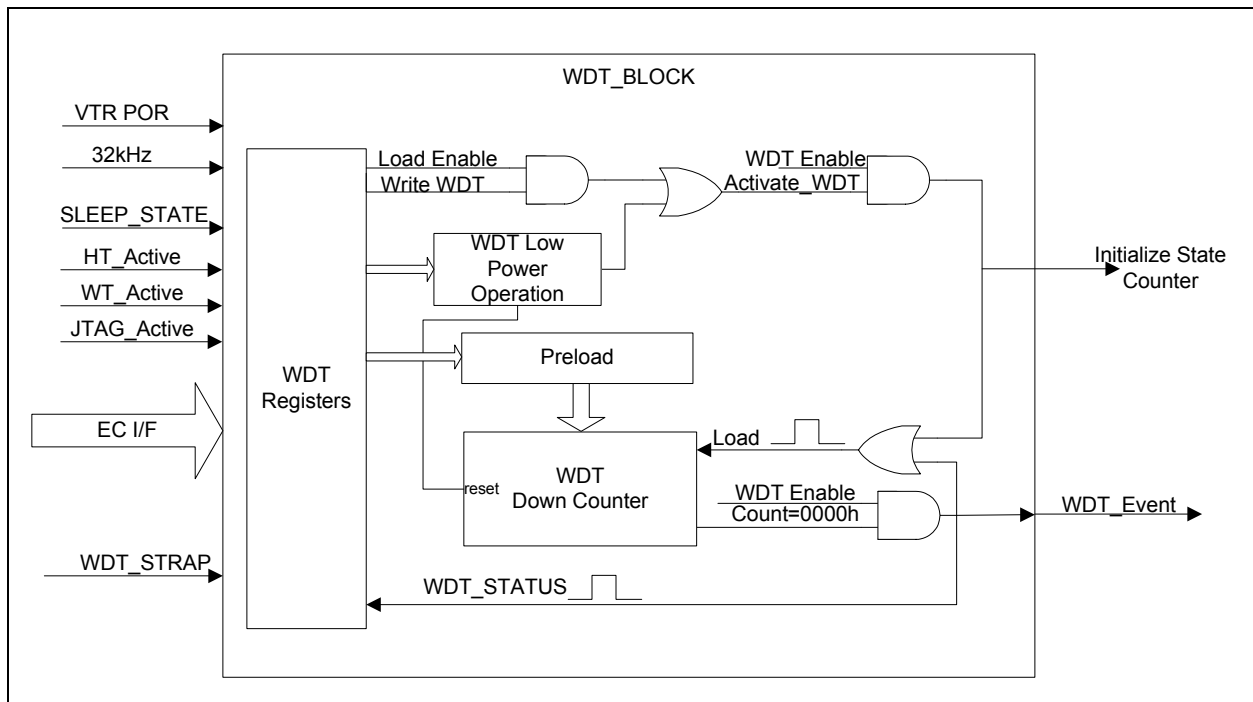
When enabled, the WATCHDOG Timer (WDT) circuit will generate a [WDT Event](#) if the user program fails to reload the WDT within a specified length of time known as the [WDT Interval](#).

This timer can be held inactive via the WDT Stall feature if the Hibernation timer, Week Timer, or the JTAG interface are enabled and active. This featured if enabled can be used to avoid unintended system resets.

Some operations can be carried out without any delay, e.g., registers can be read at any time and disabling the WDT takes effect immediately. On the other hand, 'kicking' the WDT may have a latency of up to 1 32-kHz cycle (~ 30 us). Similarly, when the load register is altered, the WDT cannot be enabled for up to 1 32-kHz cycle. Note that the ring oscillator must not be stopped within one 32-kHz clock following register write events.

### 16.2 Block Diagram

**FIGURE 16-1: Watchdog Timer Interface BLOCK DIAGRAM**



### 16.3 Watchdog Timer Interface Signal List

**TABLE 16-1: Watchdog Timer Interface SIGNAL LIST**

Signal Name	Direction	Description
<a href="#">nSYS_RST</a>	INPUT	Power on Reset to the block
32kHz	INPUT	<a href="#">X32K_CLK</a> , Clock source for WDT logic
HT_Active	INPUT	Signal indicating the Hibernation Timer is active and counting. See <a href="#">Section 19.0, "Hibernation Timer,"</a> on page 389.

# MEC1618/MEC1618i

**TABLE 16-1:** Watchdog Timer Interface SIGNAL LIST (CONTINUED)

Signal Name	Direction	Description
WT_Active	INPUT	Signal indicating the Week Timer is active and counting. See <a href="#">Section 20.0, "Week Alarm Interface," on page 393</a> .
JTAG_Active	INPUT	Signal indicating the JTAG interface is active. See <a href="#">Section 41.0, "JTAG and XNOR," on page 604</a> .
SPB Interface	I/O Bus	Bus used by microprocessor to access the registers in this block.
WDT Event	OUTPUT	Pulse generated when WDT expires ( <a href="#">Note 16-1</a> )

**Note 16-1** In the MEC1618/MEC1618i, the [WDT Event](#) output is routed to the [WDT\\_ALERT](#) input (see [Table 5-4, "Power, Clocks, and Resets Port List," on page 75](#)). Asserting the [WDT Event](#) output causes a [Watch-Dog Timer Forced Reset](#) (see [Section 5.6.12, "Watch-Dog Timer Forced Reset," on page 110](#)).

## 16.4 Power, Clocks and Reset

### 16.4.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

### 16.4.2 CLOCKS

This block has two clock inputs, the [EC Bus Clock](#) and [X32K\\_CLK](#). The [EC Bus Clock](#) is used in the interface to the embedded controller accessible registers. The 32.768kHz [X32K\\_CLK](#) is the clock source for the Watchdog Timer functional logic, including the counter.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

### 16.4.3 RESET

This block is reset on a [nSYS\\_RST](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

### 16.4.4 POWER MANAGEMENT

In all cases, [X32K\\_CLK](#) does not affect [Power Management](#); i.e., the [Watchdog Timer Interface](#) operates normally when the [MCLK](#) is stopped. The sleep enable inputs have no effect on the [Watchdog Timer Interface](#) and the clock required outputs are only asserted during register read/write cycles for as long as necessary to propagate updates to the block core ([Table 16-2](#)).

**TABLE 16-2:** WDT POWER MANAGEMENT

SLEEP_EN	Bus Access Cycle?	CLK_REQ	Description
X	Yes	1	CLK_REQ is only asserted for as long as necessary to propagate updates to the block core
	No	0	CLK_REQ is not asserted when the EC is not accessing the register interface. (Note that this block <i>cannot</i> prevent the chip from entering the system deepest sleep states.)

## 16.5 WDT Event Output Routing

The WDT Event (output) causes the [Section 5.6.12, "Watch-Dog Timer Forced Reset," on page 110](#).

The WDT Event state is also retained through a [Watch-Dog Timer Forced Reset](#) in the [WDT](#) bit of the [Power-Fail and Reset Status Register on page 131](#). The [Power-Fail and Reset Status Register](#) can generate a interrupt via the [PFR interrupt GIRQ23 Source Register on page 342](#).

The WDT Event output is not directly connected to an EC interrupt.

## 16.6 WDT Operation

### 16.6.1 WDT ACTIVATION MECHANISM

The WDT is activated by the following sequence of operations during normal operation. Note that the [WDT Load Register](#) can be programmed only when WDT is disabled ([WDT Enable](#) = '1')

1. Load the [WDT Load Register](#) with the count value. '0' is an invalid load value.
2. Set the [WDT Enable](#) bit in the [WDT Control Register](#).

The [WDT Activation Mechanism](#) starts the WDT decrementing counter.

### 16.6.2 WDT DEACTIVATION MECHANISM

The WDT is deactivated by the clearing the [WDT Enable](#) bit in the [WDT Control Register](#). The [WDT Deactivation Mechanism](#) places the WDT in a low power state in which clock are gated and the counter stops decrementing.

### 16.6.3 WDT RELOAD MECHANISM

The WDT must be reloaded within periods that are shorter than the programmed watchdog interval; otherwise the WDT will underflow and a [WDT Event](#) will be generated and the [WDT Status](#) bit will be set in the [WDT Control Register](#). It is the responsibility of the user program to continually execute sections of code which reload the watchdog timer (WDT) causing the counter to be reloaded

There are two methods of reloading the WDT: a write to the [WDT Kick Register](#) or the [WDT Activation Mechanism](#).

### 16.6.4 WDT INTERVAL

The [WDT Interval](#) is the time it takes for the WDT to decrements from the [WDT Load Register](#) value to 0000h. The [WDT Count Register](#) value takes 1.007ms to decrement by 1 count.

### 16.6.5 WDT STALL OPERATION

The WDT has several events that can cause the WDT STALL. When a WDT STALL event is asserted, the WDT stops decrementing, and the WDT enters a low power state. When a WDT STALL event is de-asserted, the counter resumes decrementing from the count at which it stopped.

The WDT STALL feature has been implemented for convenience. If the system designer chooses not to utilize the WDT STALL feature, the WDT defaults with the WDT STALL feature disabled.

There are three Stall inputs to the WDT: [HT\\_Active](#), [WT\\_Active](#), and [JTAG\\_Active](#), corresponding to the Hibernation Timer, the Week Alarm Timer, & the J-TAG interface being active. The Stall inputs have individual enable bits: [HT\\_STALL\\_EN](#), [WT\\_STALL\\_EN](#), [JTAG\\_STALL\\_EN](#) bits in the [WDT Control Register on page 354](#).

**Note 16-2** Only a single instance of the Hibernation Timer ([Hibernation Timer.0 on page 391](#)) is routed to the [HT\\_Active](#) stall input of the [Watchdog Timer Interface](#).

# MEC1618/MEC1618i

**TABLE 16-3: WDT STALL EVENT BEHAVIOR**

WDT Stall Input  (Activity Indicator)	WDT Control Register on page 354		WDT Behavior	WDT Event Output
	STALL_EN Bit	WDT Enable Bit		
X	X	0	Counter is reset and not active. Clock source to counter is gated to save power.	0
X	0	1	Count is active. If counter > 0000h	0
0	1	1		
X	X	1	Count is decremented to 0000h	1
1	1	1	Counter is not active. Clock source to counter is gated to save power.	0

**Note 16-3** When the counter reaches 0000h it wraps to the preload value and starts counting down again. This creates a pulse on the [WDT Event](#) output.

## 16.7 Instance Description

There is one instance of the [Watchdog Timer Interface](#) block implemented in the MEC1618/MEC1618i.

Each instance of the [Watchdog Timer Interface](#) has its own Logical Device Number, and Base Address as indicated in [Table 16-4](#).

**TABLE 16-4: Watchdog Timer Interface BASE ADDRESS TABLE**

Watchdog Timer Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
<a href="#">Watchdog Timer</a>	1h	<a href="#">F0_0400h</a>

The [Table 16-5](#) is a register summary for one instance of the [Watchdog Timer Interface](#). Each EC address is indicated as an SPB Offset from its AHB address.



**TABLE 16-5: Watchdog Timer Interface REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
WDT Load Register	00h	0	R/W	
		1		
WDT Control Register	04h	0	R/W	
WDT Kick Register	08h	0	W	
WDT Count Register	0Ch	0	R	
		1		

**Note:** All Registers listed in Table 16-5 are powered by VTR and reset by nSYS\_RST.

**Note 16-4** All register are clocked by the EC Bus Clock.

## 16.8 Detailed Register Descriptions

### 16.8.1 WDT LOAD REGISTER

**TABLE 16-6: WDT LOAD REGISTER**

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	00h				16-bit			EC SIZE	
POWER	VTR				FFFFh			nSYS_RST DEFAULT	
BUS	EC SPB								
BIT	D15	D14	D13	<div></div>	<div></div>	D2	D1	D0	
HOST TYPE	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	WDT load[15:0]								

#### WDT LOAD[15:0]

Writing this field reloads the Watch Dog Timer counter.

WDT Load Register can be programmed only when WDT Enable = '0'.

'0' is not a valid load value.

To verify that load has taken place, it is recommended that software polls the WDT Count Register until its value reflects that of the new load value.

# MEC1618/MEC1618i

## 16.8.2 WDT CONTROL REGISTER

TABLE 16-7: WDT CONTROL REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	04h			8-bit			EC SIZE	
POWER	VTR			00h (Note 16-5)			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/WC	R/W
BIT NAME	Reserved			JTAG STALL_ EN	WT STALL_ EN	HT STALL_ EN	WDT Status	WDT Enable

**Note 16-5** The default for the [WDT Control Register](#) changes depending on the state of the [WDT Status](#) bit.

### WDT ENABLE

**Note:** The default of the WDT is inactive.

In [WDT Operation](#), the WDT is activated by the sequence of operations defined in [Section 16.6.1, "WDT Activation Mechanism"](#) and deactivated by the sequence of operations defined in [Section 16.6.2, "WDT Deactivation Mechanism"](#). In [WDT STALL Operation](#), hardware may be enabled to automatically activate and deactivate the WDT.

### WDT STATUS

[WDT Status](#) is set by hardware if the last reset of MEC1618/MEC1618i was caused by an underflow of the WDT. See [Section 16.6.3, "WDT Reload Mechanism," on page 351](#) for more information.

This bit must be cleared by the EC firmware writing a '1' to this bit. Writing a '0' to this bit has no effect.

### JTAG STALL\_EN

This bit is used to enable the [JTAG\\_Active](#) (JTAG\_RST# pin not asserted) [WDT STALL Operation on page 351](#).

0= [JTAG\\_Active WDT STALL Operation](#) not enabled

1= [JTAG\\_Active WDT STALL Operation](#) enabled

### WT STALL\_EN

This bit is used to enable the [WT\\_Active](#) (Week Timer) [WDT STALL Operation on page 351](#).

0= [WT\\_Active WDT STALL Operation](#) events not enabled

1= [WT\\_Active WDT STALL Operation](#) events enabled.

### HT STALL\_EN

This bit is used to enable the [HT\\_Active](#) (Hibernation Timer) [WDT STALL Operation on page 351](#).

0= [HT\\_Active WDT STALL Operation](#) events not enabled

1= [HT\\_Active WDT STALL Operation](#) events enabled.

## 16.8.3 WDT KICK REGISTER

TABLE 16-8: WDT KICK REGISTER

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	08h				8-bit			EC SIZE	
POWER	VTR				n/a			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	
EC TYPE	W	W	W	W	W	W	W	W	
BIT NAME	Kick								

**KICK**

The [WDT Kick Register](#) is a strobe. Reads of the [WDT Kick Register](#) return 0.

Writes to the [WDT Kick Register](#) cause the WDT to reload the [WDT Load Register](#) value and start decrementing when the [WDT Enable](#) bit in the [WDT Control Register](#) is set to '1'. When the [WDT Enable](#) bit in the [WDT Control Register](#) is cleared to '0', writes to the [WDT Kick Register](#) have no effect.

## 16.8.4 WDT COUNT REGISTER

TABLE 16-9: WDT COUNT REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	0Ch			16-bit			EC SIZE	
POWER	VTR			FFFFh			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D15	D14	D13	■	■	D2	D1	D0
HOST TYPE	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	WDT COUNT[15:0]							

**WDT COUNT[15:0]**

This read-only register provide the current WDT count.

# MEC1618/MEC1618i

## 17.0 HDMI-CEC INTERFACE CONTROLLER

### 17.1 Overview

This chapter describes an implementation for an HDMI Consumer Electronics Control (CEC) Interface as defined in [References \[1\]](#).

The [HDMI-CEC Interface Controller](#) described in this document handles in hardware the transfer of CEC data across the physical interface; including, all initiator/follower data/framing bit timing, logical address decoding, contention detection/[Lost Arbitration](#), [Line Error Handling](#) and message block acknowledgement.

Firmware is required to configure physical addressing and the CEC high level protocol, which is beyond the scope of this document.

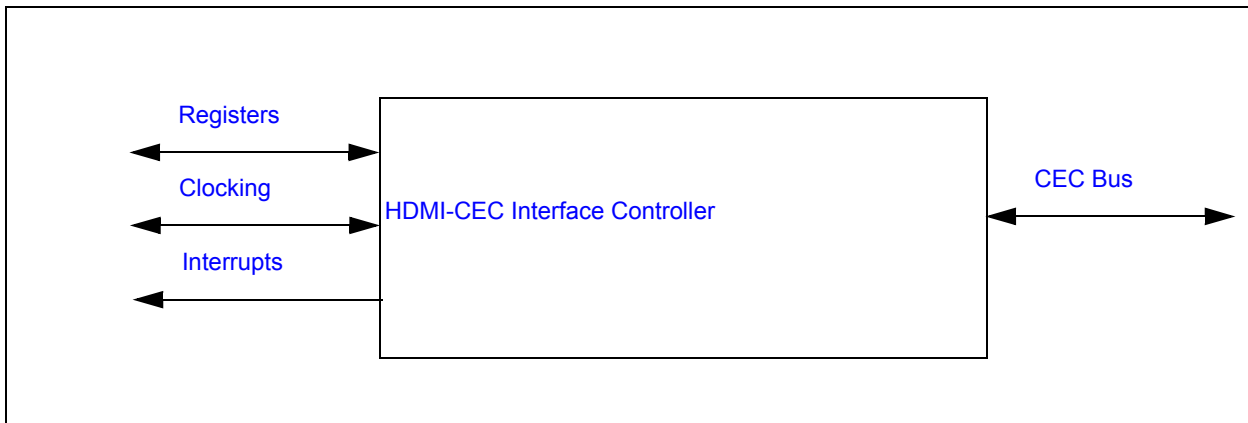
### 17.2 References

1. High-Definition Multimedia Interface Specification Version 1.3a, Supplement 1 CEC, November 10, 2006.

**Note:** Italicized text in this chapter typically refers to content defined in [References \[1\]](#).

### 17.3 Block Diagram

FIGURE 17-1: [HDMI-CEC Interface Controller Block Diagram](#)



### 17.4 CEC Bus

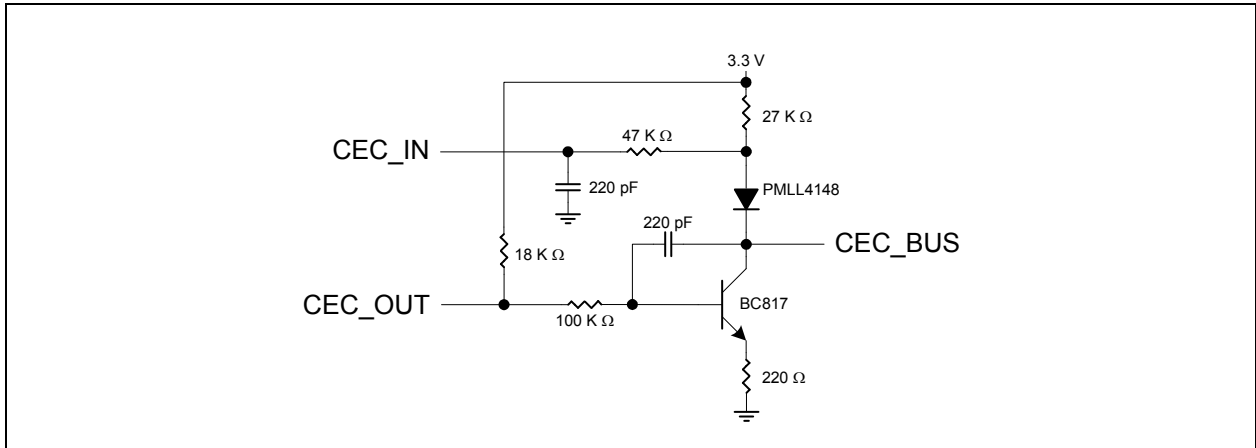
#### 17.4.1 OVERVIEW

The [CEC Bus](#) is a single wire bus that provides high-level control functions between all of the various audiovisual products in a user's environment (see 3, "Overview" in [References \[1\]](#)). The MEC1618/MEC1618i [CEC Bus](#) interface includes CEC\_IN and CEC\_OUT pins as defined in MEC1618/MEC1618i Pin Configuration.

## 17.4.2 EXTERNAL INTERFACE

Figure 17-2 illustrates an example of an external CEC Bus interface network. CEC\_IN and CEC\_OUT pin characteristics, including polarity, direction and buffer type, can be programmed using the GPIO Interface Pin Control Registers (Table 17-1).

**FIGURE 17-2: CEC Bus APPLICATION EXAMPLE**



**TABLE 17-1: EXTERNAL INTERFACE PIN CONFIGURATION**

Signal Function	Direction	Output Buffer Type	Polarity
CEC_IN	INPUT	X	NON-INVERTED
CEC_OUT	OUTPUT	OPEN DRAIN	INVERTED

## 17.4.3 FILTERING

When enabled, the HDMI-CEC Interface Controller filters the CEC\_IN pin as defined in Table 17-2. Filtering is enabled using the FILTEN bit in the CEC Control Register.

**TABLE 17-2: HDMI-CEC INTERFACE INPUT FILTER**

	Parameter	Symbol	Value			Units	Notes
			Min	Typ	Max		
1.	Filtered Pulse Width	$t_{FPW}$	100	–	–	ns	

# MEC1618/MEC1618i

## 17.4.4 PAD

Table 17-3 is taken from CEC 4, “Electrical Specification” in [References \[1\]](#). These requirements are to be satisfied using an external electrical network.

**TABLE 17-3: ELECTRICAL SPECIFICATIONS**

	Parameter	Symbol	Value			Units	Notes
			Min	Typ	Max		
1.	Leakage Current in Powered-off State	$I_{IL}$	–	–	1.8	$\mu A$	<a href="#">Note 17-1</a>
2.	Output Voltage Logic ‘0’	$V_{OL}$	0.0	–	0.6	V	<a href="#">Note 17-3</a>
3.	Output Voltage Logic ‘1’	$V_{OH}$	2.5	–	3.63	V	–
4.	Output Current Logic ‘0’	$I_{OL}$	–	–	20	mA	–
5.	High to Low Input Voltage Threshold Logic ‘0’	$V_{IL}$	–	–	0.8	V	–
6.	Low to High Input Voltage Threshold Logic ‘1’	$V_{IH}$	–	–	2.0	V	–
7.	Input Hysteresis	$V_{HYS}$	–	400	–	mV	<a href="#">Note 17-2</a>
8.	Rise Time (10% to 90%)	$t_R$	–	–	250	$\mu s$	–
9.	Fall Time (10% to 90%)	$t_F$	–	–	50	$\mu s$	–
10.	Internal Device Pull-up	$R_{PU}$	23.4	26	28.6	K Ohms	–
11.	Bus Capacitance	$C_{LOAD}$	–	–	7200	pF	<a href="#">Note 17-4</a>

**Note 17-1** This effectively requires that the internal pull-up circuit shall be disconnected from the CEC line when the device is off. For example, this can be implemented by connecting an isolating diode between the CEC input pin and the internal pull-up circuit, such that diode is reverse-biased in the off state with an external device pulling-up the CEC line.

**Note 17-2** Input hysteresis is handled by the pad.

**Note 17-3** During transition from Logic ‘1’ to Logic ‘0’ a negative overshoot with maximum 300mV and up to 150 $\mu s$  duration is allowed.

**Note 17-4** The device shall remain within specification under the full-range of load conditions.

## 17.5 Interrupts

If any of the [Interrupt Interface](#) (Table 17-4) status bits are asserted when the corresponding enable bit is asserted, an external interrupt is asserted.

**Note:** [CEC\\_INT](#) in the [GIRQ16 Source Register](#) is the [HDMI-CEC Interface Controller](#) interrupt source bit in the MEC1618/MEC1618i (see [Section 15.0, "EC Interrupt Aggregator,"](#) on page 296).

**TABLE 17-4: INTERRUPT INTERFACE**

	Status Bits		Enable Bits	
	Name	Register	Name	Register
1.	<a href="#">IFDONE</a>	<a href="#">Initiator Status Register</a>	<a href="#">IFDONE_EN</a>	<a href="#">Initiator Control Register</a>
2.	<a href="#">IFE</a>		<a href="#">IFE_EN</a>	
3.	<a href="#">FFDONE</a>	<a href="#">Follower Status Register</a>	<a href="#">FFDONE_EN</a>	<a href="#">Follower Control Register</a>
4.	<a href="#">FDR</a>		<a href="#">FDR_EN</a>	
5.	<a href="#">FFF</a>		<a href="#">FFF_EN</a>	

## 17.6 Power

### 17.6.1 INTERFACE

The [HDMI-CEC Interface Controller](#) requires a single power plane (VTR). [Power Management](#) can be controlled by the [Clocking](#) interface as described in [Section 17.6.2](#).

### 17.6.2 POWER MANAGEMENT

**Note:** The [CEC](#) bit in the [EC Blocks Sleep Enables Register 2](#) and the [EC Blocks Clock Required Status Register 2](#) are the sleep enable and clock required status bits for the [HDMI-CEC Interface Controller](#) in the MEC1618/MEC1618i (see [Section 5.0, "Power, Clocks, and Resets,"](#) on page 74).

**TABLE 17-5: HDMI-CEC Interface Controller Power Management**

<a href="#">ACTIVATE</a> ( <a href="#">Note 17-5</a> )	<a href="#">External SLEEP_EN Input</a> ( <a href="#">Note 17-6</a> )	<a href="#">Internal Idle</a> ( <a href="#">Note 17-7</a> )	<a href="#">Core Clock Required Status Output</a> ( <a href="#">Note 17-6</a> )	Mode	Power	Description
0	X	X	0	DISABLED	MINIMUM	The <a href="#">HDMI-CEC Interface Controller</a> is disabled by firmware and the core clock is gated 'off' internally. Note: it is up to the host to guarantee that the block is not in use before the internal enable bit is asserted.

# MEC1618/MEC1618i

**TABLE 17-5: HDMI-CEC Interface Controller Power Management (CONTINUED)**

ACTIVATE (Note 17-5)	External SLEEP_EN Input (Note 17-6)	Internal Idle (Note 17-7)	Core Clock Required Status Output (Note 17-6)	Mode	Power	Description
1	0	NOT IDLE	1	FULL POWER	MAXIMUM	FULL POWER mode identifies the normal operation mode where the block is neither disabled by firmware nor commanded to sleep by the <a href="#">Clocking</a> interface.
		IDLE				
	1	NOT IDLE		PREPARING TO SLEEP		A sleep command has been asserted but the core clock is still required because the block is not idle.
		IDLE	0	SLEEPING	MINIMUM	A sleep command has been asserted, the block is idle and the core clocks are stopped. In the SLEEPING mode, normal operation cannot be resumed until the external sleep command is not asserted.

**Note 17-5** The [ACTIVATE](#) bit is in the [CEC Control Register](#).

**Note 17-6** The external sleep enable input and clock required status output are part of the [Clocking](#) interface.

**Note 17-7** The INTERNAL IDLE state in part depends upon the state of the [IDLE](#) bit in the [Initiator Status Register](#).

## 17.7 Clocking

**TABLE 17-6: [Clocking](#) SIGNAL INTERFACE**

Signal Name	Description	Reference
<a href="#">MCLK_DIV203_EN</a>	Core Clock	<a href="#">Table 17-7, "Core Clock Timing"</a>
<a href="#">CEC_SLEEP_ENABLE</a>	Sleep Enable Command Input	<a href="#">Section 17.6.2, "Power Management," on page 359</a>
<a href="#">CEC_CLOCK_REQUIRED</a>	Clock Required Status Output	

**TABLE 17-7: CORE CLOCK TIMING**

Parameter	Symbol	Min	Typ	Max	Unit
Core Clock Frequency	$f_{\text{CLOCK}}$	90	100	110	kHz



## 17.8 Functional Description

### 17.8.1 OVERVIEW

The [HDMI-CEC Interface Controller](#) includes an [Initiator Interface](#) and a [Follower Interface](#) that function independently within the constraints defined in the subsections that follow. Data transfer buffering in each interface include a 9-bit wide x 16 deep FIFO and a shift register. Each interface can generate interrupts as described in [Section 17.5, "Interrupts," on page 359](#) and the [Initiator Interface](#) hardware senses the [Bus Idle Condition](#) to initiate message frame transfers.

### 17.8.2 INITIATOR INTERFACE

#### 17.8.2.1 Overview

The [Initiator Interface](#) describes data transfers in the [HDMI-CEC Interface Controller](#) as an initiator when the [ACTIVATE](#) bit is asserted. The [Initiator Interface](#) is disabled when the [ACTIVATE](#) bit is not asserted.

Specific accesses to the [Initiator Data Register](#), [Initiator Control Register](#) and [Initiator Status Register](#) are required for successful [Message Initiation](#). The [Initiator Interface](#) hardware is also capable of detecting [Lost Arbitration](#).

Note that enforcing CEC message frame size constraints in the [HDMI-CEC Interface Controller](#) is the responsibility of firmware because the [Initiator Interface](#) can send any number of message blocks in a frame.

#### 17.8.2.2 Message Initiation

[Initiator Interface](#) begins a message frame transaction when the [START](#) bit in the [Initiator Control Register](#) is asserted, the [IFE](#) bit is not asserted, and hardware detects a [Bus Idle Condition](#) as described in [Section 17.8.4 on page 362](#).

**APPLICATION NOTE:** Recommended [Message Initiation](#) procedure:

- Assert the [IFLUSH](#) bit to clear the [Initiator Interface](#) FIFO,
- Write all message data to the [Initiator Interface](#) FIFO using the [Initiator Data Register](#),
- Assert the [START](#) bit.

The message frame transaction is complete, or terminated because of an error (see [Section 17.8.2.4, "Error Handling," on page 361](#)), when the [IFDONE](#) bit in the [Initiator Status Register](#) is asserted.

To transfer message frames larger than 16 blocks, additional message data can be written to the [Initiator Data Register](#) whenever the [IFE](#) interrupt is asserted. In this case, data must be written within 20 milliseconds to avoid an underrun error.

#### 17.8.2.3 Lost Arbitration

[Lost Arbitration](#) occurs when the [Initiator Interface](#) detects bus contention during CEC line arbitration, in which case the [LAB](#) bit in the [Initiator Status Register](#) is asserted. As a result of [Lost Arbitration](#) the transfer is aborted by the [Initiator Interface](#) and the task of decoding the rest of the message header is passed to the [Follower Interface](#). If the destination logical address matches one of the addresses configured in the [Claimed Logical Addresses Register](#), the [Follower Interface](#) continues decoding the rest of the message frame.

**APPLICATION NOTE:** following [Lost Arbitration](#), the [Initiator Interface](#) firmware is responsible for flushing the [Initiator Interface](#) FIFO, and re-initiating the transaction (see also [Section 17.8.2.4, "Error Handling," on page 361](#)).

#### 17.8.2.4 Error Handling

If at any time during an [Initiator Interface](#) message transfer the [UNDRN](#), [ACKERR](#), [LAB](#), or [CE](#) bits in the [Initiator Status Register](#) are asserted, the transfer is aborted by the initiator and the [IFDONE](#) bit is asserted. Note that in all cases the transfer is terminated, except during [Lost Arbitration](#) as described in [Section 17.8.2.3](#).

Following message frame errors as defined above, the [IFLUSH](#) bit in the [Initiator Control Register](#) may need to be asserted to delete residual data in the [Initiator Interface](#) FIFO.

## 17.8.3 FOLLOWER INTERFACE

### 17.8.3.1 Overview

The [Follower Interface](#) in the [HDMI-CEC Interface Controller](#) responds to data transfers when the [ACTIVATE](#) bit is asserted and the [Claimed Logical Addresses Register](#) is greater than zero. The [Follower Interface](#) is disabled when the [ACTIVATE](#) bit is not asserted; if the [ACTIVATE](#) bit is asserted, no message frames will be decoded when the [Claimed Logical Addresses Register](#) is zero.

Specific accesses to the [Follower Data Register](#), [Follower Control Register](#) and [Follower Status Register](#) are required for successful [Follower Message Response](#). The [Follower Interface](#) hardware is also capable of [Line Error Handling](#) as described in [Section 17.8.3.3](#). The [Follower Interface](#) can respond to frames that contain any number of message blocks.

### 17.8.3.2 Follower Message Response

#### 17.8.3.2.1 Overview

When the [HDMI-CEC Interface Controller](#) is activated as described in [Section 17.8.3.1](#), the [Follower Interface](#) responds to all signaling on the [CEC Bus](#) when the [Initiator Interface](#) is not active, or following [Lost Arbitration](#). The [Follower Interface](#) supports [Directed Messages](#) and [Broadcast Messages](#).

When the [FFDONE](#) bit in the [Follower Status Register](#) is asserted, the [Follower Interface](#) has successfully decoded a single message frame and placed it in the [Follower Interface](#) FIFO. It is the responsibility of firmware to read the message data using the [Follower Data Register](#) before another message arrives. If the [Follower Interface](#) tries to write received message data to the FIFO when the [FFF](#) bit is asserted, an overrun error occurs (see [Section 17.9.7.1, "OVRN," on page 369](#)), the current data block is negatively acknowledged, and the message is terminated.

In addition to detecting overrun errors, the [Follower Interface](#) implements [Line Error Handling](#) for message blocks that cannot be decoded successfully, as described in [Section 17.8.3.3 on page 362](#).

#### 17.8.3.2.2 Directed Messages

When the [Follower Interface](#) decodes a message header addressed to one of the logical device addresses configured in the [Claimed Logical Addresses Register](#) that are less than 15, the message data blocks are decoded until the [EOM](#) bit is detected. These are [Directed Messages](#).

Acknowledge bit handling for [Directed Messages](#) is different than for [Broadcast Messages](#) (see [Section 17.9.6.4, "ACK-ERR," on page 368](#)).

#### 17.8.3.2.3 Broadcast Messages

When the [Follower Interface](#) decodes a message header addressed to logical device address 15, which like directed message addresses is configured in the [Claimed Logical Addresses Register](#), the message data blocks are decoded until the [EOM](#) bit is detected. These are [Broadcast Messages](#).

Acknowledge bit handling for [Broadcast Messages](#) is different than for [Directed Messages](#) (see [Section 17.9.6.4, "ACK-ERR," on page 368](#)).

### 17.8.3.3 Line Error Handling

Whenever the [Follower Interface](#) detects spurious pulses on the [CEC Bus](#) while decoding a message frame, the [BERR](#) bit in the [Follower Status Register](#) is asserted and the initiator is notified that a potential error has occurred. Spurious pulses and initiator notification are defined in [CEC 7.4, "CEC Line Error Handling"](#) in [References \[1\]](#).

[Line Error Handling](#) in the [HDMI-CEC Interface Controller](#) also includes bus time-out detection, which is indicated by the [BTO](#) bit in the [Follower Status Register](#). In all cases, the [Follower Interface](#) response to [Line Error Handling](#) includes terminating message frame reception.

## 17.8.4 BUS IDLE CONDITION

The [Bus Idle Condition](#) is uniquely determined for each device on the [CEC Bus](#) and depends upon the state of the [CEC Bus](#) itself and the device system state. In the [HDMI-CEC Interface Controller](#), the [Bus Idle Condition](#) is considered 'asserted' when the [IDLE](#) bit in the [Initiator Status Register](#) is '1,' which occurs according to the signal free time as defined in [CEC 9, "CEC Arbitration"](#) in [References \[1\]](#).

The start of a message frame by the [HDMI-CEC Interface Controller](#) as an initiator (see [Section 17.8.2, "Initiator Interface," on page 361](#)) requires that the [Bus Idle Condition](#) be asserted, which is determined by device hardware as described in [Table 17-8](#).

**TABLE 17-8:** Bus Idle Condition ASSERTION

	SFT (Note 17-8)	Factor
1.	3	The previous message was send by the internal initiator and was not successful.
2.	5	RESET (including a hardware reset), or the previous message was send by an external initiator.
3.	7	The previous message was send by the internal initiator and was successful.

**Note 17-8** The Signal Free Time (SFT) is determined by the specified number, or greater, of nominal data bit periods where the CEC Bus remains high following the end of the last message.

For example, if the HDMI-CEC Interface Controller sends a successful message frame (A) it will not be eligible to send a subsequent frame for 16.8 ms (2.4 ms x 7) nominal following the end of frame A. In this same scenario, if an external initiator begins a message frame (B) 12 ms (2.4 ms x 5) after the end of frame A, the internal HDMI-CEC Interface Controller initiator will then be eligible to send a subsequent frame 12 ms after the end of frame B.

In all cases, the IDLE bit is de-asserted in the same bit period as a valid (or invalid) start bit. The IDLE bit does not generate an interrupt.

Immediately following the end of an Initiator Interface message transfer, successful or otherwise, the Bus Idle Condition can be temporarily overridden using the SFT5 bit in the CEC Control Register (see Section 17.9.2, "CEC Control Register," on page 364).

## 17.9 Registers

### 17.9.1 SUMMARY

**TABLE 17-9:** HDMI-CEC Interface Controller REGISTER SUMMARY

Register Name	Mnemonic	EC Interface			Notes
		SPB Offset	Byte Lane	EC Type	
CEC Control Register	CCR	00h	0	R/W	–
Claimed Logical Addresses Register	CLAR	04h	1-0	R/W	–
Initiator Data Register	IDR	08h	1-0	R/W	–
Follower Data Register	FDR	0Ch	1-0	R	–
Initiator Status Register	ISTR	10h	0	R/WC	Note 17-9
Follower Status Register	FSTR	14h	0	R/WC	Note 17-9
Initiator Control Register	ICR	18h	0	R/W	–
Follower Control Register	FCR	1Ch	0	R/W	–
MCHP Reserved Register	MCHPRSR	20h	3-0	R/W	–

**Note 17-9** R/WC type bits must be written with a '1' to clear the bit.

# MEC1618/MEC1618i

## 17.9.2 CEC CONTROL REGISTER

TABLE 17-10: CEC Control Register

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	00h				8-bit			EC SIZE	
POWER	VTR				00h			DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R/W	R/W	R/W	R/W	
BIT NAME	Reserved				SFT5	FILTEN	RESET	ACTIVATE	

### 17.9.2.1 ACTIVATE

When the **ACTIVATE** bit is asserted ('1'), the **HDMI-CEC Interface Controller** is enabled for normal operation, subject to the constraints of the **Power Management** interface as defined in [Section 17.6.2 on page 359](#).

When the **ACTIVATE** bit is not asserted (default), the **HDMI-CEC Interface Controller** is disabled and in the lowest power consumption state. The **ACTIVATE** bit does not affect the **Registers**.

### 17.9.2.2 RESET

When the **RESET** bit is asserted, the hardware state machines and **Registers**, except for the **ACTIVATE** bit, are reset to their default state. The **RESET** bit is cleared by hardware within one register access cycle.

**APPLICATION NOTE:** If there is an error condition that requires an **HDMI-CEC Interface Controller RESET**, the reset operation should be completed within the minimum bus idle time to avoid unnecessary message NACKs.

### 17.9.2.3 FILTEN

When the **FILTEN** bit is asserted ('1'), the **Filtering** on the **CEC Bus** is enabled. When **FILTEN** is not asserted (default), **Filtering** is disabled. See [Section 17.4.3, "Filtering," on page 357](#).

### 17.9.2.4 SFT5

When the **SFT5** bit is asserted ('1'), the Signal Free Time (SFT) is changed to '5' only once (see [Table 17-8, "Bus Idle Condition Assertion," on page 363](#)).

The **SFT5** bit is cleared, and normal SFT behavior resumes when the **Bus Idle Condition** is asserted.

### 17.9.2.5 Reserved

**Reserved** bits cannot be written and return '0' when read.

## 17.9.3 CLAIMED LOGICAL ADDRESSES REGISTER

**TABLE 17-11:** [Claimed Logical Addresses Register](#)

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	04h			16-bit			EC SIZE	
POWER	VTR			0000h			DEFAULT	
BUS	EC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	ADDR15	ADDR14	ADDR13	ADDR12	ADDR11	ADDR10	ADDR9	ADDR8
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	ADDR7	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0

When any bit in the [Claimed Logical Addresses Register](#) is asserted ('1'), [Directed Messages](#) and [Broadcast Messages](#) addressed to that logical address are claimed by the [Follower Interface](#). The [Claimed Logical Addresses Register](#) default effectively disables the [Follower Interface](#).

# MEC1618/MEC1618i

## 17.9.4 INITIATOR DATA REGISTER

### 17.9.4.1 Description

The [Initiator Data Register](#) is two bytes wide and is used to write the [Initiator Interface](#) FIFO for CEC header and data block values. For 8-bit access cycles, the most significant byte of the [Initiator Data Register](#) must be written first. See also [CEC 6.1, "Header/Data Block Description"](#) in [References \[1\]](#).

Reads of the [Initiator Data Register](#) return zero.

**TABLE 17-12:** [Initiator Data Register](#)

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	08h				16-bit			EC SIZE	
POWER	VTR				0000h			DEFAULT	
BUS	EC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	W	
BIT NAME	Reserved							EOM	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	W	W	W	W	W	W	W	W	
BIT NAME	INITIATOR_DATA								

### 17.9.4.2 EOM

The [EOM](#) bit is used for the End of Message bit in header/data blocks in CEC message frames. When the [EOM](#) bit is '0,' more data blocks follow; a '1' specifies that the message is complete.

### 17.9.4.3 INITIATOR\_DATA

The [INITIATOR\\_DATA](#) register is used for the Information bits in header/data blocks in CEC message frames.

### 17.9.4.4 Reserved

[Reserved](#) bits cannot be written and return '0' when read.

## 17.9.5 FOLLOWER DATA REGISTER

### 17.9.5.1 Description

The [Follower Data Register](#) is two bytes wide and is used to read the [Follower Interface](#) FIFO for CEC header and data block values. For 8-bit access cycles, the most significant byte of the [Follower Data Register](#) must be read first. See also [CEC 6.1, "Header/Data Block Description"](#) in [References \[1\]](#).

Reads of the [Follower Data Register](#) when the [FFNE](#) bit in the [Follower Status Register](#) is not asserted ('0') return undefined data.

**TABLE 17-13: [Follower Data Register](#)**

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	0Ch				16-bit			EC SIZE	
POWER	VTR				0000h			DEFAULT	
BUS	EC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved							EOM	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	FOLLOWER_DATA								

### 17.9.5.2 EOM

The [EOM](#) bit is the received End of Message bit from header/data blocks in CEC message frames. When the [EOM](#) bit is '0,' more data blocks follow; a '1' specifies that the message is complete.

### 17.9.5.3 FOLLOWER\_DATA

The [FOLLOWER\\_DATA](#) register is used for the received Information bits from header/data blocks in CEC message frames.

### 17.9.5.4 Reserved

[Reserved](#) bits cannot be written and return '0' when read.

# MEC1618/MEC1618i

## 17.9.6 INITIATOR STATUS REGISTER

TABLE 17-14: Initiator Status Register

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	10h				8-bit			EC SIZE	
POWER	VTR				21h			DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R/WC	R	R/WC	R/WC	R/WC	R/WC	R	
BIT NAME	Reserved	IFDONE	IFE	CE	ACKERR	UNDRN	LAB	IDLE	

### 17.9.6.1 IDLE

When the **IDLE** bit is asserted ('1'), the **CEC Bus** is idle (**Bus Idle Condition**). When the **IDLE** bit is not asserted ('0'), the **CEC Bus** is busy.

See [Section 17.8.4, "Bus Idle Condition," on page 362](#) for details regarding the behavior of the **IDLE** bit.

### 17.9.6.2 LAB

The **LAB** (**Lost Arbitration**) bit is asserted when bus contention is detected by the **Initiator Interface** during CEC line arbitration (see *CEC 9, "CEC Arbitration"* in [References \[1\]](#)).

See also [Section 17.9.6.7, "IFDONE," on page 369](#).

### 17.9.6.3 UNDRN

The **UNDRN** (**Initiator Underrun**) bit is asserted ('1') when the initiator shift register requires data from the **Initiator Interface** FIFO and the **IFE** bit is asserted. See also [Section 17.9.6.7, "IFDONE," on page 369](#).

### 17.9.6.4 ACKERR

The **ACKERR** (**Acknowledge Error**) bit is asserted ('1') when the follower NACKs an acknowledge bit, both for **Directed Messages** and **Broadcast Messages**.

For example, in **Directed Messages** if the follower acknowledge bit response is '1,' the **ACKERR** bit is '1;' in **Broadcast Messages** if a follower acknowledge bit response is '0,' the **ACKERR** bit is '1.'

See also [Section 17.9.6.7, "IFDONE," on page 369](#).

### 17.9.6.5 CE

The **CE** (**Contention Error**) bit is asserted ('1') when bus contention is detected by the **Initiator Interface** at any point following CEC line arbitration. This is likely to be a consequence of CEC Line Error Handling (see *CEC 7.4, "CEC Line Error Handling"* in [References \[1\]](#)).

See also [Section 17.9.6.7, "IFDONE," on page 369](#).

### 17.9.6.6 IFE

The **IFE** (**Initiator FIFO Empty**) bit is asserted ('1') when the **Initiator Interface** retrieves the last data entry from the FIFO. The **IFE** bit can generate a level sensitive interrupt as described in [Section 17.5, "Interrupts," on page 359](#).



## 17.9.6.7 IFDONE

The **IFDONE** (Initiator Frame Done) bit is asserted ('1') when a message block with the **EOM** bit asserted ('1') has been transferred by the **Initiator Interface**, or the message frame has been terminated because an error occurred as defined by the **LAB**, **UNDRN**, **ACKERR**, and **CE** bits.

The **IFDONE** bit can generate a level sensitive interrupt as described in [Section 17.5, "Interrupts," on page 359](#).

## 17.9.6.8 Reserved

**Reserved** bits cannot be written and return '0' when read.

## 17.9.7 FOLLOWER STATUS REGISTER

**TABLE 17-15: Follower Status Register**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	14h			8-bit			EC SIZE	
POWER	VTR			00h			DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/WC	R/WC	R	R	R/WC	R/WC	R/WC
BIT NAME	Reserved	FFDONE	FDR	FFF	FFNE	BTO	BERR	OVRN

### 17.9.7.1 OVRN

The **OVRN** (Follower Overrun) bit is asserted ('1') when data from the follower shift register is transferred to the **Follower Interface** FIFO when the **FFF** bit is asserted. See also [Section 17.8.3, "Follower Interface," on page 362](#).

See also [Section 17.9.7.7, "FFDONE," on page 370](#).

### 17.9.7.2 BERR

The **BERR** (Bus Error Detected) bit is asserted ('1') when the follower detects spurious pulses on the **CEC Bus** as defined in [Section 17.8.3.3, "Line Error Handling," on page 362](#).

See also [Section 17.9.7.7, "FFDONE," on page 370](#).

### 17.9.7.3 BTO

The **BTO** (Bus Time Out Detected) bit is asserted ('1') when the follower detects that the **CEC Bus** is held high too long (see [Section 17.8.3.3, "Line Error Handling," on page 362](#)).

See also [Section 17.9.7.7, "FFDONE," on page 370](#).

### 17.9.7.4 FFNE

The **FFNE** (Follower FIFO Not Empty) bit is asserted ('1') when there is data in the **Follower Interface** FIFO. The **FFNE** bit is not asserted ('0') when the **Follower Interface** FIFO is empty.

### 17.9.7.5 FFF

The **FFF** (Follower FIFO Full) bit is asserted ('1') when there is no room in the **Follower Interface** FIFO for more data. The **FFF** bit can generate a level sensitive interrupt as described in [Section 17.5, "Interrupts," on page 359](#).

# MEC1618/MEC1618i

## 17.9.7.6 FDR

The **FDR** (Follower Data Ready) bit is asserted ('1') whenever received data is written to the **Follower Interface** FIFO. **FDR** bit can generate a level sensitive interrupt as described in

## 17.9.7.7 FFDONE

The **FFDONE** (Follower Frame Done) bit is asserted ('1') when a message block with the **EOM** bit asserted ('1') has been received by the **Follower Interface**, or the message frame has been terminated because an error occurred as defined by the **OVRN**, **BERR**, and **BTO** bits.

The **FFDONE** bit can generate a level sensitive interrupt as described in [Section 17.5, "Interrupts," on page 359](#).

## 17.9.7.8 Reserved

**Reserved** bits cannot be written and return '0' when read.

## 17.9.8 INITIATOR CONTROL REGISTER

**TABLE 17-16:** [Initiator Control Register](#)

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	18h				8-bit			EC SIZE	
POWER	VTR				00h			DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R/W	R/W	R	R	R	R/W	R/W	
BIT NAME	Reserved	IFDONE_EN	IFE_EN	Reserved			START	IFLUSH	

### 17.9.8.1 IFLUSH

When the **IFLUSH** bit is asserted ('1'), data in the **Initiator Interface** FIFO is cleared. When set, the **IFLUSH** bit is automatically cleared by hardware within one register access cycle. See also [Section 17.8.2.4, "Error Handling," on page 361](#).

The **IFLUSH** bit can be used to terminate an **Initiator Interface** message frame transfer.

### 17.9.8.2 START

When the **START** bit is asserted ('1'), the **Initiator Interface** begins a message frame transfer if the **IFE** bit is not asserted. If the **Initiator Interface** FIFO is empty when the **START** bit is asserted, the message frame transfer begins as soon as a write to the **Initiator Data Register** occurs. In both cases, the **START** bit is automatically cleared by hardware within one register access cycle.

The **START** bit only needs to be asserted once per message frame.

### 17.9.8.3 IFE\_EN

Enable bit for the **IFE** interrupt (see [Section 17.9.6.6, "IFE," on page 368](#)).

### 17.9.8.4 IFDONE\_EN

Enable bit for the **IFDONE** interrupt (see [Section 17.9.6.7, "IFDONE," on page 369](#)).

## 17.9.8.5 Reserved

Reserved bits cannot be written and return '0' when read.

## 17.9.9 FOLLOWER CONTROL REGISTER

TABLE 17-17: Follower Control Register

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	1Ch			8-bit			EC SIZE	
POWER	VTR			00h			DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R/W	R	R	R	R/W
BIT NAME	Reserved	FFDONE_EN	FDR_EN	FFF_EN	Reserved			FFLUSH

## 17.9.9.1 FFLUSH

When the FFLUSH bit is asserted ('1'), data in the Follower Interface FIFO is cleared. When set, the FFLUSH bit is automatically cleared by hardware within one register access cycle.

## 17.9.9.2 FFF\_EN

Enable bit for the FFF interrupt (see Section 17.9.7.5, "FFF," on page 369).

## 17.9.9.3 FDR\_EN

Enable bit for the FDR interrupt (see Section 17.9.7.6, "FDR," on page 370).

## 17.9.9.4 FFDONE\_EN

Enable bit for the FFDONE interrupt (see Section 17.9.7.7, "FFDONE," on page 370).

## 17.9.9.5 Reserved

Reserved bits cannot be written and return '0' when read.

## 17.9.10 MCHP RESERVED REGISTER

This register is reserved for use by Microchip. Writes to this register may produce undesirable results.

# MEC1618/MEC1618i

## 18.0 16-BIT TIMER INTERFACE

### 18.1 General Description

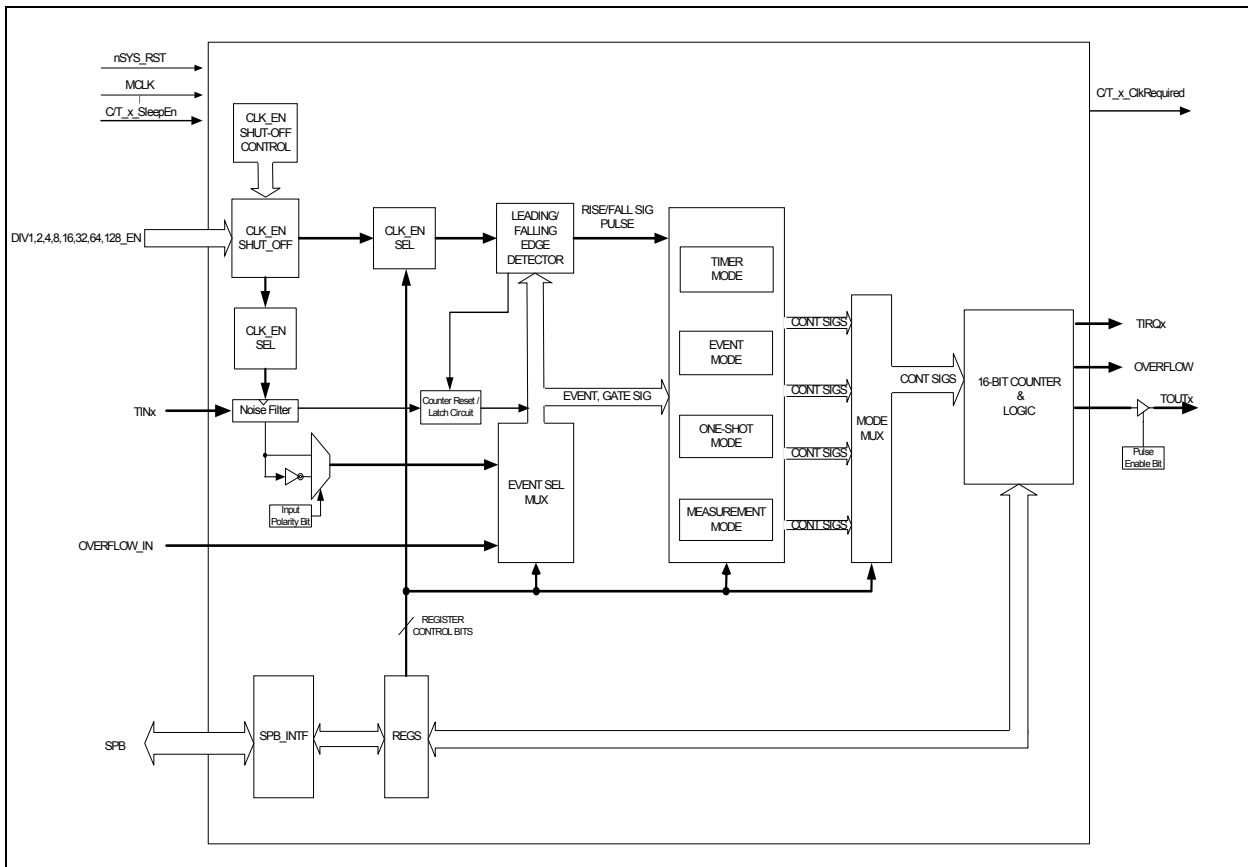
The MEC1618/MEC1618i [16-Bit Timer Interface](#) implements four 16-bit auto-reloading timer/counters. Each timer/counter is categorized as one of three types: General Purpose, Input-Only and Input/Output. All timer/counters have four modes of operation: Timer, One-Shot, Event and Measurement. In addition, each timer/counter can generate a unique wake-up interrupt to the EC. The clock for each timer/counter is derived from the system clock and can be divided down by a prescaler. Input-Only and Input/Output timers can also use an external input pin to clock or gate the counter. To aid operation in noisy environments the external input pin also has a selectable noise filter. If large counts are required, the output of each timer/counter can be internally connected to the next timer/counter.

The following section defines terms used in this chapter.

Term	Definition
Overflow	When the timer counter transitions from FFFFh to 0000h
Underflow	When the timer counter transitions from 0000h to FFFFh.
Timer Tick Rate	This is the rate at which the timer is incremented or decremented.

### 18.2 Block Diagram

FIGURE 18-1: BLOCK DIAGRAM FOR TIMER X



### 18.3 Signal List for Block Diagram

**TABLE 18-1: BLOCK DIAGRAM SIGNAL LIST DESCRIPTION**

Signal Name	Direction	Description
VTR POR	INPUT	<a href="#">nSYS_RST</a>
<a href="#">MCLK</a>	INPUT	20.27 MHz clock source to block.
DIV1,2,4,8,16,32,64,128_EN	INPUT	Clock Enables for supporting Filter and Timer frequencies.
TINx	INPUT	Timer x Input signal
TIRQx	OUTPUT	Timer x Interrupt Request
C/T_x_SleepEn, x=0-3	INPUT	Sleep Enable signals to counters 1-4
C/T_x_ClkRequired, x=0-3	INPUT	Clock required signals from counters 1-4
TOUTx	OUTPUT	Timer x Output signal
SPB_IF	I/O Bus	Bus used by microprocessor to access the registers in this block.

### 18.4 Timer Connections

For external inputs/outputs ([TINx/TOUTx](#)) to/from timers, please see [Section 3.6.15, "16-Bit Counter/Timer Interface," on page 63](#).

**TABLE 18-2: TIMER CASCADING DESCRIPTION**

Timer Name	Timer Type	Over-Flow/ Under-flow Input's Connection
Timer 0	General Purpose	from Timer 3
Timer 1	General Purpose	from Timer 0
Timer 2	General Purpose	from Timer 1
Timer 3	General Purpose	from Timer 2

**Note:** The cascading connections are independent of the [TINx/TOUTx](#) connections.

### 18.5 Power, Clocks and Reset

#### 18.5.1 POWER DOMAIN

This block is powered by the VTR power supply.

#### 18.5.2 CLOCKS

There is a clock enable input for each of the supported frequencies listed in [Table 18-12, "Timer Clock Frequencies," on page 387](#). Any of these enables may be selected for the Timer Clock Frequency. Independently, any of these clock frequencies may be selected for the filter clock via the FCLK[3:0] bits located in [Section 18.11.2, "Timer x Clock and Event Control Register," on page 386](#).

# MEC1618/MEC1618i

The Event input is synchronized to FCLK and (if enabled) filtered by a three stage filter. The resulting recreated clock is used to clock the timer in Event mode. In Bypass Mode (Sync Only), the pulse width of the external signal must be at least 2x the pulse width of the FCLK source. If the Event input not in Bypass Mode (Sync and Filter), the pulse width of the external signal must be at least 4x the pulse width of the sync and filter clock

## 18.5.3 RESET

This block is reset on a [nSYS\\_RST](#). On [nSYS\\_RST](#) all timers are reset to their default values. The timers are also reset by the [RESET](#) bit in each [Timer x Control Register](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 18.6 Interrupts

The timers in the MEC1618/MEC1618i can be used to generate interrupts when the timer overflows or underflows. The timer interrupts are routed to the [TIMER3](#), [TIMER2](#), [TIMER1](#), and [TIMER0](#) bits in [GIRQ15 Source Register](#).

**Note:** No interrupts are generated while the ENABLE bit is cleared.

## 18.7 Low Power Modes

This block is designed to conserve power when it is either sleeping or a clock source is not required.

During normal operation, if the timer is disabled via the PD bit the [TIMERx\\_CLK\\_REQ](#) signal is de-asserted. This indicates to the clock generator logic that this timer does not require the [MCLK](#) clock source.

During Sleep modes the clock input is gated, the [TIMERx\\_CLK\\_REQ](#) signal is asserted, and the interrupt output goes to the inactive state. When the block returns from sleep, if enabled, it will be restarted from the preload value.

**Note:** The timer is terminated one TCLK after the [SLEEP ENABLE](#) is asserted.

The following table illustrates the low power mode options.

**TABLE 18-3: BLOCK CLOCK GATING IN LOW POWER MODES**

Power Down (PD) Bit	<a href="#">SLEEP ENABLE</a>	Block Idle Status	<a href="#">TIMERx_CLK_REQ</a>	State	Description
1	X	NOT IDLE	1	PREPARING to SLEEP	The core clock is still required for up to one Timer Clock period.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.
0	0	X	1	NORMAL OPERATION	The block is neither disabled by firmware nor commanded to SLEEP
	1	NOT IDLE	1	PREPARING to SLEEP	The core clock is still required for up to one Timer Clock period.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.

## 18.8 Noise Filter

The noise filter uses Filter Clock (FCLK) to filter the signal on the [TINx](#) pins. An external [TINx](#) pin must remain in the same state for three FCLK ticks before the internal state changes. The [Filter Bypass](#) bit in the [Timer x Control Register](#) is used to bypass the noise filter.

- The signal TIN may be optionally only synchronized, or synchronized and filtered depending on the filter bypass bit
- The minimum FCLK period must be at least 2X the duration of the TIN signal so that signal can be reliably captured in the bypass mode
- The minimum FCLK period must be at least 4X the duration of the TIN signal so that signal can be reliably captured in the non-bypass mode
- In One-Shot mode, the TIN duration could be smaller than a TCLK period. The filtered signal is latched until the signal is seen in the TCLK domain. This also applies in the filter bypass mode

## 18.9 Operating Modes

### 18.9.1 STARTING AND STOPPING

The MEC1618/MEC1618i timers can be started and stopped by setting and clearing the Timer Enable bit in the Timer Control Register in all modes, except one-shot.

### 18.9.2 TIMER MODE

The Timer mode of the MEC1618/MEC1618i is used to generate periodic interrupts to the EC. When operating in this mode the timer always counts down based on one of the internally generated clock sources. The Timer mode is selected by setting the Timer Mode Select bits in the Timer Control Register. See [Section 18.11.1, "Timer x Control Register," on page 384](#).

The period between timer interrupts and the width of the output pulse is determined by the speed of the clock source, the clock divide ratio and the value programmed into the Timer Reload Register. The timer clock source and clock rate are selected using the Clock Source Select bits (TCLK) in the [Timer x Clock and Event Control Register](#). See [Section 18.11.2, "Timer x Clock and Event Control Register," on page 386](#).

**TABLE 18-4: TIMER MODE OPERATIONAL SUMMARY**

Item	Description
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in <a href="#">Table 18-12, "Timer Clock Frequencies," on page 387</a>
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in <a href="#">Table 18-12, "Timer Clock Frequencies," on page 387</a>
Count Operation	Down Counter
Reload Operation	When the timer underflows: RLOAD = 1, timer reloads from Timer Reload Reg RLOAD = 0, timer rolls over to FFFFh.
Count Start Condition	UPDN = 0 (timer only mode): <a href="#">ENABLE</a> = 1 UPDN = 1 (timer gate mode): <a href="#">ENABLE</a> = 1 & TIN = 1;
Count Stop Condition	UPDN = 0: <a href="#">ENABLE</a> = 0; UPDN = 1: ( <a href="#">ENABLE</a> = 0   TIN = 0)
Interrupt Request Generation Timing	When timer underflows from 0000h to reload value (as determined by RLOAD) an interrupt is generated.
TINx Pin Function	Provides timer gate function
TOUTx Pin Function	TOUT toggles each time the timer underflows (if enabled).
Read From Timer	Current count value can be read by reading the Timer Count Register

# MEC1618/MEC1618i

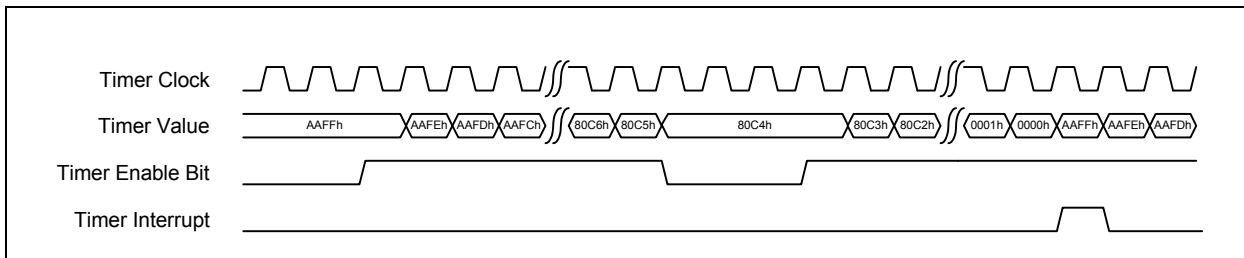
**TABLE 18-4: TIMER MODE OPERATIONAL SUMMARY (CONTINUED)**

Item	Description
Write to Preload Register	After the firmware writes to the Timer Reload Register asserting the RESET loads the timer with the new value programmed in the Timer Reload Register. Note: If the firmware does not assert RESET, the timer will automatically load the Timer Reload Register value when the timer underflows. When the timer is running, values written to the Timer Reload Register are written to the timer counter when the timer underflows. The assertion of Reset also copies the Timer Reload Register into the timer counter.
Selectable Functions	<ul style="list-style-type: none"> <li>Reload timer on underflow with programmed Preload value (Basic Timer)</li> <li>Reload timer with FFFFh in Free Running Mode (Free-running Timer)</li> <li>Timer can be started and stopped by the TINx input pin (Gate Function)</li> <li>The TOUTx pin changes polarity each time the timer underflows (Pulse Output Function)</li> </ul>

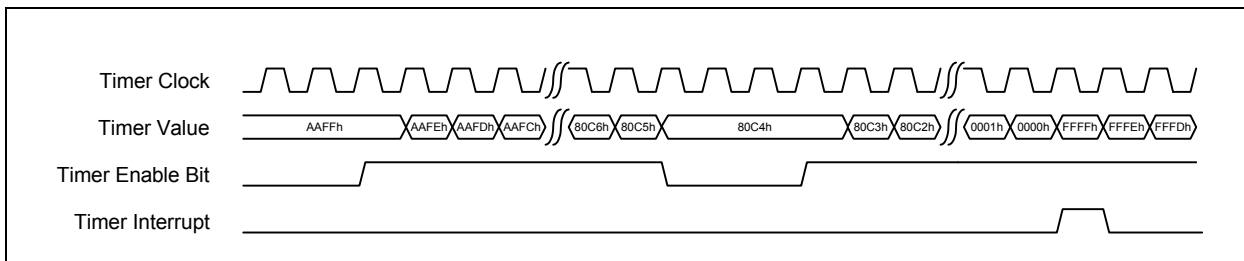
## 18.9.2.1 Timer Mode Underflow

The MEC1618/MEC1618i timers operating in Timer mode can underflow in two different ways. One method, the Reload mode shown in [Figure 18-2](#), is to reload the value programmed into the Reload register and continue counting from this value. The second method, Free Running mode [Figure 18-3](#), is to set the timer to FFFFh and continue counting from this value. The underflow behavior is controlled by the [RLOAD](#) bit in the Timer Control Register.

**FIGURE 18-2: RELOAD MODE BEHAVIOR**



**FIGURE 18-3: FREE RUNNING MODE BEHAVIOR**

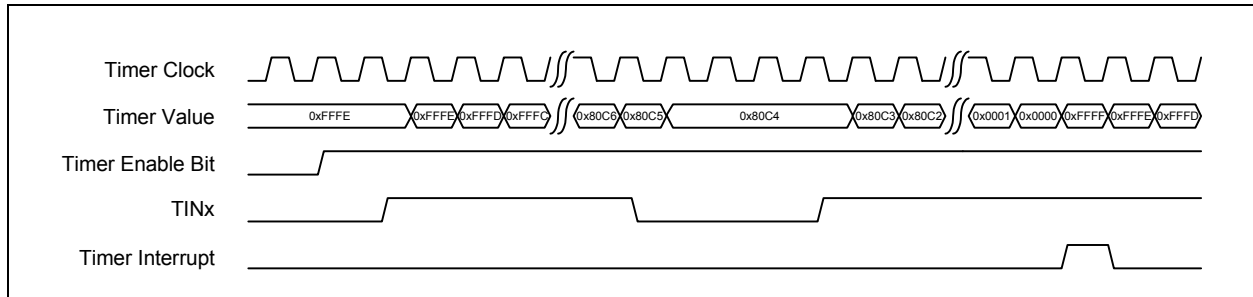




## 18.9.2.2 Timer Gate Function

The TINx pin on each timer can be used to pause the timer's operation when the timer is running. The timer will stop counting when the TINx pin is deasserted and count when the TINx pin is asserted. [Figure 18-4](#) shows the timer behavior when the TINx pin is used to gate the timer function. The UPDN bit is used to enable and disable the Timer Gate function when in the Timer mode.

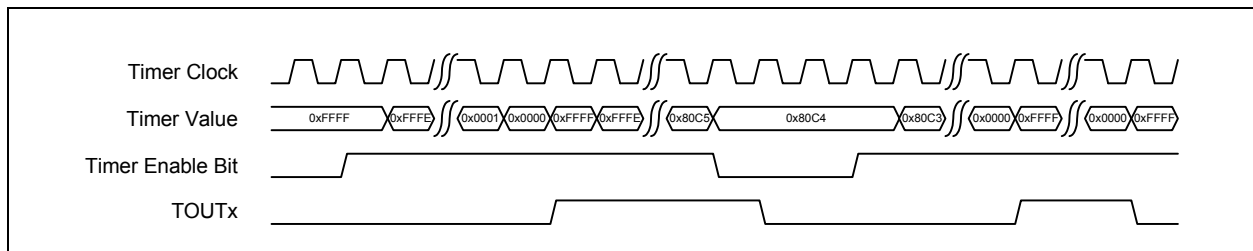
**FIGURE 18-4: TIMER GATE OPERATION**



## 18.9.2.3 Timer Mode Pulse Output

The four Timers can be used to generate a periodic output pulse. The output pulse changes state each time the timer underflows. The output is also cleared when the EN bit is cleared. [Figure 18-5](#) shows the behavior of the TOUTx pin when it is used as a pulse output pin.

**FIGURE 18-5: TIMER PULSE OUTPUT**



## 18.9.3 EVENT MODE

Event mode is used to count events that occur external to the timer. The timer can be programmed to count the overflow output from the previous timer or an edge on the TINx pin. The direction the timer counts in Event mode is controlled by the UPDN bit in the Timer Control Register. When the timer is in Event mode, the TOUTx signal can be used to generate a periodic output pulse when the timer overflows or underflows. [Figure 18-5](#) illustrates the pulse output behavior of the TOUTx pin in event mode when the timer underflows.

The timer can be programmed using the Clock and Event Control register to respond to the following events using the **EVENT** bits and the **EDGE** bits: rising edge of TINx, falling edge of TINx, rising and falling edge of TINx, rising edge of overflow input, falling edge of the overflow input, and the rising and falling edges of the overflow input.

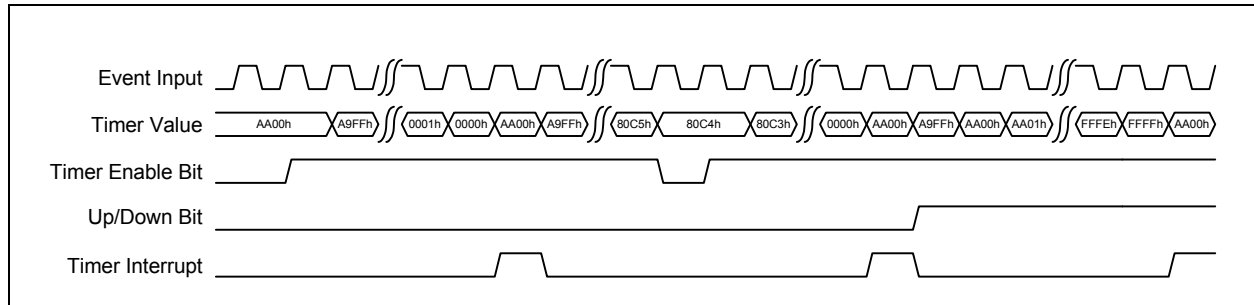
**TABLE 18-5: EVENT MODE OPERATIONAL SUMMARY**

Item	Description
Count Source	<ul style="list-style-type: none"> <li>External signal input to TINx pin (effective edge can be selected by software)</li> <li>Timer x-1 overflow</li> </ul>
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in <a href="#">Table 18-12, “Timer Clock Frequencies,” on page 387</a>
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in <a href="#">Table 18-12, “Timer Clock Frequencies,” on page 387</a>
Count Operation	Up/Down Counter
Reload Operation	<ul style="list-style-type: none"> <li>When the timer underflows: RLOAD = 1, timer reloads from Timer Reload Reg RLOAD = 0, timer rolls over to FFFFh.</li> <li>When the timer overflows: RLOAD = 1, timer reloads from Timer Reload Reg RLOAD = 0, timer rolls over to 0000h.</li> </ul>
Count Start Condition	Timer Enable is set ( <a href="#">ENABLE</a> = 1)
Count Stop Condition	Timer Enable is cleared ( <a href="#">ENABLE</a> = 0)
Interrupt Request Generation Timing	When timer overflows or underflows
TINx Pin Function	Event Generation
TOUTx Pin Function	TOUT toggles each time the timer underflows/overflows (if enabled).
Read From Timer	Current count value can be read by reading the Timer Count Register
Write to Preload Register	After the firmware writes to the Timer Reload Register, asserting the RESET loads the timer with the new value programmed in the Timer Reload Register. Note: If the firmware does not assert RESET, the timer will automatically load the Timer Reload Register value when the timer underflows.
Selectable Functions	<ul style="list-style-type: none"> <li>The direction of the counter is selectable via the UPDN bit.</li> <li>Reload timer on underflow/overflow with programmed Preload value (Basic Timer)</li> <li>Reload timer with FFFFh in Free Running Mode (Free-running Timer)</li> <li>Pulse Output Function The TOUTx pin changes polarity each time the timer underflows or overflows.</li> </ul>

## 18.9.3.1 Event Mode Operation

The timer starts counting events when the [ENABLE](#) bit in the Timer Control Register is set and continues to count until the [ENABLE](#) bit is cleared. When the [ENABLE](#) bit is set, the timer continues counting from the current value in the timer except after a reset event. After a reset event, the timer always starts counting from the value programmed in the Reload Register if counting down or from 0000h if counting up. [Figure 18-6](#) shows an example of timer operation in Event mode. The RLOAD bit controls the behavior of the timer when it underflows or overflows.

**FIGURE 18-6: EVENT MODE OPERATION**



## 18.9.4 ONE-SHOT MODE

The One-Shot mode of the timer is used to generate a single interrupt to the EC after a specified amount of time. The timer can be configured to start using the **ENABLE** bit (Figure 18-7) or on a timer overflow event from the previous timer. See Section 18.11.2, "Timer x Clock and Event Control Register," on page 386 for configuration details. The **ENABLE** bit must be set for an event to start the timer. The **ENABLE** bit is cleared one clock after the timer starts. The timer always starts from the value in the Reload Register and counts down in One-Shot mode.

**TABLE 18-6: ONE SHOT MODE OPERATIONAL SUMMARY**

Item	Description
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in Table 18-12, "Timer Clock Frequencies," on page 387
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in Table 18-12, "Timer Clock Frequencies," on page 387
Count Operation	Down Counter
Reload Operation	When the timer underflows the timer will stop.  When the timer is enabled timer starts counting from value programmed in Timer Reload Register. (RLOAD has no effect in this mode)
Count Start Condition	Setting the <b>ENABLE</b> bit to 1 starts One-Shot mode. The timer clock automatically clears the enable bit one timer tick later. <b>Note:</b> One-Shot mode may be enabled in Event Mode. In Event mode an overflow from the previous timer is used for timer tick rate.
Count Stop Condition	<ul style="list-style-type: none"> <li>Timer is reset (RESET = 1)</li> <li>Timer underflows</li> </ul>
Interrupt Request Generation Timing	When an underflow occurs.
TINx Pin Function	One Shot External input
TOUTx Pin Function	The TOUTx pin is asserted when the timer starts and de-asserted when the timer stops
Read From Timer	Current count value can be read by reading the Timer Count Register

# MEC1618/MEC1618i

TABLE 18-6: ONE SHOT MODE OPERATIONAL SUMMARY (CONTINUED)

Item	Description
Write to Preload Register	After the firmware writes to the Timer Reload Register, asserting the RESET loads the timer with the new value programmed in the Timer Reload Register. Note: If the firmware does not assert RESET, the timer will automatically load the Timer Reload Register value when the timer underflows.
Selectable Functions	<ul style="list-style-type: none"><li>Pulse Output Function The TOUTx pin is asserted when the timer starts and de-asserted when the timer stops.</li></ul>

FIGURE 18-7: TIMER START BASED ON ENABLE BIT

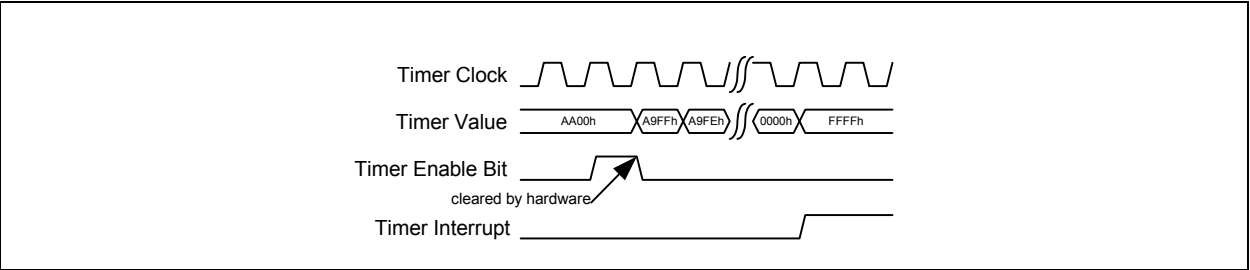


FIGURE 18-8: TIMER START BASED ON EXTERNAL EVENT

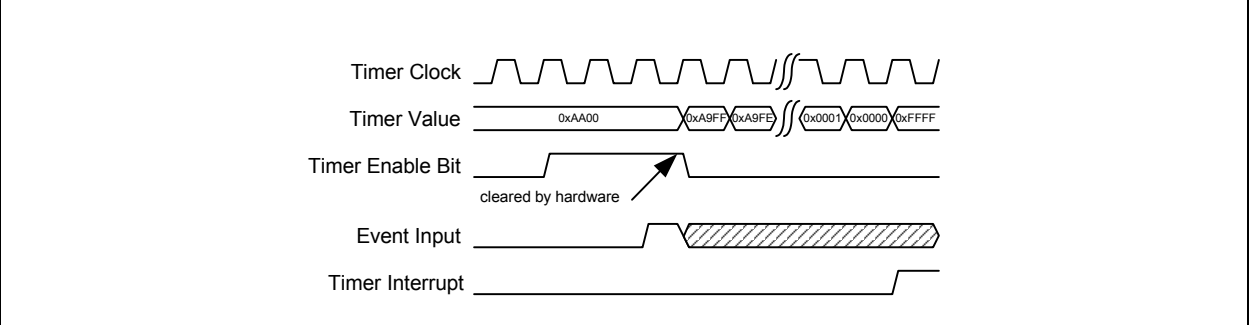
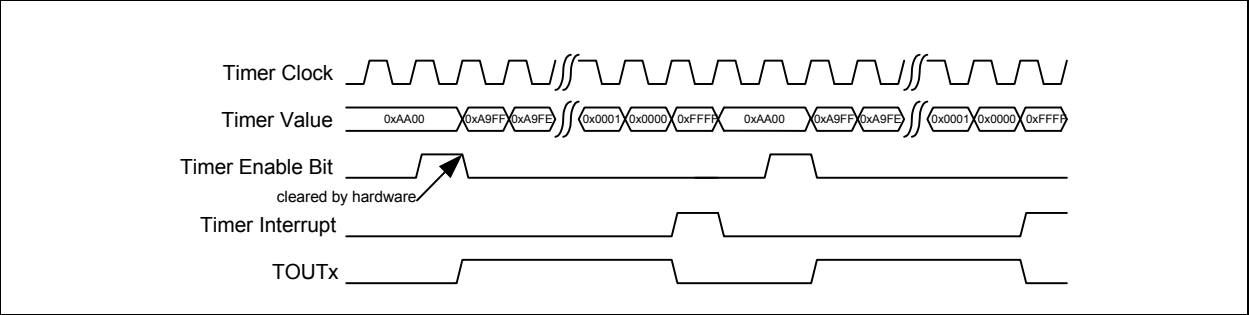


FIGURE 18-9: ONE SHOT TIMER WITH PULSE OUTPUT



### 18.9.5 MEASUREMENT MODE

The Measurement mode is used to measure the pulse width or period of an external signal. An interrupt to the EC is generated after each measurement or if the timer overflows and no measurement occurred. The timer measures the pulse width or period by counting the number of clock between edges on the TINx pin. The timer always starts counting at zero and counts up to 0xFFFF. The accuracy of the measurement depends on the speed of the clock being used. The speed of the clock also determines the maximum pulse width or period that can be detected.

**TABLE 18-7: MEASUREMENT MODE OPERATIONAL SUMMARY**

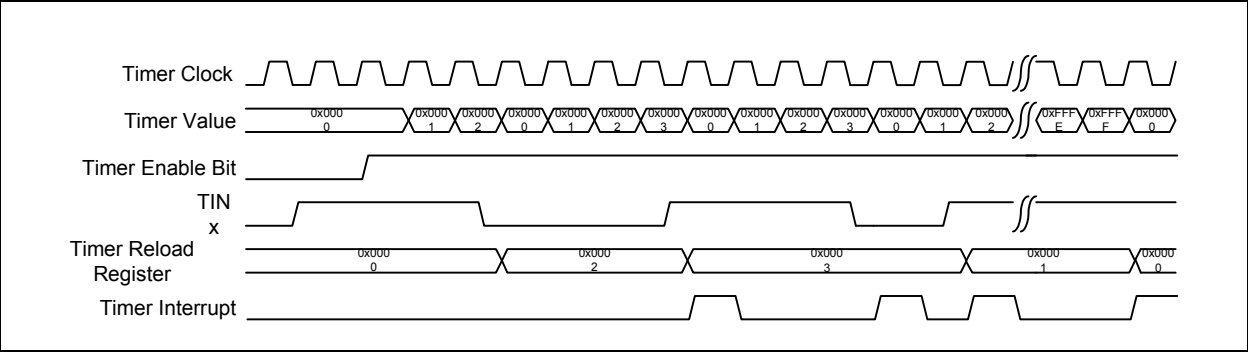
Item	Description
Timer Clock Frequencies	This mode supports all the programmable frequencies listed in <a href="#">Table 18-12, "Timer Clock Frequencies," on page 387</a>
Filter Clock Frequencies	This mode supports all the programmable frequencies listed in <a href="#">Table 18-12, "Timer Clock Frequencies," on page 387</a>
Count Operation	<ul style="list-style-type: none"> <li>Up Count</li> <li>At measurement pulse's effective edge, the count value is transferred to the Timer Reload Register and the timer is loaded with 0000h and continues counting.</li> </ul>
Count Start Condition	<ul style="list-style-type: none"> <li>Timer enable is set (<a href="#">ENABLE</a> = 1)</li> </ul>
Count Stop Condition	<ul style="list-style-type: none"> <li>Timer is reset (RESET = 1)</li> <li>Timer overflows</li> <li>Timer enable is cleared (<a href="#">ENABLE</a> = 0)</li> </ul>
Interrupt Request Generation Timing	<ul style="list-style-type: none"> <li>When timer overflows</li> <li>When a measurement pulse's effective edge is input. (An interrupt is not generated on the first effective edge after the timer is started.)</li> </ul>
TINx Pin Function	Programmable Input port or Measurement input
Read From Timer	When the <a href="#">Timer x Reload Register</a> is read it indicates the measurement result from the last measurement made. The <a href="#">Timer x Reload Register</a> reads 0000h if the timer overflows before a measurement is made.
Write to Timer	<a href="#">Timer x Reload Register</a> is Read-Only in Measurement mode

#### 18.9.5.1 Pulse Width Measurements

The timers measure pulse width by counting the number of timer clocks since the last rising or falling edge of the TINx input. To measure the pulse width of a signal on the TINx pin, the [EDGE](#) bits in the Clock and Event Control Register, must be set to start counting on rising and falling edges. The timer starts measuring on the next edge (rising or falling) on the TINx pin after the [ENABLE](#) bit is set. The Reload register stores the result of the last measurement taken. If the timer overflows, 0x0000 is written to the Reload register and the [ENABLE](#) bit is cleared stopping the timer. [Figure 18-10](#) shows the timer behavior when measuring pulse widths.

The timer will not assert an interrupt in Pulse Measurement mode until the timer detects both a rising and a falling edge.

FIGURE 18-10: PULSE WIDTH MEASUREMENT

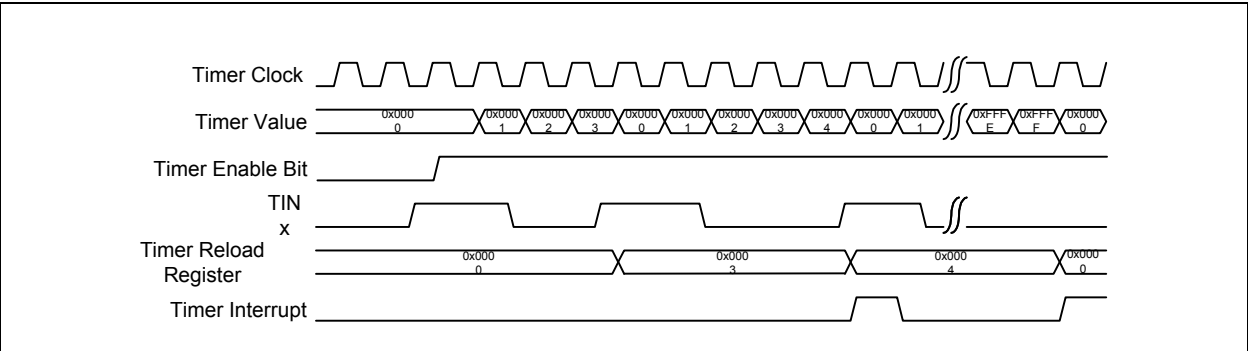


18.9.5.2 Period Measurements

The timers in the MEC1618/MEC1618i measure the period of a signal by counting the number of timer clocks between either rising or falling edges of the TINx input. The measurement edge is determined by the **EDGE** bits in the Clock and Event Control Register. The timer starts measuring on the next edge (rising or falling) on the TINx pin after the **ENABLE** bit is set. The reload register stores the result of the last measurement taken. If the timer overflows, 0x0000 is written to the reload register. Figure 18-11 shows the timer behavior when measuring the period of a signal.

The timer will not signal an interrupt in period measurement mode until the timer detects either two rising edges or two falling edges.

FIGURE 18-11: PULSE PERIOD MEASUREMENT



18.10 16-Bit Counter/Timer Interface Register Summary

There are four instances of the **16-Bit Timer Interface** block implemented in the MEC1618/MEC1618i enumerated as [0:3] with an overflow/underflow interface. Each instance of the **16-Bit Timer Interface** has its Base Address as indicated in Table 18-8.

**TABLE 18-8: 16-BIT COUNTER/TIMER INTERFACE BASE ADDRESS TABLE**

16-Bit Timer Interface Instance	LDN from (Table 3-2 on page 51)	AHB Base Address
16-bit Timer.0	3h	F0_0C00h
16-bit Timer.1		F0_0C80h = F0_0C00h + 80h
16-bit Timer.2		F0_0D00h = F0_0C00h + 100h
16-bit Timer.3		F0_0D80h = F0_0C00h + 180h

Table 18-9 is a register summary for one instance of the 16-Bit Timer Interface.

**TABLE 18-9: 16-BIT COUNTER/TIMER INTERFACE REGISTER SUMMARY**

Register Name	EC Interface		Notes
	SPB Offset	EC Type	
Timer x Control Register	00h	R/W	
Timer x Clock and Event Control Register	04h	R/W	
Timer x Reload Register	08h	R/W	
Timer x Count Register	0Ch	R	

# MEC1618/MEC1618i

## 18.11 Detailed Register Descriptions

### 18.11.1 TIMER X CONTROL REGISTER

TABLE 18-10: TIMER X CONTROL REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	00h			16-bit			EC SIZE	
POWER	VTR			0200h			VTR POR DEFAULT	
	EC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R/W	R/W	R/W
BIT NAME	Reserved			TIMERx_CLK_REQ	SLEEP_ENABLE	TOUT Polarity	PD	Filter Bypass
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	RLOAD	TOUT_EN	UPDN	INPOL	MODE		RESET	ENABLE

#### ENABLE

Timer Enable - This bit is used to start and stop the timer. This bit does not reset the timer count but does reset the timer pulse output. This bit will be cleared when the timer starts counting in One-Shot mode.

0=Timer is disabled

1=Timer is enabled

**Note:** The **ENABLE** bit is cleared after the RESET cycle is done. Firmware must poll the RESET bit.

#### RESET

Timer Reset - This bit stops the timer and resets the internal counter to the value in the Timer Reload Register. This bit also clears the Timer Enable bit if it is set. This bit is self clearing after the timer is reset. Firmware must poll this RESET bit.

0=Normal timer operation

1=Timer reset

**APPLICATION NOTE:** When the RESET takes effect interrupts are blocked. Interrupts are not blocked until RESET takes effect and the **ENABLE** bit is cleared. If interrupts are not desired, firmware must mask interrupt in the interrupt block.



## **MODE**

Timer Mode Select - These bits control the timer mode.

00=Timer Mode

01=Event Mode

10=One Shot Mode

11=Measurement Mode

## **INPOL**

Timer Input Polarity. This bit selects the polarity of the TINx input

0=TINx input is active low (inverted)

1=TINx input is active high (non-inverted)

## **UPDN**

Up/Down. In Event mode this bit selects the timer count direction.

Event Mode:

0=The timer counts down

1=The timer counts up

Timer Mode:

0=TINx pin has no effect on the timer

1=TINx pin pauses the timer when deasserted

## **TOUT\_EN**

TOUT Enable

0=TOUT pin is pin in the inactive state (driven low)

1=TOUT function is enabled

## **RLOAD**

Reload Control. This bit controls how the timer is reloaded on overflow or underflow in Event and Timer modes, it has no effect in One Shot mode.

0=Roll timer over to FFFFh and continue counting when counting down and rolls over to 0000h and continues counting when counting up.

1=Reload timer from Timer Reload Register and continue counting.

## **FILTER BYPASS**

Filter Bypass permits TINx to bypass the noise filter and go directly into the timer

0=Filter enabled on TINx (default)

1=Filter bypassed on TINx

## **PD**

Power Down.

0=The timer is in a running state.

1=The timer is powered down and all clocks are gated (default).

## **TOUT POLARITY**

This bit determines the polarity of the TOUT signal. In timer modes that toggle the TOUT signal, this polarity bit will not have a perceivable difference, except to determine the inactive state. in One-Shot mode this determines if the pulsed output is active high or active low.

0=Active high (default)

1=Active low

# MEC1618/MEC1618i

## SLEEP ENABLE

This bit is a read-only bit that reflects the state of the [SLEEP ENABLE](#) signal. This signal stops the timer and resets the internal counter to the value in the Timer Reload Register. After the timer is disabled, the [TIMERX\\_CLK\\_REQ](#) bits will be deasserted. This signal does not clear the Timer Enable bit if it is set. If the timer is enabled, the counter will resume operation when the [SLEEP ENABLE](#) signal is deasserted. The timer is held in reset as long as the input signal is asserted.

0=Normal timer operation. In Normal Mode, the timer operates as configured. When returning from a sleep mode, if enabled, the counter will be restarted from the preload value.

1=Sleep Mode Requested. In Sleep Mode, the timer is reset, the counter is disabled, and the [TIMERX\\_CLK\\_REQ](#) outputs are deasserted.

## TIMERX\_CLK\_REQ

The [TIMERX\\_CLK\\_REQ](#) bit is a read-only bit that reflects the state of the [TIMERX\\_CLK\\_REQ](#) output signal.

0=Indicates the [MCLK](#) clock domain can be turned 'off' when appropriate

1=Indicates the [MCLK](#) clock domain is required to be 'on.'

## 18.11.2 TIMER X CLOCK AND EVENT CONTROL REGISTER

**TABLE 18-11: TIMER X CLOCK AND EVENT CONTROL REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	04h			16-bit			EC SIZE	
POWER	VTR			0000h			VTR POR DEFAULT	
	EC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R/W	R/W	R/W	R/W
BIT NAME	Reserved				FCLK			
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
BIT NAME	EVENT	EDGE		Reserved	TCLK			

## TCLK

This field is the Timer Clock Select, used to determine the clock source to the 16-bit timer. Available frequencies are shown in [Table 18-12](#).

TABLE 18-12: TIMER CLOCK FREQUENCIES

Timer Clock Select	Frequency Selected
0000	MCLK
0001	MCLK/2
0010	MCLK/4
0011	MCLK/8
0100	MCLK/16
0101	MCLK/32
0110	MCLK/64
0111	MCLK/128
1xxx	Reserved

**EDGE**

Edge Type Select. These bits are used to select the edge type that the timer counts. In One-Shot mode these bits select which edge starts the timer.

**Event Mode:**

00=Counts falling edges

01=Counts rising edges

10=Counts rising and falling edges

11=No event selected

**One-Shot Mode:**

00=Starts counting on a falling edge

01=Starts counting on a rising edge

10=Starts counting on a rising or falling edge

11=Start counting when the Enable bit is set

**Measurement Mode:**

00=Measures the time between falling edges

01=Measures the time between rising edges

10=Measures the time between rising edges and falling edges and the time between falling edges and rising edges

11=No event selected

**EVENT**

Event Select - This bit is used to select the count source when the timer is operating in event mode.

0=Timer x-1 overflow is count source

1=TINx is count source

**FCLK**

This field is the Filter Clock Select, used to determine the clock source for the TINx noise filter. Available frequencies are the same as the Timer clock and are shown in Table 18-12.

# MEC1618/MEC1618i

## 18.11.3 TIMER X RELOAD REGISTER

TABLE 18-13: TIMER X RELOAD REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	08h			16-bit			EC SIZE	
POWER	VTR			FFFFh			VTR POR DEFAULT	
	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Timer Reload [15:0]							

### TIMER RELOAD

The Timer Reload register is used in Timer and One-Shot modes to set the lower limit of the timer. In Event mode the Timer Reload register sets either the upper or lower limit of the timer depending on if the timer is counting up or down. Valid [Timer Reload](#) values are 0001h - FFFFh. If the timer is running, the reload value will not be updated until the timer overflows or underflows.

**Note:** Programming a 0000h as a preload value is not a valid count value.

## 18.11.4 TIMER X COUNT REGISTER

TABLE 18-14: TIMER X COUNT REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	0Ch			16-bit			EC SIZE	
POWER	VTR			FFFFh			VTR POR DEFAULT	
	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Timer Count [15:0]							

### TIMER COUNT

The Timer Count register returns the current value of the timer in all modes.

## 19.0 HIBERNATION TIMER

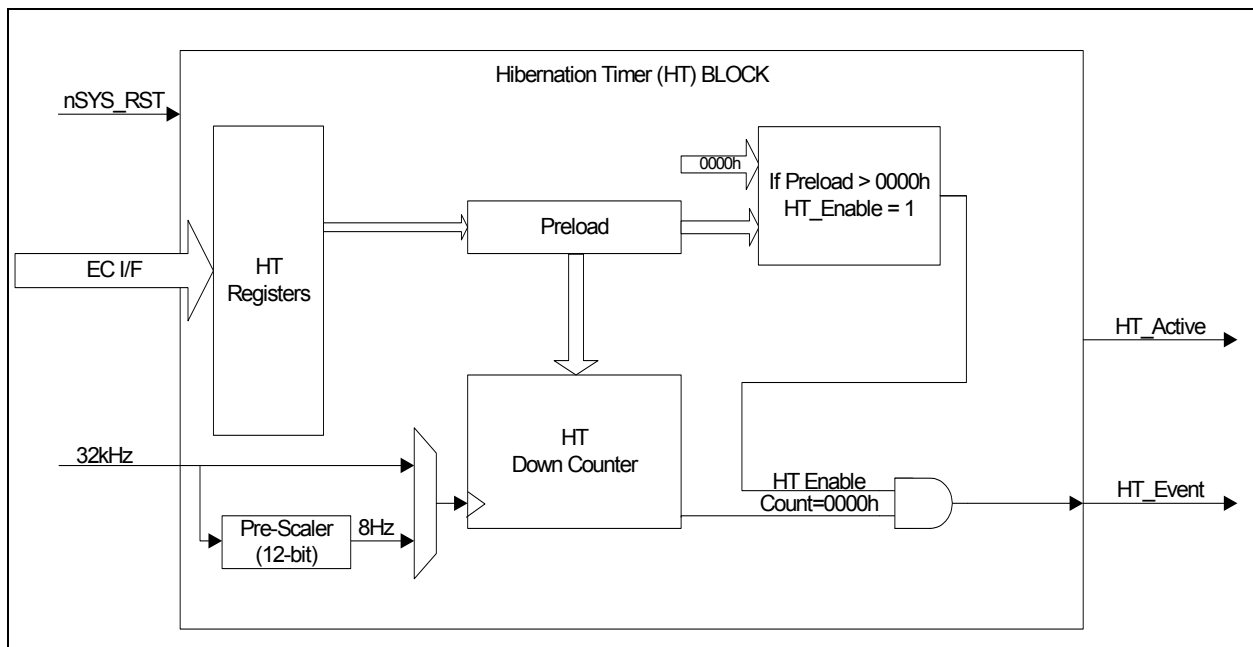
### 19.1 General Description

The Hibernation Timer can generate a wake event to the Embedded Controller (EC) when it is in a hibernation mode. This block supports wake events up to 2 hours in duration. The timer is a 16-bit binary count-down timer that can be programmed in 30.5us and 0.125 second increments for period ranges of 30.5us to 2s or 0.125s to 136.5 minutes, respectively. Writing a non-zero value to this register starts the counter from that value. A wake-up interrupt is generated when the count reaches zero.

See [GIRQ23 Source Register on page 342](#) for details on enabling the Hibernation Timer wake-up event.

### 19.2 Block Diagram

**FIGURE 19-1: HIBERNATION TIMER BLOCK DIAGRAM**



### 19.3 Block Diagram Signal List

**TABLE 19-1: BLOCK DIAGRAM SIGNAL LIST DESCRIPTION**

Signal Name	Direction	Description
<a href="#">nSYS_RST</a>	INPUT	VTR Power on Reset.
<a href="#">X32K_CLK</a>	INPUT	32Khz, Clock Source for Hibernation Timer
HT_Active	OUTPUT	Signal indicating that the timer is enabled and actively counting
HT_Event	OUTPUT	Signal indicating that the timer is enabled and has expired. This signal is used to generate an Hibernation Timer interrupt event.
E/C IF	I/O Bus	Bus used by microprocessor to access the registers in this block.

# MEC1618/MEC1618i

**TABLE 19-1: BLOCK DIAGRAM SIGNAL LIST DESCRIPTION (CONTINUED)**

Signal Name	Direction	Description
SLEEP_EN	INPUT	Sleep Enable input from the Block Sleep Enable Registers. See also <a href="#">Section 19.7, "Sleep Interface," on page 392</a> .
CLOCK_REQ	OUTPUT	Clock Required output to the Clock Required Status Registers. See also <a href="#">Section 19.7, "Sleep Interface," on page 392</a> .

## 19.4 Power, Clocks and Reset

### 19.4.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.5, "Power Configuration," on page 101](#) for details on power domains.

### 19.4.2 CLOCKS

This block has two clock inputs, the [EC Bus Clock](#), and [X32K\\_CLK](#). The [EC Bus Clock](#) is used in the interface to the embedded controller accessible registers. The 32.768kHz [X32K\\_CLK](#) is the clock source for the [Hibernation Timer](#) functional logic, including the counters.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

### 19.4.3 RESET

This block is reset on a [nSYS\\_RST](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

### 19.4.4 POWER MANAGEMENT

In all cases, [X32K\\_CLK](#) does not affect [Power Management](#); i.e., the [Hibernation Timer](#) operates normally when [MCLK](#) is stopped. The sleep enable inputs have no effect on the [Hibernation Timer](#) and the clock required outputs are only asserted during register read/write cycles for as long as necessary to propagate updates to the block core ([Table 19-2](#)).

**TABLE 19-2: HIBERNATION TIMER POWER MANAGEMENT**

SLEEP_EN	Bus Access Cycle?	CLK_REQ	Description
X	Yes	1	CLK_REQ is only asserted for as long as necessary to propagate updates to the block core
	No	0	CLK_REQ is not asserted when the EC is not accessing the register interface. (Note that this block <i>cannot</i> prevent the chip from entering the system deepest sleep states.)

## 19.5 Interrupts

Each instance of the Hibernation Timer in the MEC1618/MEC1618i can be used to generate interrupts and wake-up events when the [Hibernation Timer x Count Register](#) value transitions from '1' to '0'. The Hibernation Timer interrupts are routed to the [HTIMER1](#) & [HTIMER0](#) bits in [GIRQ23 Source Register on page 342](#).

## 19.6 Registers

There are two instances of [Hibernation Timer](#) block implemented in the MEC1618/MEC1618i enumerated as [Hibernation Timer.0](#) & [Hibernation Timer.1](#). Each instance of the [Hibernation Timer](#) has its own Logical Device Number, and Base Address as indicated in [Table 19-3](#).

**TABLE 19-3: Hibernation Timer BASE ADDRESS TABLE**

Hibernation Timer Instance	LDN from (Table 3-2 on page 51)	AHB Base Address
Hibernation Timer.0	0h	F0_0000h
Hibernation Timer.1		F0_0000h + 80h

The Table 19-4 is a register summary for one instance of the Hibernation Timer. Each EC address is indicated as an SPB Offset from its AHB base address.

**TABLE 19-4: Hibernation Timer REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
HTimer x Preload Register	00h	1-0	R/W	
Hibernation Timer x Control Register	04h	0	R/W	
Hibernation Timer x Count Register	08h	1-0	R	

#### 19.6.1 HTIMER X PRELOAD REGISTER

**TABLE 19-5: HTIMER X PRELOAD REGISTER**

HOST ADDRESS	n/a			n/a		HOST SIZE		
EC OFFSET	00h			16-bit		EC SIZE		
POWER	VTR			0000h		nSYS_RST DEFAULT		
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	HT Preload[15:0]							

#### HT PRELOAD[15:0]

This register is used to set the Hibernation Timer Preload value. Writing this register to a non-zero value resets the down counter to start counting down from this programmed value. Writing this register to 0000h disables the hibernation counter. The resolution of this timer is determined by the CTRL bit in the HTimer x Control Register.

Writes to the HTimer x Control Register are completed with an EC SPB bus cycle.

# MEC1618/MEC1618i

## 19.6.2 HTIMER X CONTROL REGISTER

TABLE 19-6: HIBERNATION TIMER X CONTROL REGISTER

HOST ADDRESS	n/a			n/a		HOST SIZE		
EC OFFSET	04h			16-bit		EC SIZE		
POWER	VTR			0000h		nSYS_RST DEFAULT		
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R							R/W
BIT NAME	Reserved							CTRL

### CTRL

0= The Hibernation Timer has a resolution of 30.5us per LSB, which yields a maximum time of ~2seconds.

1= The Hibernation Timer has a resolution of 0.125s per LSB, which yields a maximum time in excess of 2 hours.

## 19.6.3 HTIMER X COUNT REGISTER

TABLE 19-7: HIBERNATION TIMER X COUNT REGISTER

HOST ADDRESS	n/a			n/a		HOST SIZE		
EC OFFSET	08h			16-bit		EC SIZE		
POWER	VTR			0000h		nSYS_RST DEFAULT		
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R							
BIT NAME	Count[15:0]							

### COUNT[15:0]

The current state of the Hibernation Timer.

## 19.7 Sleep Interface

The [Hibernation Timer](#) CLOCK\_REQ output is only asserted during a register read/write cycle. The [Hibernation Timer](#) 32kHz clock requirement does not influence the CLOCK\_REQ output.



## 20.0 WEEK ALARM INTERFACE

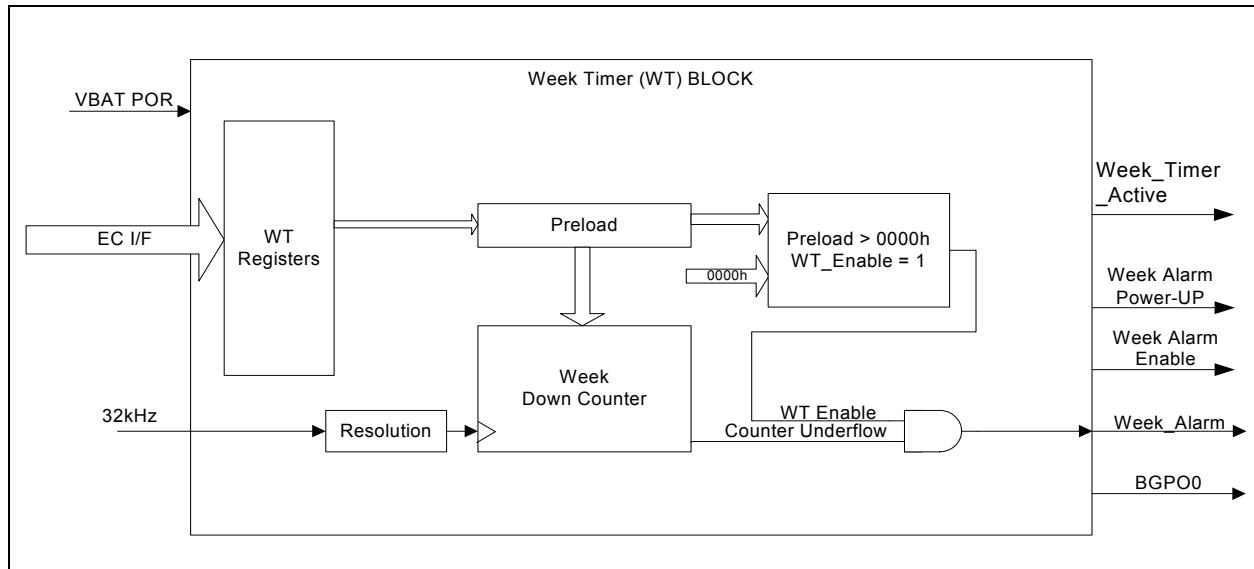
### 20.1 General Description

The [Week Alarm Interface](#) provides a 16-bit, battery-powered, Week Timer that supports 1 ms, 1 second and 1 minute resolution and auto reloads following a counter underflow ([Figure 20-1](#)). In addition, this block interfaces directly with the [VBAT-Powered Control Interface](#) and includes a [VBAT](#)-backed general-purpose output pin ([BGPO0](#)).

It takes up to two [X32K\\_CLK](#) clock period for registers to get updated after register writes. The [20 MHz Oscillator](#) must not be stopped for at least one [X32K\\_CLK](#) clock period following a register write.

### 20.2 Block Diagram

**FIGURE 20-1: WEEK TIMER BLOCK DIAGRAM**



### 20.3 Signal List for Block Diagram

**TABLE 20-1: WEEK ALARM INTERFACE SIGNAL LIST**

Signal Name	Direction	Description
<a href="#">VBAT_POR</a>	Input	VTR Power on Reset.
<a href="#">EC Bus Clock</a>	Input	Bus Clock (part of the <a href="#">EC Interface</a> )
<a href="#">X32K_CLK</a>	Input	Core logic clock
Week Alarm Power-Up Output	Output	Week Timer wake up event signal
Week_alarm_en	Output	Output to control the function of the Week Timer Output
Week_alarm	Output	Week Timer Interrupt indicating that the timer has expired.
Week_Timer_Active	Output	The Week_Timer_Active output is asserted when the counter is enabled and counting. It is cleared when the Week_Timer is disabled or not counting.

**TABLE 20-1: WEEK ALARM INTERFACE SIGNAL LIST (CONTINUED)**

Signal Name	Direction	Description
EC Interface	I/O Bus	Bus used by microprocessor to access the registers in this block.
BGPO0	Output	VBAT-powered General Purpose Output (see the BGPO0 bit D5 in the <a href="#">Week Timer Control Register</a> ).

## 20.4 Power, Clocks and Reset

### 20.4.1 POWER DOMAIN

This block is powered by the VBAT power supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

### 20.4.2 CLOCKS

This block has two clock inputs: [EC Bus Clock](#) and the [X32K\\_CLK](#). [EC Bus Clock](#) is used by the EC Data Memory Bus to interface to the embedded controller accessible registers. The 32.768kHz [X32K\\_CLK](#) clock source is the clock source for the week alarm logic.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

### 20.4.3 POWER ON RESET

This block is reset on [VBAT\\_POR](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 20.5 Interrupts

The [Week Alarm Interface](#) generates an interrupt and wakeup event following a Week Timer underflow. The Interrupt is routed [WEEK\\_ALR](#) in [GIRQ23 Source Register](#). The [WEEK\\_ALR](#) wake and interrupt event can be asserted when VBAT is powered and VTR is unpowered. The [WEEK\\_ALR](#) wake event and Interrupt event is retained during VBAT and is detected after the next VTR POR power sequence.

## 20.6 Week Timer

The [Week\\_Timer\\_Active](#) output is asserted when the counter is enabled and counting. It is cleared when the [Week\\_Timer](#) is disabled or not counting.

## 20.7 Week Alarm Power-Up Output

The internal Week Alarm Power-Up Output signal drives an input to the [VBAT-Powered Control Interface](#) as described in [Section 31.0, "VBAT-Powered Control Interface," on page 519](#). The [Week Alarm Power-Up Output](#) signal is driven even when the VTR supply is unpowered.

The [WEEK\\_ALRM\\_EN](#) bit in the [Week Timer Control Register](#) enables the [Week Alarm Power-Up Output](#) function in the [VBAT-Powered Control Interface](#) (see [FIGURE 31-1: on page 519](#)). After the internal [Week Alarm Power-Up Output](#) signal drives the input to the [VBAT-Powered Control Interface](#) as a result of an [Interrupts](#), the [WEEK\\_ALRM\\_EN](#) bit must be cleared to reset the [Week Alarm Interface](#).

**APPLICATION NOTE:** The [WEEK\\_ALRM\\_EN](#) bit defaults to '0,' disabling the ability to power the system up by the [Week Alarm Interface](#). This is necessary to avoid an uninitialized [Week Alarm Interface](#) from causing unintended power-up events.

## 20.8 Registers

The [Week Alarm Interface](#) has its own Logical Device Number, and Base Address as indicated in [Table 20-2](#). [Table 20-3](#) is a register summary for the [Week Alarm Interface](#) block. See [Note 3-1 on page 52](#).

**TABLE 20-2: [Week Alarm Interface](#) BASE ADDRESS TABLE**

<a href="#">Week Alarm Interface</a> Blocks	LDN from ( <a href="#">Table 3-2 on</a> <a href="#">page 51</a> )	AHB Base Address
<a href="#">Week Alarm Timer</a>	33h	F0_CC80h

[Table 20-3](#) is a register summary for the [Week Alarm Interface](#) block. Each EC address is indicated as an SPB Offset from its AHB base address.

**TABLE 20-3: [Week Alarm Interface](#) REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
<a href="#">Week Timer Control Register</a>	00h	0	R/W	
<a href="#">Week Timer Reload Register</a>	04h	0:1	R/W	
<a href="#">Week Timer Data Register</a>	08h	0:1	R	

### 20.8.1 WEEK TIMER CONTROL REGISTER

The [Week Timer Control Register](#) is used to configure the [Week Timer](#).

**TABLE 20-4: WEEK TIMER CONTROL REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	00h			8-bit			EC SIZE	
POWER	VBAT			01h			VBAT_POR DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R	R	R/W (Note 20-1)		R/W
BIT NAME	Reserved	WEEK_ALARM_EN	BGPO0	Reserved		RESOLUTION		WT_EN ABLE

# MEC1618/MEC1618i

## WT\_ENABLE

Week Timer Enable - This bit is used to start and stop the Week Timer. The Week timer is held when the timer is disabled and starts counting from the value in the Week Timer Reload register when enabled.

0 - Week Timer is disabled.

1 - Week Timer is enabled.

(See [Note 20-1 on page 397](#))

## RESOLUTION

Week Timer Resolution - These bits are used to control the resolution of the Week Timer counter.

00 - 1 minute resolution.

01 - 1 Second resolution

10 - 1 Millisecond resolution.

## BGPO0

VBAT-powered General Purpose Output Control that is used as part of the [VBAT-Powered Control Interface](#).

0= output low (default)

1= output high

## WEEK\_ALARM\_EN

VCI Week Alarm Enable- This bit controls the routing of the [Week Alarm Power-Up Output](#) output to [VBAT-Powered Control Interface](#) to assert the VCI signal. After the internal [Week\\_alm\\_en](#) signal drives the input to the [VBAT-Powered Control Interface](#) as a result of [Interrupts](#), this bit must be cleared to reset the [Week Alarm Interface](#).

0 - Disable routing (Default)

1 - Enable routing

**Note:** the Week Timer Enable bit [D0] must be cleared ('0') when changing the Week Timer Resolution bits. For example to change the resolution of the Week Timer two writes to the [Week Timer Control Register](#) are required: the first write de-asserts the [WT\\_ENABLE](#) bit, the second write modifies the [RESOLUTION](#) bits and asserts the [WT\\_ENABLE](#) bit.

## 20.8.2 WEEK TIMER RELOAD REGISTER

TABLE 20-5: WEEK TIMER RELOAD REGISTER

HOST ADDRESS	n/a			n/a		HOST SIZE		
EC OFFSET	04h			16-bit		EC SIZE		
POWER	VBAT			2760h		VBAT_POR DEFAULT		
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W <a href="#">Note 20-1</a>							
BIT NAME	WEEK TIMER RELOAD[15:0]							

**WEEK TIMER RELOIAD[15:0]**

This register contains the value that is used to reload the [Week Timer Data Register](#) when the latter underflows. A Reload value of 0000h is equivalent to a reload value of FFFFh. In both cases the Week Timer will count  $2^{16}$  times before triggering an interrupt.

**Note 20-1** The EC must clear the Enable bit in the [Week Timer Control Register](#) to perform a write access to the [Week Timer Reload Register](#).

**Note 20-2** A write to the [Week Timer Reload Register](#) of 0000h will be treated as a full count (FFFFh +1) and start downcounting when the [WT\\_ENABLE](#) bit in the [Week Timer Control Register](#) is set to '1'.

## 20.8.3 WEEK TIMER DATA REGISTER

**TABLE 20-6: WEEK TIMER DATA REGISTER**

HOST ADDRESS	n/a				n/a		HOST SIZE	
EC OFFSET	08h				16-bit		EC SIZE	
POWER	VBAT				2760h		VBAT_POR DEFAULT	
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R							
BIT NAME	WEEK TIMER Counter[15:0]							

**WEEK TIMER COUNTER[15:0]**

The current state of the Week Timer.

# MEC1618/MEC1618i

---

## 21.0 GPIO INTERFACE

### 21.1 General Description

The MEC1618/MEC1618i [GPIO Interface](#) provides general purpose input monitoring and output control, as well as managing many aspects of pin functionality; including, multi-function [Pin Multiplexing Control](#), [GPIO Direction](#) control, [PU/PD \(PU\\_PD\)](#) resistors, asynchronous wakeup and synchronous [Interrupt Detection \(int\\_det\)](#), [GPIO Direction](#), and [Polarity](#) control.

Features of the [GPIO Interface](#) include:

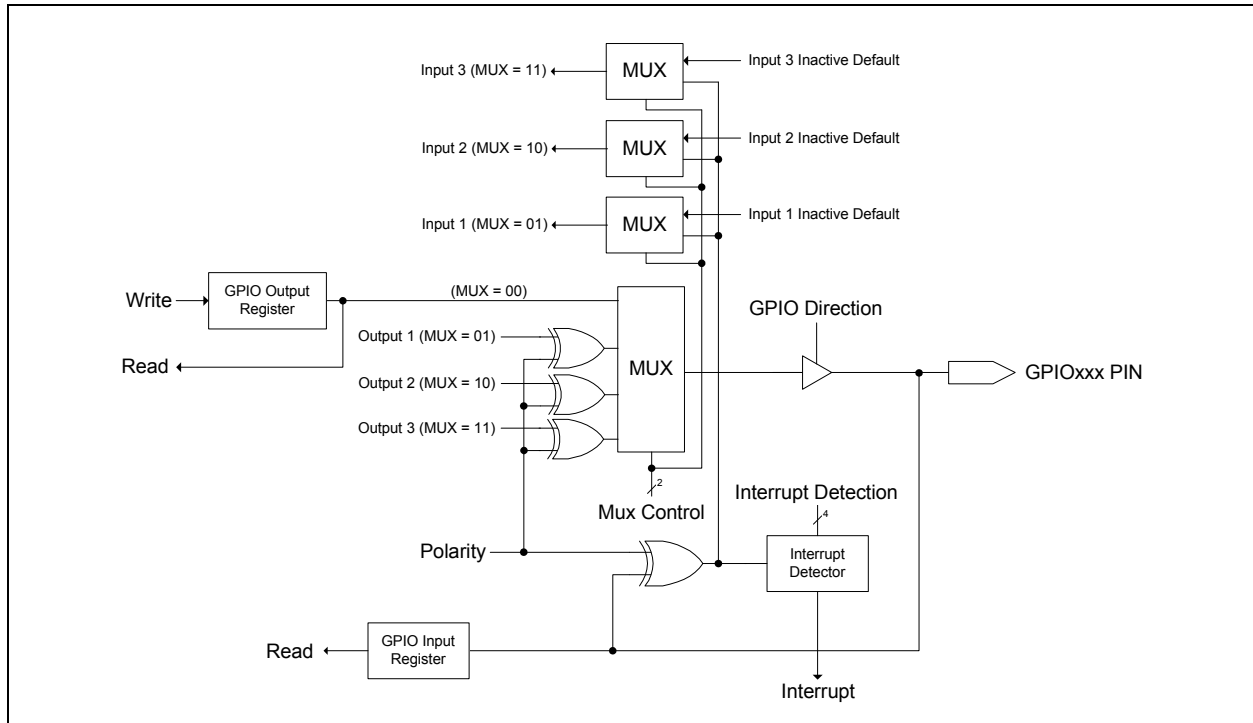
- Inputs:
  - Asynchronous rising and falling edge wakeup detection
  - Interrupt High or Low Level
- Outputs:
  - Push Pull or Open Drain output
  - Programmable power well emulation
- Pull up or pull down resistor control
- Interrupt and wake capability available for all GPIOs
- 8 [GPIO Pass-Through Ports](#)
- Group- or individual control of GPIO data. See [Section 21.3](#) and [Section 21.10](#)
- Multi-function pin multiplexing is controlled by the [GPIO Interface](#)
- Power consumption in the [GPIO Interface](#) is reduced by automatically disabling pull-up resistors when GPIO outputs are driven 'low,' and disabling pull-down resistors when GPIO outputs are driven 'high.'

### 21.2 Block Diagram

The [GPIO Interface Block Diagram](#) shown in [Figure 21-1](#) illustrates the functionality of a single MEC1618/MEC1618i [GPIO Interface](#) pin. The source for the [Pin Multiplexing Control](#), [Interrupt Detection \(int\\_det\)](#), [GPIO Direction](#), and [Polarity](#) controls in [Figure 21-1](#) is a [Pin Control Register](#) that is associated with each pin (see [Section 21.9.1](#), "Pin Control Register," on page 406).

The MEC1618/MEC1618i supports up to four independent signal functions per pin including the GPIO signal function itself, which is always positioned at [MUX Control](#) = '00.' The [GPIO Input Registers](#) and the [GPIO Output Registers](#) provide the [GPIO Interface](#) 'Read' and 'Write' functionality illustrated in [Figure 21-1](#) (see [Section 21.9.3](#), "GPIO Input Registers," on page 415 and [Section 21.9.2](#), "GPIO Output Registers," on page 411).

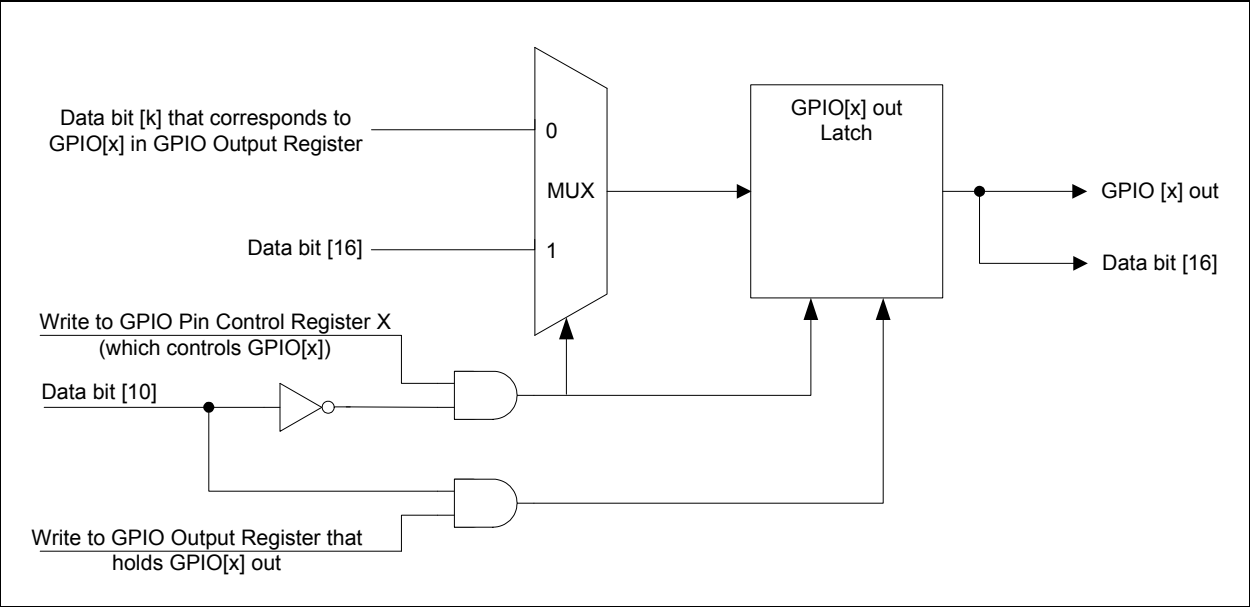
**FIGURE 21-1: GPIO Interface BLOCK DIAGRAM**



## 21.3 Accessing GPIOs

There are two ways to specify GPIO input and output port data. In the legacy approach that maintains compatibility with earlier generation devices, outputs to individual GPIO ports are grouped into four 32-bit [GPIO Output Registers](#) (see [Table 21-9](#).) It is incumbent on firmware to modify particular bit(s) while not disturbing the others. The MEC1618/MEC1618i supports an alternative approach in which each port's output is individually specified, i.e., Bit [16] in the port [Pin Control Register](#) is used for output data. Bit [10] [Output GPIO Write Enable](#) is used to enable this alternative write to the GPIO on a per-bit basis. [Figure 21-2](#) illustrates the concept. On reads, Bit [16] returns the programmed value while Bit [24] reflects the state of GPIO input from the pad regardless of setting of Bit [10].

FIGURE 21-2: OUTPUT DATA TO GPIO



21.4 GPIO Indexing

Each GPIO signal function name consists of a 4-character prefix (“GPIO”) followed by a 3-digit octal-encoded index number. [GPIO Indexing](#) is done sequentially starting from ‘GPIO000’ ([Table 21-1](#)). There is a unique index number for each GPIO pin function. Index numbers for the [NORM Exception Bits](#) are skipped; e.g., there is no GPIO037, GPIO077, GPIO137 or GPIO177 (see [Section 21.9.3, “GPIO Input Registers,” on page 415](#)).

Note that [Table 21-1](#) illustrates an example of [GPIO Indexing](#). Not all GPIOs are available in all parts. See [Section 21.5, “Pin Multiplexing Control,” on page 401](#) for a list of the MEC1618/MEC1618i GPIO pin functions.

TABLE 21-1: [GPIO Indexing](#) EXAMPLE

Bit Position	GPIO[000:036]	GPIO[040:076]	GPIO[100:136]	GPIO[140:176]	GPIO[200:236]
Bit 0	GPIO000	GPIO040	GPIO100	GPIO140	GPIO200
Bit 1	GPIO001	GPIO041	GPIO101	GPIO141	GPIO201
Bit 2	GPIO002	GPIO042	GPIO102	GPIO142	GPIO202
Bit 3	GPIO003	GPIO043	GPIO103	GPIO143	GPIO203
Bit 4	GPIO004	GPIO044	GPIO104	GPIO144	GPIO204
Bit 5	GPIO005	GPIO045	GPIO105	GPIO145	GPIO205
Bit 6	GPIO006	GPIO046	GPIO106	GPIO146	GPIO206
Bit 7	GPIO007	GPIO047	GPIO107	GPIO147	GPIO207
Bit 8	GPIO010	GPIO050	GPIO110	GPIO150	GPIO210
Bit 9	GPIO011	GPIO051	GPIO111	GPIO151	GPIO211
Bit 10	GPIO012	GPIO052	GPIO112	GPIO152	GPIO212



**TABLE 21-1: GPIO Indexing EXAMPLE (CONTINUED)**

Bit Position	GPIO[000:036]	GPIO[040:076]	GPIO[100:136]	GPIO[140:176]	GPIO[200:236]
Bit 11	GPIO013	GPIO053	GPIO113	GPIO153	GPIO213
Bit 12	GPIO014	GPIO054	GPIO114	GPIO154	GPIO214
Bit 13	GPIO015	GPIO055	GPIO115	GPIO155	GPIO215
Bit 14	GPIO016	GPIO056	GPIO116	GPIO156	GPIO216
Bit 15	GPIO017	GPIO057	GPIO117	GPIO157	GPIO217
Bit 16	GPIO020	GPIO060	GPIO120	GPIO160	GPIO220
Bit 17	GPIO021	GPIO061	GPIO121	GPIO161	GPIO221
Bit 18	GPIO022	GPIO062	GPIO122	GPIO162	GPIO222
Bit 19	GPIO023	GPIO063	GPIO123	GPIO163	GPIO223
Bit 20	GPIO024	GPIO064	GPIO124	GPIO164	GPIO224
Bit 21	GPIO025	GPIO065	GPIO125	GPIO165	GPIO225
Bit 22	GPIO026	GPIO066	GPIO126	GPIO166	GPIO226
Bit 23	GPIO027	GPIO067	GPIO127	GPIO167	GPIO227
Bit 24	GPIO030	GPIO070	GPIO130	GPIO170	GPIO230
Bit 25	GPIO031	GPIO071	GPIO131	GPIO171	GPIO231
Bit 26	GPIO032	GPIO072	GPIO132	GPIO172	GPIO232
Bit 27	GPIO033	GPIO073	GPIO133	GPIO173	GPIO233
Bit 28	GPIO034	GPIO074	GPIO134	GPIO174	GPIO234
Bit 29	GPIO035	GPIO075	GPIO135	GPIO175	GPIO235
Bit 30	GPIO036	GPIO076	GPIO136	GPIO176	GPIO236
Bit 31	Reserved or Norm Exception Bit	Reserved or Norm Exception Bit	Reserved or Norm Exception Bit	Reserved or Norm Exception Bit	Reserved or Norm Exception Bit

## 21.5 Pin Multiplexing Control

As described above in [Section 21.2, "Block Diagram"](#), pin multiplexing depends upon the [MUX Control](#) bits in the [Pin Control Register](#). There is a [Pin Control Register](#) for each GPIO signal function.

The MEC1618/MEC1618i [Pin Control Register](#) address offsets shown in the following tables depends on the GPIO Index number. [Pin Control Register](#) defaults are also shown in these tables. Pin Multiplexing Control notes that are cited in the "Notes" column can be found in [Table 2.6 on page 82](#),

GPIO signal function names in parentheses in the [Pin Control Register](#) tables represent interrupt/wake-only GPIO signal functions. The [MUX Control](#) bits in the [Pin Control Register](#) for these pins should not be programmed '00.'

# MEC1618/MEC1618i

**TABLE 21-2: PIN CONTROL MUXING PIN REFERENCE NUMBERS [1:32]**

Pin Ref. Number	Ball	GPIO Name (Octal)	Pin Control Reg. Offset (Hex)	Pin Control Reg. POR Value (Hex)	POR Default Signal Function	Mux Control = 00	Mux Control = 01	Mux Control = 10	Mux Control = 11	Notes
1	D4	GPIO165	01D4	00000000	GPIO165	GPIO165	32KHZ_IN	Reserved	Reserved	
2	E3			N/A	BGND					
3	C4			N/A	VBAT					
4	C1	None		N/A	BGPO0	BGPO0	Reserved	Reserved	Reserved	
5	D1	None		N/A	VCL_OUT	VCL_OUT	Reserved	Reserved	Reserved	
6	C2	GPIO161	01C4	00001000	VCL_IN2#	GPIO161	VCL_IN2#	Reserved	Reserved	
7	G1	GPIO162	01C8	00001000	VCL_IN1#	GPIO162	VCL_IN1#	Reserved	Reserved	
8	D2	GPIO163	01CC	00001000	VCL_IN0#	GPIO163	VCL_IN0#	Reserved	Reserved	
9	E2	GPIO164	01D0	00001000	VCL_OVRD_IN	GPIO164	VCL_OVRD_IN	Reserved	Reserved	
10	E1	GPIO000	0000	00001000	VCL_IN3#	GPIO000	VCL_IN3#	Reserved	Reserved	
11	C3	GPIO234	0270	00001000	VCL_IN4#	GPIO234	VCL_IN4#	Reserved	Reserved	
12	H2	GPIO235	0274	00001000	VCL_IN5#	GPIO235	VCL_IN5#	Reserved	Reserved	
13	F4	None		N/A	RESET#	RESET#	Reserved	Reserved	Reserved	
14	G2	GPIO062	00C8	00000200	GPIO062	GPIO062	Reserved	Reserved	Reserved	Note 20
15	F3	(GPIO057)	00BC	00001000	VCC_PWRGD	Reserved	VCC_PWRGD	Reserved	Reserved	Note 17
16	F2	GPIO106	0118	00000000	GPIO106	GPIO106	nRESET_OUT	Reserved	Reserved	
17	H5			N/A	VSS_RO					
18	G4			N/A	VTR					
19	J4			N/A	VSS0					
20	H1	GPIO033	006C	00000000	GPIO033	GPIO033	RC_ID	Reserved	Reserved	
21	H3	GPIO021	0044	00000000	GPIO021	GPIO021	KSI2	Reserved	Reserved	
22	J5			N/A	VTR_REG					
23	H4			N/A	VR_CAP					
24	F1	GPIO060	00C0	00000000	GPIO060	GPIO060	KBRST	Reserved	Reserved	
25	J1			N/A	AVTR_ADC					
26	J3			N/A	VREF_ADC					
27	K3	GPIO200	0200	00001000	ADC0	GPIO200	ADC0	Reserved	Reserved	
28	J2	GPIO210	0220	00001000	ADC8	GPIO210	ADC8	Reserved	Reserved	
29	K1	GPIO201	0204	00001000	ADC1	GPIO201	ADC1	Reserved	Reserved	
30	K2	GPIO211	0224	00001000	ADC9	GPIO211	ADC9	Reserved	Reserved	
31	G3			N/A	VTR3					
32	L2	GPIO202	0208	00001000	ADC2	GPIO202	ADC2	Reserved	Reserved	

**TABLE 21-3: PIN CONTROL MUXING PIN REFERENCE NUMBERS [33:64]**

Pin Ref. Number	Ball	GPIO Name (Octal)	Pin Control Reg. Offset (Hex)	Pin Control Reg. POR Value (Hex)	POR Default Signal Function	Mux Control = 00	Mux Control = 01	Mux Control = 10	Mux Control = 11	Notes
33	L1	GPIO212	0228	00001000	ADC10	GPIO212	ADC10	Reserved	Reserved	
34	M1	GPIO203	020C	00001000	ADC3	GPIO203	ADC3	Reserved	Reserved	
35	L3	GPIO213	022C	00001000	ADC11	GPIO213	ADC11	Reserved	Reserved	
36	L4	GPIO204	0210	00001000	ADC4	GPIO204	ADC4	Reserved	Reserved	
37	M2	GPIO214	0230	00001000	ADC12	GPIO214	ADC12	Reserved	Reserved	
38	N1	GPIO205	0214	00001000	ADC5	GPIO205	ADC5	Reserved	Reserved	
39	N2	GPIO215	0234	00001000	ADC13	GPIO215	ADC13	Reserved	Reserved	
40	M3	GPIO206	0218	00001000	ADC6	GPIO206	ADC6	Reserved	Reserved	
41	M4	GPIO216	0238	00001000	ADC14	GPIO216	ADC14	Reserved	Reserved	
42	N3	GPIO207	021C	00001000	ADC7	GPIO207	ADC7	Reserved	Reserved	
43	N4	GPIO217	023C	00001000	ADC15	GPIO217	ADC15	Reserved	Reserved	
44	K4			N/A	VSS_ADC					
45	M5	(GPIO064)	00D0	00001000	LRESET#	Reserved	LRESET#	Reserved	Reserved	Note 17
46	L5	(GPIO067)	00DC	00001000	CLKRUN#	Reserved	CLKRUN#	Reserved	Reserved	Note 17
47	L6	(GPIO066)	00D8	00001000	LFRAME#	Reserved	LFRAME#	Reserved	Reserved	Note 17
48	N5	(GPIO063)	00CC	00001000	SER_IRQ	Reserved	SER_IRQ	Reserved	Reserved	Note 17
49	K6			N/A	VTR1					
50	N6	(GPIO065)	00D4	00001000	PCI_CLK	Reserved	PCI_CLK	Reserved	Reserved	Note 17
51	M6	None		N/A	LAD0	LAD0	Reserved	Reserved	Reserved	
52	M7	None		N/A	LAD1	LAD1	Reserved	Reserved	Reserved	
53	L7	None		N/A	LAD2	LAD2	Reserved	Reserved	Reserved	
54	K7	None		N/A	LAD3	LAD3	Reserved	Reserved	Reserved	
55	K5	GPIO100	0100	00000000	GPIO100	GPIO100	nEC_SCI	Reserved	Reserved	
56	N7	GPIO011	0024	00000000	GPIO011	GPIO011	nSMI	Reserved	Reserved	
57	L8	GPIO061	00C4	00000000	GPIO061	GPIO061	LPCPD#	Reserved	Reserved	
58	M8	GPIO050	00A0	00000000	GPIO050	GPIO050	FAN_TACH0	Reserved	Reserved	
59	N9	GPIO051	00A4	00000000	GPIO051	GPIO051	FAN_TACH1	Reserved	Reserved	
60	M9	GPIO052	00A8	00000000	GPIO052	GPIO052	FAN_TACH2	Reserved	Reserved	
61	L9	GPIO016	0038	00000000	GPIO016	GPIO016	GPTP-IN7	FAN_TACH3	Reserved	
62	J9	GPIO230	0260	00000000	GPIO230	GPIO230	ECGP_SCLK	Reserved	Reserved	
63	N13	GPIO053	00AC	00000000	GPIO053	GPIO053	PWM0	Reserved	Reserved	
64	K9	GPIO231	0264	00000000	GPIO231	GPIO231	ECGP_SOUT	Reserved	Reserved	

TABLE 21-4: PIN CONTROL MUXING PIN REFERENCE NUMBERS [65:96]

Pin Ref. Number	Ball	GPIO Name (Octal)	Pin Control Reg. Offset (Hex)	Pin Control Reg. POR Value (Hex)	POR Default Signal Function	Mux Control = 00	Mux Control = 01	Mux Control = 10	Mux Control = 11	Notes
65	M10	GPIO054	00B0	00000000	GPIO054	GPIO054	PWM1	Reserved	Reserved	
66	K8	GPIO233	026C	00000000	GPIO233	GPIO233	ECGP_SIN	Reserved	Reserved	
67	M11	GPIO055	00B4	00000000	GPIO055	GPIO055	PWM2	Reserved	Reserved	
68	L13	GPIO056	00B8	00000000	GPIO056	GPIO056	PWM3	Reserved	Reserved	
69	M12	GPIO001	0004	00000000	GPIO001	GPIO001	PWM4	Reserved	Reserved	
70	L10	GPIO002	0008	00000000	GPIO002	GPIO002	PWM5	Reserved	Reserved	
71	L12	GPIO014	0030	00000000	GPIO014	GPIO014	GPTP-IN6	PWM6	Reserved	
72	K12	GPIO015	0034	00000000	GPIO015	GPIO015	GPTP-OUT6	PWM7	Reserved	
73	N12	GPIO151	01A4	00000000	GPIO151	GPIO151	GPTP-IN3	FAN_TACH4	KSO15	
74	N10	GPIO152	01A8	00000000	GPIO152	GPIO152	GPTP-OUT3	FAN_TACH5	KSO16	
75	J8			N/A	VTR2					
76	K13	GPIO003	000C	00000000	GPIO003	GPIO003	SMB00_DATA	Reserved	Reserved	
77	H13	GPIO004	0010	00000000	GPIO004	GPIO004	SMB00_CLK	Reserved	Reserved	
78	K11	GPIO005	0014	00000000	GPIO005	GPIO005	SMB01_DATA	Reserved	Reserved	
79	J12	GPIO006	0018	00000000	GPIO006	GPIO006	SMB01_CLK	Reserved	Reserved	
80	J11	GPIO012	0028	00000000	GPIO012	GPIO012	SMB07_DATA	Reserved	Reserved	
81	H11	GPIO013	002C	00000000	GPIO013	GPIO013	SMB07_CLK	Reserved	Reserved	
82	H12	GPIO130	0160	00000000	GPIO130	GPIO130	SMB10_DATA	Reserved	Reserved	
83	G10	GPIO131	0164	00000000	GPIO131	GPIO131	SMB10_CLK	Reserved	Reserved	
84	G11	GPIO132	0168	00000000	GPIO132	GPIO132	SMB06_DATA	KSO14	Reserved	
85	G12	GPIO140	0180	00000000	GPIO140	GPIO140	SMB06_CLK	PWM13	Reserved	
86	H9			N/A	VTR_FLASH					
87	J10	GPIO141	0184	00000000	GPIO141	GPIO141	SMB05_DATA	PWM14	FLSCLK	
88	K10	GPIO142	0188	00000000	GPIO142	GPIO142	SMB05_CLK	PWM15	FLSOUT	
89	H10	GPIO143	018C	00000000	GPIO143	GPIO143	SMB04_DATA	PWM12	FLSIN	
90	L11	GPIO144	0190	00000000	GPIO144	GPIO144	SMB04_CLK	Reserved	FLSCS#	
91	F11	GPIO007	001C	00000000	GPIO007	GPIO007	SMB03_DATA	PS2_CLK0B	Reserved	
92	F10	GPIO010	0020	00000000	GPIO010	GPIO010	SMB03_CLK	PS2_DAT0B	Reserved	
93	F12	GPIO154	01B0	00000000	GPIO154	GPIO154	SMB02_DATA	PS2_CLK1B	Reserved	
94	E12	GPIO155	01B4	00000000	GPIO155	GPIO155	SMB02_CLK	PS2_DAT1B	Reserved	
95	E11	GPIO110	0120	00000000	GPIO110	GPIO110	PS2_CLK2	GPTP-IN5	Reserved	
96	E10	GPIO111	0124	00000000	GPIO111	GPIO111	PS2_DAT2	GPTP-OUT5	Reserved	

TABLE 21-5: PIN CONTROL MUXING PIN REFERENCE NUMBERS [96:128]

Pin Ref. Number	Ball	GPIO Name (Octal)	Pin Control Reg. Offset (Hex)	Pin Control Reg. POR Value (Hex)	POR Default Signal Function	Mux Control = 00	Mux Control = 01	Mux Control = 10	Mux Control = 11	Notes
97	D12	GPIO112	0128	00000000	GPIO112	GPIO112	PS2_CLK1A	KSO5	Reserved	
98	N11	GPIO113	012C	00000000	GPIO113	GPIO113	PS2_DAT1A	KSO6	Reserved	
99	M13	GPIO114	0130	00000000	GPIO114	GPIO114	PS2_CLK0A	Reserved	Reserved	
100	J13	GPIO115	0134	00000000	GPIO115	GPIO115	PS2_DAT0A	Reserved	Reserved	
101	J6			N/A	VSS1					
102	G13	GPIO070	00E0	00000000	GPIO070	GPIO070	Reserved	Reserved	Reserved	
103	F13	GPIO071	00E4	00000000	GPIO071	GPIO071	Reserved	Reserved	Reserved	
104	E13	GPIO072	00E8	00000000	GPIO072	GPIO072	Reserved	Reserved	Reserved	
105	D13	GPIO073	00EC	00000000	GPIO073	GPIO073	Reserved	Reserved	Reserved	
106	C13	GPIO074	00F0	00000000	GPIO074	GPIO074	Reserved	Reserved	Reserved	
107	B13	GPIO075	00F4	00000000	GPIO075	GPIO075	Reserved	Reserved	Reserved	
108	F9	GPIO041	0084	00000000	GPIO041	GPIO041	Reserved	Reserved	Reserved	
109	A13	GPIO076	00F8	00000000	GPIO076	GPIO076	Reserved	Reserved	Reserved	
110	E9	GPIO220	0240	00000000	GPIO220	GPIO220	Reserved	Reserved	Reserved	
111	E8	GPIO035	0074	00000000	GPIO035	GPIO035	PWM8	Reserved	Reserved	
112	D8	GPIO170	01E0	00000000	GPIO170	GPIO170	MSCLK	Reserved	Reserved	
113	D7	GPIO171	01E4	00000001	GPIO171	GPIO171	MSDATA	Reserved	Reserved	Note 18
114	N8	GPIO133	016C	00000000	GPIO133	GPIO133	PWM9	Reserved	Reserved	
115	D11	GPIO134	0170	00000000	GPIO134	GPIO134	PWM10	Reserved	Reserved	
116	D10	GPIO135	0174	00000000	GPIO135	GPIO135	PWM11	Reserved	Reserved	
117	C12	GPIO145	0194	00000000	GPIO145	GPIO145	SMB09_DATA	JTAG_TDI	Reserved	Note 14
118	B12	GPIO146	0198	00000000	GPIO146	GPIO146	SMB09_CLK	JTAG_TDO	Reserved	Note 14
119	B11	GPIO147	019C	00000000	GPIO147	GPIO147	SMB08_DATA	Reserved	JTAG_CLK	Note 14
120	A12	GPIO150	01A0	00000000	GPIO150	GPIO150	SMB08_CLK	Reserved	JTAG_TMS	Note 14
121	A11	None		N/A	JTAG_RST#	JTAG_RST#	Reserved	Reserved	Reserved	Note 14
122	C11	GPIO104	0110	00000000	GPIO104	GPIO104	UART_TX	Reserved	Reserved	Note 19
123	A10	GPIO105	0114	00000000	GPIO105	GPIO105	UART_RX	Reserved	Reserved	
124	B10	GPIO025	0054	00000000	GPIO025	GPIO025	UART_CLK	TIN0	EM_INT	
125	B9	GPIO026	0058	00000000	GPIO026	GPIO026	GPTP-IN0	TIN1	KSI3	
126	D9	GPIO027	005C	00000000	GPIO027	GPIO027	GPTP-OUT0	TIN2	KSI4	
127	C10	GPIO030	0060	00000000	GPIO030	GPIO030	GPTP-IN1	TIN3	KSI5	
128	A9	GPIO107	011C	00000000	GPIO107	GPIO107	KSO4	Reserved	Reserved	

# MEC1618/MEC1618i

**TABLE 21-6: PIN CONTROL MUXING PIN REFERENCE NUMBERS [129:156]**

Pin Ref. Number	Ball	GPIO Name (Octal)	Pin Control Reg. Offset (Hex)	Pin Control Reg. POR Value (Hex)	POR Default Signal Function	Mux Control = 00	Mux Control = 01	Mux Control = 10	Mux Control = 11	Notes
129	C9	GPIO120	0140	00000000	GPIO120	GPIO120	KSO7	Reserved	Reserved	
130	B8	GPIO124	0150	00000000	GPIO124	GPIO124	GPTP-OUT4	KSO11	Reserved	
131	A8	GPIO125	0154	00000000	GPIO125	GPIO125	GPTP-IN4	KSO12	Reserved	
132	D6	GPIO031	0064	00000000	GPIO031	GPIO031	GPTP-OUT1	TOUT0	KSI6	
133	B7	GPIO032	0068	00000000	GPIO032	GPIO032	GPTP-IN2	TOUT1	KSI7	
134	C8	GPIO040	0080	00000000	GPIO040	GPIO040	GPTP-OUT2	TOUT2	KSO0	
135	A7	GPIO017	003C	00000000	GPIO017	GPIO017	GPTP-OUT7	TOUT3	KSI0	
136	D5	GPIO022	0048	00000000	GPIO022	GPIO022	BCM_B_CLK	Reserved	Reserved	
137	C5	GPIO023	004C	00000000	GPIO023	GPIO023	BCM_B_DAT	Reserved	Reserved	
138	E4	GPIO024	0050	00000000	GPIO024	GPIO024	BCM_B_INT#	Reserved	Reserved	
139	C7	GPIO127	015C	00000000	GPIO127	GPIO127	A20M	Reserved	Reserved	
140	F5	GPIO034	0070	00000000	GPIO034	GPIO034	CEC_OUT	Reserved	Reserved	
141	E5	GPIO036	0078	00000000	GPIO036	GPIO036	CEC_IN	Reserved	Reserved	
142	A6	GPIO045	0094	00000000	GPIO045	GPIO045	LSBCM_D_INT#	KSO1	Reserved	
143	E6	GPIO046	0098	00000000	GPIO046	GPIO046	LSBCM_D_DAT	KSO2	Reserved	
144	B6	GPIO047	009C	00000000	GPIO047	GPIO047	LSBCM_D_CLK	KSO3	Reserved	
145	C6	GPIO121	0144	00000000	GPIO121	GPIO121	BCM_A_INT#	KSO8	Reserved	
146	A5	GPIO122	0148	00000000	GPIO122	GPIO122	BCM_A_DAT	KSO9	Reserved	
147	B5	GPIO123	014C	00000000	GPIO123	GPIO123	BCM_A_CLK	KSO10	Reserved	
148	A3	GPIO042	0088	00000000	GPIO042	GPIO042	Reserved	PECL_DAT	SB-TSI_DAT	
149	A2	GPIO043	008C	00000000	GPIO043	GPIO043	Reserved	Reserved	SB-TSI_CLK	
150	A1	GPIO044	0090	00000000	GPIO044	GPIO044	VREF_VTT	Reserved	Reserved	
151	B1	GPIO126	0158	00000000	GPIO126	GPIO126	KSO13	Reserved	Reserved	
152	A4	GPIO020	0040	00000000	GPIO020	GPIO020	KSI1	Reserved	Reserved	
153	B2	GPIO156	01B8	00000000	GPIO156	GPIO156	LED0	Reserved	Reserved	
154	B3	GPIO157	01BC	00000000	GPIO157	GPIO157	LED1	Reserved	Reserved	
155	B4	GPIO153	01AC	00000000	GPIO153	GPIO153	LED2	Reserved	Reserved	
156	D3	GPIO175	01F4	00000000	GPIO175	GPIO175	32KHZ_OUT	KSO17	Reserved	

## 21.6 Power, Clocks and Reset

### 21.6.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

### 21.6.2 CLOCKS

This block uses the [EC Bus Clock](#) and [MCLK](#). [EC Bus Clock](#) is used for access to registers. The [MCLK](#) is used for synchronizing the GPIO inputs.

### 21.6.3 RESET

This block is reset on a [nSYS\\_RST](#). On reset, all Registers are reset to their default values.

## 21.7 Interrupts

Each pin in the [GPIO Interface](#) has both an interrupt and/or a Wake-up event ([Table 21-7](#)). The interrupt source is routed onto the GPIO Status Bits corresponding to the specific GPIO identified in the [GIRQ8 Source Register - GIRQ11 Source Register](#), and the [GIRQ22 Source Register](#). The [GPIO Interface](#) can generate an interrupt on a high level, low level, rising edge and falling edge, as configured by the [Interrupt Detection \(int\\_det\)](#) bits in the [Pin Control Register](#) associated with the GPIO signal function.

**TABLE 21-7: GPIO INTERRUPT/WAKE EVENT TIMING PARAMETERS**

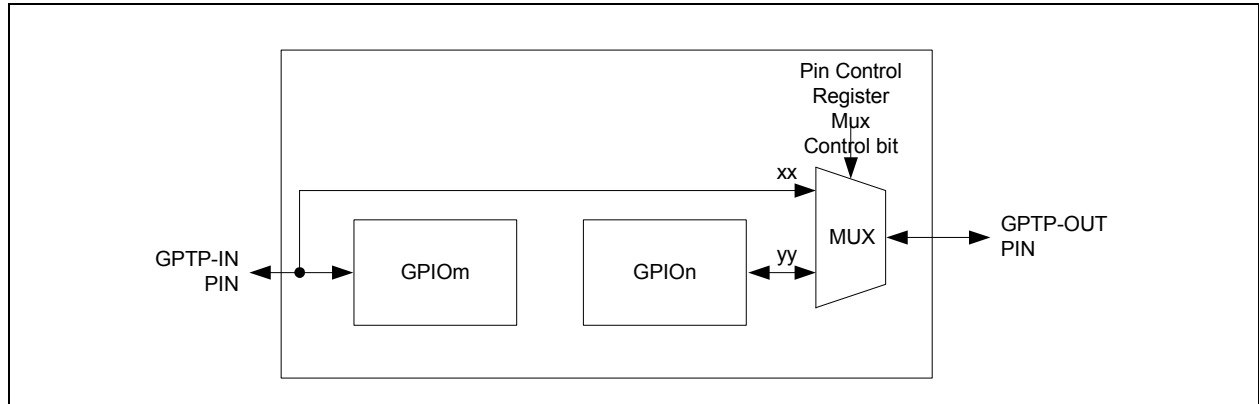
Symbol	Parameter	Limits			Unit	Comments
		Min	Nom	Max		
t <sub>WAKE</sub>	Wake Event Time	5	—	—	ns	Minimum pulse required on a GPIO pin to wake the system.
t <sub>INTERRUPT</sub>	Interrupt Status Assertion Time	300	—	—	ns	Minimum pulse required on a GPIO pin to assert an IRQ Status bit.

## 21.8 GPIO Pass-Through Ports

**GPIO Pass-Through Ports** (GPTP) can multiplex two general purpose I/O pins as shown in [Figure 21-3](#). **GPIO Pass-Through Ports** connect the GPTP-IN pin to the GPTP-OUT pin. The GPTP are sequentially assigned values 0:7. The GPTP port assignment have no relation to the [GPIO Indexing](#) assignments. The GPTP ports are controlled by the **MUX Control** bits in the **Pin Control Register** associated with the GPTP-OUT signal function.

In order to enable the GPTP Pass-Through Mode, the GPTP-IN (GPIOm in [Figure 21-3](#)) **Pin Control Register** must assign **MUX Control** = 00 (GPIO) and the **GPIO Direction** bit = 0 (input); the GPTP-OUT (GPIOn in [Figure 21-3](#)) **Pin Control Register** must assign **MUX Control** = the GPTP\_OUT signal function and **GPIO Direction** bit = 1 (output). The **MUX Control** = GPTP-OUT signal function can differ from pin to pin See [Section 21.4, "GPIO Indexing," on page 400](#).

**FIGURE 21-3: GPIO PASS-THROUGH PORT EXAMPLE**



**Note:** The Pin Control Register Mux Control fields shown in [Figure 21-3](#) are illustrated as 'xx' and 'yy' because this figure is an example, it does not represent the actual GPIO multiplexing configuration. The GPIO Multiplexing tables in this chapter must be used to determine the correct values to use to select between a GPIO and the pass-through.

**Note 21-1** When Pass-Through Mode is enabled, the GPIOn output is disconnected from the GPIOn pin and the GPIOm pin signal appears on GPIOn pin. Note that in this case the GPIOm input register still reflects the state of the GPIOm pin.

## 21.9 Registers

The **GPIO Interface Registers** include **GPIO Input Registers**, **GPIO Output Registers** and a **Pin Control Register** for each signal function. The **GPIO Interface** has its own Logical Device Number and Base Address as indicated in [Table 21-8](#).

[Table 21-9](#) is a register summary for the **GPIO Interface**. Each EC address is indicated as an SPB Offset from its AHB base address.

**TABLE 21-8: GPIO Interface BASE ADDRESS TABLE**

GPIOs Blocks	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
GPIOs	31h	F0_C400h

# MEC1618/MEC1618i

TABLE 21-9: GPIO Interface REGISTER SUMMARY

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
Pin Control Register	000h - 200h	0-3	R/W	
GPIO[000:036] Output Register	280h	0-3	R/W	
GPIO[040:076] Output Register	284h	0-3	R/W	
GPIO[100:136] Output Register	288h	0-3	R/W	
GPIO[140:176] Output Register	28Ch	0-3	R/W	
GPIO[200:236] Output Register	290h	0-3	R/W	
GPIO[000:036] Input Register	300h	0-3	R/W	
GPIO[040:076] Input Register	304h	0-3	R/W	
GPIO[100:136] Input Register	308h	0-3	R/W	
GPIO[140:176] Input Register	30Ch	0-3	R/W	
GPIO[200:236] Input Register	310h	0-3	R/W	

**APPLICATION NOTE:** Bit31 in the five GPIO input registers (GPIO[200:236] Input Register, GPIO[140:176] Input Register, GPIO[100:136] Input Register, GPIO[040:076] Input Register, GPIO[000:036] Input Register) is a single bit register that can be set or cleared by software. It is provided to enable the use of the NORM instruction in the ARC instruction set in order to quickly find the first set or cleared bit in the register. Setting Bit31 to '0b' makes the NORM instruction find the first set bit; setting Bit31 to '1b' makes the NORM instruction find the first cleared bit. For example, if Bit31 of GPIO[000:036] Input Register is '0b' and Bit17 is the highest numbered bit position for which a GPIO signal is set high, then the instruction sequence:

```
LD      R0, [GPIO[000:036] Input Register]
NORM    R0, R0
```

will return the value 17 in ARC register R0.

## 21.9.1 PIN CONTROL REGISTER

TABLE 21-10: PIN CONTROL REGISTER

Offset	See Note 21-2			
Bits	Description	Type	Default	Reset Event
31:25	RESERVED	RES	-	-
24	GPIO input from pad On reads, Bit [24] reflects the state of GPIO input from the pad regardless of setting of Bit [10].	R	Note 21-2	nSYS_ RST

TABLE 21-10: PIN CONTROL REGISTER (CONTINUED)

Offset	See Note 21-2			
Bits	Description	Type	Default	Reset Event
23:17	RESERVED	RES	-	-
16	<b>Alternative GPIO Data</b> The <a href="#">Alternative GPIO Data</a> bit can be used to determine the level on the GPIO pin when configured as an output depending on the state of the <a href="#">Output GPIO Write Enable</a> (see also <a href="#">Table 21-11</a> ).  If <a href="#">Output GPIO Write Enable</a> = 1, the GPIO[x] output is unaffected by this bit.  If <a href="#">Output GPIO Write Enable</a> = 0: 0 = GPIO[x] out = '0' 1 = GPIO[x] out = '1'  Reads of the <a href="#">Alternative GPIO Data</a> bit return the value on the pin.	R/W	<a href="#">Note 21-2</a>	nSYS_RST
15:14	RESERVED	RES	-	-
13:12	<b>MUX Control</b> The MUX Control field determines the active signal function for a pin.  00 = GPIO Function Selected 01 = Signal Function 1 Selected 10 = Signal Function 2 Selected 11 = Signal Function 3 Selected	R/W	<a href="#">Note 21-2</a>	nSYS_RST
11	<b>Polarity</b> 0 = Non-inverted 1 = Inverted  When the Polarity bit is set to '1' and the <a href="#">MUX Control</a> bits are greater than '00,' the selected signal function outputs are inverted and <a href="#">Interrupt Detection (int_det)</a> sense defined in <a href="#">Table 21-12</a> , " <a href="#">Edge Enable and Interrupt Detection Bits Definition</a> " is inverted. When the <a href="#">MUX Control</a> field selects the GPIO signal function (MUX = '00'), the Polarity bit does not effect the output. Regardless of the state of the <a href="#">MUX Control</a> field and the Polarity bit, the state of the pin is always reported without inversion in the GPIO input register. See <a href="#">FIGURE 21-1: GPIO Interface Block Diagram on page 399</a>	R/W	<a href="#">Note 21-2</a>	nSYS_RST

# MEC1618/MEC1618i

TABLE 21-10: PIN CONTROL REGISTER (CONTINUED)

Offset	See <a href="#">Note 21-2</a>			
Bits	Description	Type	Default	Reset Event
10	<p><b>Output GPIO Write Enable</b> Every GPIO has two mechanisms to set a GPIO data output: Output GPIO Bit and <a href="#">Alternative GPIO Data</a> bit.</p> <p><b>Note:</b> See <a href="#">Section 21.9.2, "GPIO Output Registers,"</a> on <a href="#">page 411</a> for a description of the GPIO Output registers.</p> <p>The <a href="#">Output GPIO Write Enable</a> bit is used to determine the source of the GPIO output (<a href="#">Table 21-11</a>).</p> <p>0 = <a href="#">Alternative GPIO Data</a> write enabled When this bit is zero the <a href="#">Alternative GPIO Data</a> write is enabled and the Output GPIO is disabled.</p> <p>1 = Output GPIO enable When this bit is one the <a href="#">Alternative GPIO Data</a> write is disabled and the Output GPIO is enabled.</p> <p><b>Note:</b> See description in <a href="#">Section 21.10, "Programmer's Notes,"</a> on <a href="#">page 418</a>.</p>	R/W	<a href="#">Note 21-2</a>	nSYS_RST
9	<p><b>GPIO Direction</b> 0 = Input 1 = Output</p> <p>The <a href="#">GPIO Direction</a> bit controls the buffer direction only when the <a href="#">MUX Control</a> field is '00' selecting the pin signal function to be GPIO. When the <a href="#">MUX Control</a> field is greater than '00' (i.e., a non-GPIO signal function is selected) the <a href="#">GPIO Direction</a> bit has no affect and the selected signal function logic directly controls the pin direction.</p>	R/W	<a href="#">Note 21-2</a>	nSYS_RST
8	<p><b>Output Buffer Type</b> 0 = Push-Pull 1 = Open Drain</p> <p><b>Note:</b> Unless explicitly stated otherwise, pins with (I/O/OD) or (O/OD) in their buffer type column in the tables in are compliant with the following Programmable OD/PP Multiplexing Design Rule: Each compliant pin has a programmable open drain/push-pull buffer controlled by the Output Buffer Type bit in the associated <a href="#">Pin Control Register</a>. The state of this bit controls the mode of the interface buffer for all selected functions, including the GPIO function.</p>	R/W	<a href="#">Note 21-2</a>	nSYS_RST



**TABLE 21-10: PIN CONTROL REGISTER (CONTINUED)**

Offset	See <a href="#">Note 21-2</a>			
Bits	Description	Type	Default	Reset Event
7	<b>Edge Enable (edge_en)</b> 0 = Edge detection disabled 1 = Edge detection enabled  <b>Note:</b> See <a href="#">Table 21-12</a> , "Edge Enable and Interrupt Detection Bits Definition" and <a href="#">Table 21-13</a> , "GPIO Interrupt Programming Procedure".  <b>Note:</b> In order to put the pin in its lowest power state, the Edge Enable bit should be set to '0', and the <a href="#">Interrupt Detection (int_det)</a> field set to one of the edge triggered values (101b - 111b). This combination guarantees that no interrupt will be generated and that no wakeup function will be enabled.	R/W	<a href="#">Note 21-2</a>	nSYS_RST
6:4	<b>Interrupt Detection (int_det)</b>  The interrupt detection bits determine the event that generates a GPIO event.  <b>Note:</b> See <a href="#">Table 21-12</a> , "Edge Enable and Interrupt Detection Bits Definition" and <a href="#">Table 21-13</a> , "GPIO Interrupt Programming Procedure".	R/W	<a href="#">Note 21-2</a>	nSYS_RST
3:2	<b>Power Gating Signals</b> The GPIO pin will be tristated when the selected power well is off. 00 = VTR 01 = VCC 10 = RESERVED 11 = RESERVED	R/W	<a href="#">Note 21-2</a>	nSYS_RST
1:0	<b>PU/PD (PU_PD)</b> These bits are used to enable an internal pull-up. 00 = None 01 = Pull Up Enabled 10 = Pull Down Enabled 11 = None	R/w	<a href="#">Note 21-2</a>	nSYS_RST

**Note 21-2** See [Section 21.5](#), "Pin Multiplexing Control," on [page 401](#) for the offset and default values for each Pin Control Register.

**TABLE 21-11: GPIO OUTPUT SOURCE SELECTION**

Output GPIO Write Enable	GPIO Output Register ( <a href="#">Note 21-3</a> )	Alternative GPIO Data	Description
0	READ-ONLY	READ/WRITE	State of the <a href="#">Alternative GPIO Data</a> bit determines the state of the GPIO output pin.
1	READ/WRITE	READ-ONLY	State of the appropriate bit in the <a href="#">GPIO Output Registers</a> determines the state of the GPIO output pin.

# MEC1618/MEC1618i

**Note 21-3** See [Section 21.9.2, "GPIO Output Registers,"](#) on page 411.

**TABLE 21-12: EDGE ENABLE AND INTERRUPT DETECTION BITS DEFINITION**

D7	D6	D5	D4	Selected Function
0	0	0	0	Low Level Sensitive
0	0	0	1	High Level Sensitive
0	0	1	0	Reserved
0	0	1	1	Reserved
0	1	0	0	Interrupt events are disabled
0	1	0	1	Reserved
0	1	1	0	Reserved
0	1	1	1	Reserved
1	1	0	1	Rising Edge Triggered
1	1	1	0	Falling Edge Triggered
1	1	1	1	Either edge triggered

**Note:** Only edge triggered interrupts can wake up the [20 MHz Oscillator](#). The GPIO must be configured for edge-triggered interrupts (Interrupt Detection set to 101b - 111b), edge-triggered interrupts must be enabled (Edge Enable set to 1b) and the GPIO interrupt must be enabled in the interrupt aggregator to generate a wake event.

**APPLICATION NOTE:** To prevent any adverse affects from spurious interrupts when changing the [Edge Enable \(edge\\_en\)](#) and [Interrupt Detection \(int\\_det\)](#) bits, follow the steps defined in [Table 21-13, "GPIO Interrupt Programming Procedure"](#).

**TABLE 21-13: GPIO INTERRUPT PROGRAMMING PROCEDURE**

	Step	Description
1.	Disable	Disable the ARC interrupts to prevent false ISR calls
2.	Configure	Configure the appropriate GPIO interrupts using the <a href="#">Edge Enable (edge_en)</a> and <a href="#">Interrupt Detection (int_det)</a> bits
3.	Sync	Execute an ARC Sync instruction to allow I/O operations to complete before executing subsequent instructions
4.	Clear	Clear the appropriate GPIO interrupt source bits in the <a href="#">EC Interrupt Aggregator</a>
5.	Enable	Enable the ARC interrupts

## 21.9.2 GPIO OUTPUT REGISTERS

See [Table 21-11, “GPIO Output Source Selection,”](#) on page 409.

### 21.9.2.1 Output GPIO[000:036]

**TABLE 21-14: GPIO[000:036] OUTPUT REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> and <a href="#">Table 21-9 on page 406</a>			32		EC SIZE		
POWER	VTR			0000_0000h		<a href="#">nSYS_RST</a> DEFAULT		
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	See <a href="#">Table 21-15 on page 411</a>							
BIT NAME	GPIO[000:036] Output							

**TABLE 21-15: BIT DEFINITIONS GPIO[000:036] OUTPUT REGISTER**

Bit	EC Type	Signal Name	Description
[7:0]	R/W	GPIO[007:000]	Each bit monitors the corresponding pins status.
[15:8]	R/W	GPIO[017:010]	
[23:16]	R/W	GPIO[027:020]	
[26:24]	R/W	GPIO[032:030]	
[30:27]	R	GPIO[036:033]	Reserved
31	R	Reserved	This bit is always reserved

# MEC1618/MEC1618i

## 21.9.2.2 Output GPIO[040:076]

**TABLE 21-16: GPIO[040:076] OUTPUT REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32		EC SIZE		
POWER	VTR			0000_000h		<a href="#">nSYS_RST</a> DEFAULT		
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	See <a href="#">Table 21-17 on page 412</a>							
BIT NAME	GPIO[040:076] Output							

**TABLE 21-17: BIT DEFINITIONS GPIO[040:076] OUTPUT REGISTER**

Bit	EC Type	Signal Name	Description
[7:0]	R/W	GPIO[047:040]	Each bit monitors the corresponding pins status
[15:8]	R/W	GPIO[057:050]	
[23:16]	R/W	GPIO[067:060]	
[30:24]	R	GPIO[076:070]	Reserved
31	R	Reserved	This bit is always reserved

## 21.9.2.3 Output GPIO[100:136]

**TABLE 21-18: GPIO[100:136] OUTPUT REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32		EC SIZE		
POWER	VTR			0000_0000h		<a href="#">nSYS_RST</a> DEFAULT		
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	See <a href="#">Table 21-19 on page 413</a>							
BIT NAME	GPIO[100:136] Output							

**TABLE 21-19: BIT DEFINITIONS GPIO[100:136] OUTPUT REGISTER**

Bit	EC Type	Signal Name	Description
[7:0]	R/W	GPIO[107:100]	Each bit monitors the corresponding pins status.
[15:8]	R/W	GPIO[117:110]	
[23:16]	R/W	GPIO[127:120]	
[26:24]	R/W	GPIO[132:130]	
[30:27]	R	GPIO[136:133]	Reserved
31	R	Reserved	This bit is always reserved

# MEC1618/MEC1618i

## 21.9.2.4 Output GPIO[140:176]

**TABLE 21-20: GPIO[140:176] OUTPUT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	See <a href="#">Table 21-21 on page 414</a>							
BIT NAME	GPIO[140:176] Output							

**TABLE 21-21: BIT DEFINITIONS GPIO[140:176] OUTPUT REGISTER**

Bit	EC Type	Signal Name	Description
[7:0]	R/W	GPIO[147:140]	Each bit monitors the corresponding pins status.
[15:8]	R/W	GPIO[157:150]	
[20:16]	R/W	GPIO[164:160]	
[23:21]	R	GPIO[167:165]	Reserved
[30:24]	R	GPIO[176:170]	
31	R	Reserved	This bit is always reserved

## 21.9.2.5 Output GPIO[200:236]

TABLE 21-22: GPIO[200:236] OUTPUT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32			EC SIZE	
POWER	VTR			0000_0000h			nSYS_RST DEFAULT	
	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	See <a href="#">Table 21-23 on page 415</a>							
BIT NAME	GPIO[200:236] Output							

TABLE 21-23: BIT DEFINITIONS GPIO[200:236] OUTPUT REGISTER

Bit	EC Type	Signal Name	Description
[7:0]	R/W	GPIO[207:200]	Each bit monitors the corresponding pins status.
[15:8]	R/W	GPIO[217:210]	
[23:16]	R	GPIO[227:220]	Reserved
[30:24]	R	GPIO[236:230]	
31	R	Reserved	This bit is always reserved

## 21.9.3 GPIO INPUT REGISTERS

## 21.9.3.1 Overview

The MEC1618/MEC1618i [GPIO Interface](#) includes four [GPIO Input Registers](#) which are always active as illustrated in [FIGURE 21-1: GPIO Interface Block Diagram on page 399](#). The [GPIO Input Registers](#) can always be used to read the state of a pin (excluding the [NORM Exception Bits](#)), even when the pin is in an output mode and/or when a signal function other than the GPIO signal function is selected; i.e., the [Pin Control Register MUX Control](#) bits are not equal to '00.'

## 21.9.3.2 NORM Exception Bits

There can be up to 31 GPIO signal function names in each of the [GPIO Input Registers](#) (Bits D0 - D30). Bit D31 in each of these registers, the [NORM Exception Bits](#), are single bit registers that can be set or cleared by software. The [NORM Exception Bits](#) are provided to enable the use of the NORM instruction in the ARC instruction set in order to quickly find the first set or cleared bit in the register. Setting Bit D31 to '0b' makes the NORM instruction find the first set bit; setting Bit D31 to '1b' makes the NORM instruction find the first cleared bit. For example, if Bit D31 of [GPIO\[000:036\] Input Register](#) is '0b' and Bit17 is the highest numbered bit position for which a GPIO signal is set high, then the following instruction sequence will return the value 17 in ARC register R0.

```
LD      R0, [GPIO\[000:036\] Input Register]
NORM    R0, R0
```

# MEC1618/MEC1618i

## 21.9.3.3 Input GPIO[000:036]

TABLE 21-24: GPIO[000:036] INPUT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32			EC SIZE	
POWER	VTR			0000_0000h			<a href="#">nSYS_RST</a> DEFAULT	
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[000:036] INPUT						

## 21.9.3.4 Input GPIO[040:076]

TABLE 21-25: GPIO[040:076] INPUT REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32			EC SIZE	
POWER	VTR			0000_0000h			<a href="#">nSYS_RST</a> DEFAULT	
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[040:076] INPUT						



## 21.9.3.5 Input GPIO[100:136]

**TABLE 21-26: GPIO[100:136] INPUT REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32		EC SIZE		
POWER	VTR			0000_0000h		<a href="#">nSYS_RST</a> DEFAULT		
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[100:136] INPUT						

## 21.9.3.6 Input GPIO[140:176]

**TABLE 21-27: GPIO[140:176] INPUT REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32		EC SIZE		
POWER	VTR			0000_0000h		<a href="#">nSYS_RST</a> DEFAULT		
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[140:176] INPUT						

# MEC1618/MEC1618i

## 21.9.3.7 Input GPIO[200:236]

**TABLE 21-28: GPIO[200:236] INPUT REGISTER**

HOST ADDRESS	N/A					HOST SIZE		
EC OFFSET	See <a href="#">Table 21-8 on page 405</a> - <a href="#">Table 21-9 on page 406</a>			32		EC SIZE		
POWER	VTR			0000_0000h		<a href="#">nSYS_RST</a> DEFAULT		
	<a href="#">EC SPB</a>							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	NORM	GPIO[200:236] INPUT						

## 21.10 Programmer's Notes

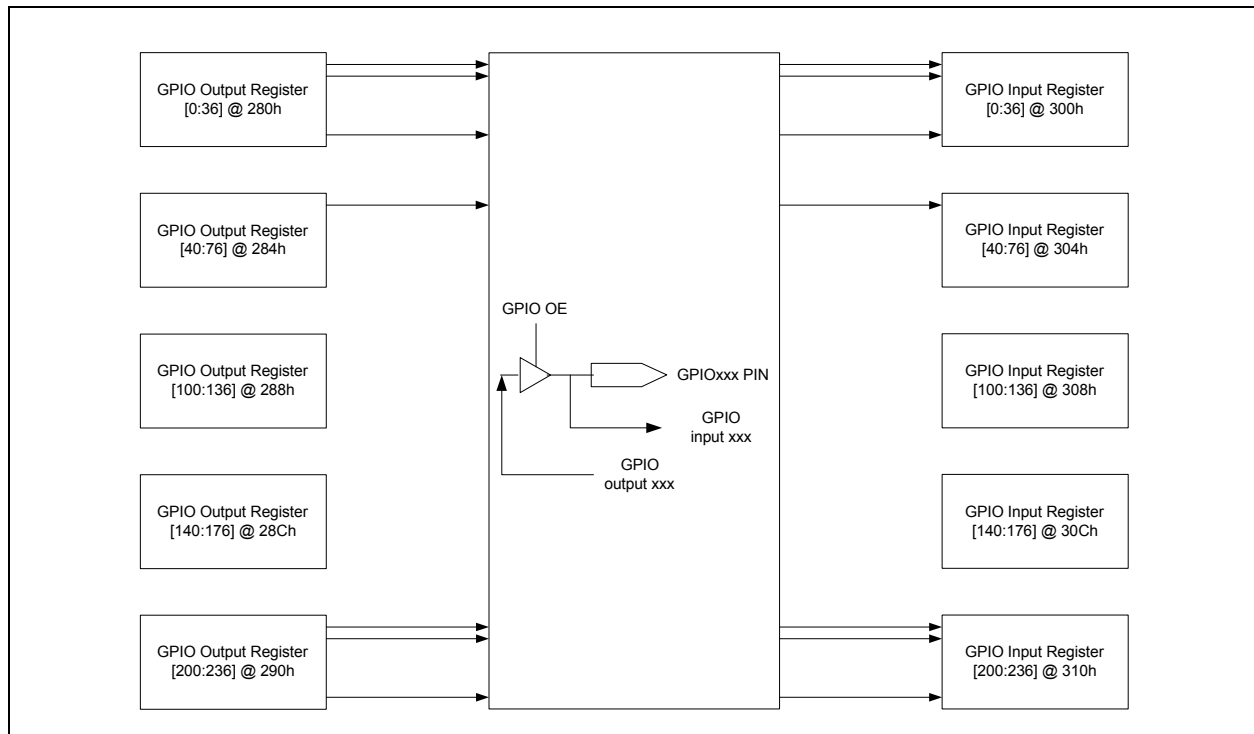
As mentioned in [Section 21.1](#) there are two ways to access GPIO input and output pin signals. This note aims to describe these access mechanisms and their relative merits. The two schemes can be employed concurrently.

In the following description, the terms pins, ports, lines, signals are used interchangeably.

### 21.10.1 ACCESSING GPIO - MECHANISM 1

#### 21.10.1.1 Overview

In the legacy approach, referred to hereafter as Mechanism 1, GPIO lines are grouped into 5 logical groups of 32 each. Associated with each group is an output register that controls the value to be driven out to the GPIO lines in the group and an input register whose value reflects the value seen on the group's GPIO lines' input side. Each non-reserved bit in the input/output register corresponds to a GPIO line. Since the number of GPIO lines that are implemented is less than that indicated by the logical labeling (0 to 236 in octal notation), some register bits are reserved. [Figure 21-4](#) illustrates the arrangements.

**FIGURE 21-4: ACCESSING GPIOs - SCHEME 1 (LEGACY SCHEME)**

### 21.10.1.2 Access Mechanism

To change the output value of a particular GPIO line, the software first reads the GPIO Output Register associated with the group the line belongs to. The output register's value is logically (bitwise) ORed with a 32-bit value formed by setting the bit corresponding the GPIO line to the desired value while keeping all other bits zero. The result is then written to the same output register. Note that the value on GPIO output is as specified by its register bit value and is not affected by setting of **Polarity** in the **Pin Control Register**, which affects only the polarity of Alternate Function signal multiplexed on the GPIO pin. GPIO lines belonging to the same group can be changed together in one step.

To monitor the value on the input side of GPIO buffer, read the GPIO Input Register associated with the group the GPIO line belongs to. For example, GPIO102 corresponds to Bit 2 in GPIO Input Register at offset 308h. Bit value can be extracted by applying appropriate mask and/or shift. In the current example, the mask would be 0x4; the shift, >>2.

Bit 31 in each GPIO Input Register, the **NORM Exception Bits**, is not associated with a GPIO line, rather it is to be used in conjunction with the ARC's NORM instruction to quickly locate the first bit in the register that is set or cleared. See [Section 21.9.3.2](#) for details.

### 21.10.1.3 Applicability

Scheme 1 is simple and allows updates to multiple GPIO lines with one register update. However, when there is more than one software process accessing GPIO lines that share the same register, there is the potential for conflicts. In such cases the test-and-set sequence described in [Section 21.10.1.2](#) must be an atomic operation. Since there is no support for this, software must implement a synchronization mechanism, e.g., mutex, semaphore, or spin lock, to serialize accesses.

## 21.10.2 ACCESSING GPIO - MECHANISM 2

### 21.10.2.1 Overview

In this mechanism each GPIO port is individually - as opposed to in groups of 32 as in Mechanism 1 -accessed via its [Pin Control Register](#). The mechanism is enabled on a per-port basis. After being enabled, it overrides Mechanism 1, i.e., a write to a GPIO Output Register will not affect ports for which Mechanism 2 have been enabled. The mechanism can be dynamically enabled and disabled.

### 21.10.2.2 Access Mechanism

[Figure 21-2](#) shows how relevant bits in the [Pin Control Register](#) are used to effect GPIO accesses in Mechanism 2.

To change a GPIO output port, the [Output GPIO Write Enable](#) bit in its [Pin Control Register](#) is set to enable Mechanism 2, and the [Alternative GPIO Data](#) bit is set to the port's desired output value. Both bits can be updated with a single register write. As long as [Output GPIO Write Enable](#) is cleared ('0'), Mechanism 2 is in effect. This bit must be set ('1') to switch to Mechanism 1.

The state of GPIO input port is reflected by the [GPIO input from pad](#) bit in the [Pin Control Register](#). This holds true regardless of whether Mechanism 2 is enabled.

The port's input, output, and programmed output values are available in the same [Pin Control Register](#).

### 21.10.2.3 Applicability

Mechanism 2 provides accesses at the port level and hence greatly reduces the unnecessary sharing of GPIO ports among different processes or threads. Even when a port is shared, the per-port control eliminates the need for atomic test-and-set.

Unless the applications call for the ability to update more than one GPIO port at the same time, it is recommended that Mechanism 2 be used. Alternatively, the two mechanisms can be together, with Mechanism 1 be applied to ports that do not need exclusive accesses.

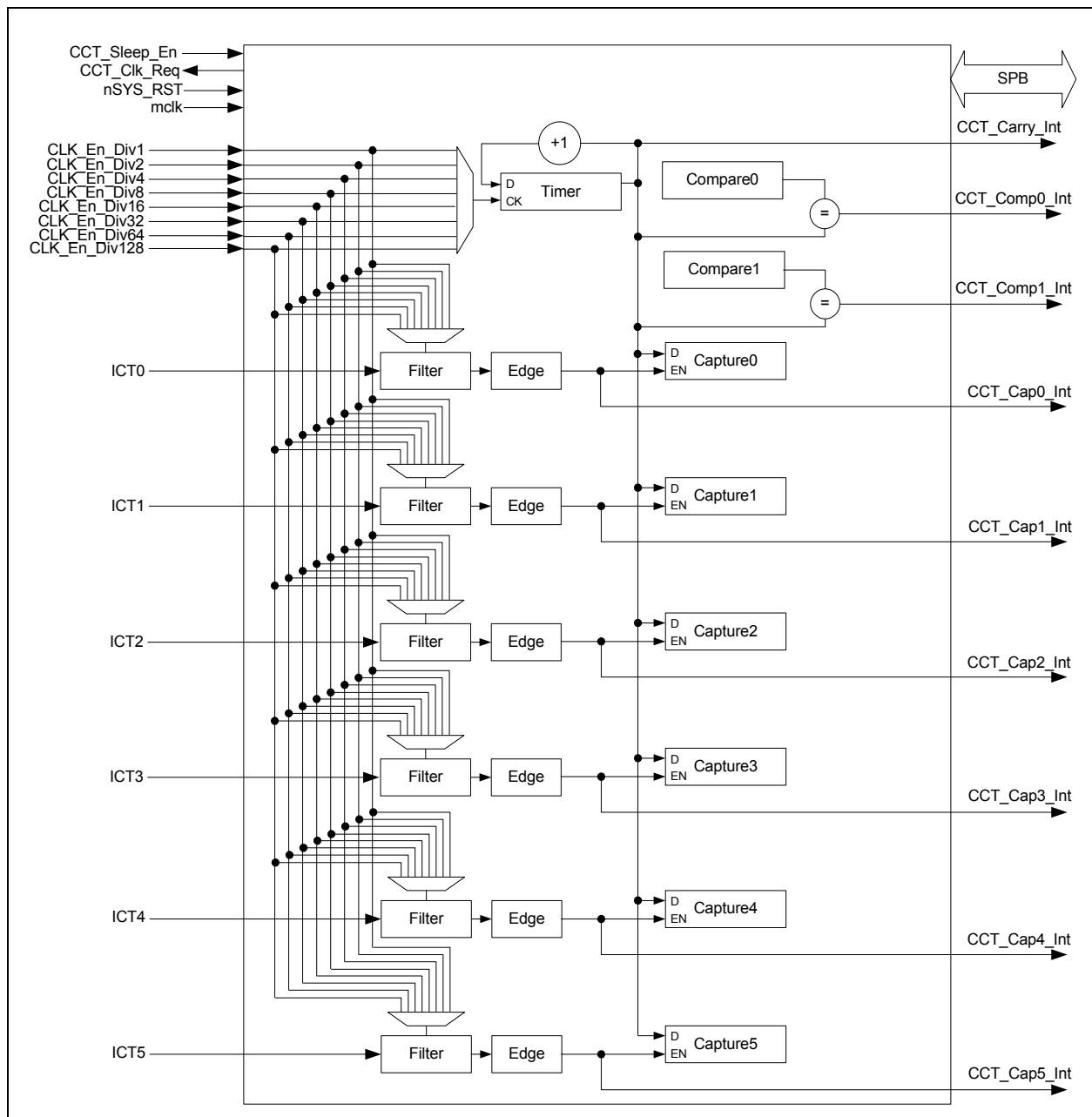
## 22.0 INPUT CAPTURE AND COMPARE TIMER

### 22.1 General Description

The Input Capture and Compare Timers block contains a 32-bit timer running at the main system clock frequency. The timer is free-running and is associated with six 32-bit capture registers and two compare registers. Each capture register can record the value of the free-running timer based on a programmable edge of its associated input pin. An interrupt can be generated for each capture register each time it acquires a new timer value. The timer can also generate an interrupt when it automatically resets and can additionally generate two more interrupts when the timer matches the value in either of two 32-bit compare registers.

### 22.2 Capture and Compare Timer Block Diagram

FIGURE 22-1: CAPTURE AND COMPARE TIMER BLOCK DIAGRAM



# MEC1618/MEC1618i

## 22.3 Block Diagram Signal List

TABLE 22-1: Input Capture and Compare Timer SIGNAL LIST

Signal Name	Direction	Description
SPB	I/O Bus	MEC1618/MEC1618i peripheral bus
MCLK	INPUT	Master MEC1618/MEC1618i clock
nSYS_RST	INPUT	Block reset signal
CLK_EN_DIV1	INPUT	MCLK clock enable
CLK_EN_DIV2	INPUT	MCLK/2 clock enable
CLK_EN_DIV4	INPUT	MCLK/4 clock enable
CLK_EN_DIV8	INPUT	MCLK/8 clock enable
CLK_EN_DIV16	INPUT	MCLK/16 clock enable
CLK_EN_DIV32	INPUT	MCLK/32 clock enable
CLK_EN_DIV64	INPUT	MCLK/64 clock enable
CLK_EN_DIV128	INPUT	MCLK/128 clock enable
ICT[5:0]	INPUT	External capture trigger signals (Note 22-1)
CCT_CARRY_INT	OUTPUT	Free-running timer wraparound interrupt
CCT_COMP[1:0]_INT	OUTPUT	Timer compare interrupts
CCT_CAP[5:0]_INT	OUTPUT	Timer capture interrupts
CCT_SLEEP_EN	INPUT	External enable/disable signal used to put the block in the lowest power consumption state. 0=No Sleep Requested. The block should operate as configured. 1=Sleep Requested. The block enters sleep mode. See <a href="#">Low Power Mode on page 423</a> .
CCT_CLK_REQ	OUTPUT	This output indicates when this block requires this clock input. 0= MCLK can be turned 'off' when appropriate 1= MCLK is required to be 'on.'

**Note 22-1** External capture trigger signal inputs ICT0 - ICT5 are identical to FAN\_TACH0 - FAN\_TACH5.

## 22.4 Power, Clocks and Reset

### 22.4.1 POWER DOMAIN

This block is powered by the VTR power supply.

### 22.4.2 CLOCKS

The timer in this unit is driven by mclk, the main system clock.

### 22.4.3 RESET

This block is reset on a nSYS\_RST. On nSYS\_RST the timer and all Capture and Compare registers are reset to their default values. The timer is also reset by the [Free\\_Reset](#) bit in the [Capture and Compare Timer Control Register](#).

## 22.5 Interrupts

Interrupts from the Input Capture and Compare Timer block are routed to [GIRQ23 Source Register](#) of the Interrupt Aggregator. There are a total of nine interrupts from this block: one each for the six capture, one each for the two compare registers and one when the Free Running Timer wraps around. The interrupt signals are always generated by this block and can be queried or enabled through the Source and Enable registers in the Interrupt Aggregator.

## 22.6 Low Power Mode

This block is designed to conserve power when it is either sleeping or disabled. There are two ways to put the Capture and Compare block into a low power mode: Disable the [Activate](#) Bits or assert the CCT\_SLEEP\_EN signal to the Capture and Compare Timer block. The following table summarizes the Capture and Compare Timer behavior for each of these low power modes.

**TABLE 22-2: BLOCK CLOCK GATING IN LOW POWER MODES**

Activate Bits	CCT_SLEEP_En	Block Idle Status	CCT_CLK_REQ	State	Description
All 0	X	NOT IDLE	1	PREPARING to SLEEP	The core clock is required until the current transaction is completed and the Block is IDLE.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.
Any 1	0	X	1	NORMAL OPERATION	The block is neither disabled by firmware nor commanded to SLEEP
	1	NOT IDLE	1	PREPARING to SLEEP	The core clock is required until the current transaction is completed and the Block is IDLE.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.

## 22.7 Noise Filter

The noise filter uses Filter Clock (FCLK) to filter the signal on the ICTx pins. An external ICTx pin must remain in the same state for three FCLK ticks before the internal state changes. The Filter Bypass bit is used to bypass the noise filter. Each ICT input capture register can individually bypass the filter, but all ICT input capture registers that use the filter use the same Filter Clock.

- The signal ICT may be optionally only synchronized, or synchronized and filtered depending on the filter bypass bit.
- The minimum FCLK period must be at least 2X the duration of the ICT signal so that signal can be reliably captured in the bypass mode.
- The minimum FCLK period must be at least 4X the duration of the ICT signal so that signal can be reliably captured in the non-bypass mode.

## 22.8 Operation

### 22.8.1 INPUT CAPTURE

The Input Capture block consists of a free-running 32-bit timer and 6 capture registers. Each of the capture registers is associated with an input pin as well as an interrupt source bit in the Interrupt Aggregator: [Table 22-3, "Pin Capture Interrupt Assignments"](#) shows the assignment of pins to the Capture registers:

**TABLE 22-3: PIN CAPTURE INTERRUPT ASSIGNMENTS**

Capture Register	Capture Pin	Interrupt
<a href="#">Capture 0 Register</a>	ICT0	<a href="#">CAPTURE 0, GIRQ23 Source Register</a>
<a href="#">Capture 1 Register</a>	ICT1	<a href="#">CAPTURE 1, GIRQ23 Source Register</a>
<a href="#">Capture 2 Register</a>	ICT2	<a href="#">CAPTURE 2, GIRQ23 Source Register</a>
<a href="#">Capture 3 Register</a>	ICT3	<a href="#">CAPTURE 3, GIRQ23 Source Register</a>
<a href="#">Capture 4 Register</a>	ICT4	<a href="#">CAPTURE 4, GIRQ23 Source Register</a>
<a href="#">Capture 5 Register</a>	ICT5	<a href="#">CAPTURE 5, GIRQ23 Source Register</a>

The Capture registers store the current value of the Free Running timer whenever the associated input signal changes, according to the programmed edge detection. An interrupt is also generated to the EC. The Capture registers are read-only. The registers are updated every time an edge is detected. If software does not read the register before the next edge, the value is lost.

### 22.8.2 COMPARE INTERRUPT GENERATION

There are two 32-bit Compare registers. Each of these registers can independently generate an interrupt to the EC when the 32-bit free running Capture timer matches the contents of the Compare register.

## 22.9 Input Capture and Compare Timers Register Summary

There is one instance of the [Input Capture and Compare Timer](#) block implemented in the MEC1618/MEC1618i.

**TABLE 22-4: [Input Capture and Compare Timer](#) BASE ADDRESS TABLE**

<a href="#">Input Capture and Compare Timer Instance</a>	LDN	AHB Base Address
<a href="#">Input Capture and Compare Timer</a>	2h	<a href="#">F0_0800h</a>



TABLE 22-5: Input Capture and Compare Timer REGISTER SUMMARY

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
Capture and Compare Timer Control Register	00h	3-0	R/W	
Capture Control 0 Register	04h	3-0	R/W	
Capture Control 1 Register	08h	3-0	R/W	
Free Running Timer Register	0Ch	3-0	R/W	
Capture 0 Register	10h	3-0	R	
Capture 1 Register	14h	3-0	R	
Capture 2 Register	18h	3-0	R	
Capture 3 Register	1Ch	3-0	R	
Capture 4 Register	20h	3-0	R	
Capture 5 Register	24h	3-0	R	
Compare 0 Register	28h	3-0	R/W	
Compare 1 Register	2Ch	3-0	R/W	

**Note:** The registers in this block with 32-bit values (Free Running Timer Register, Capture 0 Register, Capture 1 Register, Capture 2 Register, Capture 3 Register, Capture 4 Register, Capture 5 Register, Compare 0 Register, Compare 1 Register) should be read with 32-bit accesses. If these registers are read with multiple 8-bit or 16-bit accesses, the register values could change between the multiple accesses to the registers.

# MEC1618/MEC1618i

## 22.10 Detailed Register Descriptions

### 22.10.1 CAPTURE AND COMPARE TIMER CONTROL REGISTER

TABLE 22-6: CAPTURE AND COMPARE TIMER CONTROL REGISTER

HOST ADDRESS	n/a				n/a		HOST SIZE	
EC OFFSET	00h				32-bit		EC SIZE	
POWER	VTR				0000h		VTR POR DEFAULT	
BUS	EC SPB							
BYTE[3-2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R/W	R/W
BIT NAME	Reserved						Compare Enable1	Compare Enable0
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R/W	R	R/W	R/W	R/W
BIT NAME	Reserved	TCLK			Reserved	Free_Reset	Free_Enable	Activate

#### ACTIVATE

0=The timer block is powered down and all clocks are gated. (default).

1=The timer block is in a running state

#### FREE\_ENABLE

Free-Running Timer Enable. This bit is used to start and stop the free running timer. This bit does not reset the timer count. The timer starts counting at 0000\_0000h on reset and wraps around back to 0000\_0000h after it reaches FFFF\_FFFFh.

0=Timer is disabled. The [Free Running Timer Register](#) is writable.

1=Timer is enabled. The [Free Running Timer Register](#) is read-only.

**Note:** The [Free\\_Enable](#) bit is cleared after the RESET cycle is done. Firmware must poll the [Free\\_Reset](#) bit to determine when it is safe to re-enable the timer.

## FREE\_RESET

Free Running Timer Reset. This bit stops the timer and resets the internal counter to 0000\_0000h. This bit does not affect the [Free\\_Enable](#) bit. This bit is self clearing after the timer is reset.

0=Normal timer operation

1=Timer reset

## TCLK

This 3-bit field sets the clock source for the Free Running Counter (see [Section 22.10.4, "Free Running Timer Register," on page 431](#)). The available frequencies are shown in [Table 22-7](#):

**TABLE 22-7: FREE RUNNING TIMER CLOCK FREQUENCIES**

Timer Clock Select	Frequency Selected
000	<a href="#">MCLK</a>
001	<a href="#">MCLK/2</a>
010	<a href="#">MCLK/4</a>
011	<a href="#">MCLK/8</a>
100	<a href="#">MCLK/16</a>
101	<a href="#">MCLK/32</a>
110	<a href="#">MCLK/64</a>
111	<a href="#">MCLK/128</a>

## COMPARE\_ENABLE0

Compare Enable for [Compare 0 Register](#). If this bit is 1, a match between the [Compare 0 Register](#) and the [Free Running Timer Register](#) will cause an interrupt to be generated. If this bit is 0, no interrupt will be generated.

## COMPARE\_ENABLE1

Compare Enable for [Compare 1 Register](#). If this bit is 1, a match between the [Compare 1 Register](#) and the [Free Running Timer Register](#) will cause an interrupt to be generated. If this bit is 0, no interrupt will be generated.

# MEC1618/MEC1618i

## 22.10.2 CAPTURE CONTROL 0 REGISTER

TABLE 22-8: CAPTURE CONTROL 0 REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	04h			32-bit			EC SIZE	
POWER	VTR			0000h			VTR POR DEFAULT	
BUS	EC SPB							
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R	R/W	R/W	R/W
BIT NAME	FCLK_SEL3			Reserved		Filter Byp3	Capture_Edge3	
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R	R/W	R/W	R/W
BIT NAME	FCLK_SEL2			Reserved		Filter Byp2	Capture_Edge2	
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R	R/W	R/W	R/W
BIT NAME	FCLK_SEL1			Reserved		Filter Byp1	Capture_Edge1	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R	R/W	R/W	R/W
BIT NAME	FCLK_SEL0			Reserved		Filter Byp0	Capture_Edge0	

### CAPTURE\_EDGE0

Capture Timer value Edge Type Select. This field selects the edge type that triggers the capture of the free-running timer into [Capture 0 Register](#). See [Table 22-9](#).

**TABLE 22-9: CAPTURE EDGE SELECTION**

Capture Edge Select	Edge that Triggers Capture
00	Falling edges
01	Rising edges
10	Both rising and falling edges
11	Capture event disabled

## **FILTER BYP0**

Filter Bypass permits ICT0 to bypass the noise filter and go directly into the timer. A 0 enables the filter and a 1 bypasses the filter.

## **FCLK\_SEL0**

Filter Clock Select, used to determine the clock source to the input filter. Available frequencies are shown in [Table 22-10](#):

**TABLE 22-10: FILTER CLOCK FREQUENCIES**

Timer Clock Select	Frequency Selected
000	MCLK
001	MCLK/2
010	MCLK/4
011	MCLK/8
100	MCLK/16
101	MCLK/32
110	MCLK/64
111	MCLK/128

## **CAPTURE\_EDGE1**

Capture Timer value Edge Type Select. This field selects the edge type that triggers the capture of the free-running timer into [Capture 1 Register](#). See [Table 22-9](#).

## **FILTER BYP1**

Filter Bypass permits ICT1 to bypass the noise filter and go directly into the timer. A 0 enables the filter and a 1 bypasses the filter.

## **FCLK\_SEL1**

Filter Clock Select, used to determine the clock source to the input filter. Available frequencies are shown in [Table 22-10](#).

## **CAPTURE\_EDGE2**

Capture Timer value Edge Type Select. This field selects the edge type that triggers the capture of the free-running timer into [Capture 2 Register](#). See [Table 22-9](#).

## **FILTER BYP2**

Filter Bypass permits ICT2 to bypass the noise filter and go directly into the timer. A 0 enables the filter and a 1 bypasses the filter.

# MEC1618/MEC1618i

## FCLK\_SEL2

Filter Clock Select, used to determine the clock source to the input filter. Available frequencies are shown in [Table 22-10](#).

## CAPTURE\_EDGE3

Capture Timer value Edge Type Select. This field selects the edge type that triggers the capture of the free-running timer into [Capture 3 Register](#). See [Table 22-9](#).

## FILTER\_BYP3

Filter Bypass permits ICT3 to bypass the noise filter and go directly into the timer. A 0 enables the filter and a 1 bypasses the filter.

## FCLK\_SEL3

Filter Clock Select, used to determine the clock source to the input filter. Available frequencies are shown in [Table 22-10](#).

## 22.10.3 CAPTURE CONTROL 1 REGISTER

TABLE 22-11: CAPTURE CONTROL 1 REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	08h			32-bit			EC SIZE	
POWER	VTR			0000h			VTR POR DEFAULT	
BUS	EC SPB							
BYTE[3-2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R	R/W	R/W	R/W
BIT NAME	FCLK_SEL5			Reserved		Filter Byp5	Capture_Edge5	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R	R	R/W	R/W	R/W
BIT NAME	FCLK_SEL4			Reserved		Filter Byp4	Capture_Edge4	

## CAPTURE\_EDGE4

Capture Timer value Edge Type Select. This field selects the edge type that triggers the capture of the free-running timer into [Capture 4 Register](#). See [Table 22-9](#).

## FILTER BYP4

Filter Bypass permits ICT4 to bypass the noise filter and go directly into the timer. A 0 enables the filter and a 1 bypasses the filter.

## FCLK\_SEL4

Filter Clock Select, used to determine the clock source to the input filter. Available frequencies are shown in [Table 22-10](#).

## CAPTURE\_EDGE5

Capture Timer value Edge Type Select. This field selects the edge type that triggers the capture of the free-running timer into [Capture 5 Register](#). See [Table 22-9](#).

## FILTER BYP5

Filter Bypass permits ICT5 to bypass the noise filter and go directly into the timer. A 0 enables the filter and a 1 bypasses the filter.

## FCLK\_SEL5

Filter Clock Select, used to determine the clock source to the input filter. Available frequencies are shown in [Table 22-10](#).

## 22.10.4 FREE RUNNING TIMER REGISTER

**TABLE 22-12: FREE RUNNING TIMER REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	0Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Free Running Timer[31:0]							

## FREE RUNNING TIMER[31:0]

This register contains the current value of the Free Running Timer. A Capture Timer interrupt is signaled to the Interrupt Aggregator when this register transitions from FFFF\_FFFFh to 0000\_0000h.

When [Free\\_Enable](#) in [Capture and Compare Timer Control Register](#) is 1, this register is read-only. When [Free\\_Enable](#) is 0, this register may be written.

# MEC1618/MEC1618i

## 22.10.5 CAPTURE 0 REGISTER

TABLE 22-13: CAPTURE 0 REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	10h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Capture 0[31:0]							

### CAPTURE 0[31:0]

This register saves the value copied from the Free Running timer on a programmed edge of ICT0.

## 22.10.6 CAPTURE 1 REGISTER

TABLE 22-14: CAPTURE 1 REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	14h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Capture 1[31:0]							

### CAPTURE 1[31:0]

This register saves the value copied from the Free Running timer on a programmed edge of ICT1.



## 22.10.7 CAPTURE 2 REGISTER

**TABLE 22-15: CAPTURE 2 REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	18h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Capture 2[31:0]							

### CAPTURE 2[31:0]

This register saves the value copied from the Free Running timer on a programmed edge of ICT2.

## 22.10.8 CAPTURE 3 REGISTER

**TABLE 22-16: CAPTURE 3 REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	1Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Capture 3[31:0]							

### CAPTURE 3[31:0]

This register saves the value copied from the Free Running timer on a programmed edge of ICT3.

# MEC1618/MEC1618i

## 22.10.9 CAPTURE 4 REGISTER

TABLE 22-17: CAPTURE 4 REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	20h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Capture 4[31:0]							

### CAPTURE 4[31:0]

This register saves the value copied from the Free Running timer on a programmed edge of ICT4.

## 22.10.10 CAPTURE 5 REGISTER

TABLE 22-18: CAPTURE 5 REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	24h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Capture 5[31:0]							

### CAPTURE 5[31:0]

This register saves the value copied from the Free Running timer on a programmed edge of ICT5.

## 22.10.11 COMPARE 0 REGISTER

**TABLE 22-19: COMPARE 0 REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
BUS	EC SPB							
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Compare 0[31:0]							

### COMPARE 0[31:0]

A Compare 0 interrupt is generated when this register matches the value in the Free Running Timer.

## 22.10.12 COMPARE 1 REGISTER

**TABLE 22-20: COMPARE 1 REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	2Ch			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BIT	D31	D30	D29	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Compare 1[31:0]							

### COMPARE 1[31:0]

A Compare 1 interrupt is generated when this register matches the value in the Free Running Timer.

# MEC1618/MEC1618i

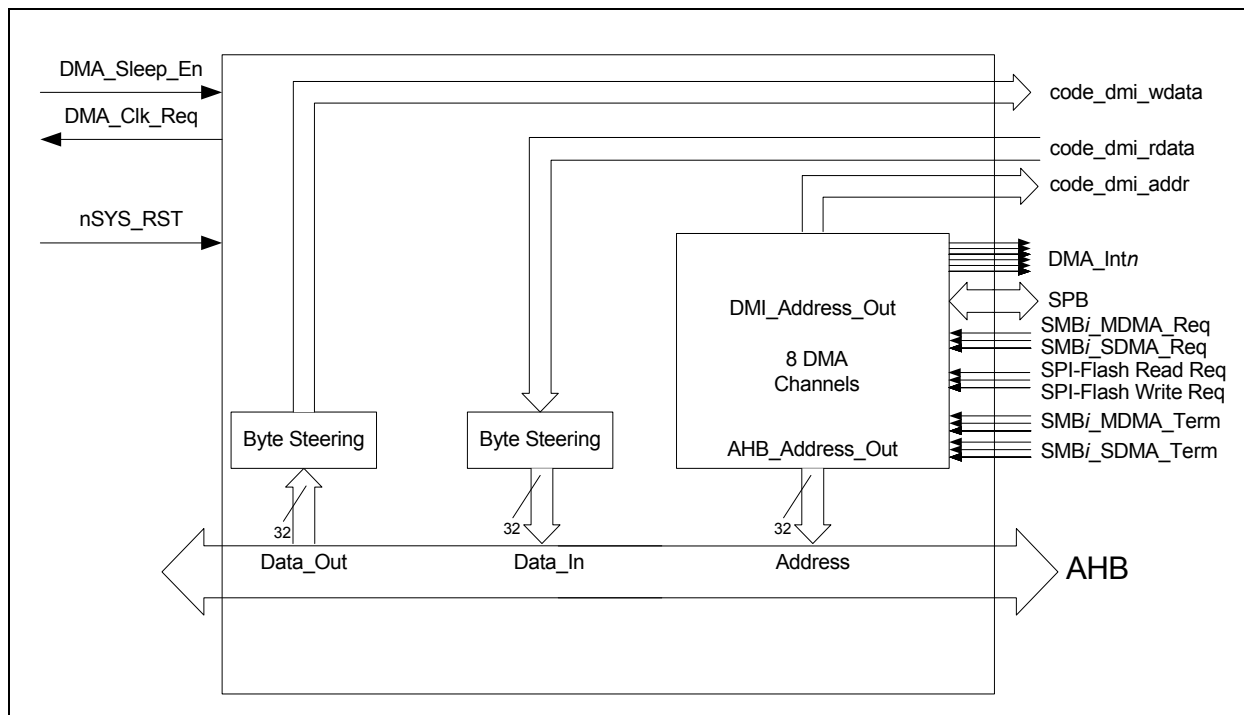
## 23.0 DMA CONTROLLER

### 23.1 General Description

This block describes the MEC1618/MEC1618i DMA controller. The DMA controller is designed to move data between the SMBus and SPI Flash controllers and the EC closely-coupled SRAM memory. There are eight independent channels that each move byte-wide data between the SMBus and SPI Flash controllers and the SRAM in either direction.

### 23.2 DMA Block Diagram

**FIGURE 23-1: DMA BLOCK DIAGRAM**



### 23.3 Block Diagram Signal List

**TABLE 23-1: DMA Controller SIGNAL LIST**

Signal Name	Direction	Description
AHB	I/O Bus	MEC1618/MEC1618i system bus
SPB	I/O Bus	MEC1618/MEC1618i peripheral bus
code_dmi_wdata, code_dmi_rdata, code_dmi_addr	I/O Bus	Direct Memory Interface (DMI) to ARC ICCM memory
MCLK	INPUT	Master MEC1618/MEC1618i clock
nSYS_RST	INPUT	Block reset signal
DMA_Int[7:0]	OUTPUT	DMA Interrupt signals
SMB[2:0]_MDMA_Req	INPUT	DMA request control from SMBus Master channel.

TABLE 23-1: DMA Controller SIGNAL LIST (CONTINUED)

Signal Name	Direction	Description
SMB[2:0]_SDMA_Req	INPUT	DMA request control from SMBus Slave channel.
SMB[2:0]_MDMA_Term	INPUT	DMA termination control from SMBus Master channel.
SMB[2:0]_SDMA_Term	INPUT	DMA termination control from SMBus Slave channel.
SPIFLASH_Read_Req	INPUT	DMA request control from SPI Flash receive Buffer register
SPIFLASH_Write_Req	INPUT	DMA request control from SPI Flash transmit Buffer register
DMA_SLEEP_EN	INPUT	External enable/disable signal used to put the block in the lowest power consumption state. 0=No Sleep Requested. The block should operate as configured. 1=Sleep Requested. The block enters sleep mode. See <a href="#">Low Power Mode on page 437</a> .
DMA_CLK_REQ	OUTPUT	This output indicates when this block requires this clock input. 0= <a href="#">MCLK</a> can be turned 'off' when appropriate 1= <a href="#">MCLK</a> is required to be 'on.'

## 23.4 Power, Clocks and Reset

### 23.4.1 POWER DOMAIN

This block is powered by the VTR power supply with a separate Analog supply (AVDD).

### 23.4.2 CLOCKS

This block has three clock inputs: [MCLK](#), the AHB bus clock enable, used for AHB transfers, and the EC clock enable, used for DMI transfers.

**APPLICATION NOTE:** The DMA Controller requires the EC clock in order to write into the EC SRAM. The EC, therefore, must not be in sleep mode at any time during a DMA transfer.

### 23.4.3 RESET

This block is reset on a [nSYS\\_RST](#).

## 23.5 DMA Interrupts

Each channel of the DMA controller generates an interrupt event to the EC which indicate a DMA transfer is complete.

## 23.6 Low Power Mode

This block is designed to conserve power when it is either sleeping or disabled. There are two ways to put the DMA Controller into a low power mode: Disable all DMA channels via the [Activate](#) Bits or assert the DMA\_SLEEP\_EN signal to the DMA Controller. The following table summarizes the DMA Controller behavior for each of these low power modes.

**Note:** The DMA Controller is not [Busy](#) if the Activate bit is 0 or all the Channel State Machines are in the IDLE state as illustrated in [Figure 23-3, "Channel State Machine"](#).

**TABLE 23-2: BLOCK CLOCK GATING IN LOW POWER MODES**

Activate	DMA_SLEEP_EN	Busy	DMA_CLK_REQ	State	Description
All 0	X	X	0	INACTIVE	All channels are disabled
Any 1	0	X	1	NORMAL OPERATION	The block is neither disabled by firmware nor commanded to SLEEP
Any 1	1	1	1	PREPARING to SLEEP	The core clock is required until the current transaction is completed and the Block is IDLE.
Any 1	1	0	0	SLEEPING	The block is idle and the core clock can be stopped.

**APPLICATION NOTE:** The DMA controller will keep its Clock\_Required status output asserted as long as the Run bit is on in any of the controller's channels. This means that the part will not enter a heavy sleep mode, and the main oscillator will stay on, as long as any controller channel is in the Run state. Before setting the SLEEP\_FLAG in order to enter the heavy sleep states, firmware must make sure that the Run bits are clear in all channels. If a DMA channel is required for any activity that is initiated by a Wake interrupt, the Wake interrupt Interrupt Service Routine should assert the Run bit in that channel.

For example, if the system uses SMB Controller 1 as a Slave SMBus port, and DMA Controller channel 7 as the SMB Slave DMA channel, then in order to enter a heavy sleep state, firmware must clear the Run bit in Channel 7 and enable Wake interrupts on the Clock and Data pins for SMB Controller 1 before asserting SLEEP\_FLAG. The Interrupt Service Routines for Wake interrupts for the Clock and Data pins for SMB Controller 1 must assert the Run bit for DMA Channel 7, so that the SMB network layer controller can pull data out of the pin interface without additional processor intervention.

## 23.7 Operation

The MEC1618/MEC1618i features a eight channel DMA controller. The DMA controller can autonomously move data from I/O devices to and from EC local memory without EC intervention.

The DMA has the following characteristics:

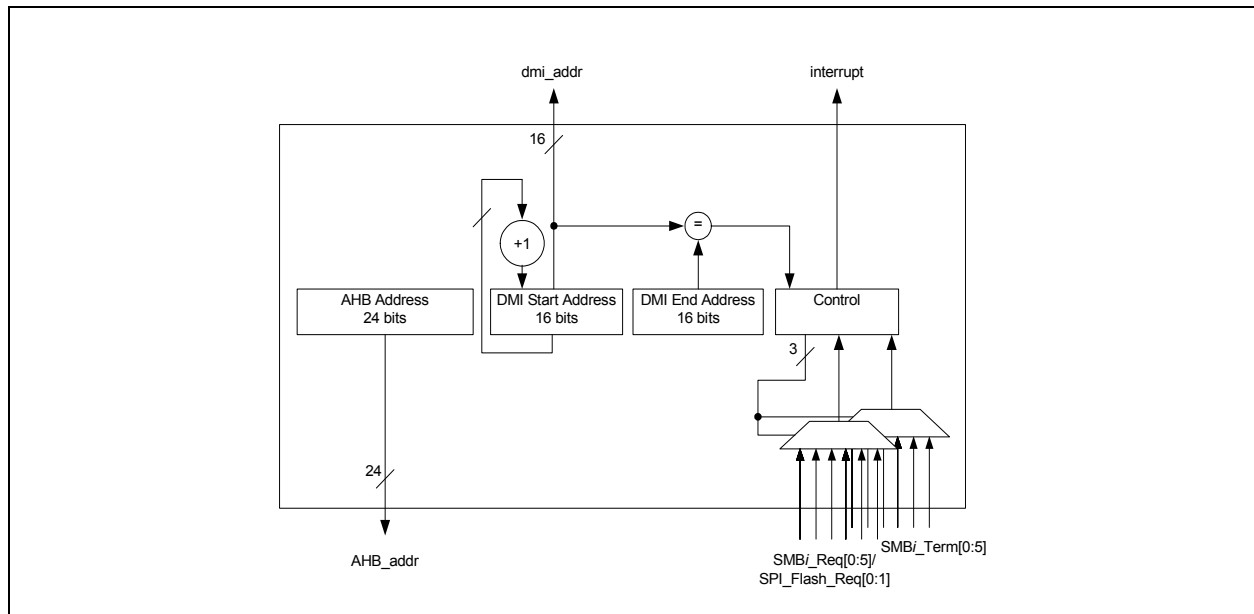
- Data is only moved 1 byte (8-bits) at a time
- Data only moves between devices on the AHB bus, or devices connected to an AHB bus bridge, and the EC SRAM. Since the SRAM is dual-ported to be both ICCM and DCCM, the DMA can be interpreted as moving data into and out of the DCCM as well as the ICCM.
- The number of DMA channels can be less than the number of I/O devices that can use the DMA for data transfers, so the DMA channels are shareable and can be assigned to any device.

The DMA controller is not designed to communicate with I/O devices with more than an 8-bit interface. The controller will access SRAM buffers only with incrementing addresses (that is, it cannot start at the top of a buffer, nor does it handle circular buffers automatically). The controller does not handle chaining (that is, automatically starting a new DMA transfer when one finishes).

### 23.7.1 DMA CHANNELS

Each DMA channel is capable of bi-directional data movement between an logical device and the EC closely-coupled memory. A single DMA channel is illustrated in Figure 23-2, "DMA Channel":

**FIGURE 23-2: DMA CHANNEL**



There are 8 possible logical devices that may connect to a DMA Controller channel. There are three SMBus controllers, each of which has a separate read request and write request. In addition, the SPI Flash controller has a read request and a write request. The SMBus controllers also provide a termination signal for each direction of each device. The SPI Flash device does not provide termination signals, so the two termination inputs corresponding to the SP Flash Read request and the SPI Flash Write request are always held to 0.

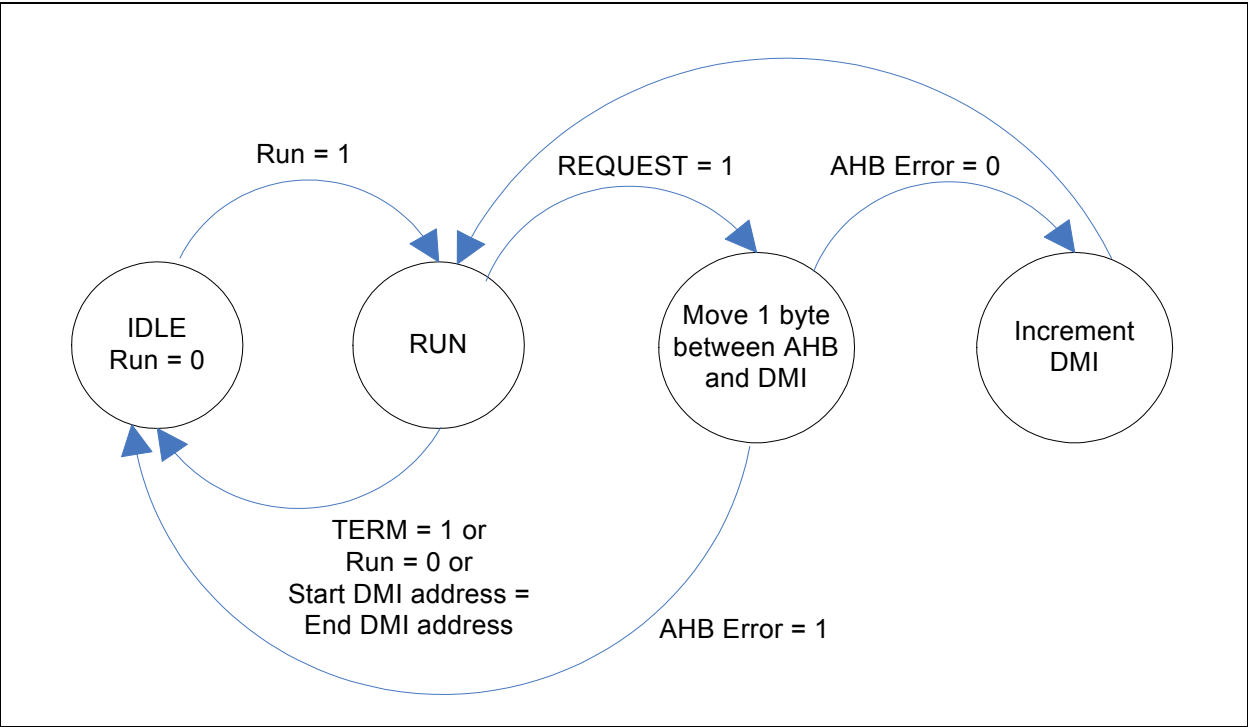
Based on the [DIR](#) in the channel's [DMA Control Register](#), bytes are copied from a device's Receive Buffer to the ICCM/DCCM, or from the ICCM/DCCM to a device's Transmit Buffer. The [AHB Address Register](#) is programmed to be the address of the buffer required for the transfer. Software is responsible for insuring that this address is associated with the correct SRAM buffer defined by the [DMI Start Address Register](#) and by the correct SMBus as selected by the [DEVICE](#) field in the Control register.

The DMI Start Address is an offset from the base of the SRAM. It is an offset from both the base of the DCCM and the base of the ICCM, since the two memories are different aliases of a single dual-ported SRAM. This register is loaded into a 16-bit counter, which increments by 1 under state machine control in order to generate the current DMI address.

The End Address register contains the address one greater than the last byte to transfer. The DMA transfer terminates when the current DMI address equals the End Address. If the DMA channel is configured with the Start Address and End Address registers set to the same address, no bytes will be transferred. If the End Address is configured with an address that is less than the Start Address, the DMI Start Address register will wrap around from FFFFh to 0000h. The DMA Controller always sends 16 address bits to the SRAM. If fewer than 16 bits are required to address the ICCM/DCCM, then the SRAM will ignore the upper bits of the Start Address and End Address registers. For example, if the ICCM/DCCM is 16KB, bits 14 to 15 in the Start and End Address registers will be ignored by the SRAM.

The state machine that runs each DMA channel is illustrated in [Figure 23-3, "Channel State Machine"](#). When software sets the [bit](#) to 1, the state machine enters the RUN state and waits for an assertion of the selected REQUEST input. As long as REQUEST is asserted when the state machine is in the RUN state the DMA controller starts a 1-byte wide AHB bus transaction, in the direction defined by [DIR](#). After the AHB transaction completes, the current DMI address counter is incremented by 1 and compared to the End Address register. If the current address is not equal to the End Address, the state machine returns to the RUN state. If the current address is equal to the End Address, or if the TERM input is asserted, the DMA transaction is terminated and the state machine returns to the IDLE state. If there is an AHB bus error on the transfer between the AHB and the DMI, the DMI address counter increment is inhibited. The state machine returns to the IDLE state and the Status is set to AHB Bus Error.

FIGURE 23-3: CHANNEL STATE MACHINE



23.7.2 I/O DEVICES

The DMA Controller is configured to work with any of the three SMBus controllers and the SPI Flash controller in the MEC1618/MEC1618i. Table 23-3, "DMA Device Selection" shows the mapping between the AHB Address and **DEVICE** and **DIR** fields for each I/O device.

TABLE 23-3: DMA DEVICE SELECTION

DEVICE	DIR	Device Name	Data AHB Address	Request Signals
0	0	SMBUS Controller 0	F0_184Ch (SMBus Slave Receive Buffer)	SMB0_SDMA_Req
0	1		F0_1848h (SMBus Slave Transmit Buffer)	
1	0		F0_1854h (SMBus Master Receive Buffer)	SMB0_MDMA_Req
1	1		F0_1850h (SMBus Master Transmit Buffer)	



**TABLE 23-3: DMA DEVICE SELECTION (CONTINUED)**

DEVICE	DIR	Device Name	Data AHB Address	Request Signals
2	0	SMBUS Controller 1	F0_18CCh (SMBus Slave Receive Buffer)	SMB1_SDMA_Req
2	1		F0_18C8h (SMBus Slave Transmit Buffer)	
3	0		F0_18D4h (SMBus Master Receive Buffer)	SMB1_MDMA_Req
3	1		F0_18D0h (SMBus Master Transmit Buffer)	
4	0	SMBUS Controller 2	F0_194Ch (SMBus Slave Receive Buffer)	SMB2_SDMA_Req
4	1		F0_1948h (SMBus Slave Transmit Buffer)	
5	0		F0_1954h (SMBus Master Receive Buffer)	SMB2_MDMA_Req
5	1		F0_1950h (SMBus Master Transmit Buffer)	
6	0	SPI Flash Controller	FF_3C10h (SPI RX_Data Register)	SPIFLASH_Read_Req
6	1		FF_3C0Ch (SPI TX_Data Register)	SPIFLASH_Write_Req
7	0/1	Reserved	–	–

**Note 23-1** See the [DEVICE](#) field in the [DMA Control Register](#) on page 445.

**Note 23-2** See the [DIR](#) bit in the [DMA Control Register](#) on page 445.

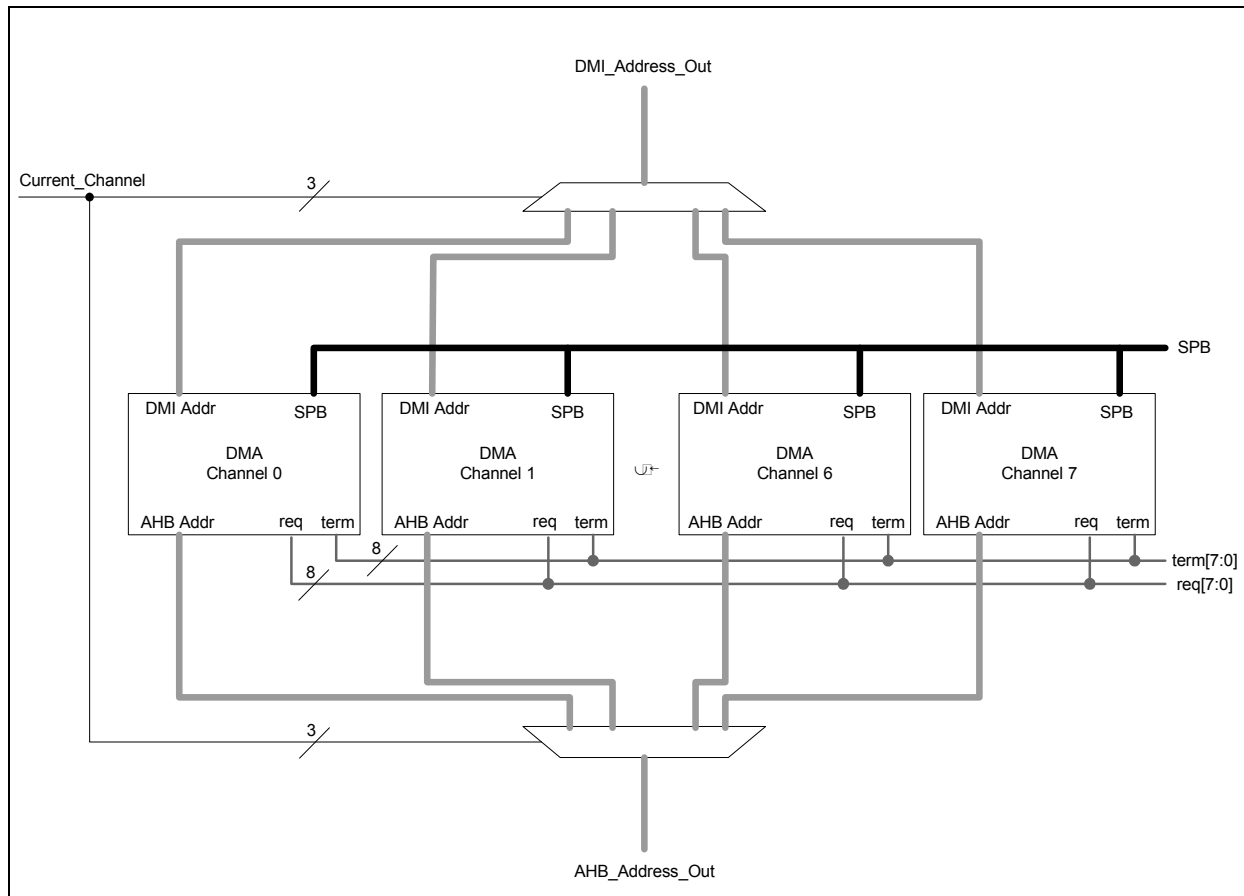
The Request signals from the devices are the Data register status signals. In the read (0, from the device to SRAM) direction, a DMA request is asserted as long as the Receive Data register is not empty. In the transmit (1, from the SRAM to the device) direction, a DMA request is asserted as long as the Transmit Data register is not full. For the SMBus controllers, the Terminate signal is asserted if the SMBus controller detects an error condition during an SMBus transaction, or if software shuts down the SMBus controller. There is no Terminate signal for the SPI Flash controller.

# MEC1618/MEC1618i

## 23.7.3 DMA CHANNEL ARBITRATION AND MULTIPLEXING

The eight DMA channels share the DMI interface and the AHB bus. Figure 23-4, "DMA Channel Multiplexing" illustrates the multiplexing of the channels onto the busses.

**FIGURE 23-4: DMA CHANNEL MULTIPLEXING**



A DMA Channel is ready to run as long as it is in the Run state and its selected Request input is asserted. The DMA controller services DMA Channels on a first-come first-served basis. If two channels become ready to run simultaneously, they are served in numerical order (channel 0 before channel 1, etc.). If multiple channels are continuously ready (that is, their respective Request inputs are always asserted), then they will be served in round-robin order.

## 23.8 DMA Registers

The base address for the DMA Controller block in the AHB address space is listed in [Table 23-4, "DMA Controller Base Address Table"](#).

**TABLE 23-4: DMA Controller BASE ADDRESS TABLE**

DMA Controller Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
DMA Channel0	9h	F0_2400h + 000h
DMA Channel1		F0_2400h + 020h
DMA Channel2		F0_2400h + 040h
DMA Channel3		F0_2400h + 060h
DMA Channel4		F0_2400h + 080h
DMA Channel5		F0_2400h + 0A0h
DMA Channel6		F0_2400h + 0C0h
DMA Channel7		F0_2400h + 0E0h

The following table summarizes the registers allocated for the DMA Controller. The offset field in the following table is the offset from the AHB Base Address defined in [Table 23-4 on page 443](#).

**TABLE 23-5: DMA Controller REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
DMA Control Register	0h	1-0	R/W	
DMI End Address Register	4h	1-0	R/W	
DMI Start Address Register	8h	1-0	R/W	
AHB Address Register	Ch	3-0	R/W	
DMA Activate Register	10h	0	R/W	

# MEC1618/MEC1618i

## 23.8.1 DMA CONTROL REGISTER

The [DMA Control Register](#) is used to control the behavior of the DMA controller.

**TABLE 23-6: DMA CONTROL REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	00h				16-bit			EC SIZE	
POWER	VTR				0000h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R/W	R/W	R/W	R/W	
BIT NAME	Reserved				Device			DIR	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R/W	
BIT NAME	Reserved		Busy	Status		Done	Request	Run	

### RUN

When this bit is 1, the state machine is active and the channel continually tries to move a byte between the I/O device at AHB Address and SRAM at DMA Address. When this bit is 1, software cannot modify any of the other registers in the channel, or any other bits in this register besides , in order to insure that no AHB transaction is modified while it is in progress.

Setting this bit to 0 will halt the DMA function. If there is an AHB transfer in progress when this bit is set to 0 the transfer will complete before the DMA state machine returns to the IDLE state. Firmware should query the [Busy](#) bit after setting [Run](#) to 0 in order to determine when the DMA transaction has terminated.

The DMA\_Int signal is only asserted when is 1.

### REQUEST

Read-only. This bit is always 0 when [Run](#) is 0, and is set when the DMA request input is 1. The DMA request input is selected by [DEVICE](#) and [DIR](#).

### DONE

Read-only. This bit is always 0 when is [Run](#) is 0, and is 1 when the DMA Controller state machine returns to the IDLE state. The DMA Controller state machine will transition back to IDLE when [DMI Start Address Register](#) equals the [DMI End Address Register](#), when the DMA Termination input is 1 or if the AHB transaction is terminated with a bus error. The DMA Termination input is selected by [DEVICE](#) and [DIR](#). This bit is routed to the interrupt controller.

### STATUS

Read-only. This field is updated whenever [Done](#) goes from 0 to 1 or when [Run](#) goes from 1 to 0, and indicates why a DMA transfer completed. Status values are:

00: [Run](#) is set to 0. This field is always 0 when [Run](#) is 0.

01: Start Address matched End Address

10: DMA\_Term input asserted

11: An AHB bus error occurred on the transfer

Status values are shown in order of priority. If more than one condition caused a return to the IDLE state, the condition with the lowest Status value is reported.

## BUSY

Read-only. This bit is 1 when the DMA State Machine is not in the IDLE state and 0 when the DMA State Machine is in the IDLE state.

## DIR

DMA transfer direction. 0 for reads from the AHB device to DCCM memory, 1 for writes from DCCM memory to the AHB device. When combined with [DEVICE](#), determines which DMA\_request input is used to start a DMA transfer

## DEVICE

The [DEVICE](#) field selects which I/O device is assigned to this DMA channel. See [Table 23-3, "DMA Device Selection," on page 440](#) for the [DEVICE](#) field to Device Name mapping.

### 23.8.2 DMI END ADDRESS REGISTER

This address defines the DMA stops transferring. When the incrementer that was loaded from the [DMI Start Address Register](#) is equal to this register, the DMA completes.

**TABLE 23-7: DMI END ADDRESS REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	04h				16-bit			EC SIZE	
POWER	VTR				00h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	DMI_End_Address[15:8]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	DMI_End_Address[7:0]								

## DMI\_END\_ADDRESS

This field contains the address one past the last byte to be transferred for the DMA channel. The DMA transfer stops when the current DMI address is equal to this register. If the End Address register is equal to the Start Address register when is set to 1, no data are transferred and the DMA terminates immediately.

# MEC1618/MEC1618i

## 23.8.3 DMI START ADDRESS REGISTER

**Note:** This register is 16-bit only. It does not support 8-bit accesses.

**TABLE 23-8: DMI START ADDRESS REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC ADDRESS	08h				16bit			EC SIZE	
POWER	VTR				0000h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	DMI_Start_Address[15:8]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	DMI_Start_Address[7:0]								

### DMI\_START\_ADDRESS[15:0]

This field defines an offset from the base of the SRAM, which is an offset from both the base of the DCCM and the base of the ICCM, since the two memories are different aliases of a single dual-ported SRAM. This register is loaded into a 16-bit counter, which increments by 1 under state machine control, when is set to 1. This register defines the initial byte address for bytes to be transferred on the associated DMA channel.

When first written by software, this register contains the start address in the DCCM for the DMA transfer. While a DMA transfer is in progress, this register contains the address of the next byte to be transferred. When the DMA transfer completes, this register is one greater than the address of the last byte transferred. Software can determine how many bytes were transferred overall by subtracting the value it used to configure this register initially from the value of this register when the transfer completes.

## 23.8.4 AHB ADDRESS REGISTER

The [AHB Address Register](#) is the address of the I/O device that is the source or sink of the DMA transfer.

**TABLE 23-9: AHB ADDRESS REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC ADDRESS	0Ch				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	AHB_Address[23:16]								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	AHB_Address[15:8]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	AHB_Address[7:0]								

### AHB\_ADDRESS[23:0]

This is the address of the I/O port in the AHB address space. Software is responsible for insuring that this address is the correct address for the I/O device assigned to the channel.

# MEC1618/MEC1618i

## 23.8.5 DMA ACTIVATE REGISTER

The [DMA Activate Register](#) is used to gate clocks to a DMA channel, in order to conserve power. Software must set the [Activate](#) bit to '1b' in order for a channel to operate.

**TABLE 23-10: DMA ACTIVATE REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	10h				8-bit			EC SIZE	
POWER	VTR				00h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R/W	
BIT NAME	Reserved							Activate	

### ACTIVATE

When this bit is 0, the [MCLK](#) is gated to this channel, so the channel will not operate. When this bit is 1, the channel is provided with the system clock and the channel can operate.



## 24.0 SMB DEVICE INTERFACE

### 24.1 General Description

The MEC1618/MEC1618i [SMB Device Interface](#) includes three instances of an SMBus controller core: SMBus[2:0]. This chapter describes aspects of the [SMB Device Interface](#) that are unique to the MEC1618/MEC1618i instantiations of this core; including, [Power Domain](#), [Resets](#), [Clocks](#), [Interrupts](#), [Registers](#) and the [Physical Interface](#). For a *General Description*, *Features*, *Block Diagram*, *Functional Description*, *Registers Interface* and other core-specific details, see Ref [1] (note: in this chapter, *italicized text* typically refers to SMBus controller core interface elements as described in Ref [1]).

#### 24.1.1 REFERENCES

1. SMBus Controller Core Interface, Revision 2.0 (10 MHz), v3.20, Core-Level Architecture Specification, SMSC, 4/22/11

#### 24.1.2 SMB PIN SIGNAL INTERFACE DESCRIPTION

The pin signals are defined in [Table 3.6.13, “SMBus Interface,” on page 62](#).

### 24.2 Power, Clocks and Reset

#### 24.2.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.5, “Power Configuration,” on page 101](#) for details on power domains. For more detail about the SMBus controller core Power Domain, see [Section 3.1, “Power Configuration”](#) in Ref [1].

#### 24.2.2 CLOCKS

[SMB Device Interface Clocking](#) is described below in [Table 24-1](#). Use this table when programming the SMBus controller core bus clock and timing values as specified in Ref [1].

**TABLE 24-1: [SMB Device Interface CLOCKING](#)**

Clock Source ( <a href="#">Note 24-1</a> )	sMBus Controller Core Clock ( <a href="#">Note 24-2</a> )	Frequency	Description
<a href="#">MCLK</a>	CORE_CLK	20.27 MHz	–
<a href="#">MCLK_DIV2_EN</a>	BAUD_CLK_EN	10.14 MHz	Use this frequency when programming the <i>Bus Clock Register</i> , <i>Data Timing Register</i> and the <i>Time-Out Scaling Register</i> described in Ref [1].
<a href="#">EC_BUS_CLK_EN</a>	SPB_CLK_EN	Programmable	EC Bus Clock.

**Note 24-1** See [Section 5.4.9, “MCLK Sourced Clocking,” on page 99](#).

**Note 24-2** For more detail about SMBus controller core Clocking see [Chapter 2, “Hardware Interface”](#) and [Section 3.3, “Clocking”](#) in Ref [1].

#### 24.2.3 RESETS

Each of the SMBus controller core instances in the MEC1618/MEC1618i [SMB Device Interface](#) are reset by [nSYS\\_RST](#). See [Section 5.6, “Reset Interface,” on page 104](#) for details on resets in the MEC1618/MEC1618i. For more detail about SMBus controller core Resets, see [Section 3.2, “Reset Interface”](#) in Ref [1].

# MEC1618/MEC1618i

## 24.3 Interrupts

Each EC SMB Controller has both an activity interrupt event and a START Bit detection Wake-up event. The SMB activity interrupt events are routed to the [SMB0](#), [SMB1](#) and [SMB2](#) bits in the [GIRQ12 Source Register on page 320](#). The START Bit detection Wake-up events are routed to the [SMB00 WK](#), [SMB01 WK](#), [SMB02 WK](#), [SMB03 WK](#), [SMB04 WK](#), [SMB05 WK](#), [SMB06 WK](#), [SMB07 WK](#), [SMB08 WK](#), [SMB09 WK](#), [SMB10 WK](#) and [SB\\_TSI](#) bits in the [GIRQ12 Source Register on page 320](#). The edge detection of the interrupt and wake events are controlled by their associated pin control registers in the [Section 21.0, "GPIO Interface," on page 398](#)

**APPLICATION NOTE:** The pin control registers for GPIOs that are associated with the SDAT pins for ports supporting wake events should be programmed to Input, Falling Edge Triggered, non-inverted polarity detection.

## 24.4 DMA

Each EC SMB Controller can utilize two [DMA Controller](#) channels as defined in Ref [1]. DMA Channel configuration is defined in [Table 23-3, "DMA Device Selection," on page 440](#).

## 24.5 Registers

Each SMBus controller core instance in the MEC1618/MEC1618i [SMB Device Interface](#) has unique Register Interface Addressing, defined by a base address as indicated in [Table 24-2](#). For more detail about SMBus controller core registers, see [Chapter 5, "Registers Interface"](#) in Ref [1].

**TABLE 24-2: SMB Device Interface BASE ADDRESS TABLE**

SMB Device Interface Instance	LDN from (Table 3-3 on page 52)	AHB Base Address
<a href="#">SMBus.0</a>	6h	<a href="#">F0_1800h</a>
<a href="#">SMBus.1</a>		<a href="#">F0_1880h</a> = <a href="#">F0_1800h</a> + 80h
<a href="#">SMBus.2</a>		<a href="#">F0_1900h</a> = <a href="#">F0_1800h</a> + 100h

## 24.6 Physical Interface

### 24.6.1 OVERVIEW

The [Physical Interface](#) for the SMB controller core includes 12 ports as defined below in [Section 24.6.2, "SMBus Port Selection"](#).

Each of the 3 SMBus controllers can be connected to any of the 12 ports. The *PORT SEL [3:0]* bits in each controller will appear the same ([Table 24-3](#)). The default for each field is 15, Reserved (which means that the SMB controller is not connected to a port).

An SMB port should be connected to a single controller. An attempt to configure the *PORT SEL [3:0]* bits in one controller to a value already assigned to another controller may result in unexpected results.

The port signal-function names and pin numbers are defined in [Section 3.6.13, "SMBus Interface," on page 62](#)." The [SMB Device Interface](#) port selection is made using the *PORT SEL [3:0]* bits in the *Configuration Register* as described in Ref [1] and in the subsections that follow.

For [SMB Device Interface](#) port signal functions that are alternate functions of GPIO pins, the buffer type for these pins must be configured as open-drain outputs when the port is selected as defined in [Section 24.6.2, "SMBus Port Selection"](#). For more information regarding the SMBus controller core [Physical Interface](#), see [Section 2.2, "Physical Interface"](#) in Ref [1].

## 24.6.2 SMBUS PORT SELECTION

SMBus Port Selection is defined below in [Table 24-3](#).

**TABLE 24-3: SMBUS PORT SELECTION**

Port SEL [3:0]				Port ( <a href="#">Note 24-3</a> )
3	2	1	0	
0	0	0	0	SMB00
0	0	0	1	SMB01
0	0	1	0	SMB02
0	0	1	1	SMB03
0	1	0	0	SMB04
0	1	0	1	SMB05
0	1	1	0	SMB06
0	1	1	1	SMB07
1	0	0	0	SMB08
1	0	0	1	SMB09
1	0	1	0	SMB10
1	0	1	1	SB-TSI
1100b - 1111b				Reserved

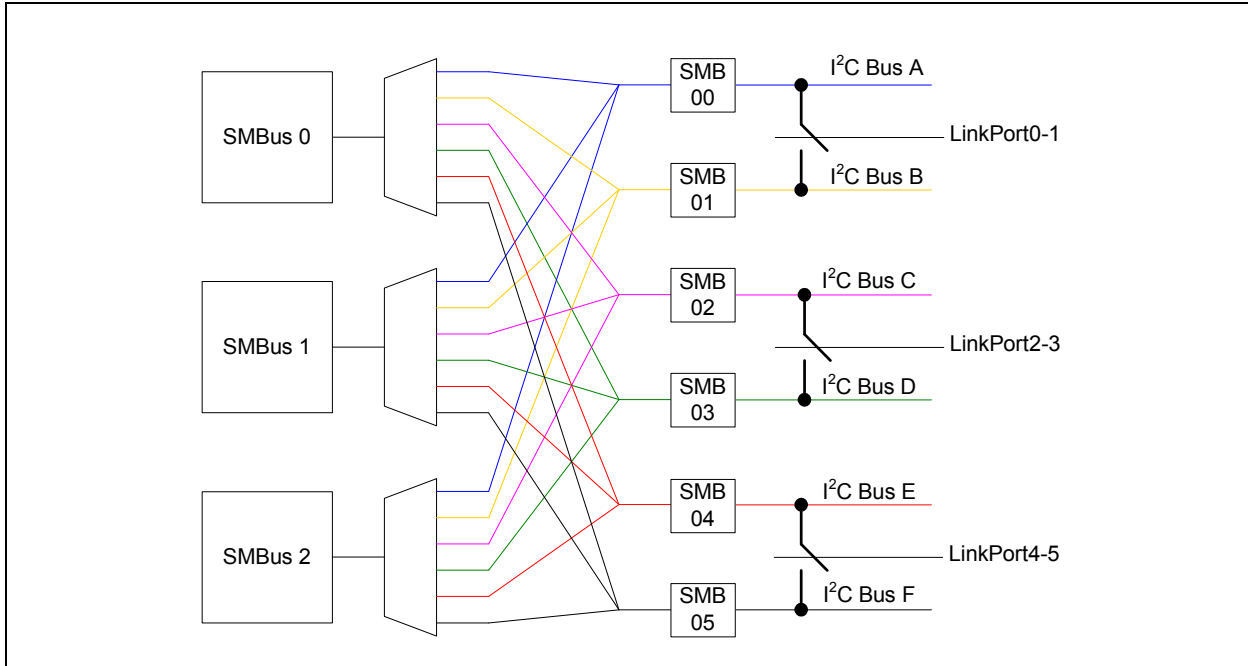
**Note 24-3** See [Section 3.6.13, "SMBus Interface,"](#) on page 62 for the [SMB Device Interface](#) pin configuration.

## 24.7 Port Isolation

- The SMBus [Port Isolation](#) feature in the MEC1618/MEC1618i enables electrical connectivity, or isolation between pairs of I<sup>2</sup>C/SMBus ports.
- SMBus [Port Isolation](#) is only present between three pairs of ports as shown in [FIGURE 24-1: SMBus Port Connectivity Example on page 452](#). SMBus ports SMB06 through SMB10, as well as the SB-TSI port, are not affected by the SMBus [Port Isolation](#) feature.
- The [Port Isolation](#) feature is controlled by the LinkPort bits in the [Port Isolation Register](#) (see [Section 24.7.1, "Port Isolation Register,"](#) on page 452).

# MEC1618/MEC1618i

**FIGURE 24-1: SMBUS PORT CONNECTIVITY EXAMPLE**



**Note:** For simplicity, only 6 of the 12 SMBus ports are shown in [Figure 24-1](#).

## 24.7.1 PORT ISOLATION REGISTER

The [Port Isolation Register](#) is located in LDN 3Fh ([F0\\_FC00h](#)) in the [System Registers](#) block.

Offset	40h			
Bits	Description	Type	Default	Reset Event
31:3	Reserved	RES	0	–
2	LinkPort4-5 (LP45) When this bit is asserted ('1'), the SMBus04 port is electrically connected the SMBus05 port. When LP45 is not asserted ('0') (default), the SMBus04 port is electrically isolated from the SMBus05 port.	RW	0	VTR POR
1	LinkPort2-3 (LP23) When this bit is asserted ('1'), the SMBus02 port is electrically connected the SMBus03 port. When LP23 is not asserted ('0') (default), the SMBus02 port is electrically isolated from the SMBus03 port.			
0	LinkPort0-1 (LP01) When this bit is asserted ('1'), the SMBus00 port is electrically connected the SMBus01 port. When LP01 is not asserted ('0') (default), the SMBus00 port is electrically isolated from the SMBus01 port.			

**TABLE 24-4: REGISTER BIT ACCESS TYPES**

Register Bit Type	Description
R	<b>Read:</b> A register or bit with this attribute can be read.
W	<b>Write:</b> A register or bit with this attribute can be written.
RO	<b>Read Only:</b> A register or bit with this attribute is read only, writes have no effect.
RS	<b>Read to Set:</b> A register or bit with this attribute is set on read.
RC	<b>Read to Clear:</b> A register or bit with this attribute is cleared after the read, writes have no effect.
WO	<b>Write Only:</b> A register or bit with this attribute is write only, reads return zero.
WC	<b>Write One to Clear:</b> Writing a one to a bit with this attribute clears ('0') the value, writing a zero has no effect.
WS	<b>Write One to Set:</b> Writing a one to a bit with this attribute sets the value to '1', writing a zero has no effect.
WZS	<b>Write Zero to Set:</b> Writing a zero to a bit with this attribute sets the value to '1', writing a one has no effect.
RES	<b>Reserved:</b> Reads of a register or bit with this attribute return zero, writes are ignored.

# MEC1618/MEC1618i

## 25.0 PECI INTERFACE

### 25.1 Overview

The MEC1618/MEC1618i includes a [PECI Interface](#) to allow the EC to retrieve temperature readings from PECT-compliant devices. The [PECI Interface](#) implements the PHY and Link Layer of a PECT host controller as defined in [References](#) [1] and includes hardware support for the PECT 3.0 command set.

The block has a 32 Byte FIFO for PECT 3.0 compliance.

This chapter focuses on MEC1618/MEC1618i specific [PECI Interface](#) configuration information such as [Register Addressing](#), [Power Domain](#), [Resets](#), [Physical Interface](#), [Interrupts](#) and [Clocking](#). For a functional description of the MEC1618/MEC1618i [PECI Interface](#) refer to [References](#) [1].

### 25.2 References

1. PECT Interface Core, Rev. 1.31, Core-Level Architecture Specification, SMC Confidential, 4/15/11.

### 25.3 Register Addressing

The [PECI Interface](#) module is attached to [EC SPB](#). It is assigned EC LDN 19h with base address [F0\\_6400h](#); register addresses are aligned on 4-byte boundaries. The [PECI Interface](#) registers are summarized in [Table 25-1](#). For register details see [References](#) [1].

**TABLE 25-1: PECT Interface REGISTERS SUMMARY**

Address Offset	Mnemonic	Register Description	NL Access (Note 25-1)
0x00	SSTWRBUF	<a href="#">Write Data Register</a>	RW
0x04	SSTRDBUF	<a href="#">Read Data Register</a>	RW
0x08	SSTCTL	<a href="#">Control Register</a>	RW
0x0C	SSTSTA1	<a href="#">Status Register 1</a>	RWC
0x10	SSTSTA2	<a href="#">Status Register 2</a>	RWC
0x14	SSTERR	<a href="#">Error Register</a>	RWC
0x18	SSTINTEN1	<a href="#">Interrupt Enable 1 Register</a>	RW
0x1C	SSTINTEN2	<a href="#">Interrupt Enable 2 Register</a>	RW
0x20	SSTOBT1	<a href="#">Optimal Bit Time Register (Low Byte)</a>	RW
0x24	SSTOBT2	<a href="#">Optimal Bit Time Register (High Byte)</a>	RW
0x28	SSTRTR1	<a href="#">Request Timer Register (Low Byte)</a>	RW
0x2C	SSTRTR2	<a href="#">Request Timer Register (High Byte)</a>	RW
0x30-0x3C	–	<a href="#">Reserved</a>	R
0x40	SSTBLKID	<a href="#">Block ID Register</a>	R
0x44	SSTREV	<a href="#">Revision Register</a>	R
0x48 - 0x7C	Reserved and Test Registers	MCHP Reserved. MCHP Reserved registers are reserved for use by Microchip, only. Reading and Writing MCHP Reserved registers may cause undesirable results.	RW

**Note 25-1** “R” means the register is read-only, writes have no affect; “RW” means the register can written and read; “RWC” means the register can written and read but that a ‘1’ must be written to a register bit to clear it.

## 25.4 Block Interface Parameters

### 25.4.1 SIGNAL LIST

**TABLE 25-2:** [PECI Interface Signal List](#)

Signal Name	Type	Description
VREF_VTT	INPUT	PECI Voltage Reference pin
PECI_DAT	INPUT/OUTPUT	PECI Data signal pin ( <a href="#">VREF_VTT</a> )
<a href="#">EC SPB</a>	I/O Bus	EC MEC1618/MEC1618i peripheral bus
<a href="#">MCLK</a>	INPUT	Master Clock
SPB_CLK_EN	INPUT	MEC1618/MEC1618i clock enable signal for Host interface clock
<a href="#">nSYS_RST</a>	INPUT	Synchronous block reset signal
PECI_INT	OUTPUT	Interrupt signal from Peci controller to EC
SLEEP_EN	INPUT	External sleep enable control
CLOCK_REQ	OUTPUT	Clock required status
VTR	POWER	Digital logic voltage supply
GND		Ground

### 25.4.2 POWER DOMAIN

The [PECI Interface](#) core logic is powered by [VTR](#); the [Physical Interface Power Domain](#) is [VREF\\_VTT](#).

### 25.4.3 RESETS

The [PECI Interface](#) is reset on a [nSYS\\_RST](#). The [PECI Interface](#) core also includes soft reset capabilities which reset control logic and part of registers. See [References](#) [1] for details.

### 25.4.4 CLOCKING

The [PECI Interface Clocking](#) requirement is defined below in [Table 25-3](#).

**TABLE 25-3:** [Clocking](#)

Domain	Clock	Type	Frequency
CORE	<a href="#">MCLK</a>	Fixed Clock	20.27 MHz
HOST	<a href="#">MCLK</a>	Fixed Clock	20.27 MHz
	<a href="#">SPB_CLK_EN</a>	Variable Clock Enable	20.27 MHz and slower.

### 25.4.5 INTERRUPTS

The interrupt from the [PECI Interface](#) module is routed to the [PECI\\_INT](#) bit of [GIRQ16 Source Register](#).

# MEC1618/MEC1618i

---

## 25.4.6 PHYSICAL INTERFACE

The pin configuration for the MEC1618/MEC1618i PECL [Physical Interface](#) is defined in [Section 3.6.14, "PECL Interface," on page 63](#).

## 25.4.7 SLEEP ENABLE/CLOCK REQUIRED POWER STATE CONTROLS

For a description of the [PECL Interface Sleep Enable/Clock Required Power State Controls](#) see the [PECL Interface Core, Rev. 1.31, Core-Level Architecture Specification, SMSC Confidential, 4/15/11](#).



## 26.0 ANALOG TO DIGITAL CONVERTER

### 26.1 General Description

This block is designed to convert external analog voltage readings into digital values. It consists of a single successive-approximation Analog-Digital Converter that can be shared among sixteen inputs. The characteristics of this interface are shown in [Table 26-1](#).

**APPLICATION NOTE:** Transitions on ADC GPIOs are not permitted when [Analog to Digital Converter](#) readings are being taken.

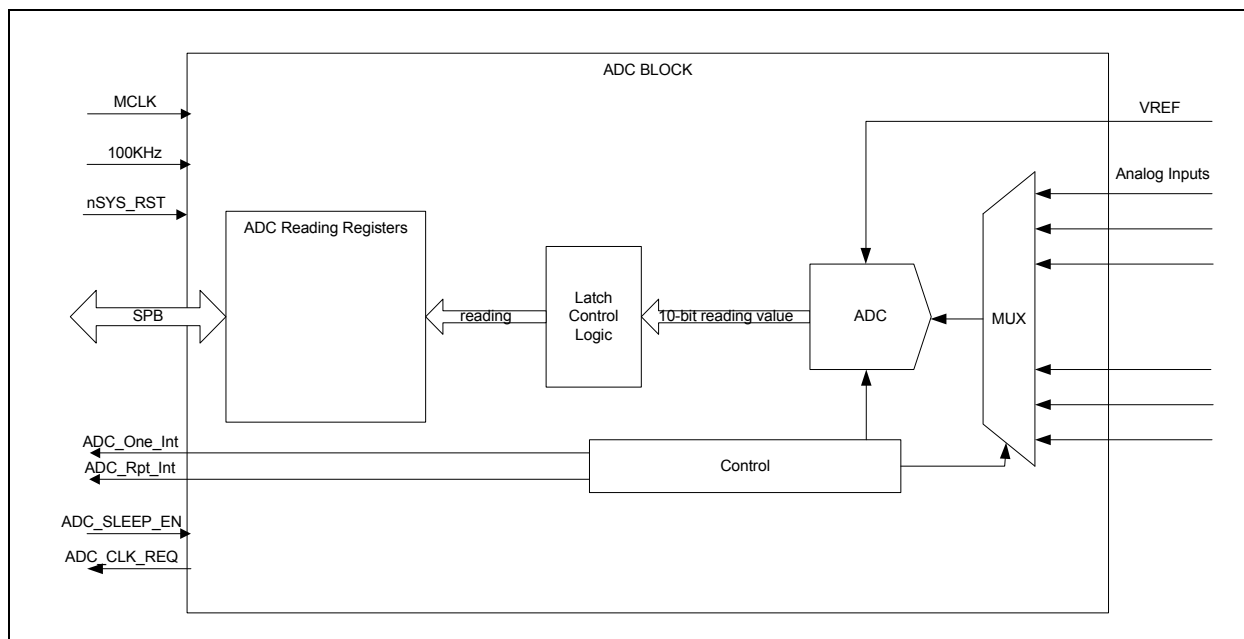
**TABLE 26-1: ADC CHARACTERISTICS**

Parameter	Min	Typ	Max	Unit
Resolution	–	–	10	Bits
Total Inputs (Programmable)	–	–	16	Channel
Conversion Time	–	–	10	μs/channel
Absolute Accuracy	–	2	4	LSB
Integral Non-Linearity	-0.5	–	+0.5	LSB
Differential Non-Linearity	-0.5	–	+0.5	LSB
Input Impedance	7	10	–	MOhms
Analog Input Range	0	–	VREF_ADC	Volts
VREF_ADC	2.97	–	AVTR_ADC	Volts
VREF_ADC Impedance	14 K	16 K	–	Ohms
AVTR_ADC	2.97	3.3	3.63	Volts

# MEC1618/MEC1618i

## 26.2 ADC Block Diagram

FIGURE 26-1: ADC BLOCK DIAGRAM



## 26.3 Block Diagram Signal List

TABLE 26-2: Analog to Digital Converter SIGNAL LIST

Signal Name	Direction	Description
VREF_ADC	INPUT	Analog Voltage Reference
AVTR_ADC	POWER	Analog Supply
VSS_ADC	POWER	Analog Ground
SPB	I/O Bus	EC MEC1618/MEC1618i peripheral bus
MCLK	INPUT	Master MEC1618/MEC1618i clock
MCLK_DIV203_EN	INPUT	100 kHz Clock enable derived from MCLK
nSYS_RST	INPUT	Block reset signal
ADC_One_Int	OUTPUT	Interrupt signal from ADC controller to EC for One-shot ADC conversion
ADC_Rpt_Int	OUTPUT	Interrupt signal from ADC controller to EC for Repeated ADC conversion
Analog Inputs	INPUT	16 analog voltage inputs from pins

**TABLE 26-2: Analog to Digital Converter SIGNAL LIST (CONTINUED)**

Signal Name	Direction	Description
ADC_SLEEP_EN	INPUT	External enable/disable signal used to put the block in the lowest power consumption state. 0=No Sleep Requested. The block should operate as configured. 1=Sleep Requested. The block enters sleep mode. See <a href="#">Low Power States on page 459</a> .
ADC_CLK_REQ	OUTPUT	This output indicates when this block requires this clock input. 0= <a href="#">MCLK</a> can be turned 'off' when appropriate 1= <a href="#">MCLK</a> is required to be 'on.'

## 26.4 Power, Clocks and Reset

### 26.4.1 POWER DOMAIN

This block is powered by the VTR power supply with a separate Analog supply (AVDD).

### 26.4.2 CLOCKS

This block has two clock inputs: [MCLK](#) and the 100 kHz clock enable ([MCLK\\_DIV203\\_EN](#)). The latter signal is used to derive the 10ms period used for delay generation in the block. The block internally generates a 1.2 MHz clock with a 50% duty cycle from the Master clock ([MCLK](#)).

### 26.4.3 RESET

This block is reset on a [nSYS\\_RST](#).

## 26.5 ADC Interrupts

The ADC generates an interrupt/wake-up events to the EC which indicate an ADC conversion cycle is complete. The [ADC\\_OneStat](#) bit and the [ADC\\_RptStat](#) bit in the [ADC Control Register](#) are set when conversion cycles complete. The two status bits are routed to the ADC bits in the [GIRQ16 Source Register](#).

## 26.6 Low Power States

The [Analog to Digital Converter](#) is designed to conserve power when it is either sleeping or disabled. The ADC is disabled via the [Activate](#) Bit and sleeps when the [ADC\\_SLEEP\\_EN](#) signal is asserted. The sleeping state only controls clocking in the ADC and does not power down the analog circuitry. For lowest power consumption, the ADC [Activate](#) bit must be set to '0.' The following table summarizes the ADC behavior for each of these [Low Power States](#).

**TABLE 26-3: BLOCK CLOCK GATING IN LOW POWER STATES**

Activate Bit	<a href="#">ADC_SLEEP_EN</a>	Block Idle Status	<a href="#">ADC_CLK_REQ</a>	State	Description
0	X	NOT IDLE	1	PREPARING to DISABLE	The core clock is required until the current transaction is completed and the Block is IDLE.
		IDLE	0	DISABLED	The block is idle and the core clock can be stopped.

# MEC1618/MEC1618i

TABLE 26-3: BLOCK CLOCK GATING IN LOW POWER STATES (CONTINUED)

Activate Bit	ADC_SLEEP_EN	Block Idle Status	ADC_CLK_REQ	State	Description
1	0	X	1	NORMAL OPERATION	The block is neither disabled by firmware nor commanded to SLEEP.
	1	NOT IDLE	1	PREPARING to SLEEP	The core clock is required until the current transaction is completed and the Block is IDLE.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.

## 26.7 Operation

The MEC1618/MEC1618i features a sixteen channel successive approximation Analog to Digital Converter. The ADC architecture features excellent linearity and converts analog signals to 10-bit words. Conversions take 10 microseconds per 10-bit word. The sixteen channels are implemented with a single high speed ADC fed by a sixteen input analog multiplexer. The multiplexer cycles through the sixteen voltage channels, starting with the lowest-numbered channel and proceeding to the highest-number channel, selecting only those channels that are programmed to be active.

The input range on the voltage channels spans from 0V to the external voltage reference. With a voltage reference of 3.3V, this provides resolutions of 3.2mV. The range can easily be extended with the aid of resistor dividers. The accuracy of any voltage reading depends on the accuracy and stability of the voltage reference input.

The ADC conversion cycle starts either when the [Start\\_Once](#) bit in the [ADC Control Register](#) is set to 1 or when the ADC Repeat Timer counts down to 0. When the [Start\\_Once](#) is set to 1 the conversion cycle converts channels enabled by configuration bits in the [ADC One Shot Register](#). When the Repeat Timer counts down to 0 the conversion cycle converts channels enabled by configuration bits in the [ADC Repeat Register](#). When both the [Start\\_Once](#) bit and the Repeat Timer request conversions the [Start\\_Once](#) conversion is completed first.

**Note:** If software repeatedly sets [Start\\_Once](#) to 1 at a rate faster than the Repeat Timer count down interval, the conversion cycle defined by the [ADC Repeat Register](#) will not be executed.

### 26.7.1 INPUT TERMINATION

A 0.1μF capacitor is recommended for the ADC inputs. If the ADC source impedance is greater than 10KΩ, then the 0.1 capacitor is required.

## 26.8 ADC Registers

The base address for the ADC block in the AHB address space is listed in [Table 26-4, "Analog to Digital Converter Base Address Table"](#).

TABLE 26-4: [Analog to Digital Converter](#) BASE ADDRESS TABLE

ADC Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
ADC	1Ah	F0_6800h

The following table summarizes the registers allocated for the ADC. The offset field in the following table is the offset from the Embedded Controller (EC) AHB Base Address.

- All register accesses require a single 16, or 32 bit read of the ADC Channel Reading Registers, because the data holding registers have been removed; i.e., two 8-bit reads cannot guarantee data coherency.

**TABLE 26-5:** Analog to Digital Converter REGISTER SUMMARY

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
ADC Control Register	0h	3-0	R/W	
ADC Delay Register	4h	3-0	R/W	
ADC Status Register	8h	3-0	R/W	
ADC One Shot Register	Ch	3-0	R/W	
ADC Repeat Register	10h	3-0	R/W	
ADC Channel 0 Reading Register	14h	3-0	R	Table 26-11 Note 26-1
ADC Channel 1 Reading Register	18h	3-0	R	
ADC Channel 2 Reading Register	1Ch	3-0	R	
ADC Channel 3 Reading Register	20h	3-0	R	
ADC Channel 4 Reading Register	24h	3-0	R	
ADC Channel 5 Reading Register	28h	3-0	R	
ADC Channel 6 Reading Register	2Ch	3-0	R	
ADC Channel 7 Reading Register	30h	3-0	R	
ADC Channel 8 Reading Register	34h	3-0	R	
ADC Channel 9 Reading Register	38h	3-0	R	
ADC Channel 10 Reading Register	3Ch	3-0	R	
ADC Channel 11 Reading Register	40h	3-0	R	
ADC Channel 12 Reading Register	44h	3-0	R	
ADC Channel 13 Reading Register	48h	3-0	R	
ADC Channel 14 Reading Register	4Ch	3-0	R	
ADC Channel 15 Reading Register	50h	3-0	R	

# MEC1618/MEC1618i

## 26.8.1 ADC CONTROL REGISTER

The [ADC Control Register](#) is used to control the behavior of the Analog to Digital Converter.

**TABLE 26-6: ADC CONTROL REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC OFFSET	00h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BYTE[3-1] BIT	D31	D30	D29	...		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R	R	R/W	R/W	R/W	R/W
BIT NAME	ADC_OneStat	ADC_RptStat.	Reserved		Power_Saver	Start_Repeat	Start_Once	Activate

### ACTIVATE

- 0: The ADC is disabled and placed in a low power state. Any conversion cycle in process will complete before the block is shut down, so that the reading registers will contain valid data but no new conversion cycles will begin.
- 1: [Start\\_Once](#) or [Start\\_Repeat](#) can begin data conversions by the ADC. A one cycle reset pulse is sent to the ADC core when this bit changes from 0 to 1.

### START\_ONCE

Writing this bit with a 1 will start a single conversion cycle of all ADC channels enabled by bits [Single\\_En\[15:0\]](#) in the [ADC One Shot Register](#). The conversion cycle will begin without a delay. Every channel that is enabled will be converted in 10  $\mu$ s. After all channels enabled by [Single\\_En\[15:0\]](#) are complete, [ADC\\_OneStat](#) will be set to 1. When the conversion cycle begins the bit is cleared.

If [Start\\_Once](#) is written with a 1 while a conversion cycle is in progress because [Start\\_Repeat](#) is set, the conversion cycle will complete, followed immediately by a conversion cycle using [Single\\_En\[15:0\]](#) to control the channel conversions.

Writing this bit with a 0 has no effect.

### START\_REPEAT

Writing this bit with a 1 will start a conversion cycle of all ADC channels enabled by bits [Rpt\\_En\[15:0\]](#) in the [ADC Repeat Register](#). The conversion cycle will begin after a delay determined by [Start\\_Delay\[15:0\]](#) in the [ADC Delay Register](#). The [Start\\_Delay\[15:0\]](#) value is loaded into an internal Repeat Timer register and the conversion cycle begins when the Repeat Timer counts down to 0. Every channel that is enabled will be converted in 10  $\mu$ s. After all channels enabled by [Rpt\\_En\[15:0\]](#) are complete, [ADC\\_RptStat](#) will be set to 1. As long as [Start\\_Repeat](#) is 1 when the Repeat Timer counts down to 0, the Repeat Timer will be reloaded with [Repeat\\_Delay\[15:0\]](#), so that the ADC will repeatedly begin conversion cycles with a period defined by [Repeat\\_Delay\[15:0\]](#). If the delay period expires and a conversion cycle is already in progress because [Start\\_Once](#) was written with a 1, the cycle in progress will complete, followed immediately by a conversion cycle using [Rpt\\_En\[15:0\]](#) to control the channel conversions.

Setting this bit to 0 will not terminate any conversion cycle in process, but will clear the Repeat Timer and inhibit any further periodic conversions.

## **POWER\_SAVER**

The [Analog to Digital Converter](#) includes a optional power saving feature to keep the ADC analog section off for as long as possible. To do this, the controller powers down the ADC between conversions sequences.

The [Power\\_Saver](#) feature is not optimized for power cycling. For example, if a one-shot conversion sequence follows immediately after a repeat conversion sequence (or vice versa), the ADC controller will still disable the ADC after the first conversion sequence, and re-enable it before starting the second conversion sequence.

## **ADC\_ONESTAT**

This bit is cleared whenever an ADC conversion cycle is begun when [Start\\_Once](#) is written with a 1 and is set to 1 when the conversion cycle started by writing [Start\\_Once](#) completes.

This bit is also cleared when it is written with a 1. Writing a 0 to this bit has no effect.

This bit can be used to generate an EC interrupt.

## **ADC\_RPTSTAT**

This bit is cleared whenever an ADC conversion cycle is begun when [Start\\_Repeat](#) is 1 and is set to 1 when a repeating conversion cycle completes.

This bit is also cleared when it is written with a 1. Writing a 0 to this bit has no effect.

This bit can be used to generate an EC interrupt.

## 26.8.2 ADC DELAY REGISTER

The ADC Delay register determines the delay from setting [Start\\_Repeat](#) in the [ADC Control Register](#) and the start of a conversion cycle. This register also controls the interval between conversion cycles in repeat mode.

# MEC1618/MEC1618i

**TABLE 26-7: ADC DELAY REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC OFFSET	04h				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Repeat_Delay[15:8]								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Repeat_Delay[7:0]								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Start_Delay[15:9]								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Start_Delay[7:0]								

## START\_DELAY[15:0]

This field determines the starting delay before a conversion cycle is begun when [Start\\_Repeat](#) is written with a 1. The delay is in units of 40μs. A value of 0 means no delay before the start of a conversion cycle, and a value of 0xFF means a delay of 2.6 seconds.

This field has no effect when [Start\\_Once](#) is written with a 1.

## REPEAT\_DELAY[15:0]

This field determines the interval after one conversion cycle completes and the next cycle begins when [Start\\_Repeat](#) is 1. The delay is in units of 40μs. A value of 0 means no delay between conversion cycles, and a value of 0xFF means a delay of 2.6 seconds.

This field has no effect when [Start\\_Once](#) is written with a 1.



**Note:** If the Repeat Timer counts down to 0 more than once while a conversion cycle is in progress, only one periodic conversion cycle will be requested.

## 26.8.3 ADC STATUS REGISTER

The [ADC Status Register](#) indicates whether the ADC has completed a conversion cycle.

**TABLE 26-8: ADC STATUS REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	08h			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	ADC_Ch_Status15	ADC_Ch_Status14	ADC_Ch_Status13	ADC_Ch_Status12	ADC_Ch_Status11	ADC_Ch_Status10	ADC_Ch_Status9	ADC_Ch_Status8
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	ADC_Ch_Status7	ADC_Ch_Status6	ADC_Ch_Status5	ADC_Ch_Status4	ADC_Ch_Status3	ADC_Ch_Status2	ADC_Ch_Status1	ADC_Ch_Status0

### ADC\_CH\_STATUS[15:0]

Each bit in this field reports the conversion status of the corresponding ADC channel. All bits are cleared either by being written with a '1,' or following a system reset ([nSYS\\_RST](#)). Each bit is set when the conversion on the corresponding channel is complete. When [ADC\\_CH\\_Status\[15:0\]](#) matches [Single\\_En\[15:0\]](#) after a conversion cycle initiated by a write to the [Start\\_Once](#) bit in the [ADC Control Register](#), bit [ADC\\_OneStat](#) in the [ADC Control Register](#) is set and an interrupt to the EC will occur (if the interrupt is enabled). When [ADC\\_CH\\_Status\[15:0\]](#) matches [Rpt\\_En\[15:0\]](#) after a conversion cycle initiated by a value of 1 in bit [Start\\_Repeat](#) in the [ADC Control Register](#), bit [ADC\\_RptStat](#) in the [ADC Control Register](#) is set and an interrupt to the EC will occur (if the interrupt is enabled).

Conversions always start with the lowest-numbered enabled channel and proceed to the highest-numbered enabled channel.

# MEC1618/MEC1618i

## 26.8.4 ADC ONE SHOT REGISTER

The [ADC One Shot Register](#) is used to control which ADC channels are captured during a one-shot conversion cycle initiated by the [Start\\_Once](#) bit in the [ADC Control Register](#).

**TABLE 26-9: ADC ONE SHOT REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC ADDRESS	0Ch				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Single_En15	Single_En14	Single_En13	Single_En12	Single_En11	Single_En10	Single_En9	Single_En8	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Single_En7	Single_En6	Single_En5	Single_En4	Single_En3	Single_En2	Single_En1	Single_En0	

### SINGLE\_EN[15:0]

Each bit in this field enables the corresponding ADC channel when a single cycle of conversions is started when the [Start\\_Once](#) bit in the [ADC Control Register](#) is written with a 1. If a [Single\\_En\[i\]](#) bit is 1, the channel is enabled. If a [Single\\_En\[i\]](#) bit is 0, the channel is disabled. At least one channel must be enabled before a conversion cycle can be initiated. Conversions start with the lowest-numbered channel that is enabled and proceed to the highest-numbered enabled channel. If this register is changed while a conversion cycle is in progress the conversion cycle will use the new values for channels that have not yet been examined, but will not rescan channels that have already been checked.

## 26.8.5 ADC REPEAT REGISTER

The [ADC Repeat Register](#) is used to control which ADC channels are captured during a one-shot conversion cycle initiated by the [Start\\_Repeat](#) bit in the [ADC Control Register](#).

**TABLE 26-10: ADC REPEAT REGISTER**

HOST OFFSET	N/A				N/A			HOST SIZE	
EC ADDRESS	10h				32-bit			EC SIZE	
POWER	VTR				0000_0000h			VTR POR DEFAULT	
BUS	EC SPB								
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Rpt_En15	Rpt_En14	Rpt_En13	Rpt_En12	Rpt_En11	Rpt_En10	Rpt_En9	Rpt_En8	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Rpt_En7	Rpt_En6	Rpt_En5	Rpt_En4	Rpt_En3	Rpt_En2	Rpt_En1	Rpt_En0	

### RPT\_EN[15:0]

Each bit in this field enables the corresponding ADC channel for each pass of the Repeated ADC Conversion that is controlled by bit [Start\\_Repeat](#) in the [ADC Control Register](#). If a Rpt\_En[*i*] bit is 1, the channel is enabled. If a Rpt\_En[*i*] bit is 0, the channel is disabled. At least one channel must be enabled before a conversion cycle can be initiated. Conversions start with the lowest-numbered channel that is enabled and proceed to the highest-numbered enabled channel. If this register is changed while a conversion cycle is in progress the conversion cycle will use the new values for channels that have not yet been examined, but will not rescan channels that have already been checked.

# MEC1618/MEC1618i

## 26.8.6 ADC CHANNEL READING REGISTERS

All 16 ADC channels return their results into a 32-bit reading register. In each case the low 10-bits of the reading register return the result of the Analog to Digital conversion and the upper 22 bits return 0. [Table 26-11](#) shows the format of all the reading registers. [Table 26-5, "Analog to Digital Converter Register Summary," on page 461](#) shows the addresses of all the reading registers.

**Note 26-1** The [ADC Channel Reading Registers](#) access require single 16-, or 32-bit reads; that is, two 8-bit reads cannot guarantee data coherency.

**TABLE 26-11: ADC CHANNEL X READING REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC ADDRESS	xxh			32-bit			EC SIZE	
POWER	VTR			0000_0000h			VTR POR DEFAULT	
BUS	EC SPB							
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved						ADCx_[9:8]	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	ADCx_[7:0]							

### ADCX\_[9:0]

This read-only field reports the 10-bit output reading of ADCx.

## 27.0 TACH MONITOR

### 27.1 General Description

This block is designed to monitor tach output signals or locked rotor signals from various types of fans to determine their speed. One mode returns the value in number of **CLOCK\_LOW** pulses. Another mode returns the value in pulses per programmed amount of time. This second mode can use the raw tach input. Each Tach is associated with a pair of limit registers that define maximum and minimum acceptable Tach counter values. If the readings on a Tach is outside these limits an interrupt to the EC can be generated.

In typical systems the fans are powered by the main power supply. Firmware may disable this block when it detects the main power rail has been turned off.

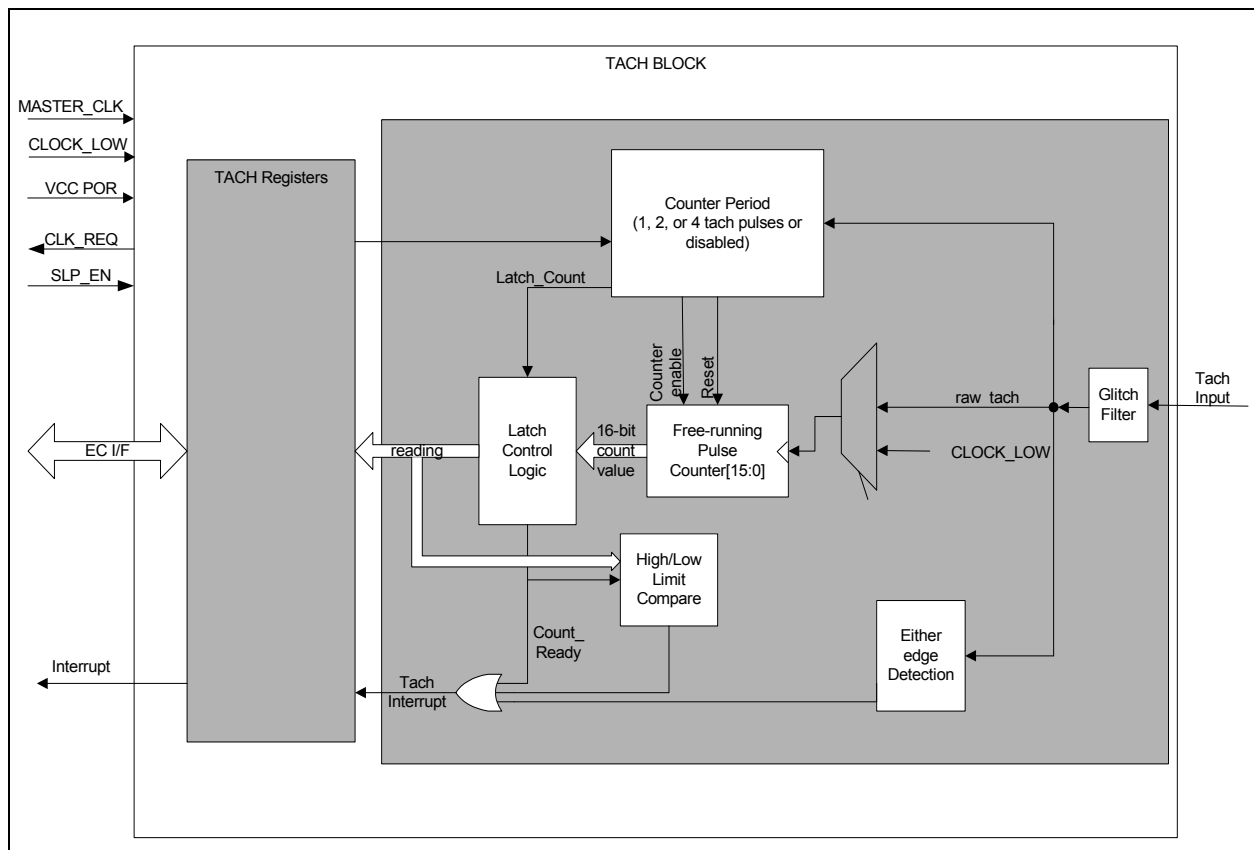
**APPLICATION NOTE:** This block can be utilized with Fans running at the following speed range:100 to 30K RPM.

The TACH Monitor performs the following functions:

- Count the number of pulses detected on the raw tach input.
- Count the number of clocks for a programmed number of pulses.
- Generate an interrupt when the count value is latched into the reading register.
- Generate a programmable either-edge triggered interrupt for detecting when the tach input changes state. This may be used for Locked Rotor detection.
- Generate an interrupt when the count value latched into the reading register is greater than the high limit or less than the low limit.

### 27.2 TACH Monitor Block Diagram

**FIGURE 27-1: BLOCK DIAGRAM OF TACH MONITOR**



# MEC1618/MEC1618i

**Note:** After the counter is enabled it is a 16-bit free-running counter. Latch count value on a read or when number of tach pulses is detected (if enabled) for 1, 2, or 4 pulses. Counter is reset to 0000h if the count value is latched by a programmed number of tach pulses and on a VCC POR. Counter enable is software controlled signal.

## 27.3 Block Diagram Signal List

TABLE 27-1: TACH PORT LIST

Signal Name	Direction	Description
VTR POR	INPUT	<a href="#">nSYS_RST</a>
Master_Clock	INPUT	<a href="#">MCLK</a>
CLOCK_LOW	INPUT	100kHz <a href="#">MCLK_DIV203_EN</a>
EC I/F	I/O Bus	EC-side SPB bus
Tach Input	INPUT	Tachometer signal from TACHx Pin
Interrupts	OUTPUT	Interrupt used to indicate that either Tach Input has changed state or the TACH reading has been updated. One per TACH
SLP_EN	INPUT	Sleep Enable input from MEC1618/MEC1618i Clock Generator Power Management Interface.
CLK_REQ	OUTPUT	Clock Required output to MEC1618/MEC1618i Clock Generator Power Management Interface.

## 27.4 Power, Clocks and Reset

### 27.4.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.5, "Power Configuration," on page 101](#) for details on power domains.

### 27.4.2 CLOCKS

This block uses the [EC Bus Clock](#) and the 100kHz [MCLK\\_DIV203\\_EN](#). [EC Bus Clock](#) is used when reading and writing the [TACH Monitor](#) control registers. The individual TACH counters are driven by Clock\_Low, the [MCLK\\_DIV203\\_EN](#).

The [TACH Monitor](#) clock required output ([CLK\\_REQ](#)) is the inversion of the sleep enable input ([SLP\\_EN](#)). The [CLK\\_REQ](#) output is not asserted when the [TACH Monitor](#) is disabled.

See also [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

#### 27.4.2.1 Clock Idle

When the internal ring oscillator is disabled or when the TACH block is disabled, the internal TACH counters are reset. The reading register is not affected. This insures that inaccurate readings are not generated if the master clock halts in the middle of a TACH reading or when the TACH starts up.

**Note:** Each Tach pin should be pulled up via an external resistor to the main power supply.

### 27.4.3 RESET

This block is reset on a [nSYS\\_RST](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 27.5 TACH Interrupts

Each [TACH Monitor](#) in the MEC1618/MEC1618i can be used to generate one interrupt event. Each [TACH Monitor](#) interrupt source is a level, active high signal. The [TACH Monitor](#) interrupts are routed to the [TACH5](#), [TACH4](#), [TACH3](#), [TACH2](#), [TACH1](#), & [TACH0](#) bits in [GIRQ17 Source Register on page 331](#). The [TACH Monitor](#) interrupts generate interrupt events.

## 27.6 TACH Circuitry

The TACH Circuitry is implemented as a pulse counter. There are two types of toggling signals that can be used to increment the counter: the raw tach input or [CLOCK\\_LOW](#). See [FIGURE 27-1: Block Diagram of TACH Monitor on page 469](#). The two modes for incrementing the counter are controlled by [Tach Reading Mode Select](#) in the [TACHx Control Register](#).

If the raw tach is used to increment the counter, the circuitry can be configured as a free-running counter that increments when a pulse from the tach is detected (i.e., input signal transitions from low-to-high). The counter is latched into the reading register ([Tachx Counter](#) in the [TACHx Control Register](#)) every time it is incremented. If this mode is selected, firmware will monitor the number of pulses detected over a period of time to determine the speed of the attached fan.

If [CLOCK\\_LOW](#) is used to increment the counter, the raw tach input will be used to determine when to latch the current count value into the reading register and reset the counter to 0000h. The counter is latched after a programmed number of tach pulses is detected. The programmed period can be configured to be 1, 2, or 4 tach pulses in duration.

Each Tach counter has comparison logic to compare the counter value with the high limit and low limit registers.

### 27.6.1 TACH INTERRUPT SOURCES

There are three interrupt source events: notify EC when reading is updated, notify EC when TACH input toggles, or notify EC when the TACH reading exceeds a programmed limit. The corresponding interrupt status bits are Count Ready Status Toggle Status Out-of-Limit Status Bits[3,1,0] in [TACHx Status Register on page 475](#).

#### 27.6.1.1 Count Reading Ready Status

This status bit is asserted when the counter value is latched. The bit is implemented in Bit D3 of the [TACHx Status Register](#).

#### 27.6.1.2 Tach Input Toggle Status

This status bit is asserted when the Tach input changes state. The bit is implemented in Bit D2 of the [TACHx Status Register](#).

#### 27.6.1.3 TACH Out-of\_Limit STATUS

To generate a TACH out-of-limit status event, the high and low limits may be programmed in the [TACHx High Limit Register](#) and [TACHx Low Limit Register](#). An out-of-limit event is triggered when the reading register ([Tachx Counter](#) in the [TACHx Control Register](#)) is set to a value less than the [TACHx Low Limit Register](#) or to a value greater than the [TACHx High Limit Register](#). If the value in the [Tachx Counter](#) violates the programmed limits the TACH limit registers a status event will be generated, indicating the out-of-limit event. This status bit is implemented in Bit D0 of the [TACHx Status Register](#). This signal may be used to interrupt the Embedded Controller, if enabled via Bit D0 of the [TACHx Control Register](#).

**Note:** If the [TACHx Low Limit Register](#) is set to 0000h, no out-of-limit event will be triggered by a Tachx Counter value that is below the limit. If the [TACHx High Limit Register](#) is set to FFFFh, no out-of-limit event will be triggered by a Tachx Counter value that is above the limit.

**APPLICATION NOTE:** Out-of-Limit checks are typically only used when the tach counter is incremented in Mode 1 (in which the counter counts the number of [CLOCK\\_LOW](#) until a programmed number of pulses occur on the raw tach input).

# MEC1618/MEC1618i

## 27.7 Registers

There are six block instances defined in this chapter: TACH[5:0].

Each instance of the [TACH Monitor](#) has its own Logical Device Number, and Base Address as indicated in [Table 27-2](#).

**TABLE 27-2: [TACH Monitor](#) BASE ADDRESS TABLE**

<a href="#">TACH Monitor Instances</a>	<b>LDN from</b> ( <a href="#">Table 3-3 on page 52</a> )	<b>AHB Base Address</b>
<a href="#">TACH0</a>	18h	<a href="#">F0_6000h</a>
<a href="#">TACH1</a>		<a href="#">F0_6080h</a>
<a href="#">TACH2</a>		<a href="#">F0_6100h</a>
<a href="#">TACH3</a>		<a href="#">F0_6180h</a>
<a href="#">TACH4</a>		<a href="#">F0_6200h</a>
<a href="#">TACH5</a>		<a href="#">F0_6280h</a>

[Table 27-3](#) is a register summary for one instance of the [TACH Monitor](#).

**TABLE 27-3: [TACHX REGISTER SUMMARY](#)**

<b>Register Name</b>	<b>EC Interface</b>			<b>Notes</b>
	<b>SPB Offset</b>	<b>Byte Lane</b>	<b>EC Type</b>	
<a href="#">TACHx Control Register</a>	00h	3-0	R/W	
<a href="#">TACHx Status Register</a>	04h	0	R/W	
<a href="#">TACHx High Limit Register</a>	08h	1-0	R/W	
<a href="#">TACHx Low Limit Register</a>	0Ch	1-0	R/W	



## 27.7.1 DETAILED DESCRIPTION OF TACHOMETER REGISTER VALUES

This section describes the parameters that must be stored in hardware registers that will be used by the TACH logic.

### 27.7.1.1 TACHx Control Register

**TABLE 27-4: TACHX CONTROL REGISTER**

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	00h				32-bit			EC SIZE	
POWER	VTR				0000_0000h			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE3 BIT	D31	D30	D29	D28	D27	D26	D25	D24	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	TACHx Counter [15:8] Register								
BYTE2 BIT	D23	D22	D21	D20	D19	D18	D17	D16	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	TACHx Counter [7:0] Register								
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
BIT NAME	Tach Input INT_EN	Count Ready INT_EN	Reserved	Tach Edges		Tach Reading Mode Select	Reserved	Filter Enable	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R/W	R/W	
BIT NAME	Reserved						TACH Enable	Tach Out-of- Limit Enable	

## TACH OUT-OF-LIMIT ENABLE

The TACH Out-of\_Limit Enable is used to enable Bit[0] TACH Out-of\_Limit Status bit to generate an interrupt event.

0=disable interrupt output from tach block (default)

1=enable interrupt output from tach block

## TACH ENABLE

This bit enables the TACH logic.

0=TACH Idle (default)

This mode gates the clocks to the TACH block. The TACHx pin is tristate in the idle mode.

1=TACH Monitoring enabled.

**APPLICATION NOTE:** This bit gates the clocks into the block. When re-enabled, the internal counters will continue from the last known state and stale status events may still be pending. Firmware should discard any status or reading values until the reading value has been updated at least one time after the enable bit is set.

## FILTER ENABLE

The TACH glitch filter rejects input pulses that are less than three [CLOCK\\_LOW](#) periods wide.

0=Filter disabled (default)

1=Filter enabled

## TACH READING MODE SELECT

0 = Counter is incremented when Tach Input transitions from low-to-high state (default)

1 = Counter is incremented on the rising edge of the [CLOCK\\_LOW](#) input. The counter is latched into [Tachx Counter](#) and reset when the programmed number of edges is detected.

## TACH EDGES

A tach signal is a square wave with a 50% duty cycle. Typically, two tach periods represents one revolution of the fan. A tach period consists of three tach edges.

This programmed value represents the number of tach edges that will be used to determine the interval for which the number of [CLOCK\\_LOW](#) pulses will be counted

00 = 2 Tach edges (1/2 tach period)

01 = 3 Tach edges (1 tach period)

10 = 5 Tach edges (2 tach periods)

01 = 9 Tach edges (4 tach periods)

## COUNT READY INT\_EN

0=disable interrupt output from tach block (default)

1=enable interrupt output from tach block

## TACH INPUT INT\_EN

0=disable interrupt output from tach block (default)

1=enable interrupt output from tach block

## TACHX COUNTER

This 16-bit field contains the latched value of the tach counter, which may be configured to operate as a free-running counter or to be gated by the tach input signal.

If the counter is free-running (Mode 0), it increments (if enabled) at the rate determined by the raw tach signal and latched into this field every time it is incremented. The act of reading this field will not reset the counter, which rolls over to 0000h after FFFFh. The firmware will compute the delta between the current count reading and the previous count reading, to determine the number of pulses detected over a programmed period.

If the counter is being gated by the tach input and clocked by the [CLOCK\\_LOW](#) (Mode 1), the counter will be latched into the reading register when the programmed number of edges is detected or when the counter reaches FFFFh and the counter will be reset to zero.

**APPLICATION NOTE:** In Mode 1, a counter rate of FFFFh means that the tach did not detect the programmed number of edges in 655ms. A stuck fan can be detected by setting the [TACHx High Limit Register](#) to a number less than FFFFh. If the counter then reaches FFFFh, the reading register will be set to FFFFh and an out-of-limit interrupt can be sent to the EC.

## 27.7.1.2 TACHx Status Register

**TABLE 27-5: TACHX STATUS REGISTER**

HOST ADDRESS	n/a				n/a		HOST SIZE	
EC OFFSET	04h				32-bit		EC SIZE	
POWER	VTR				0000h		nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3-1 BIT	D31	D32	D31	...		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R/WC	R/WC	R	R/WC
BIT NAME	Reserved				Count Ready Status	Toggle Status	TACH Pin Status	Tach Out-of-Limit Status

### TACH OUT-OF-LIMIT STATUS

This bit is set when the Tach Count value is greater than the high limit or less than the low limit. It is cleared when written with a 1. To disable this status event set the limits to their extreme values. If enabled via [TACH Out-of-Limit Enable](#) in the [TACHx Control Register](#), this status bit will assert the Interrupt signal, which may be enabled to cause an interrupt event to the embedded controller.

0=Within Limits (TACH count value is less than or equal to the high limit or greater than or equal to the low limit).

1=Out of Limits (TACH count value is greater than the high limit or less than the low limit).

### TACH PIN STATUS

This bit reflects the state of Tach Input. This bit is a read only bit that may be polled by the embedded controller.

0=Tach Input is low

1=Tach Input is high

### TOGGLE STATUS

This bit is set when Tach Input changes state. It is cleared when written with a 1. If enabled via [Tach Input INT\\_EN](#) in the [TACHx Control Register](#), this status bit will assert the Interrupt signal, which may be enabled to cause an interrupt event to the embedded controller.

0=Tach stable (default)

1=Tach Input changed state (this bit is set on a low-to-high or high-to-low transition)

# MEC1618/MEC1618i

**APPLICATION NOTE:** Some fans offer a Locked Rotor output pin that generates a level event if a locked rotor is detected. This bit may be used in combination with the tach pin status bit to detect a locked rotor signal event from a fan.

**APPLICATION NOTE:** Tach Input may come up as active for Locked Rotor events. This would not cause an interrupt event because the pin would not toggle. Firmware must read the status events as part of the initialization process, if polling is not implemented.

## COUNT READY STATUS

The [Count Ready Status](#) bit remains cleared to '0' when the [Tach Reading Mode Select](#) bit in the [TACHx Control Register](#) is clear to '0'.

When the [Tach Reading Mode Select](#) bit in the [TACHx Control Register](#) is set to '1', The [Count Ready Status](#) bit is set when the counter value is latched by the hardware. It is cleared when written with a 1. If enabled via the [Count Ready INT\\_EN](#) bit in the [TACHx Control Register](#), this status bit will assert the Interrupt signal, which may be enabled to cause an interrupt event to the embedded controller.

0=Reading not ready

1=Reading ready

## 27.7.1.3 TACHx High Limit Register

**TABLE 27-6: TACHX HIGH LIMIT REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSETS	08h			32-bit			EC SIZE	
POWER	VTR			FFFFh			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE[3:2] BIT	D31	D30	D29	...		D18	D17	D16
HOST TYPE	R	R	R	R	R	R	R	R
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BYTE1-0 BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	TACHx High Limit [15:0] Register							

The TACHx High Limit [15:0] value is compared with the value in the [TACHx Control Register](#). If the value in the [TACHx Control Register](#) is greater than the value programmed in the [TACHx High Limit Register](#) the TACH Out-of-Limit STATUS bit will be set. The TACH Out-of-Limit status event may be enabled to generate an interrupt to the embedded controller via Bit[0] of the [TACHx Control Register](#).

**Note:** To disable this event program FFFFh into this register.

## 27.7.1.4 TACHx Low Limit Register

TABLE 27-7: TACHX LOW LIMIT REGISTER

HOST ADDRESS	n/a				n/a		HOST SIZE	
EC OFFSETS	0Ch				32-bit		EC SIZE	
POWER	VTR				0000h		nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3-2 BIT	D31	D29	D28	...		D18	D17	D16
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	TACHx Low Limit [15:0] Register							

The TACHx Low Limit [15:0] value is compared with the value in the [Tachx Counter](#) Field of the [TACHx Control Register](#). If the value in the [Tachx Counter](#) Field is less than the value programmed in the [TACHx Low Limit Register](#) the TACH Out-of\_Limit STATUS bit will be set. The TACH Out-of-Limit status event may be enabled to generate an interrupt to the embedded controller via Bit[0] of the [TACHx Control Register](#).

To disable this event program 0000h into this register.

## 28.0 PWM CONTROLLER

### 28.1 General Description

The function of this block is to generate a PWM output that may be used to control 4-wire fans, blink LEDs, etc. Each PWM can generate an arbitrary duty cycle output at frequencies from 0.095 Hz to 10 MHz.

The PWMx Counter ON Time registers and PWMx Counter OFF Time registers determine the operation of the PWM\_OUTPUT signal. See [Section 28.3.1, "PWMx Counter ON/OFF Time Registers," on page 482](#) for a description of the PWM\_OUTPUT signal.

#### 28.1.1 PWM\_OUTPUT

The PWM\_OUTPUT signal is used to generate a duty cycle at a frequency. This block has been designed such that the PWM signal may be programmed to hold PWM\_OUTPUT high, to hold PWM\_OUTPUT low, or to toggle PWM\_OUTPUT. If the PWM is configured to toggle, then PWM\_OUTPUT will alternate high and low for the programmed duration in the [PWMx Counter ON/OFF Time Registers](#) registers as defined in the register description. The PWM equations are described in [Figure 28-2](#).

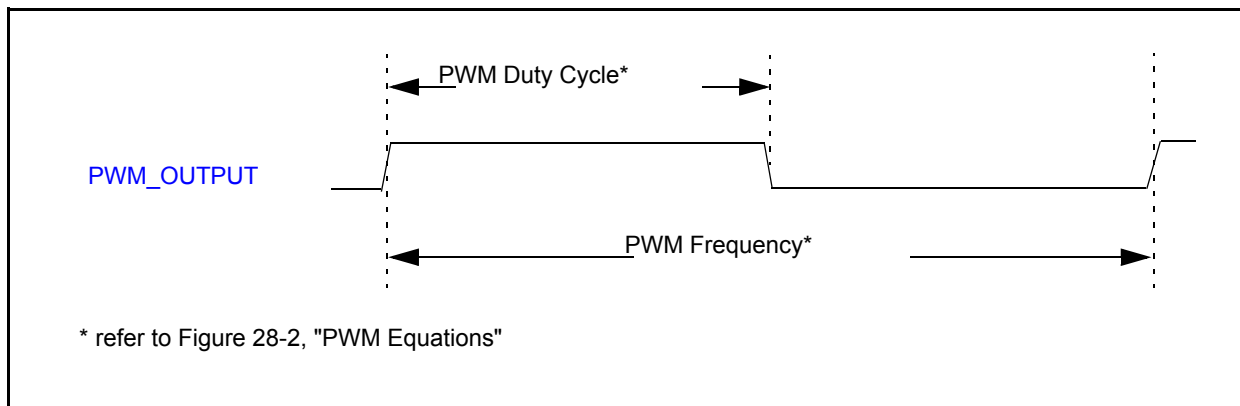
The PWM pin signal functions are routed pins described in [Table 3.12, "Fan PWM & Tachometer Interface," on page 59](#).

#### 28.1.2 PWM FEATURES

**APPLICATION NOTE:** Each PWM pin signal functions is MUXed with a GPIO pin signal function. The pin's default signal function is GPIO input as controlled by the associated [Pin Control Register](#). (See [Section 21.0, "GPIO Interface," on page 398](#)). At VTR POR or when a WDT event occurs (see [Section 16.0, "Watchdog Timer Interface," on page 349](#)), the pin will tristate. For fan applications, an external resistor termination can provide the pin state to force the external fans to the full on state, thereby protecting the system from overheating.

#### 28.1.3 PWM CONTROLLER BLOCK DIAGRAM

**FIGURE 28-1: PWM FUNCTIONAL DIAGRAM**



**FIGURE 28-2: PWM EQUATIONS**

$$\text{PWM Frequency} = \frac{1}{(\text{PreDivider} + 1)} \times \frac{(\text{Clock Select} * 20.27 \text{ MHz} \& \text{Clock Select} * (20.27 \text{ MHz}/203))}{(\text{PWM Counter OFF Time}[15:0] + 1) + (\text{PWM Counter ON Time}[15:0] + 1)}$$

$$\text{PWM Duty Cycle} = \frac{(\text{PWM Counter ON Time}[15:0] + 1)}{(\text{PWM Counter OFF Time}[15:0] + 1) + (\text{PWM Counter ON Time}[15:0] + 1)}$$

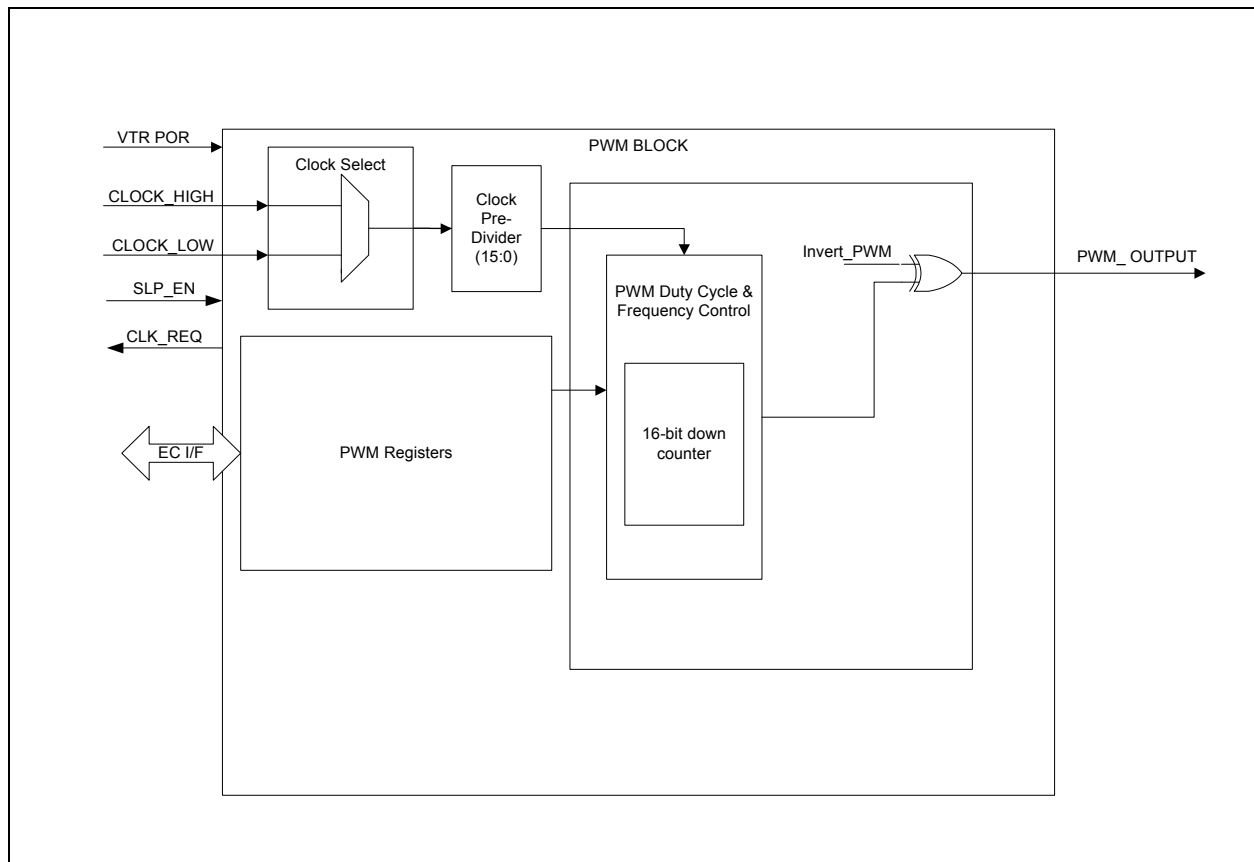
**Legend:**

\* = a Boolean AND operator  
& = a Boolean OR operator

**Notes:**

Clock Select, PreDivider, PWM Counter OFF Time[15:0], and PWM Counter ON Time[15:0] are register values defined below.

**FIGURE 28-3: BLOCK DIAGRAM OF PWM CONTROLLER**



# MEC1618/MEC1618i

## 28.1.3.1 Block Diagram Signal List

**TABLE 28-1: BLOCK DIAGRAM SIGNAL LIST DESCRIPTION**

Signal Name	Direction	Description
VTR POR	INPUT	VTR Power on Reset.
CLOCK_HIGH	INPUT	20.27MHz MCLK.
CLOCK_LOW	INPUT	100kHz MCLK_DIV203_EN.
PWM_OUTPUT	OUTPUT	Pulse Width Modulated signal to PWMx pin.
E/C IF	I/O Bus	EC-side SPB bus.
SLP_EN	INPUT	Sleep Enable input.
CLK_REQ	OUTPUT	Clock Required output.

## 28.2 Power, Clocks and Reset

### 28.2.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.5, "Power Configuration," on page 101](#) for details on power domains.

### 28.2.2 CLOCKS

This block uses the [EC Bus Clock](#), the 20.27MHz MCLK and the 100kHz MCLK\_DIV203\_EN. The [EC Bus Clock](#) is used when reading and writing the [PWM Controller](#) control registers. The individual PWM counters can be driven either by MCLK or MCLK\_DIV203\_EN.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

#### 28.2.2.1 Pre-Divider

The clock source to the PWM Down Counter used to generate a duty cycle and frequency on the PWM\_OUTPUT may be pre-divided via bits D6:D3 in the [PWMx Configuration Register](#). This results in a wide range of frequencies for the pwm output. [Table 28-2](#) shows examples of frequencies supported:

**TABLE 28-2: EXAMPLE OF PWM FREQUENCIES**

Clock	Clock Select	Clock Re-divider	High Count	Low Count	PWM Output Frequency (Hz)	PWM Output Duty Cycle
20.27 MHz	0	0	32767	32767	309	50%
20.27 MHz	0	0	192	192	52,500	50%
20.27 MHz	0	0	382	2	52,500	99%
20.27 MHz	0	0	960	960	10,500	50%
20.27 MHz	0	0	32767	32767	309	50%
100 KHz	1	1	32767	32767	0.76	50%
100 KHz	1	11	32767	32767	0.13	50%



## 28.2.2.2 Sleep Enable

The Embedded Controller can put each PWM into a sleep state. When a PWM is in the sleep state the internal counters are reset to 0 and the internal state of the PWM and thus the PWM\_OUTPUT signal is set to the OFF state.

The [PWM Controller](#) clock required output (CLK\_REQ) is the inversion of the sleep enable input (SLP\_EN). The CLK\_REQ output is not asserted when the [PWM Controller](#) is disabled.

The PWM participation in the sleep state is controlled by the PWMx bits in the [EC Blocks Sleep Enables Register 1](#) and [EC Blocks Sleep Enables Register 2](#).

## 28.2.3 RESET

This block is reset by nSYS\_RST. After the assertion of nSYS\_RST, PWM\_OUTPUT is held in the OFF state and the hardware resets the pwm counter registers to their default value.

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 28.3 Registers

There are 16 instances of the [PWM Controller](#) block implemented in the MEC1618/MEC1618i enumerated [15:0]. Each instance of the [PWM Controller](#) has its Base Address as indicated in [Table 28-3, "PWMx Controller Base Address Table"](#):

**TABLE 28-3: PWMX CONTROLLER BASE ADDRESS TABLE**

PWM Controller Instance	LDN from ( <a href="#">Table 3-3 on page 52</a> )	AHB Base Address
<a href="#">PWM(7-0).0</a>	16h	<a href="#">F0_5800h</a>
<a href="#">PWM(7-0).1</a>		F0_5880h
<a href="#">PWM(7-0).2</a>		F0_5900h
<a href="#">PWM(7-0).3</a>		F0_5980h
<a href="#">PWM(7-0).4</a>		F0_5A00h
<a href="#">PWM(7-0).5</a>		F0_5A80h
<a href="#">PWM(7-0).6</a>		F0_5B00h
<a href="#">PWM(7-0).7</a>		F0_5B80h
<a href="#">PWM(15-8).8</a>	17h	<a href="#">F0_5C00h</a>
<a href="#">PWM(15-8).9</a>		F0_5C80h
<a href="#">PWM(15-8).10</a>		F0_5D00h
<a href="#">PWM(15-8).11</a>		F0_5D80h
<a href="#">PWM(15-8).12</a>		F0_5E00h
<a href="#">PWM(15-8).13</a>		F0_5E80h
<a href="#">PWM(15-8).14</a>		F0_5F00h
<a href="#">PWM(15-8).15</a>		F0_5F80h

[Table 28-4](#) summarizes the registers allocated for each Instance. The offset field in the following table is the offset from the Embedded Controller (EC) Base Address.

# MEC1618/MEC1618i

TABLE 28-4: PWMX REGISTER SUMMARY

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	LC Type	
PWMx Counter ON Time Register	00h	1-0	R/W	
PWMx Counter OFF Time Register	04h	1-0	R/W	
PWMx Configuration Register	08h	1-0	R/W	

TABLE 28-5: PWMX EC ACCESSIBLE REGISTERS

Offset	Register Name	VTR POR (Suspend)
0h	PWMx Counter ON Time Register	0000h
4h	PWMx Counter OFF Time Register	FFFFh
8h	PWMx Configuration Register	0000h

## 28.3.1 PWMX COUNTER ON/OFF TIME REGISTERS

TABLE 28-6: PWMX COUNTER ON TIME REGISTER

HOST ADDRESS	n/a				n/a		HOST SIZE	
EC OFFSET	00h				16-bit		EC SIZE	
POWER	VTR				0000h		nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	PWMx Counter ON Time[15:0]							

**TABLE 28-7: PWMX COUNTER OFF TIME REGISTER**

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	04h			16-bit			EC SIZE	
POWER	VTR			FFFFh			nSYS_RST DEFAULT	
BUS	EC SPB							
BIT	D15	D14	D13	...		D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	PWMx Counter OFF Time[15:0]							

The PWMx Counter ON/OFF Time registers determine both the duty cycle and frequency of the signal generated on PWM\_OUTPUT. See [FIGURE 28-2: PWM Equations on page 479](#).

If the PWMx Counter OFF Time[15:0] is set to zero, PWM\_OUTPUT is held high (Full On). If the PWMx Counter ON Time is set to zero and the PWMx Counter OFF Time[15:0] is not set to zero, PWM\_OUTPUT is held low (Full Off). Note that the default case is full off. Otherwise, both the high and low count registers will contain a value that will be used to determine the length of time PWM\_OUTPUT will be held high and low. See [Table 28-8, "PWM\\_OUTPUT State"](#).

**TABLE 28-8: PWM\_OUTPUT STATE**

<b>PWM Count On Time</b>	<b>PWM Count Off Time</b>	<b>State Of PWM_OUTPUT</b>
Don't Care	0000h	Full On
0000h	Non-Zero Value	Full Off
Non-Zero Value	Non-Zero Value	Toggling On and Off

The counter values preload a 16-bit down-counter that is clocked by either the high frequency clock source or the low frequency clock source (see bit[1] CLK\_Select of [PWMx Configuration Register](#)). The firmware will program the on and off count values that correspond to the PWM Current Duty Cycle and PWM Frequency. When PWM\_OUTPUT is OFF and the internal counter is zero, the PWMx Counter ON Time is loaded into the counter. The PWM\_OUTPUT signal will transition to the ON state and the internal counter will count down to zero at the programmed frequency for the duration of the programmed on time. Similarly, when the PWM\_OUTPUT is in the ON state and the internal counter is zero, the PWMx Counter OFF Time is loaded into the counter. The PWM\_OUTPUT signal will transition OFF and the internal counter will count down to zero at the programmed frequency for the duration of the programmed off time.

The [PWMx Counter ON/OFF Time Registers](#) may be updated at any time. Values written into the two registers are kept in holding registers. The holding registers are transferred into the [PWMx Counter ON/OFF Time Registers](#) when all four bytes have been written with new values and the internal counter completes the OFF time count. If the PWM is in the Full On state then the [PWMx Counter ON/OFF Time Registers](#) are updated from the holding registers as soon as all four bytes have been written. After the two registers have been updated the holding registers are marked empty, and all four bytes must again be written before the holding registers will be reloaded into the [PWMx Counter ON/OFF Time Registers](#). Reads of the [PWMx Counter ON/OFF Time Registers](#) return the current contents of the registers that are used to load the counter and not the holding registers.

# MEC1618/MEC1618i

## 28.3.2 PWMX CONFIGURATION REGISTER

TABLE 28-9: PWMX CONFIGURATION REGISTER

HOST ADDRESS	n/a			n/a			HOST SIZE	
EC OFFSET	08h			16-bit			EC SIZE	
POWER	VTR			0000h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R							
BIT NAME	Reserved							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved	Clock Pre-Divider				Invert	Clock Select	PWM Enable

### PWM ENABLE

0= disabled (gates clocks to save power) (default)

1= enabled

**Note:** When the PWM enable bit is set to 0 the internal counters are reset and the internal state machine is set to the OFF state. In addition, the PWM\_OUTPUT signal is set to the inactive state as determined by the Invert bit. The [PWMx Counter ON/OFF Time Registers](#) are not affected by the PWM enable bit and may be read and written while the PWM enable bit is 0.

### CLOCK SELECT

The Clk\_Select bit determines the clock source used by the PWM duty cycle and frequency control logic.

0= 20.27MHz [MCLK](#) (default)

1= 100kHz [MCLK\\_DIV203\\_EN](#)

### INVERT

0= PWM\_OUTPUT ON State is active high

1= PWM\_OUTPUT ON State is active low

### CLOCK PRE-DIVIDER

The Clock source for the 16-bit down counter (see [PWMx Counter ON/OFF Time Registers](#)) is determined by bit D1 of this register. The Clock source is then divided by the value of Pre-Divider+1 and the resulting signal determines the rate at which the down counter will be decremented. For example, a Pre-Divider value of 1 divides the input clock by 2 and a value of 2 divides the input clock by 3. A Pre-Divider of 0 will disable the Pre-Divider option.

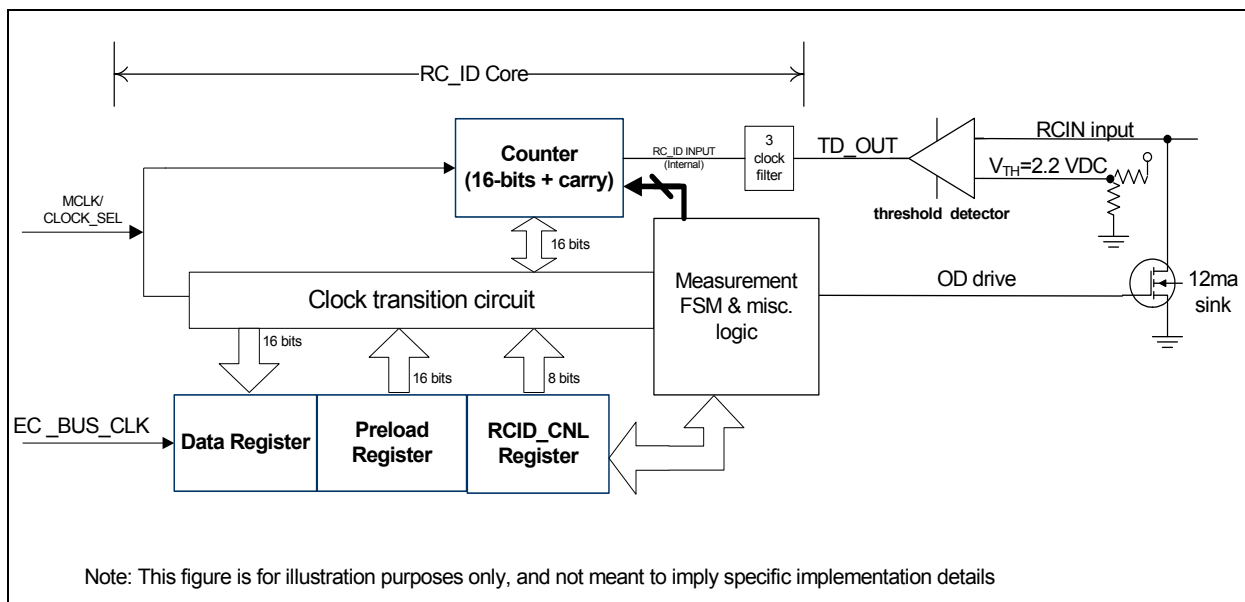
## 29.0 RC IDENTIFICATION DETECTION (RC\_ID)

### 29.1 General Description

The Resistor/Capacitor Identification Detection (RC\_ID) interface provides a single pin interface which can discriminate a number of quantized RC constants. The judicious selection of RC values can provide a low cost means for system element configuration identification. The RC\_ID I/O pin measures the charge/discharge time for an RC circuit connected to the pin as shown in [Figure 29-1](#).

### 29.2 Block Diagram

**FIGURE 29-1: BLOCK DIAGRAM OF RC Identification Detection (RC\_ID)**



### 29.3 Power, Clocks and Reset

#### 29.3.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

#### 29.3.2 CLOCKS

This block uses the [EC Bus Clock](#) and [MCLK](#). The [EC Bus Clock](#) is used to access the [Registers](#) described in this block. [MCLK](#) is divided down to provide a sampling clock.

See [Section 5.1.2, "Clock Generator," on page 74](#) for details on clocks.

#### 29.3.3 POWER ON RESET

This block is reset on a [nSYS\\_RST](#). Following a system reset, all Register are reset to 00h and the state machines are set to [Reset](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

### 29.4 Interrupts

The [RC Identification Detection \(RC\\_ID\)](#) can generate an RCID\_DONE interrupt when the [DONE](#) bit in the [RCID\\_CTL Register](#) is set. The interrupt source is routed onto the [RCID](#) bit in [GIRQ16 Source Register on page 329](#) and is a level, active high signal.

# MEC1618/MEC1618i

## 29.5 Time Constants

This section lists a set of R and C values which can be connected to the RC\_ID pin. Note that risetime generally follow RC time Tau. Firmware should use the Max and Min Limits to create quantized states.

### 29.5.1 SPECIFIC TIME CONSTANTS

**TABLE 29-1: SAMPLE RC VALUES (C=2200 PF. R VARIED)**

LIMITS				External Circuit Components		
MIN count	MAX count	Range	Band Gap	C (pF)	R (K)	
Avg-20%	Avg+10%	max-min	between RC values	10%	5%	
38	53	15		2200	1	
75	105	30	22	2200	2	
162	224	62	57	2200	4.3	
309	426	117	85	2200	8.2	
1202	1654	452	776	2200	33	
2244	3086	842	590	2200	62	
4538	6241	1703	1452	2200	130	
7999	11000	3001	1758	2200	240	
MIN	MAX	Range		C (pF)	R (K)	Tau =RxC
Risetime (usec)	Risetime (usec)	max-min		10%	5%	(usec)
1.88	3.00	1.12		2200	1	2.20
3.74	5.14	1.40		2200	2	4.40
8.01	11.02	3.01		2200	4.3	9.46
15.28	21.02	5.73		2200	8.2	18.04
59.31	81.56	22.24		2200	33	72.60
110.72	152.24	41.52		2200	62	136.40
223.89	307.84	83.96		2200	130	286.00
394.67	542.67	148.00		2200	240	528.00

**TABLE 29-2: SAMPLE RC VALUES (C=3000 PF. R VARIED)**

LIMITS				External Circuit Components		
MIN count	MAX count	Range	Band Gap	C (pF)	R (K)	
Avg-20%	Avg+10%	max-min	between RC values	10%	5%	
51	72	21		3000	1	
102	142	40	30	3000	2	
222	306	84	80	3000	4.3	
423	583	160	117	3000	8.2	
1682	2314	632	1099	3000	33	
3155	4339	1184	841	3000	62	
6344	8724	2380	2005	3000	130	
11156	15340	4184	2432	3000	240	
MIN Risettime (usec)	MAX Risettime (usec)	Range max-min		C (pF) 10%	R (K) 5%	Tau =RxC (usec)
2.55	4.00	1.45		3000	1	3.00
5.08	6.99	1.91		3000	2	6.00
10.96	15.07	4.11		3000	4.3	12.90
20.91	28.75	7.84		3000	8.2	24.60
83.02	114.16	31.13		3000	33	99.00
155.65	214.02	58.37		3000	62	186.00
313.01	430.38	117.38		3000	130	390.00
550.38	756.78	206.39		3000	240	720.00

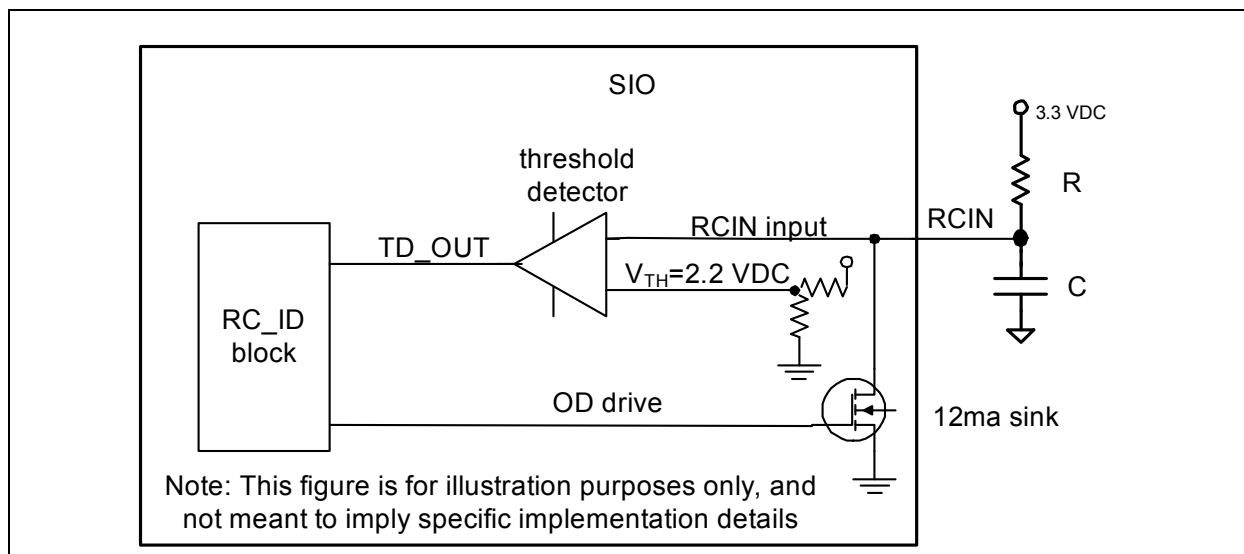
# MEC1618/MEC1618i

**TABLE 29-3: SAMPLE RC VALUES (C=4700 PF. R VARIED)**

LIMITS				External Circuit Components		
MIN count	MAX count	Range	Band Gap	C (pF)	R (K)	
Avg-20%	Avg+10%	max-min	between RC values	10%	5%	
80	112	32		4700	1	
160	222	62	48	4700	2	
344	474	130	122	4700	4.3	
659	907	248	185	4700	8.2	
2617	3600	983	1710	4700	33	
4862	6686	1824	1262	4700	62	
9866	13567	3701	3180	4700	130	
17384	23904	6520	3817	4700	240	
MIN	MAX	Range		C (pF)	R (K)	Tau = RxC
Risetime (usec)	Risetime (usec)	max-min		10%	5%	(usec)
3.99	5.60	1.61		4700	1	4.70
7.94	10.95	3.01		4700	2	9.40
17.00	23.65	6.65		4700	4.3	20.21
32.53	44.88	12.35		4700	8.2	38.54
129.14	176.55	47.41		4700	33	155.10
239.86	328.79	88.93		4700	62	291.40
486.74	648.23	161.49		4700	130	611.00
857.64	1140.70	283.06		4700	240	1128.00

## 29.6 Block Diagram

**FIGURE 29-2: RC\_ID CIRCUIT.INTERFACING TO THE RC\_ID BLOCK**





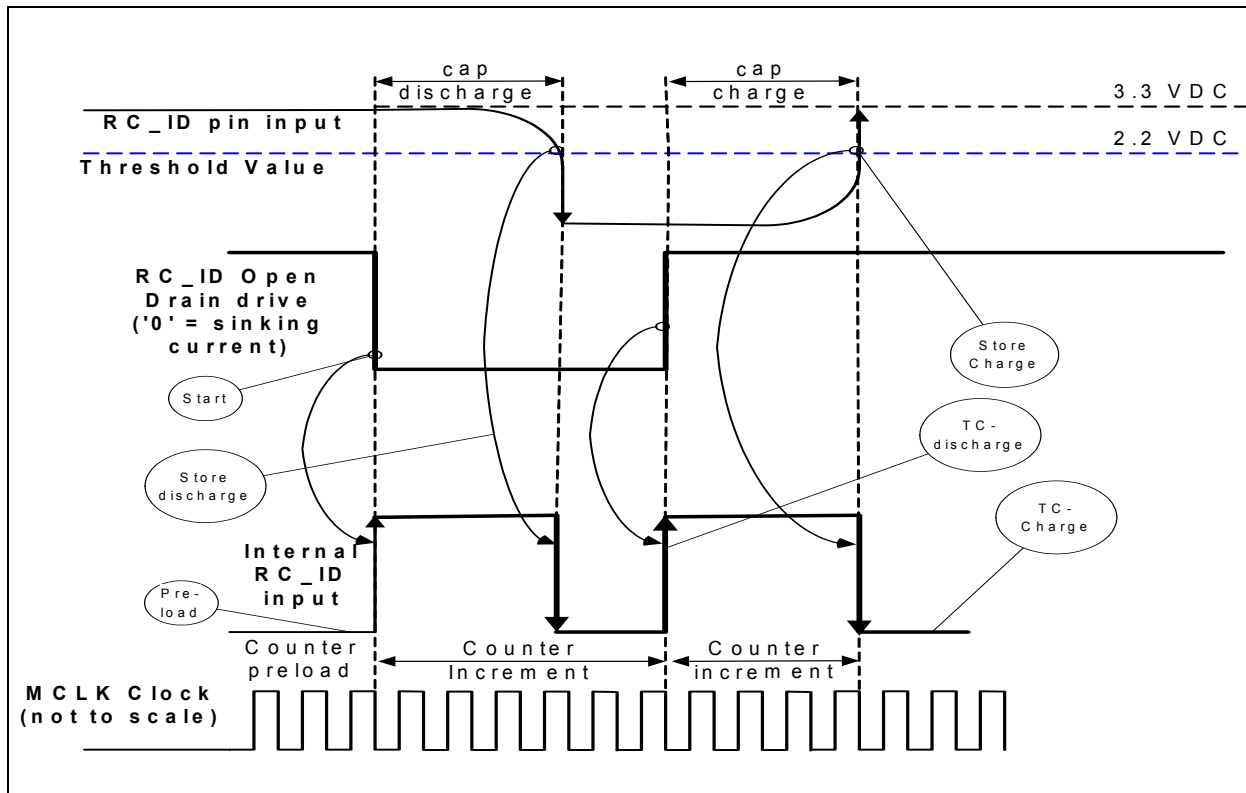
The RC\_ID block initiates the discharging followed by the charging of the external RC circuit (see [Figure 29-1](#)). At the same time, the RCID input goes through a threshold detector set at 68% of 3.3 VDC. The TD\_OUT gates the 16 bit Counter clocked by [MCLK](#) divided down by the [Clock Select Field](#) in the [RCID\\_CTL Register on page 491](#). The input has an input glitch rejection filter. Any change in input less than 3 [MCLK](#)-wide is ignored.

**APPLICATION NOTE:** After completion of a measurement cycle with the [DONE](#) bit set to '1' in the [RCID\\_CTL Register](#), the programmer must place the RC\_ID back into the reset state to before starting a new measurement.

**TABLE 29-4: RC\_ID MEASUREMENT STATES**

State	Description
Reset	<p>The <a href="#">RCID_CTL Register ENABLE</a> bit is cleared to '0' by <a href="#">nSYS_RST</a> or a write. The pin OD output driver is tristated, the RCID blocked is placed in low power mode. The <a href="#">DONE</a>, <a href="#">CY_ER</a>, <a href="#">TC</a> bits in the <a href="#">RCID_CTL Register</a> are autonomously cleared to '0'.</p> <p><b>Note:</b> Clearing the <a href="#">ENABLE</a> bit with a value other than 00h to the <a href="#">RCID_CTL Register</a> is not defined and may create unpredictable results.</p>
Enabled	<p>Setting the <a href="#">ENABLE</a> bit to '1' in the <a href="#">RCID_CTL Register</a> places the RC_ID interface active high power state.</p> <p><b>Note:</b> A 300us delay is required between the <a href="#">Enabled</a> state and the <a href="#">Start</a> state.</p>
Start	<p>The <a href="#">Start</a> state is initiated by a write to the <a href="#">RCID_CTL Register</a> setting the <a href="#">START</a> bit to '1'. The counter is initiated to the preload value of 0000h and starts incrementing. The pin OD driver begins to sink current and external capacitor starts discharging. The <a href="#">DONE</a>, <a href="#">CY_ER</a>, <a href="#">TC</a> bits in the <a href="#">RCID_CTL Register</a> are autonomously cleared to '0'.</p>
Detect Dis-charge	<p>The pin voltage decays as the external capacitor discharges and the counter continues to increment until the terminal count is reached. The pin voltage is monitored to detect the discharge voltage reaches below the threshold voltage before the counter reaches terminal count.</p>
TC-Dis-charged	<p>The incrementing counter reaches the terminal count value of FFFFh. The <a href="#">TC</a> bit in the <a href="#">RCID_CTL Register</a> is autonomously set to '1'. If the pin voltage fails to discharge below the threshold voltage before the counter reaches terminal count during <a href="#">Detect Discharge</a>, then <a href="#">CY_ER</a> bit in the <a href="#">RCID_CTL Register</a> is autonomously set to '1'; otherwise, then <a href="#">CY_ER</a> bit remains clear.</p> <p>The pin OD output driver is tristated and the counter starts incrementing from 0000h.</p>
Detect Charge	<p>The pin voltage rises to the threshold voltage, counter stops counting, and the present value of the counter is stored in the <a href="#">RC_ID Data Register</a>. The <a href="#">DONE</a> bit in the <a href="#">RCID_CTL Register</a> is autonomously set to '1'.</p>
TC-Charged	<p>The incrementing counter reaches the terminal count value of FFFFh and the pin voltage is below the threshold value. The <a href="#">CY_ER</a> bit and <a href="#">DONE</a> bits in the <a href="#">RCID_CTL Register</a> are autonomously set to '1'.</p> <p><b>Note:</b> The <a href="#">Detect Charge</a> and the <a href="#">TC-Discharged</a> STATES are mutually exclusive.</p>

**FIGURE 29-3: RC\_ID TIMING DIAGRAM**



All registers are VTR powered and are placed in reset when `nSYS_RST` (internal signal) is '0'.

## 29.7 Registers

Each instance of the [RC Identification Detection \(RC\\_ID\)](#) has its own Logical Device Number, and Base Address as indicated in [Table 29-5](#).

**TABLE 29-5: RC Identification Detection (RC\_ID) BASE ADDRESS TABLE**

RC ID Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
RC ID	4h	F0_1000h

[Table 29-6](#) is a register summary for this instance of the [RC Identification Detection \(RC\\_ID\)](#).

**TABLE 29-6: RC Identification Detection (RC\_ID) REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
RCID_CTL Register	00h	0	R/W	
RC_ID Data Register	04h	0	R	
		1		

## 29.8 Runtime Registers

### 29.8.1 RC\_ID CONTROL REGISTER

**TABLE 29-7: RCID\_CTL REGISTER**

HOST OFFSET	N/A						HOST SIZE	
EC OFFSET	00h						16-Bit	EC SIZE
POWER	VTR						00h	nSYS_RST DEFAULT
BUS	EC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R/W	R/W
BIT NAME	Reserved						Clock_Sel	
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R	R	R	R	R	R
BIT NAME	ENABLE	START	Reserved			CY_ER	TC	DONE

#### DONE

This read only status is set when the RCID completes a measurement and enters the “Detect Charge” or “TC-Charged” Measurement States described in [Table 29-4, “RC\\_ID Measurement States,” on page 489](#).

This bit is cleared when RCID enters the “Reset” or “Start” Measurement States described in [Table 29-4, “RC\\_ID Measurement States,” on page 489](#).

#### TC

This read only status bit is set when the RCID enters the “TC-Discharged” or “TC-Charged” Measurement State described in [Table 29-4, “RC\\_ID Measurement States,” on page 489](#).

This bit is cleared when RCID enters the “Reset” or “Start” Measurement States described in [Table 29-4, “RC\\_ID Measurement States,” on page 489](#).

# MEC1618/MEC1618i

## CY\_ER

This bit is a read only status bit and indicates when set to '1' that the counter reached terminal count during the Capacitive Discharge or Charge phases without crossing the voltage threshold. This is an error condition.

## START

Setting this bit to '1' causes the RCID to enter the "Start" RCID Measurement State described in [Table 29-4, "RC\\_ID Measurement States," on page 489](#). A 300us delay is required between the [Enabled](#) state and the [Start](#) state.

All writes to this register during other RC\_ID states should clear this bit to 00h. See [Note 29-1](#).

## ENABLE

Clearing this bit to '0' causes the RCID to enter the "Reset" RCID Measurement State described in [Table 29-4, "RC\\_ID Measurement States," on page 489](#).

Setting this bit to '1' starts the clock input to the RCID and arms the counter.

**APPLICATION NOTE:** The [ENABLE](#) bit should remain set during the entire measurement; therefore all writes to the [RCID\\_CTL Register](#) during a measurement should set this bit.

**Note 29-1** When writing to the [RCID\\_CTL Register](#) to clear the [ENABLE](#) bit, the [START](#) bit should be cleared to '0'. Clearing the [ENABLE](#) bit with a value other than 00h is not defined and may create unpredictable results.

## CLOCK\_SEL

This field selects the frequency of the Counter circuit clock. [Table 29-8, "Clock Select Field"](#) shows the clock frequencies that can be selected:

**TABLE 29-8: CLOCK SELECT FIELD**

<a href="#">Clock_Sel</a>	Counter Clock		Resolution per Bit	Full Count Duration
0	DIVIDE BY 1	20.27 MHz	49.33 ns	3.23 ms
1	DIVIDE BY 2	10.14 MHz	98.67 ns	6.47 ms
2	DIVIDE BY 4	5.07 MHz	197.34 ns	12.93 ms
3	DIVIDE BY 8	2.53 MHz	394.67 ns	25.86 ms

The values in the [Clock Select Field](#) should only be changed when the [ENABLE](#) bit in the [RCID\\_CTL Register](#) is cleared to '0'.

## 29.8.2 RC\_ID DATA REGISTER

The RC\_ID Data Register provides a 16 bit counter value with a 1/[MCLK](#) Resolution per bit.

Reads from this register in the [Detect Charge](#) Measurement States described in [Table 29-4, "RC\\_ID Measurement States," on page 489](#) provides the bytes of the measured result for the Charge time.

**TABLE 29-9: RC\_ID DATA REGISTER**

HOST ADDRESS							HOST SIZE	
EC OFFSET	04h			16-bit			EC SIZE	
POWER	VTR			0000h			EC SPB DEFAULT	
BUS	EC SPB							
BYTE1 BIT	D15	D14	D13	D12	D11	D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Data[15:8]							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	Data[7:0]							

## 29.9 Low Power Mode

The [RC Identification Detection \(RC\\_ID\)](#) interface is designed to conserve power when sleeping or disabled. [Table 29-10](#) summarizes the [RC Identification Detection \(RC\\_ID\)](#) interface [Low Power Mode](#) behavior.

**TABLE 29-10: BLOCK CLOCK GATING IN LOW POWER MODES**

ENABLE Bit	DONE Bit	RCID_SLEEP_EN	Block Idle Status (Note 29-2)	RCID_CLOCK_REQ	State	Description
0	X	X	X	0	SLEEPING	The block is disabled and the clock can be stopped.
1	0	0	NOT IDLE	1	NORMAL OPERATION	The block is not idle and neither disabled by firmware nor commanded to sleep.
1	0	1	NOT IDLE	1	PREPARING TO SLEEP	The block is commanded to sleep, but the clock is required until the Block is idle.
1	1	1	IDLE	0	SLEEPING	The block is commanded to sleep and idle. The clock can be stopped.

**Note 29-2** The [DONE](#) bit indicates the [RC Identification Detection \(RC\\_ID\)](#) interface 'idle' state.

## 30.0 GENERAL PURPOSE SERIAL PERIPHERAL INTERFACE (GP-SPI)

### 30.1 General Description

The SPI interfaces may be used to communicate with various peripheral devices, e.g., EEPROMS, DACs, ADCs, that use a standard Serial Peripheral Interface. There are two instances of GP-SPI controller, one located on the EC SPB bus and the other on the LPC SPB bus. The latter is intended for flash access by the host and EC; it can optionally work in conjunction with the DMA Controller to move data to and from the closely coupled SRAM with minimal software overhead.

**Characteristics of the GP-SPI Controller include:**

- 8-bit serial data transmitted and received simultaneously over two data pins in Full Duplex mode with options to transmit and receive data serially on one data pin in Half Duplex (Bidirectional) mode.
- An internal programmable clock generator and clock polarity and phase controls allowing communication with various SPI peripherals with specific clocking requirements.
- SPI cycle completion that can be determined by status polling or interrupts.
- The ability to read data in on both SPDIN and SPDOUT in parallel. This allows this SPI Interface to support dual data rate read accesses for emerging double rate SPI flashes
- Support of back-to-back reads and writes without clock stretching, provided the host can read and write the data registers within one byte transaction time.
- Hardware hooks to DMA Engine (available only in the Flash GP-SPI on the LPC SPB bus).

The MEC1618/MEC1618i SPI is a master only device and does not support multiple-master SPI configurations.

The GP-SPI controller on LPC SPB bus has its IO signals (pins) multiplexed with those from of the [EC AHB SPI Flash Read Controller](#) whose [Master Bridge Enable](#) register bit controls the multiplexer. See [Section 2.4.18, "SPI Controllers Interface," on page 23](#) for pin listings.



# MEC1618/MEC1618i

**TABLE 30-1: SPI BLOCK SIGNALS (CONTINUED)**

Signal	Direction	Description
SPDIN2	INPUT	Serial Data In 2 from SPDOUT pin. Input in bi-directional mode and Dual Read mode.
SPI_CLK	OUTPUT	SPI Clock output used to drive the SPCLK pin.
SPDOUT_Direction	OUTPUT	The SPDOUT pin may be used as an output or an input. This signal is used to determine the direction of the SPDOUT buffer. 0=output (SPDOUT pin is controlled by the <a href="#">SPDOUT</a> signal) 1=input (SPDOUT pin is an input driving the <a href="#">SPDIN2</a> signal)  The SPDOUT pin has I/O capability. The I/O capability is implemented to support the Half-Duplex mode of operation (also referred to as bi-directional mode) and the Dual Read mode.
SLEEP_EN	INPUT	External enable/disable signal used to put the block in the lowest power consumption state. 0=No Sleep Requested. The block should operate as configured. 1=Sleep Requested. The block enters sleep mode. See <a href="#">Low Power Mode on page 498</a> .
SPI_CLK_REQ	OUTPUT	This output indicates when this block requires this clock input. 0= <a href="#">MCLK</a> can be turned 'off' when appropriate 1= <a href="#">MCLK</a> is required to be 'on.'
LPC/EC SPB Bus IF	I/O Bus	Bus used by microprocessor to access the registers in this block.
SPI_CS#	OUTPUT	SPI chip select (only for the Flash GP-SPI on LPC SPB bus)
TxBE_STS	OUTPUT	Tx DMA request (only for the Flash GP-SPI on LPC SPB bus)
RxBE_STS	OUTPUT	Rx DMA request (only for the Flash GP-SPI on LPC SPB bus)

## 30.4 SPI Interface Signals

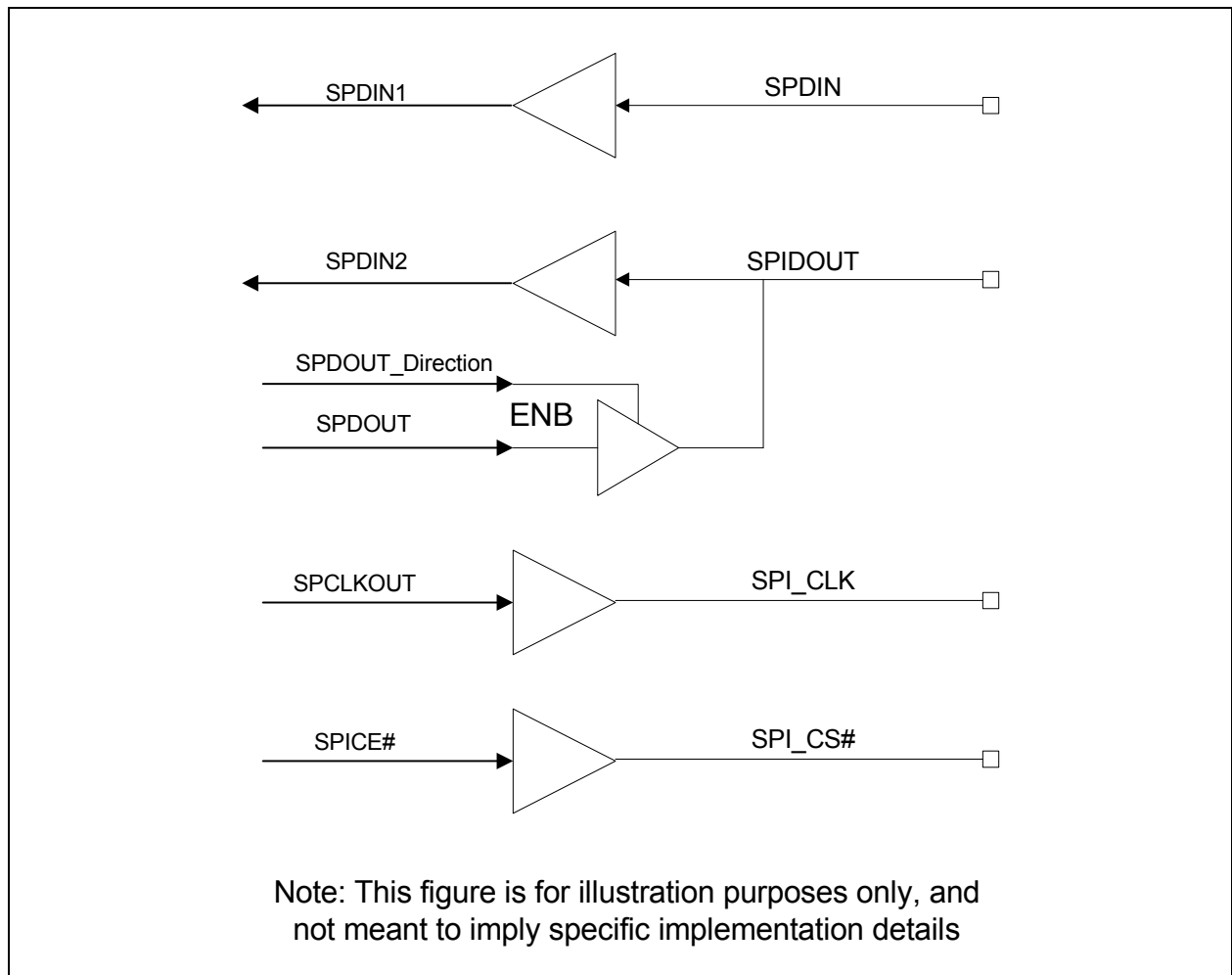
The following subsections describe the SPI Block Signals that are routed to the SPI pins. This chapter utilizes generic signal nomenclature for the Pin Signal Functions. [Table 30-2](#) is a specific lookup table for pin signal function names used in this chapter and used elsewhere. [Figure 30-2](#) show typical routing of the block interface to the pins. Look for the "Miscellaneous Functions" table in Chapter 3, Pin Configuration; this table includes the pin description for the SPI interface. Since there is only one instance of SPI port in the MEC1618/MEC1618i, there is a one-to-one correspondence.

**TABLE 30-2: PIN SIGNAL FUNCTION NOMENCLATURE LOOKUP TABLE**

Routing Figure Generic Pin Signal Name	Pin Signal Function Name Used in Other Contexts	Pin Function Signal Description
SPCLK	ECGP_SCLK	General Purpose SPI Clock
SPDOUT	ECGP_SOUT	General Purpose SPI Output
SPDIN	ECGP_SIN	General Purpose SPI Input



**FIGURE 30-2: TYPICAL BLOCK/PIN INTERFACE**



## 30.4.1 SPDOUT PIN - SERIAL PERIPHERAL DATA OUT

In Full Duplex Mode, this is the serial data output to the SPI interface. In half-duplex mode (also referred to as bi-directional mode) this is the serial data I/O port for the SPI interface.

For special SPI Flash devices that support Dual Read Modes the SPDOUT operates as in input in parallel with the SPDIN during the data portion of the Fast Dual Read command.

**Note:** In the Bi-directional mode, some slave devices may tristate the last few bits to signal a turn-around; therefore, an external weak pull-up may be required on the pin.

## 30.4.2 SPDIN PIN - SERIAL PERIPHERAL DATA IN

In Full Duplex Mode, this is the serial data input from the SPI interface. In half-duplex mode (also referred to as bi-directional mode) this pin is unused.

**Note:** Some slave device may tristate the SPDIN pin during command phase; therefore, an external weak pull-up or pull-down may be required on the pin.

# MEC1618/MEC1618i

## 30.4.3 SPCLK PIN - SERIAL PERIPHERAL CLOCK

This is the serial clock driven by the MEC1618/MEC1618i SPI (master) and connected to all SPI slaves. All data (input and output) is sampled/shifted on SPCLK according to the clock controls CLKPH and CLKPOL (See [TCLKPH](#) and [CLKPOL](#) in [Section 30.11.6, "SPICC - SPI Clock Control Register," on page 511](#)).

**Note 30-1** In the MEC1618/MEC1618i, the General Purpose Serial Peripheral Interface (GP-SPI) pins are 8 mA buffers. The maximum SPCLK pin clock frequency is  $MCLK/2$  for all modes. Limited functionality is available when SPCLK pin clock frequency is  $MCLK$  although performance is not guaranteed. See [TABLE 30-14: on page 513](#) and [Section 30.9.5.5, "Limits of SPI configurations," on page 506](#).

## 30.5 Power, Clocks and Reset

### 30.5.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

### 30.5.2 CLOCKS

This block uses the [EC Bus Clock / LPC Bus Clock](#), [MCLK](#) and the 2MHz ([MCLK\\_DIV10\\_EN](#)). [EC Bus Clock / LPC Bus Clock](#) is used when reading and writing the [General Purpose Serial Peripheral Interface \(GP-SPI\)](#) registers. The 6-bit down counter may use either [MCLK](#) or [MCLK\\_DIV10\\_EN](#) to directly decrement the counter, which is the [SPI\\_CLK](#) source.

See [Section 5.4, "Clock Generator," on page 77](#) for definition and details on clocks of: [MCLK on page 99](#), [MCLK\\_DIV10\\_EN on page 99](#), and [EC Bus Clock on page 99](#).

### 30.6 Reset

This block is reset on a [nSYS\\_RST](#). On reset, the General Purpose SPI interface defaults to disabled. The block can also be reset by software, by setting the Soft Reset bit located in the [SPICR - SPI Control Register](#). Setting this bit reinitializes the SPI Control block back to its [nSYS\\_RST](#) state.

See [Section 5.6, "Reset Interface," on page 104](#) for definition and details on reset: [nSYS\\_RST on page 108](#).

## 30.7 SPI Interrupts / DMA Requests

The [General Purpose Serial Peripheral Interface \(GP-SPI\)](#) can generate an interrupt events to the Embedded Controller (EC) to indicate that the block requires servicing. The SPI TXBE status and RXBF status bits in the [SPISR - SPI Status Register on page 509](#) are routed onto the [SPI\\_TXBE\\_GP](#) & [SPI\\_RXBF\\_GP](#) bits of the [GIRQ14 Source Register on page 324](#). In the Flash GP-SPI instance, these status bits are also connected respectively to the DMA Controller's SPI Flash Write and Read requests signals.

## 30.8 Low Power Mode

This block is designed to conserve power when it is either sleeping or disabled. There are two ways to put the SPI interface into a low power mode: Disabled the SPI Interface via the Enable Bit or Assert the [SLEEP\\_EN](#) signal to the SPI Interface. The following table summarizes the SPI behavior for each of these low power modes.

**TABLE 30-3: BLOCK CLOCK GATING IN LOW POWER MODES**

Enable Bit	<a href="#">SLEEP_EN</a>	Block Idle Status	<a href="#">SPI_CLK_REQ</a>	State	Description
0	X	NOT IDLE	1	PREPARING to SLEEP	The core clock is required until the current transaction is completed and the Block is IDLE.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.

TABLE 30-3: BLOCK CLOCK GATING IN LOW POWER MODES (CONTINUED)

Enable Bit	SLEEP_EN	Block Idle Status	SPI_CLK_REQ	State	Description
1	0	X	1	NORMAL OPERATION	The block is neither disabled by firmware nor commanded to SLEEP.
	1	NOT IDLE	1	PREPARING to SLEEP	The core clock is required until the current transaction is completed and the Block is IDLE.
		IDLE	0	SLEEPING	The block is idle and the core clock can be stopped.

### 30.8.1 DISABLING THE SPI INTERFACE BLOCK VIA THE ENABLE BIT

The enable bit is located in [Section 30.11.1, "SPIAR - SPI Enable Register," on page 507](#). When this bit is cleared the SPI interface is in its lowest power state. The [MCLK](#) clock input is gated and the [SPDOUT](#) and [SPI\\_CLK](#) pins are set to their inactive state as determined by the configuration bits.

**Note:** The SPI Interface is required to finish the current transaction and enter the Idle state before deasserting the [SPI\\_CLK\\_REQ](#) signal and gating its internal clock source.

### 30.8.2 ASSERTING THE SLEEP\_EN SIGNAL TO THE SPI INTERFACE BLOCK

When the [SLEEP\\_EN](#) signal is asserted the SPI interface completes the current transaction and then enters the low power state. In the low power state the [MCLK](#) clock input is gated, the [SPDOUT](#) and [SPI\\_CLK](#) pins are set to their inactive state as determined by the configuration bits, and the [SPI\\_CLK\\_REQ](#) signal is de-asserted.

## 30.9 Operation

The Serial Peripheral Interface (SPI) block is a master SPI block used to communicate with external SPI devices. The SPI master is responsible for generating the SPI clock and is designed to operate in Full Duplex, Half Duplex, and Dual modes of operation. The clock source may be programmed to operate at various clock speeds. The data is transmitted serially via 8-bit transmit and receive shift registers. Communication with SPI peripherals that require transactions of varying lengths can be achieved with multiple 8-bit cycles.

This block has many configuration options: The data may be transmitted and received either MSbit or LSbit first; The SPI Clock Polarity may be either active high or active low; Data may be sampled or presented on either the rising or falling edge of the clock (referred to as the transmit clock phase); and the [SPI\\_CLK SPDOUT](#) frequency may be programmed to a range of values as illustrated in [Table 30-14, "SPI\\_CLK Frequencies," on page 513](#). In addition to these many programmable options, this feature has several status bits that may be enabled to notify the host that data is being transmitted or received.

### 30.9.1 INITIATING AN SPI TRANSACTION

All SPI transactions are initiated by a write to the TX\_DATA register. No read or write operations can be initiated until the Transmit Buffer is Empty, which is indicated by a one in the TXBE status bit.

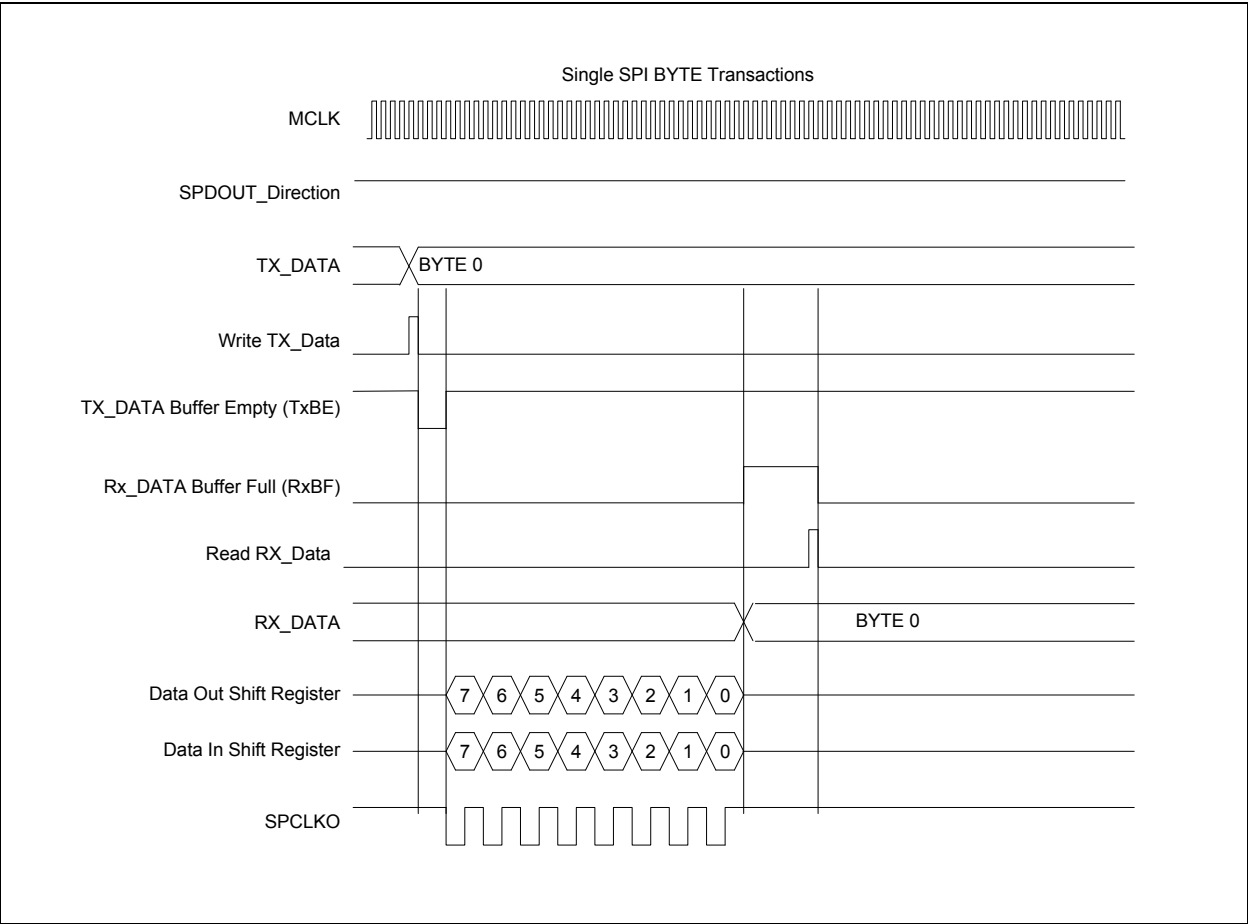
If the transaction is a write operation, the host writes the TX\_DATA register with the value to be transmitted. Writing the TX\_DATA register causes the TXBE status bit to be cleared, indicating that the value has been registered. If empty, the SPI Core loads this TX\_DATA value into an 8-bit transmit shift register and begins shifting the data out. Loading the value into the shift register causes the TXBE status bit to be asserted, indicating to software that the next byte can be written to the TX\_DATA register.

If the transaction is a read operation, the host initiates a write to the TX\_DATA register in the same manner as the write operation. Unlike the transmit command, the host must clear the RXBF status bit by reading the RX\_DATA register before writing the TX\_DATA register. This time, the host will be required to poll the RXBF status bit to determine when the value in the RX\_DATA register is valid.

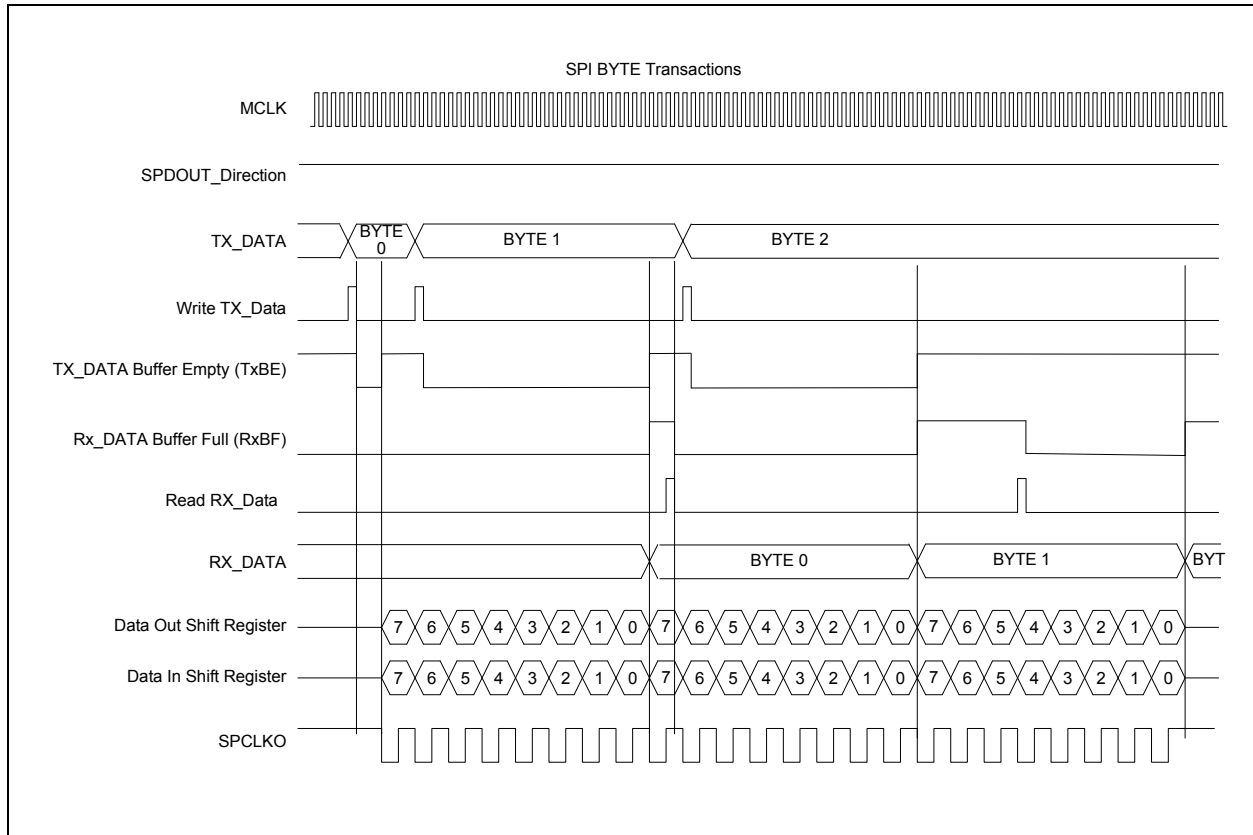
- Note:**
- If the SPI interface is configured for Half Duplex mode, the host must still write a dummy byte to receive data.
  - Since RX and TX transactions are executed by the same sequence of transactions, data is always shifted into the RX\_DATA register. Therefore, every write operation causes data to be latched into the RX\_DATA register and the RXBF bit is set. This status bit should be cleared before initiating subsequent transactions. The host utilizing this SPI core to transmit SPI Data must discard the unwanted receive bytes.
  - The length and order of data sent to and received from a SPI peripheral varies between peripheral devices. The SPI must be properly configured and software-controlled to communicate with each device and determine whether SPIRD data is valid slave data.

The following diagrams show sample single byte and multi-byte SPI Transactions.

**FIGURE 30-3: SINGLE BYTE SPI TX/RX TRANSACTIONS (FULL DUPLEX MODE)**



**FIGURE 30-4: MULTI-BYTE SPI TX/RX TRANSACTIONS (FULL DUPLEX MODE)**



The data may be configured to be transmitted MSB or LSB first. This is configured by the LSBF bit in the [Section 30.11.2, "SPICR - SPI Control Register," on page 508](#). The transmit data is shifted out on the edge as selected by the TCLKPH bit in the SPICC register. See [Section 30.11.6, "SPICC - SPI Clock Control Register," on page 511](#). All received data can be sampled on a rising or falling **SPI\_CLK** edge using RCLKPH (see [RCLKPH](#) in [Section 30.11.6, "SPICC - SPI Clock Control Register," on page 511](#) for clock controls). This clock setting must be identical to the clocking requirements of the current SPI slave.

**Note:** Common peripheral devices require a chip select signal to be asserted during a transaction. Chip selects for SPI devices may be controlled by MEC1618/MEC1618i GPIO pins.

There are three types of transactions that can be implemented for transmitting and receiving the SPI data. They are Full Duplex, Half Duplex, and Dual Mode. These modes are define in [Section 30.9.3, "Types of SPI Transactions," on page 502](#).

## 30.9.2 DMA MODE (FLASH GP-SPI ONLY)

Transmit and receive operations can use a DMA channel. Note that only one DMA channel may be enabled at a time. Setting up the DMA Controller involves specifying the device (Flash GP-SPI), direction (transmit/receive), and the start and end addresses of the DMA buffers in the closely couple memory. Please refer to the DMA Controller chapter for register programming information.

**SPI transmit / DMA write:** the GP-SPI block's transmit empty (**TxBE**) status signal is used as a write request to the DMA controller, which then fetches a byte from the DMA transmit buffer and writes it to the GP-SPI's **SPI TX Data Register (SPITD)**. As content of the latter is transferred to the internal Tx shift register from which data is shifted out onto the SPI bus bit by bit, the Tx Empty signal is again asserted, triggering the DMA fetch-and-write cycle. The process continues until the end of the DMA buffer is reached - the DMA controller stops responding to an active Tx Empty until the buffer's address registers are reprogrammed.

# MEC1618/MEC1618i

SPI receive / DMA read: the [AUTO\\_READ](#) bit in the [SPI Control Register](#) must be set. The driver first writes (dummy data) to the [SPI TX Data Register \(SPITD\)](#) to initiate the toggling of the SPI clock, enabling data to be shifted in. After one byte is received, the Rx Full ([RxBF](#)) status signal, used as a read request to the DMA controller, is asserted. The DMA controller then reads the received byte from the GP-SPI's [SPI RX Data Register \(SPIRD\)](#) and stores it in the DMA receive buffer. With [AUTO\\_READ](#) set, this read clears both the [RxBF](#) and [TxBE](#). Clearing [TxBE](#) causes (dummy) data from the [SPI TX Data Register \(SPITD\)](#) to be transferred to the internal shift register, mimicking the effect of the aforementioned write to the [SPI TX Data Register \(SPITD\)](#) by the driver. SPI clock is toggled again to shift in the second read byte. This process continues until the end of the DMA buffer is reached - the DMA controller stops responding to an active Tx Empty until the buffer's address registers are reprogrammed.

## 30.9.3 TYPES OF SPI TRANSACTIONS

The MEC1618/MEC1618i SPI can be configured to operate in three modes: Full Duplex, Half Duplex, and Dual Mode.

### 30.9.3.1 Full Duplex

In Full Duplex Mode, serial data is transmitted and received simultaneously by the SPI master over the SPDOUT and SPDIN pins. To enable Full Duplex Mode clear [SPDIN Select](#).

When a transaction is completed in the full-duplex mode, the RX\_DATA shift register always contains received data (valid or not) from the last transaction.

### 30.9.3.2 Half Duplex

In Half Duplex Mode, serial data is transmitted and received sequentially over a single data line (referred to as the [SPD-OUT](#) pin). To enable Half Duplex Mode set [SPDIN Select](#) to 01b. The direction of the [SPDOUT](#) signal is determined by the BIOEN bit (See [Section , "BIOEN," on page 508](#)).

- To transmit data in half duplex mode set the BIOEN bit before writing the TX\_DATA register.
- To receive data in half duplex mode clear the BIOEN bit before writing the TX\_DATA register with a dummy byte.

<b>Note:</b>	The Software driver must properly drive the BIOEN bit and store received data depending on the transaction format of the specific slave device.
--------------	---

### 30.9.3.3 Dual Mode of Operation

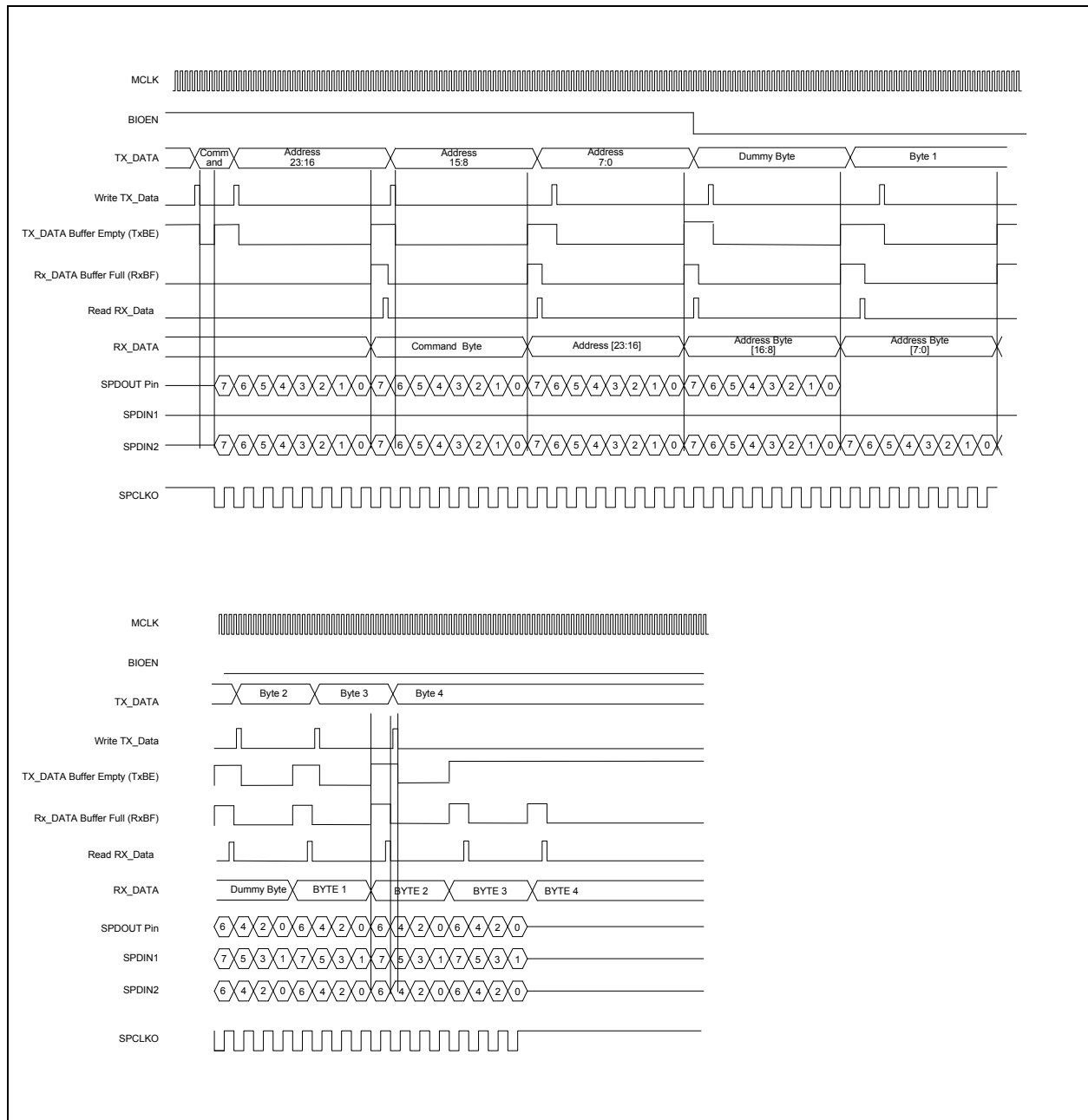
<b>Note:</b>	The Dual Mode of Operation has been implemented to support selected SPI Flash devices that support the Fast Dual Mode command.
--------------	--

In Dual Mode, serial data is transmitted sequentially from the SPDOUT pin and received in by the SPI master from the SPDOUT and SPDIN pins. This essentially doubles the received data rate. To enable Dual Mode of operation the SPI core must be configured to receive data in path on the [SPDIN1](#) and [SPDIN2](#) inputs via [SPDIN Select](#). The BIOEN bit determines if the SPI core is transmitting or receiving. The setting of this bit determines the direction of the [SPDOUT](#) signal. The [SPDIN Select](#) bits are configuration bits that remain static for the duration of a dual read command. The BIOEN bit must be toggled to indicate when the SPI core is transmitting and receiving. For a description of the BIOEN bit see [BIOEN on page 508](#).

- To transmit data in dual mode set the BIOEN bit before writing the TX\_DATA register.
- To receive data in dual mode clear the BIOEN bit before writing the TX\_DATA register with a dummy byte. The even bits (0,2,4,and 6) are received on the SPDOUT pin and the odd bits (1,3,5,and 7) are received on the SPDIN pin. The hardware assembles these received bits into a single byte and loads them into the RX\_DATA register accordingly.

The following diagram illustrates a Dual Fast Read Command that is supported by some SPI Flash devices.

**FIGURE 30-5: DUAL FAST READ FLASH COMMAND**



**Note:** When the SPI core is used for flash commands, like the Dual Read command, the host discards the bytes received during the command, address, and dummy byte portions of the transaction.

## 30.9.4 HOW BIOEN BIT CONTROLS DIRECTION OF SPDOUT BUFFER

When the SPI is configured for **Half Duplex** mode or Dual Mode the **SPDOUT** pin operates as a bi-directional signal. The **BIOEN** bit is used to determine the direction of the **SPDOUT** buffer when a byte is transmitted. Internally, the **BIOEN** bit is sampled to control the direction of the **SPDOUT** buffer when the **TX\_DATA** value is loaded into the transmit shift register. The direction of the buffer is never changed while a byte is being transmitted.

# MEC1618/MEC1618i

---

Since the TX\_DATA register may be written while a byte is being shifted out on the [SPDOUT](#) pin, the BIOEN bit does not directly control the direction of the [SPDOUT](#) buffer. An internal DIRECTION bit, which is a latched version of the BIOEN bit determines the direction of the [SPDOUT](#) buffer. The following list summarizes when the BIOEN bit is sampled.

- The DIRECTION bit is equal to the BIOEN bit when data is not being shifted out (i.e., SPI interface is idle).
- The hardware samples the BIOEN bit when it is shifting out the last bit of a byte to determine if the buffer needs to be turned around for the next byte.
- The BIOEN bit is also sampled any time the value in the TX\_DATA register is loaded into the shift register to be transmitted.

**APPLICATION NOTE:** If a TAR (Turn-around time) is required between transmitting and receiving bytes on the SPDOUT signal, software should allow all the bytes to be transmitted before changing the buffer to an input and then load the TX\_DATA register to begin receiving bytes. This allows the SPI block to operate the same as legacy Microchip SPI devices.

## 30.9.5 CONFIGURING THE CLOCK GENERATOR FOR AN SPI TRANSACTION

The SPI Core generates the [SPI\\_CLK](#) signal to the external SPI device. This clock may be configured for a range of frequencies as illustrated in [Table 30-14, “SPI\\_CLK Frequencies,” on page 513](#). The clock phase and polarity are configurable as well. The following sections define how to program these features.

**USER'S NOTE:** The clock source configuration should not be changed during an SPI transaction.

### 30.9.5.1 Configuring the Frequency of the SPI Clock

The frequency of the [SPI\\_CLK](#) signal is determined by the clock source enabled to the clock generator and the preload value of the clock generator down counter. The clock generator toggles the [SPI\\_CLK](#) output every time the counter underflows, while data is being transmitted. If the preload value is set to 0 the [MCLK](#) clock source bypasses the down counter to directly create the clock generator output.

<b>Note:</b> When the SPI interface is in the idle state and data is not being transmitted, the <a href="#">SPI_CLK</a> signal stops in the inactive state as determined by the configuration bits.
---

The clock source to the down counter is determined by Bit [CLKSRC](#). Either the [MCLK](#) clock or the 2.0 MHz clock enable ([2.0MHz\\_EN](#)) can be used to decrement the down counter in the clock generator logic.

### 30.9.5.2 Configuring the SPI Mode

In practice, there are four modes of operation that define when data should be latched. These four modes are the combinations of the [SPI\\_CLK](#) polarity ([CLKPOL](#)) and phase ([RCLKPH](#) and [TCLKPH](#)). Phase is programmable independently for the receive and transmit phases. [CLKPOL](#), [RCLKPH](#) and [TCLKPH](#) bits are programmable as defined in [Section 30.9.5.3](#) and [Section 30.9.5.4](#) below.



**TABLE 30-4: SPI MODES**

SPI Mode	CLKPOL	CLKPH	Definition	Diagram
0	0	0	data sampled on rising edge of clock	
1	0	1	data sampled on falling edge of clock	
2	1	0	data sampled on falling edge of clock	
3	1	1	data sampled on rising edge of clock	

### 30.9.5.3 Configuring the Polarity of the SPI Clock

The output of the clock generator may be inverted to create an active high or active low clock pulse. This is used to determine the inactive state of the [SPI\\_CLK](#) signal and is used for determining the first edge for shifting the data. The polarity is selected by Bit [CLKPOL](#) in the [SPI Clock Control Register \(SPICC\)](#).

### 30.9.5.4 Configuring the Phase of the SPI Clock

The SPI devices need to know when to sample the data, which may be either on the first edge of the clock or on the second edge of the clock. The phase of the clock is selected independently for receiving data and transmitting data. The receive phase is determine by Bit [RCLKPH](#) and the transmit phase is determine by [TCLKPH](#) in the [SPI Clock Control Register \(SPICC\)](#).

# MEC1618/MEC1618i

## 30.9.5.5 Limits of SPI configurations

The following limits Modes, clock frequency, & board layout (see also [Note 30-1](#)).

Master (MEC1618/MEC1618i)				Slave (Not MEC1618/MEC1618i)	
Max SPICLK Frequency	Dual Mode of Operation	Output Data Transitions	Input Data Sampled	Output Data Transitions	Input Data Sampled
MCLK	Not supported	Pos edge of clk	Pos edge of clk	Pos edge of clk	Either edge of clk
MCLK/2 or less	Supported	All combinations are valid			

## 30.10 Instance Description

There are two instances of [General Purpose Serial Peripheral Interface \(GP-SPI\)](#) block implemented in the MEC1618/MEC1618i.

Each instance of the [General Purpose Serial Peripheral Interface \(GP-SPI\)](#) has its own Logical Device Number, and Base Address as indicated in [Table 30-5](#).

**TABLE 30-5: [General Purpose Serial Peripheral Interface \(GP-SPI\)](#) BASE ADDRESS TABLE**

<a href="#">General Purpose Serial Peripheral Interface (GP-SPI)</a> Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
EC GP-SPI	7h	F0_1C00h
Flash SPI	Fh	FF_3C00h ( <a href="#">Note 30-2</a> )

The [Table 30-6](#) is a register summary for one instance of the [General Purpose Serial Peripheral Interface \(GP-SPI\)](#). Each EC address is indicated as an SPB Offset from its AHB base address.

**Note 30-2** All of the addresses in the [Detailed Register Descriptions](#) below refer to the [EC GP-SPI](#) instance register addresses which are 32-bit aligned and must be divided by four to correctly represent the addressing for the [Flash SPI](#) instance which is accessible to the LPC host.

**TABLE 30-6: General Purpose Serial Peripheral Interface (GP-SPI) REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
SPIAR - SPI Enable Register	00h	3-0	R/W	
SPICR - SPI Control Register	04h	3-0	R/W	
SPISR - SPI Status Register	08h	3-0	R	
SPITD - SPI TX_Data Register	0Ch	3-0	R/W	
SPIRD - SPI RX_Data Register	10h	3-0	R	
SPICC - SPI Clock Control Register	14h	3-0	R/W	
SPICG - SPI Clock Generator Register	18h	3-0	R/W	
SPIAR - SPI Enable Register	00h	3-0	R/W	

## 30.11 Detailed Register Descriptions

**APPLICATION NOTE:** In the SPI registers some configuration bits are assumed to be static, while others may be updated dynamically by software. The BIOEN and ENABLE bits are considered dynamic bits that can be modified by software at anytime, regardless if a transaction is active or not. These values are latched in hardware, so as to not affect the current operation being performed. All other bits are considered static and cannot be changed by Software while an SPI Transaction is in process.

### 30.11.1 SPIAR - SPI ENABLE REGISTER

**TABLE 30-7: SPI ENABLE REGISTER**

HOST ADDRESS	n/a				n/a			HOST SIZE	
EC OFFSET	00h				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R/W	
BIT NAME	Reserved							Enable	

#### ENABLE

0=Disabled. Clocks are gated to conserve power and the SPDOUT and SPI\_CLK signals are set to their inactive state  
1=Enabled. The device is fully operational.

# MEC1618/MEC1618i

## 30.11.2 SPICR - SPI CONTROL REGISTER

TABLE 30-8: SPI CONTROL REGISTER

HOST ADDRESS	n/a			8-bit			HOST SIZE	
EC OFFSET	04h			8-bit			EC SIZE	
POWER	VTR			02h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved	CE	AUTO_READ	Soft Reset	SPDIN Select		BIOEN	LSBF

### LSBF

Least Significant Bit First control.

0= The data is transferred in MSB-first order. (default)

1= The data is transferred in LSB-first order.

### BIOEN

Bidirectional Output Enable control. When the SPI is configured for [Half Duplex](#) mode or Dual Mode the [SPDOUT](#) pin operates as a bi-directional signal. The BIOEN bit is used by the internal DIRECTION bit to control the direction of the [SPDOUT](#) buffers. The direction of the buffer is never changed while a byte is being transmitted.

0=The [SPDOUT\\_Direction](#) signal configures the [SPDOUT](#) signal as an input.

1=The [SPDOUT\\_Direction](#) signal configures the [SPDOUT](#) signal as an output. (default)

**Note:** If the SPIMODE bit is configured for Full Duplex mode the BIOEN bit must be set to '1' to configure the [SPDOUT](#) signal as an output.

**APPLICATION NOTE:** Although the design supports back-to-back transmissions even when the direction of the buffer is changed, it is the software's responsibility to avoid collisions on the [SPDOUT](#) signal. The design has been implemented to support a 0 second (max) turn-around (TAR) time. If TAR greater than zero is required, the software must wait for the transmission in one direction to complete before writing the TX\_DATA register to start sending/receiving in the opposite direction.

### SPDIN SELECT

The SPDIN Select which SPI input signals are enabled when the BIOEN bit is configured as an input.

00= SPDIN1 only //Select this option for Full Duplex (default)

01=SPDIN2 only //Select this option for Half Duplex

1x=SPDIN1 and SPDIN2 //Select this option for Dual Mode

### SOFT RESET

Soft Reset is a self-clearing bit. Writing zero to this bit has no effect. Writing a one to this bit resets the entire SPI Interface, including all counters and registers back to their initial state (i.e., the same as a [nSYS\\_RST](#)).

## AUTO\_READ

When this bit is 1, a read of the SPI RX\_DATA Register will both clear the RXBF status bit, and in addition it will clear the TXBE status bit. Clearing the TXBE status bit will cause the contents of the TX\_DATA register to be copied into the 8-bit transmit shift register, which then begins shifting data out. Because shifting data out is accompanied by clocking the SPI Clock, a read of RX\_DATA will cause the next byte from the SPI device to be shifted in to RX\_DATA. The contents of the TX\_DATA register is not significant, since when reading from RX\_DATA it is just used to generate the SPI clock. When this bit is 0 (default), a read of the SPI RX\_DATA Register will clear the RXBF bit, but not the TXBE bit.

## CE

1= SPCE# output signal is asserted, i.e., driven to logic '0'

0= SPCE# output signal is asserted, i.e., driven to logic '1'

## 30.11.3 SPISR - SPI STATUS REGISTER

**TABLE 30-9: SPI STATUS REGISTER**

HOST ADDRESS	n/a				8-bit			HOST SIZE	
EC OFFSET	08h				8-bit			EC SIZE	
POWER	VTR				01h			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	Reserved					ACTIVE	RxBF	TxBE	

## TXBE

Transmit Data Buffer Empty status. This bit is a read-only bit used to indicate that the Tx\_Data buffer is empty. Writing the Tx\_DATA Buffer clears this bit. This signal may be used to generate an interrupt to the EC, if not masked.

## RXBF

Receive Data Buffer Full status. This bit is a read-only bit used to indicate when the Rx\_Data buffer is full. Reading the Rx\_DATA Buffer clears this bit. This signal may be used to generate an interrupt to the EC, if not masked.

## ACTIVE

The **ACTIVE** bit indicates that a transaction controlled by the **General Purpose Serial Peripheral Interface (GP-SPI)** is in progress. The **ACTIVE** bit is asserted ('1') when the GP-SPI controller transfers data from the **SPITD - SPI TX\_Data Register** into the DATA\_OUT Shift Register.

**ACTIVE** is de-asserted ('0') follow a system reset (**nSYS\_RST**) or when data from the DATA\_IN Shift Register is transferred to the **SPIRD - SPI RX\_Data Register**. Note that data is only transferred from the DATA\_IN Shift Register to the **SPIRD - SPI RX\_Data Register** when **RxBF** is not asserted ('0'); i.e., the **SPIRD - SPI RX\_Data Register** is not overwritten by incoming data when **RxBF** is '1.'

# MEC1618/MEC1618i

## 30.11.4 SPITD - SPI TX\_DATA REGISTER

TABLE 30-10: SPI TX DATA REGISTER (SPITD)

HOST ADDRESS	n/a			8-bit			HOST SIZE	
EC OFFSET	0Ch			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	TX_Data[7:0]							

### TX\_DATA[7:0]

A write to this register with the **TxBE** bit asserted '1' initiates an SPI transaction. If the Transmit Shift Register is empty the byte written to this register will be loaded into the shift register and the **TxBE** flag will be asserted. This indicates that the next byte can be written into the TX\_DATA register. This byte will remain in the TX\_DATA register until the SPI core has finished shifting out the previous byte. After the shift register is empty, the hardware will load the pending byte into the shift register and will again assert the **TxBE** bit.

**Note:** The TX\_DATA register must not be written when the **TxBE** bit is zero. Writing this register may overwrite the transmit data before it is loaded into the shift register.

Reading the TX\_DATA register will return the last value written to this register.

## 30.11.5 SPIRD - SPI RX\_DATA REGISTER

TABLE 30-11: SPI RX DATA REGISTER (SPIRD)

HOST ADDRESS	n/a				8-bit			HOST SIZE	
EC OFFSET	10h				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	-	-	-	-	-	
EC TYPE	R	R	R	R	R	R	R	R	
BIT NAME	RX_Data[7:0]								

## RX\_DATA[7:0]

This register is used to read the value returned by the external SPI device. At the end of a byte transfer the RX\_DATA register contains serial input data (valid or not) from the last transaction and the RXBF bit is set to one. This status bit indicates that the RX\_DATA register has been loaded with a the serial input data. The RX\_DATA register should not be read before the RXBF bit is set.

**Note:** The RX\_DATA register must be read, clearing the RXBF status bit before writing the TX\_DATA register. The data in the receive shift register is only loaded into the RX\_DATA register when this bit is cleared. If a data byte is pending in the receive shift register the value will be loaded immediately into the RX\_DATA register and the RXBF status flag will be asserted. Software should read the RX\_DATA register twice before starting a new transaction to make sure the RX\_DATA buffer and shift register are both empty.

## 30.11.6 SPICC - SPI CLOCK CONTROL REGISTER

**TABLE 30-12: SPI CLOCK CONTROL REGISTER (SPICC)**

HOST ADDRESS	n/a			8-bit			HOST SIZE	
EC OFFSET	14h			8-bit			EC SIZE	
POWER	VTR			02h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R/W	R	R/W	R/W	R/W
BIT NAME	Reserved			CLKSRC	Reserved	CLKPOL	RCLKPH	TCLKPH

**APPLICATION NOTE:** the default [CLKPOL](#), [RCLKPH](#) and [TCLKPH](#) values are appropriate for transactions with typical MODE 0 SPI Flash devices.

### TCLKPH

The TCLKPH bit determines the Transmit Clock Phase, the SPCLK edge on which the master will clock data out.

0= Valid data is clocked out on the [SPDOUT](#) signal prior to the first [SPI\\_CLK](#) edge. The slave device should sample this data on the first and following odd [SPI\\_CLK](#) edges (i.e., sample data on rising edge). (default)

1= Valid data is clocked out on the first [SPI\\_CLK](#) edge on [SPDOUT](#) signal. The slave device should sample this data on the second and following even [SPI\\_CLK](#) edges (i.e., sample data on falling edge).

**Note:** This functionality is independent of the polarity of SPCLK. See [Section 43.10, "Serial Peripheral Interface \(SPI\) Timings," on page 649](#) for timing diagrams.

### RCLKPH

The RCLKPH bit determines the Receive Clock Phase, the [SPI\\_CLK](#) edge on which the master will sample data.

0= Valid data is expected on the SPDIN signal on the first [SPI\\_CLK](#) edge. This data is sampled on the first and following odd [SPI\\_CLK](#) edges (i.e., sample data on rising edge).

1= Valid data on SPDIN signal is expected after the first [SPI\\_CLK](#) edge. This data is sampled on the second and following even [SPI\\_CLK](#) edges (i.e., sample data on falling edge). (default)

**Note:** This functionality is independent of the polarity of [SPI\\_CLK](#). See [Section 43.10, "Serial Peripheral Interface \(SPI\) Timings," on page 649](#) for timing diagrams.

# MEC1618/MEC1618i

## CLKPOL

This bit controls the polarity of the SPI clock.

0= The [SPI\\_CLK](#) is low when the interface is idle and the first clock edge is a rising edge. (default)

1= The [SPI\\_CLK](#) signal is high when the interface is idle and the first clock edge is a falling edge

## CLKSRC

This bit controls the clock source to SPI Clock Generator

0=The clock source to the SPI Clock Generator 6-bit down counter is MCLK (default)

1=The clock source to the SPI Clock Generator 6-bit down counter is 2MHz clock enable

**Note:** The CLKSRC bit should not be changed during a SPI transaction.

## 30.11.7 SPICG - SPI CLOCK GENERATOR REGISTER

**TABLE 30-13: SPI CLOCK GENERATOR REGISTER (SPICG)**

HOST ADDRESS	n/a			8-bit			HOST SIZE	
EC OFFSET	18h			8-bit			EC SIZE	
POWER	VTR			02h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved		Preload[5:0]					

## PRELOAD[5:0]

The SPI Clock Generator Preload value. The [SPI\\_CLK](#) signal is a clock output with a 50% duty cycle. The signal is generated from a 6-bit down counter that toggles the [SPI\\_CLK](#) pin every time it reaches zero and reloads the counter. This counter is decremented at the rate of the input clock enable, which is either MCLK or the 2MHz clock enable.

$$SPCKO\_FREQ = \left( \left( \frac{1}{2} \times \text{CLOCK\_ENABLE\_FREQ} \right) / \text{PRELOAD} \right)$$

The PRELOAD field contains the preload value for the counter that determines the resulting frequency. The following table outlines the ranges of frequencies possible. Notice that a preload value of zero effectively bypasses the counter and maps the [MCLK](#) clock onto the [SPI\\_CLK](#) signal.



TABLE 30-14: [SPI\\_CLK](#) FREQUENCIES

Clock Enable Frequency	Preload	<a href="#">SPI_CLK</a> Frequency	Notes
Don't Care	0	<a href="#">MCLK</a>	Max Frequency ( <a href="#">Note 30-3</a> )
<a href="#">MCLK</a>	1	<a href="#">MCLK/2</a>	
<a href="#">MCLK</a>	2	<a href="#">MCLK/4</a>	Default Frequency
<a href="#">MCLK</a>	3	<a href="#">MCLK/6</a>	
<a href="#">MCLK</a>	63	<a href="#">MCLK/126</a>	
<a href="#">2.0MHz_EN</a>	0	<a href="#">2.0MHz_EN</a>	
<a href="#">2.0MHz_EN</a>	1	<a href="#">2.0MHz_EN/2</a>	
<a href="#">2.0MHz_EN</a>	2	<a href="#">2.0MHz_EN/4</a>	
<a href="#">2.0MHz_EN</a>	3	<a href="#">2.0MHz_EN/6</a>	
<a href="#">2.0MHz_EN</a>	63	<a href="#">2.0MHz_EN/126</a>	Min Frequency

**Note 30-3** When the Preload value is programmed to zero the [MCLK](#) Clock Source is directly mapped to the [SPI\\_CLK](#) signal. Since the [MCLK](#) signal is a 50% duty cycle it may be directly used as an [SPI\\_CLK](#) frequency.

## 30.12 SPI Examples

### 30.12.1 FULL DUPLEX MODE TRANSFER EXAMPLES

#### 30.12.1.1 Read Only

The slave device used in this example is a MAXIM MAX1080 10-bit, 8 channel ADC:

- The SPI block is activated by setting the enable bit in [SPIAR - SPI Enable Register](#)
- The SPI MODE bit is de-asserted '0' to enable the SPI interface in Full Duplex mode.
- The CLKPOL and TCLKPH bits are de-asserted '0', and RCLKPH is asserted '1' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- Assert #CS using a GPIO pin.
- Write a valid command word (as specified by the slave device) to the [SPITD - SPI TX\\_Data Register](#) with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX\_DATA byte is loaded into the shift register and the SPI master reasserts the TXFE bit. After the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- A dummy 8-bit data value (any value) is written to the TX\_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the first transmit bytes:
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit command data to the slave. This particular slave device drives '0' on the SPDIN pin to the master while it is accepting command data. This SPIRD data is ignored.

- After the first SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- The final SPI cycle is initiated when another dummy 8-bit data value (any value) is written to the TX\_DATA register. Note that this value may be another dummy value or it can be a new 8-bit command to be sent to the ADC. The new command will be transmitted while the final data from the last command is received simultaneously. This overlap allows ADC data to be read every 16 SPCLK cycles after the initial 24 clock cycle. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses, the second SPI cycle is complete:
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is the first half of a valid 16 bit ADC value. SPIRD is read and stored.
  - After the second SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- After 8 [SPI\\_CLK](#) pulses, the final SPI cycle is complete, TXBF is asserted '1', and the SPINT interrupt is asserted (if enabled). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is the second half of a valid 16 bit ADC value. SPIRD is read and stored.
- If a command was overlapped with the received data in the final cycle, #CS should remain asserted and the SPI master will initiate another SPI cycle. If no new command was sent, #CS is released and the SPI is idle.

## 30.12.1.2 Read/Write

The slave device used in this example is a Fairchild NS25C640 FM25C640 64K Bit Serial EEPROM. The following subsections describe the read and write sequences.

### READ

- The SPI block is activated by setting the enable bit in [SPIAR - SPI Enable Register](#)
- The SPIMODE bit is de-asserted '0' to enable the SPI interface in Full Duplex mode.
- The CLKPOL, TCLKPH and RCLKPH bits are de-asserted '0' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- Assert CS# low using a GPIO pin.
- Write a valid command word (as specified by the slave device) to the [SPITD - SPI TX\\_Data Register](#) with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX\_DATA byte is loaded into the shift register and the SPI master reasserts the TXFE bit. After the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the [SPI\\_CLK](#) pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, EEPROM address A15-A8 is written to the TX\_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the first transmit byte (Command Byte transmitted):
- The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit command data to the slave. This particular slave device tri-states the SPDIN pin to the master while it is accepting command data. This SPIRD data is ignored.

USER'S NOTE: External pull-up or pull-down is required on the SPDIN pin if it is tri-stated by the slave device.

- After the first SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (EEPROM address A15-A8) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the [SPI\\_CLK](#) pin. Data on the SPDIN pin is also sampled on each clock. Note: The particular slave device ignores address A15-A13.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, EEPROM address A7-A0 is written to the TX\_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the second transmit byte (Address Byte (MSB) transmitted):
  - EEPROM address A15-A8 has been transmitted to the slave completing the second SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - After the second SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (EEPROM address A7-A0) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, a dummy byte is written to the TX\_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses, the third SPI cycle is complete (Address Byte (LSB) transmitted):
  - EEPROM address A7-A0 has been transmitted to the slave completing the third SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - After the third SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (dummy byte) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the SPCLK pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- If only one receive byte is required, the host would not write any more value to the TX\_DATA register until this transaction completes. If more than one byte of data is to be received, another dummy byte would be written to the TX\_DATA register (one dummy byte per receive byte is required). The SPI master automatically clears the TXFE bit when the TX\_DATA register is written, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses, the fourth SPI cycle is complete (First Data Byte received):
  - The dummy byte has been transmitted to the slave completing the fourth SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). Unlike the command and address phases, the data now contained in [SPIRD - SPI RX\\_Data Register](#) is the 8-bit EEPROM data since the last cycle was initiated to receive data from the slave.
  - After the fourth SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (if any) and loads it into the TX shift register. This process will be repeated until all the desired data is received.
- The host software will read and store the EEPROM data value in [SPIRD - SPI RX\\_Data Register](#).
- If no more data needs to be received by the master, CS# is released and the SPI is idle. Otherwise, master continues reading the data by writing a dummy value to the TX\_DATA register after every 8 [SPI\\_CLK](#) cycles.

## WRITE

- The SPI block is activated by setting the enable bit in [SPIAR - SPI Enable Register](#)
- The SPIMODE bit is de-asserted '0' to enable the SPI interface in Full Duplex mode.
- The CLKPOL, TCLKPH and RCLKPH bits are de-asserted '0' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.

# MEC1618/MEC1618i

---

- Assert **WR#** high using a GPIO pin.
- Assert **CS#** low using a GPIO pin.
- Write a valid command word (as specified by the slave device) to the [SPITD - SPI TX\\_Data Register](#) with **TXFE** asserted '1'. The SPI master automatically clears the **TXFE** bit indicating the byte has been put in the TX buffer. If the shift register is empty the **TX\_DATA** byte is loaded into the shift register and the SPI master reasserts the **TXFE** bit. After the data is in the shift register the SPI master begins shifting the data value onto the **SPDOUT** pin and drives the [SPI\\_CLK](#) pin. Data on the **SPDIN** pin is also sampled on each clock.
- After the **TXFE** bit is asserted the SPI Master is ready to receive its next byte. Before writing the next **TX\_DATA** value, software must clear the **RXBF** status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, EEPROM address A15-A8 is written to the **TX\_DATA** register. The SPI master automatically clears the **TXFE** bit, but does not begin shifting the dummy data value onto the **SPDOUT** pin. This byte will remain in the **TX\_DATA** register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the first transmit byte (Command Byte transmitted):
  - The first SPI cycle is complete, **RXBF** bit is asserted '1', and the **SPINT** interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit command data to the slave. This particular slave device tri-states the **SPDIN** pin to the master while it is accepting command data. This [SPIRD](#) data is ignored.

**USER'S NOTE:** External pull-up or pull-down is required on the **SPDIN** pin if it is tri-stated by the slave device.

- After the first SPI cycle is completed, the SPI master takes the pending data in the **TX\_DATA** register (EEPROM address A15-A8) and loads it into the TX shift register. Loading the shift register automatically asserts the **TXFE** bit, begins shifting the dummy data value onto the **SPDOUT** pin, and drives the [SPI\\_CLK](#) pin. Data on the **SPDIN** pin is also sampled on each clock. Note: The particular slave device ignores address A15-A13.
- After the **TXFE** bit is asserted the SPI Master is ready to receive its next byte. Before writing the next **TX\_DATA** value, software must clear the **RXBF** status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, EEPROM address A7-A0 is written to the **TX\_DATA** register. The SPI master automatically clears the **TXFE** bit, but does not begin shifting this data value onto the **SPDOUT** pin. This byte will remain in the **TX\_DATA** register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the second transmit byte (Address Byte (MSB) transmitted):
  - EEPROM address A15-A8 has been transmitted to the slave completing the second SPI cycle. Once again, the **RXBF** bit is asserted '1' and the **SPINT** interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - After the second SPI cycle is completed, the SPI master takes the pending data in the **TX\_DATA** register (EEPROM address A7-A0) and loads it into the TX shift register. Loading the shift register automatically asserts the **TXFE** bit, begins shifting the dummy data value onto the **SPDOUT** pin, and drives the **SPCLK** pin. Data on the **SPDIN** pin is also sampled on each clock.
- After the **TXFE** bit is asserted the SPI Master is ready to receive its next byte. Before writing the next **TX\_DATA** value, software must clear the **RXBF** status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, a data byte (D7:D0) is written to the **TX\_DATA** register. The SPI master automatically clears the **TXFE** bit, but does not begin shifting this data value onto the **SPDOUT** pin. This byte will remain in the **TX\_DATA** register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses, the third SPI cycle is complete (Address Byte (LSB) transmitted):
  - EEPROM address A7-A0 has been transmitted to the slave completing the third SPI cycle. Once again, the **RXBF** bit is asserted '1' and the **SPINT** interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated solely to transmit address data to the slave.
  - After the third SPI cycle is completed, the SPI master takes the pending data in the **TX\_DATA** register (data byte D7:D0) and loads it into the TX shift register. Loading the shift register automatically asserts the **TXFE** bit, begins shifting the dummy data value onto the **SPDOUT** pin, and drives the **SPCLK** pin. Data on the **SPDIN** pin is also sampled on each clock.
- After the **TXFE** bit is asserted the SPI Master is ready to receive its next byte. Before writing the next **TX\_DATA** value, software must clear the **RXBF** status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- If only one data byte is to be written, the host would not write any more values to the **TX\_DATA** register until this

transaction completes. If more than one byte of data is to be written, another data byte would be written to the TX\_DATA register. The SPI master automatically clears the TXFE bit when the TX\_DATA register is written, but does not begin shifting this data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.

- After 8 [SPI\\_CLK](#) pulses, the fourth SPI cycle is complete (First Data Byte transmitted):
  - The data byte has been transmitted to the slave completing the fourth SPI cycle. Once again, the RXBF bit is asserted '1' and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). Like the command and address phases, the data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid since the last cycle was initiated to transmit data to the slave.
  - After the fourth SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (if any) and loads it into the TX shift register. This process will be repeated until all the desired data is transmitted.
- If no more data needs to be transmitted by the master, CS# and WR# are released and the SPI is idle.

## 30.12.2 HALF DUPLEX (BIDIRECTIONAL MODE) TRANSFER EXAMPLE

The slave device used in this example is a National LM74 12 bit (plus sign) temperature sensor.

- The SPI block is activated by setting the enable bit in [SPIAR - SPI Enable Register](#)
- The SPI MODE bit is asserted '1' to enable the SPI interface in Half Duplex mode.
- The CLKPOL, TCLKPH and RCLKPH bits are de-asserted '0' to match the clocking requirements of the slave device.
- The LSBF bit is de-asserted '0' to indicate that the slave expects data in MSB-first order.
- BIOEN is asserted '0' to indicate that the first data in the transaction is to be received from the slave.
- Assert #CS using a GPIO pin.

//Receive 16-bit Temperature Reading

- Write a dummy command byte (as specified by the slave device) to the [SPITD - SPI TX\\_Data Register](#) with TXFE asserted '1'. The SPI master automatically clears the TXFE bit indicating the byte has been put in the TX buffer. If the shift register is empty the TX\_DATA byte is loaded into the shift register and the SPI master reasserts the TXFE bit. After the data is in the shift register the SPI master begins shifting the data value onto the SPDOUT pin and drives the [SPI\\_CLK](#) pin. This data is lost because the output buffer is disabled. Data on the SPDIN pin is sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, another dummy byte is written to the TX\_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the dummy data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the first receive byte
  - The first SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is the first half of the 16 bit word containing the temperature data.
  - After the first SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (dummy byte 2) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the [SPI\\_CLK](#) pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).

//Transmit next reading command

- BIOEN is asserted '1' to indicate that data will now be driven by the master.
- Next, a command byte is written to the TX\_DATA register. This value is the first half of a 16 bit command to be sent to temperature sensor peripheral. The SPI master automatically clears the TXFE bit, but does not begin shifting the command data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty. This data will be transmitted because the output buffer is enabled. Data on the SPDIN pin is sampled on each clock.
- After 8 [SPI\\_CLK](#) pulses from the second receive byte:
  - The second SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled.

(See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is the second half of the 16 bit word containing the temperature data.

- After the first SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (command byte 1) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the [SPI\\_CLK](#) pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Next, the second command byte is written to the TX\_DATA register. The SPI master automatically clears the TXFE bit, but does not begin shifting the command data value onto the SPDOUT pin. This byte will remain in the TX\_DATA register until the TX shift register is empty.
- After 8 [SPI\\_CLK](#) pulses from the first transmit byte:
  - The third SPI cycle is complete, RXBF bit is asserted '1', and the SPINT interrupt is asserted, if enabled. (See [Section 30.7, "SPI Interrupts / DMA Requests"](#)). The data now contained in [SPIRD - SPI RX\\_Data Register](#) is invalid, since this command was used to transmit the first command byte to the SPI slave.
  - After the first SPI cycle is completed, the SPI master takes the pending data in the TX\_DATA register (command byte 2) and loads it into the TX shift register. Loading the shift register automatically asserts the TXFE bit, begins shifting the dummy data value onto the SPDOUT pin, and drives the [SPI\\_CLK](#) pin. Data on the SPDIN pin is also sampled on each clock.
- After the TXFE bit is asserted the SPI Master is ready to transmit or receive its next byte. Before writing the next TX\_DATA value, software must clear the RXBF status bit by reading the [SPIRD - SPI RX\\_Data Register](#).
- Since no more data needs to be transmitted, the host software will wait for the RXBF status bit to be asserted indicating the second command byte was transmitted successfully.
- #CS is de-asserted.



## 31.0 VBAT-POWERED CONTROL INTERFACE

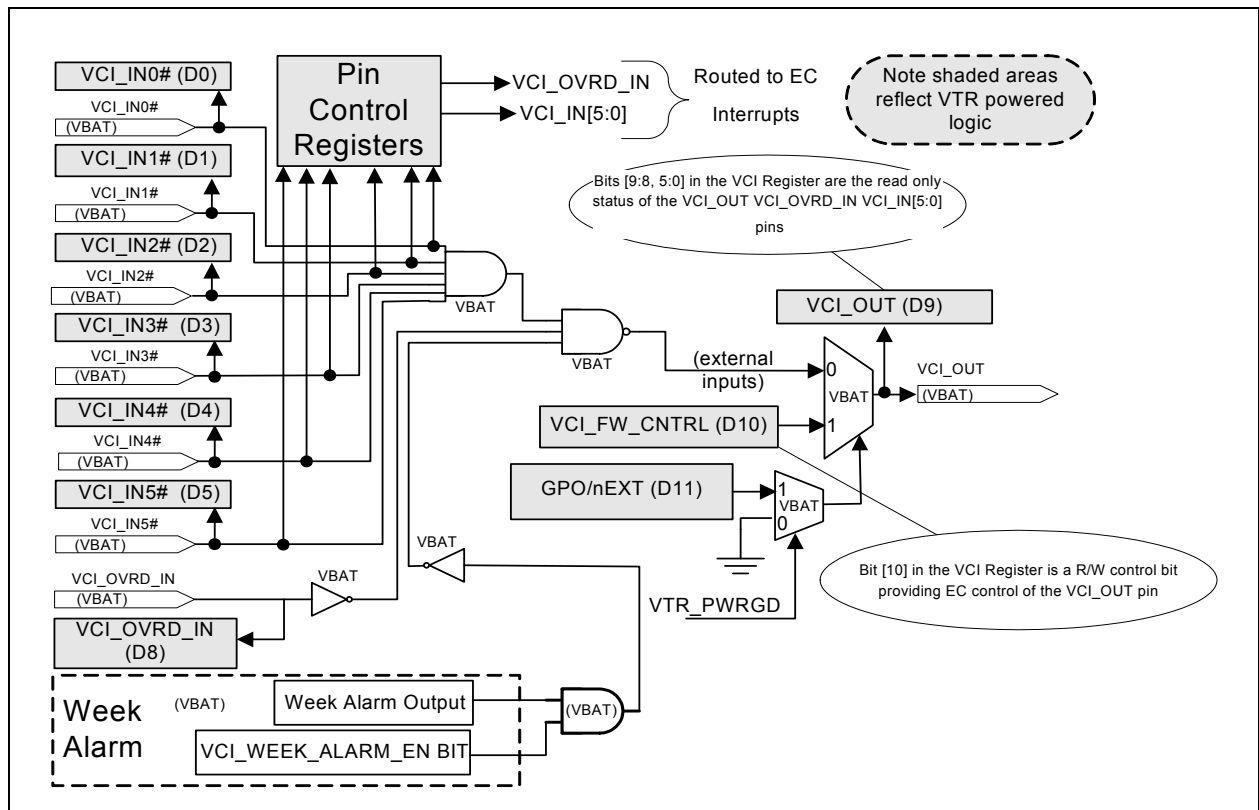
### 31.1 General Description

The [VBAT-Powered Control Interface](#) has VBAT powered combinational logic and input and output signal pins. The [VBAT-Powered Control Interface](#) has one VTR powered register (see [Section 31.10.1, "VCI Register,"](#) on page 524). The [VBAT-Powered Control Interface](#) block interfaces with the VBAT-powered [Week Alarm Interface](#) on page 393 and the VTR-powered [Week Timer Control Register](#) on page 395.

### 31.2 Features

### 31.3 Block Diagram

**FIGURE 31-1: VBAT-POWERED CONTROL INTERFACE BLOCK DIAGRAM**



**Note:** Shaded bits are located in this block in the [VCI Register](#), other logic is in the [Week Alarm Interface](#).

# MEC1618/MEC1618i

## 31.4 Block Diagram Signal List

TABLE 31-1: VBAT-POWERED CONTROL INTERFACE SIGNAL LIST

Signal Name	Direction	Description
VCI_OUT	Output	<a href="#">VBAT-Powered Control Interface</a> Pin Status Bit
VCI_IN[5:0]#	Input	Active low control inputs (see <a href="#">Section 31.9, "Registers," on page 523.</a> )
VCI_OVRD_IN	Input	MUX control in the VBAT control logic ( <a href="#">Note 31-3</a> ). See also the <a href="#">Week Alarm Interface</a> chapter.
Week Alarm Output	Input	Asserted signal enables Week Alarm terminal count to assert <a href="#">VCI_OUT</a> .
VCI_WEEK_ALARM_EN	Input	Asserted signal enables Week Alarm terminal count to assert <a href="#">VCI_OUT</a> .
Interrupts	Output	See <a href="#">Section 31.6, "Interrupts," on page 520.</a>
EC Interface	I/O Bus	Bus used by microprocessor to access the registers in this block.

This block is powered by the VBAT power supply.

## 31.5 Power, Clocks and Reset

### 31.5.1 POWER DOMAIN

This block is in the VTR power domain for EC interaction and uses the VBAT power domain for memory retention.

### 31.5.2 CLOCKS

The [VBAT-Powered Control Interface](#) has one clock input, which is used to interface to the embedded controller accessible register.

### 31.5.3 POWER ON RESET

[VBAT-Powered Control Interface](#) register bits that have default values are reset by hardware on [VBAT\\_POR](#).

## 31.6 Interrupts

The [VBAT-Powered Control Interface](#) generates interrupts and wakeup events for the following pin inputs: [VCI\\_IN\[5:0\]#](#) & [VCI\\_OVRD\\_IN](#). The Interrupts are routed to the [VCI\\_OVRD\\_IN](#), [VCI\\_IN0](#), [VCI\\_IN1](#), [VCI\\_IN2](#), [VCI\\_IN3](#), [VCI\\_IN4](#), and [VCI\\_IN5](#) bits in [GIRQ23 Source Register](#). The [VBAT-Powered Control Interface](#) Interrupt and wake events can only be asserted when VBAT and VTR are both applied; edge detection is disabled when VTR is unpowered. The edge detection for the interrupt and wake events are controlled for each pin by GPIO Pin Control Registers (see [Section 21.9.1, "Pin Control Register," on page 406](#)).

## 31.7 General Description

The MEC1618/MEC1618i [VBAT-Powered Control Interface](#) (VCI) is illustrated in [Figure 31-1](#). This block contains the [VCI Register](#), which monitors the status of the [VCI\\_IN\[5:0\]#](#), [VCI\\_OVRD\\_IN](#) and [VCI\\_OUT](#) pins and also provides firmware control of [VCI\\_OUT](#) when VTR is present.

The state of the [VCI\\_OUT](#) pin can be determined using the [Table 31-2, "VCI Output Truth Table," on page 521](#). Signals in [Table 31-2](#) are described in [Section TABLE 31-1: "VBAT-Powered Control Interface Signal List," on page 520](#) and in [Section 31.9, "Registers," on page 523](#).



**TABLE 31-2: VCI OUTPUT TRUTH TABLE**

Inputs												Output	Description
VCI_OVRD_IN Pin	VCI_IN0# Pin	VCI_IN1# Pin	VCI_IN2# Pin	VCI_IN3# Pin	VCI_IN4# Pin	VCI_IN5# Pin	Week Alarm Output	VCI_WEEK_ALARM_EN	VCI_FW_CNTRL Bit	GPO/nEXT Bit	VTRGD	VCI_OUT Pin	
X	0	X	X	X	X	X	X	X	X	X	0	1	VTR = OFF External inputs can drive VCI_OUT
X	X	0	X	X	X	X	X	X	X	X	0	1	
X	X	X	0	X	X	X	X	X	X	X	0	1	
X	X	X	X	0	X	X	X	X	X	X	0	1	
X	X	X	X	X	0	X	X	X	X	X	0	1	
X	X	X	X	X	X	0	X	X	X	X	0	1	
1	X	X	X	X	X	X	X	X	X	X	0	1	
X	X	X	X	X	X	X	1	1	X	X	0	1	
0	1	1	1	1	1	1	0	0	X	X	0	0	VTR = ON External inputs can drive VCI_OUT (GPO/nEXT = '0')
X	0	X	X	X	X	X	X	X	X	0	1	1	
X	X	0	X	X	X	X	X	X	X	0	1	1	
X	X	X	0	X	X	X	X	X	X	0	1	1	
X	X	X	X	0	X	X	X	X	X	0	1	1	
X	X	X	X	X	0	X	1	1	X	0	1	1	
1	X	X	X	X	X	0	X	X	X	0	1	1	
0	1	1	1	1	1	1	0	0	X	0	1	0	
X	X	X	X	X	X	X	X	X	0	1	1	0	VTR = ON EC drives VCI_OUT (GPO/nEXT = '1')
X	X	X	X	X	X	X	X	X	1	1	1	1	

**Note:** When VTRGD is not asserted '0,' VTR is unpowered and both the EC and the VCI Register are unavailable. When VTRGD is asserted '1,' the VTR well is powered, the VCI Register is powered, accessible to the EC, and contributes to the resultant logic driving the VCI\_OUT pin.

**APPLICATION NOTE:** The VBAT-Powered Control Interface can be used in a system as follows:

1. The initial condition is the VBAT battery is installed causing a VBAT\_POR, no AC power applied and no power button event. Therefore no power is applied to VTR power well. The VCI\_OUT pin is de-asserted, the EC is not running, the Week Alarm has not been initialized or enabled.
2. Either applying AC charger power and asserting the VCI\_OVRD\_IN input or depressing a power button asserting

# MEC1618/MEC1618i

a one or more of the [VCI\\_IN\[5:0\]#](#) input pins causes the [VCI\\_OUT](#) pin to be asserted (maybe a pulse for seconds). This powers up the VTR power well. This starts the EC and allows access to the [VCI Register](#).

3. If the [VCI Register](#) is not written then the [VBAT-Powered Control Interface](#) glue logic behaves the same way as when VTR is empowered; the [VCI\\_OUT](#) pin is asserted when any one of the an external input ([FIGURE 31-1: on page 519](#)) are asserted: the [VCI\\_OVRD\\_IN](#), or [VCI\\_IN\[5:0\]#](#), or the Week Alarm Power-up signal are asserted.
4. The EC can read the status of the [VCI\\_OVRD\\_IN](#) and [VCI\\_IN\[5:0\]#](#) in the [VCI Register](#) (similar to a GP input.) The EC can enable interrupts from these pins.
5. The EC can take programmable control of the [VCI\\_OUT](#) pin output by setting both the [VCI\\_FW\\_CNTRL](#) and the [GPO/nEXT](#) bits in the [VCI Register](#).

**Note:** BIOS should set [VCI\\_FW\\_CNTRL](#) bit to 1 prior to setting the [GPO/nEXT](#) bit to 1 to guarantee a glitch free [VCI\\_OUT](#) pin output.

6. Clearing the [VCI\\_FW\\_CNTRL](#) bit and setting the [GPO/nEXT](#) bit in the [VCI Register](#) causes the [VCI\\_OUT](#) pin to be de-asserted. The [VCI\\_OUT](#) pin remains de-asserted until VTR is empowered. When VTR=0, the [VCI\\_OVRD\\_IN](#) and [VCI\\_IN\[5:0\]#](#) inputs pins control the [VCI\\_OUT](#) pin.

**Note:** The [VCI\\_IN\[5:0\]#](#) pins have no direct effect on the [VCI\\_OUT](#) pin when [VCI\\_OVRD\\_IN](#) is asserted. However, when VTR is on and EC is running, the [VCI\\_IN\[5:0\]#](#) pins can cause an interrupt and the EC can de-assert the [VCI\\_OUT](#) pin by programming the [VCI\\_FW\\_CNTRL](#) and the [GPO/nEXT](#) bits in the [VCI Register](#). The [VCI\\_OVRD\\_IN](#) and the [VCI\\_IN\[5:0\]#](#) pins can also cause EC wake events.

The state of the [VCI\\_OUT](#) pin can be determined using the [Table 31-2, "VCI Output Truth Table," on page 521](#). Signals in [Table 31-2](#) are described in [Table 31-1, "VBAT-Powered Control Interface Signal List," on page 520](#) and in [Section 31.9, "Registers," on page 523](#).

**Note:** When [VTRGD](#) is not asserted '0,' VTR is unpowered and both the EC and the [VCI Register](#) are unavailable. When [VTRGD](#) is asserted '1,' the VTR well is powered, the [VCI Register](#) is powered, accessible to the EC, and contributes to the resultant logic driving the [VCI\\_OUT](#) pin.

## VCI pins:

- There is a latching option on all [VCI\\_IN\[5:0\]#](#) inputs. The latches are set to '1' on VBAT POR and when the corresponding bit in the [Latch Enable Register](#) is '1.' They are reset to '0' when the corresponding [VCI\\_IN#](#) pin is '0.'
- The [VCI\\_IN\[3:0\]#](#) pins are unlatched on power-on; the [VCI\\_IN\[5:4\]#](#) pins come up by default with the latches enabled (see [Section 31.10.2, "Latch Enable Register," on page 525](#)).
- Each of the [VCI\\_IN\[5:0\]#](#) pins has an optional 140 ns analog glitch filter. See also the Filters Bypass bit 12 in the [VCI Register](#).
- When the VTR power rail is fully powered, the MEC1618/MEC1618i GPIO functions in the [VBAT-Powered Control Interface](#) are full function GPIOs. When the VTR rail is not powered, The GPIO output drivers are tri-stated. Note that power for all [VBAT-Powered Control Interface](#) output drivers is sourced from the VBAT power pin.

## 31.8 Input Filtering

Input Filtering suppresses spikes on the [VBAT-Powered Control Interface](#) inputs as defined in [Table 31-3](#). Input Filtering is controlled by the [Filters Bypass](#) bit in the [VCI Register](#).

**TABLE 31-3:** [VBAT-Powered Control Interface Input Filtering](#)

Name	Description	Min	Typ	Max	Unit
t <sub>SP</sub>	Pulse width of spikes suppressed by Input Filtering.	50	—	140	ns

### 31.9 Registers

The [VBAT-Powered Control Interface](#) has its own Logical Device Number, and Base Address as indicated in [Table 31-4](#). [Table 31-5](#) is a register summary for the [VBAT-Powered Control Interface](#) block.

Each instance of the [VBAT-Powered Control Interface](#) Base Address as indicated in [Table 31-4](#).

**TABLE 31-4: VBAT-Powered Control Interface BASE ADDRESS TABLE**

<a href="#">VBAT-Powered Control Interface Instance</a>	<b>LDN from</b> ( <a href="#">Table 3-3 on page 52</a> )	<b>AHB Base Address</b>
<a href="#">VBAT-Powered Control Interface</a>	34h	F0_D000h

[Table 31-5](#) is a register summary for the [VBAT-Powered Control Interface](#) block. Each EC address is indicated as an SPB Offset from its AHB base address.

**TABLE 31-5: VBAT-POWERED CONTROL INTERFACE REGISTER SUMMARY**

<b>Register Name</b>	<b>EC Offset</b>
<a href="#">VCI Register</a>	00h
<a href="#">Latch Enable Register</a>	04h
<a href="#">Latch Resets Register</a>	08h
<a href="#">VCI Input Enable Register</a>	0Ch

### 31.10 Detailed Description of Accessible Registers

**TABLE 31-6: REGISTER BIT ACCESS TYPES**

<b>Register Bit Type</b>	<b>Description</b>
R	<b>Read:</b> A register or bit with this attribute can be read.
W	<b>Write:</b> A register or bit with this attribute can be written.
RO	<b>Read Only:</b> A register or bit with this attribute is read only, writes have no effect.
RS	<b>Read to Set:</b> A register or bit with this attribute is set on read.
RC	<b>Read to Clear:</b> A register or bit with this attribute is cleared after the read, writes have no effect.
WO	<b>Write Only:</b> A register or bit with this attribute is write only, reads return zero.
WC	<b>Write One to Clear:</b> Writing a one to a bit with this attribute clears ('0') the value, writing a zero has no effect.
WS	<b>Write One to Set:</b> Writing a one to a bit with this attribute sets the value to '1', writing a zero has no effect.

# MEC1618/MEC1618i

**TABLE 31-6: REGISTER BIT ACCESS TYPES (CONTINUED)**

Register Bit Type	Description
WZS	<b>Write Zero to Set:</b> Writing a zero to a bit with this attribute sets the value to '1', writing a one has no effect.
RES	<b>Reserved:</b> Reads of a register or bit with this attribute return zero, writes are ignored.

## 31.10.1 VCI REGISTER

Offset	00h			
Bits	Description	Type	Default	Reset Event
31:13	Reserved	RES	0	—
12	Filters Bypass The Filters Bypass bit is used to enable and disable the input filters on the VCI_IN[5:0]# pins. When the Filters Bypass bit is '0,' the input filters are enabled (default). When the Filters Bypass bit is '1,' the input filters are disabled. See also <a href="#">Section 31.8, "Input Filtering,"</a> on page 522.	RW	0	VBAT_POR
11	GPO/nEXT The GPO/nEXT bit controls selecting between the external VBAT-Powered Control Interface inputs, or the VCI_FW_CNTRL bit output to control the VCI_OUT pin. When GPO/nEXT is set ('1'), VCI_OUT follows the VCI_FW_CNTRL bit setting. When GPO/nEXT is clear ('0'), VCI_OUT follows the external inputs.	RW	0	AVTRGD
10	VCI_FW_CNTRL This bit can allow EC firmware to control the state of the VCI_OUT pin. For example, when VTR_PWRGD is asserted and the GPO/nEXT bit is '1', clearing the VCI_FW_CNTRL bit de-asserts the active high VCI_OUT pin. APPLICATION NOTE: BIOS should set the VCI_FW_CNTRL bit to 1 prior to setting the GPO/nEXT bit to 1 on power up.	RW	0	
9	VCI_OUT This bit provides the current status of the VCI_OUT pin.	R	Note 31-1	—
8	VCI_OVRD_IN This bit provides the current status of the VCI_OVRD_IN pin.	R		
7:6	Reserved	RES	0	
5:0	VCI_IN[5:0]# These bits provide the latched state of the associated VCI_IN# pin, if latching is enabled ( <a href="#">Note 31-2</a> ); or, the current state of the pin if latching is not enabled.	R	Note 31-1	

**Note 31-1** The VCI\_IN[5:0]#, VCI\_OVRD\_IN, and VCI\_OUT pin bits default to the state of their respective input pins.

**Note 31-2** Latched bits are cleared by [VBAT\\_POR](#).

## 31.10.2 LATCH ENABLE REGISTER

Offset	04h			
Bits	Description	Type	Default	Reset Event
31:6	Reserved	RES	0	–
5:0	Latching Enables (LE) When a Latching Enable bit is asserted ('1'), the latching function for the corresponding VCI_IN# pin is enabled (default for VCI_IN#[5:4]). When a LE bit is not asserted ('0'), latching for the corresponding VCI_IN# pin is not enabled (default for VCI_IN#[3:0]).	RW	30h	VBAT_POR

## 31.10.3 LATCH RESETS REGISTER

Offset	08h			
Bits	Description	Type	Default	Reset Event
31:6	Reserved	RES	0	–
5:0	Latch Resets (LS) When a Latch Resets bit is asserted ('1'), the corresponding VCI_IN# latch is de-asserted ('1'). <b>Note:</b> reads of the <a href="#">Latch Resets Register</a> are undefined.	WS	–	–

## 31.10.4 VCI INPUT ENABLE REGISTER

Offset	0Ch			
Bits	Description	Type	Default	Reset Event
31:6	Reserved	RES	0	–
5:0	Input Enables (IE) When an Input Enables bit is asserted ('1') (default for VCI_IN#[3:0]#) the corresponding input function is enabled.  When an Input Enables bit is not asserted ('0') (default for VCI_IN#[5:4]#) the corresponding VCI_IN# function is not enabled ( <a href="#">Note 31-3</a> ).  If a VCI input is not enabled, it does not affect the input status or the VCI_OUT pin, even if the input is '0.'  Latches are not asserted, even if VCI_IN# pin is low, during a VBAT power transition.	RW	Fh	VBAT POR

**Note 31-3** The VCI\_OVRD\_IN pin is not affected by the [VCI Input Enable Register](#) and cannot be disabled.

# MEC1618/MEC1618i

## 32.0 VBAT POWERED RAM

### 32.1 Abstract

#### 32.1.1 CHIP LEVEL INTERFACE CLOCK DOMAIN/POWER DOMAIN CROSSINGS

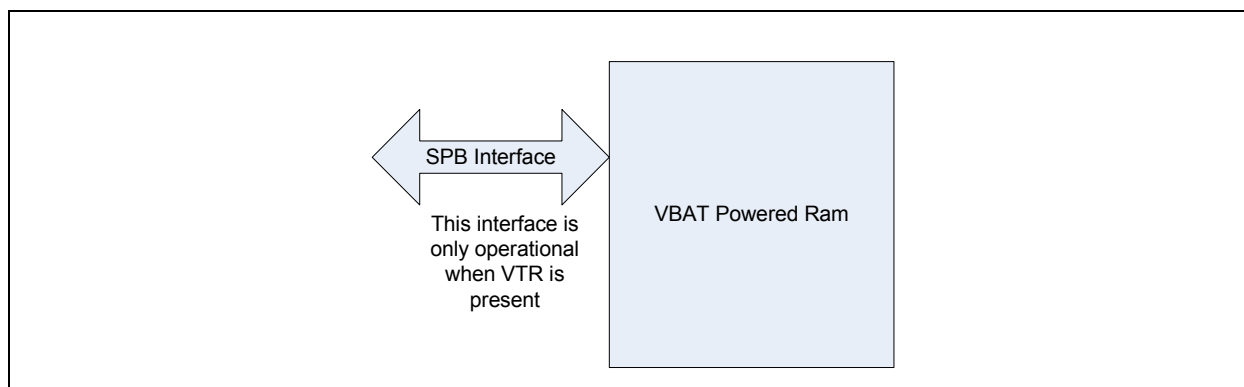
Powered by VBAT but only clocked or accessed when VTR power is available.

### 32.2 General Description

The VBAT Powered RAM provides a 64 Byte Random Accessed Memory that is operational while VTR is powered, and will retain its values while powered by VBAT powered and VTR is unpowered. The RAM is organized as a 16 “words” x 32-bit wide for a total of 64 bytes.

### 32.3 Block Diagram

**FIGURE 32-1: BLOCK DIAGRAM OF VBAT POWERED RAM**



### 32.4 Power, Clocks and Reset

#### 32.4.1 POWER DOMAIN

This block is in the VTR power domain for EC interaction and uses the VBAT power domain for memory retention. See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

#### 32.4.2 CLOCKS

The [VBAT Powered RAM](#) has one clock input., the [EC Bus Clock](#). See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

#### 32.4.3 POWER ON RESET

The [VBAT Powered RAM](#) is reset on a [VBAT\\_POR](#). See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

### 32.5 Interrupts

The [VBAT Powered RAM](#) has not interrupts.

### 32.6 Registers

The [VBAT Powered RAM](#) has it's own Logical Device Number, and Base Address as indicated in [Table 32-1](#). See [Note 3-1 on page 52](#).

**TABLE 32-1: VBAT Powered RAM BASE ADDRESS TABLE**

VBAT Powered RAM Blocks	LDN from (Table 3-2 on page 51)	AHB Base Address
VBAT Backed Memory	33h	F0_CD00h

Each 32-bit RAM location is an SPB Offset from the AHB base address.

## 33.0 BLINKING/BREATHING PWM

### 33.1 Introduction

LEDs are used in computer applications to communicate internal state information to a user through a minimal interface. Typical applications will cause an LED to blink at different rates to convey different state information. For example, an LED could be full on, full off, blinking at a rate of once a second, or blinking at a rate of once every four seconds, in order to communicate four different states.

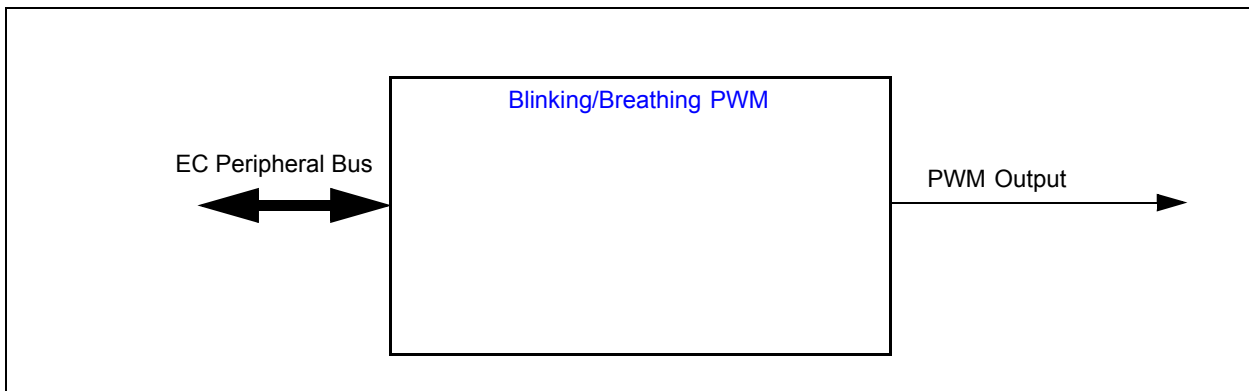
As an alternative to blinking, an LED can “breathe”, that is, oscillate between a bright state and a dim state in a continuous, or apparently continuous manner. The rate of breathing, or the level of brightness at the extremes of the oscillation period, can be used to convey state information to the user that may be more informative, or at least more novel, than traditional blinking.

The blinking/breathing hardware is implemented using a PWM. The PWM can be driven either by the [20.27 MHz clock](#) or by a [32.768 kHz clock](#) input. When driven by the [20.27 MHz clock](#), the PWM can be used as a standard 8-bit PWM in order to control a fan. When used to drive blinking or breathing LEDs, the [32.768 kHz clock](#) source is used.

### 33.2 Interface

This block is designed to drive a pin on the pin interface and to be accessed internally via a registered host interface.

**FIGURE 33-1: BLOCK DIAGRAM**



#### 33.2.1 SIGNAL DESCRIPTION

**TABLE 33-1: SIGNAL DESCRIPTION TABLE**

Name	Direction	Description
PWM Output	Output	Output of PWM
EC Peripheral Bus	Input/Output	EC Peripheral Bus
Sync_OUT	Output	Synchronize multiple instances utilizing the <a href="#">Synchronize (SYNC)</a> bit in the <a href="#">LED CONFIGURATION (LED_CFG)</a> register.
Sync_IN	Input	Synchronize multiple instances utilizing the <a href="#">Synchronize (SYNC)</a> bit in the <a href="#">LED CONFIGURATION (LED_CFG)</a> register.

#### 33.2.2 HOST INTERFACE

The blinking/breathing PWM block is accessed by a controller over the standard register interface.



## 33.3 Power, Clocks and Reset

### 33.3.1 POWER DOMAINS

**TABLE 33-2: POWER SOURCES**

Name	Description
VTR	Main power. The source of main power for the device is system dependent.

### 33.3.2 CLOCK INPUTS

**TABLE 33-3: CLOCK INPUTS**

Name	Description
MCLK_5HZ_EN	5 Hz clock
X32K_CLK	32.768 kHz clock
MCLK	20.27 MHz clock

### 33.3.3 RESETS

**TABLE 33-4: RESET SIGNALS**

Name	Description
nSYS_RST	System reset

### 33.3.4 POWER MANAGEMENT

In all cases, X32K\_CLK does not affect [Power Management](#); i.e., the [Blinking/Breathing PWM](#) operates normally when MCLK is stopped, except as a [General Purpose PWM](#). The sleep enable input has no affect on the [Blinking/Breathing PWM](#) and the clock required outputs are only asserted during register read/write cycles for as long as necessary to propagate updates to the block core ([Table 33-5](#)).

See also [Section 33.5, "Low Power Mode,"](#) on page 530.

# MEC1618/MEC1618i

TABLE 33-5: BLINKING/BREATHING PWM POWER MANAGEMENT

SLEEP_EN	PWM Mode?	Bus Access Cycle?	CLK_REQ	Description
X	No	Yes	1	CLK_REQ is only asserted for as long as necessary to propagate updates to the block core
		No	0	CLK_REQ is not asserted when the EC is not accessing the register interface. (Note that this block <i>cannot</i> prevent the chip from entering the system deepest sleep states.)
	Yes	X	1	In PWM Mode, CLK_REQ is asserted whenever the PWM is enabled.

## 33.4 Interrupts

Each PWM can generate an interrupt. The interrupt is asserted for one [20.27 MHz clock](#) period whenever the PWM WDT times out. The PWM WDT is described in [Section 33.6.3.1, "PWM WDT," on page 534](#).

**Note:** [LED0](#), [LED1](#), and [LED2](#) in the [GIRQ15 Source Register](#) are the interrupt source bits for the three instances of the [Blinking/Breathing PWM](#) in the MEC1618/MEC1618i (see [Section 15.0, "EC Interrupt Aggregator," on page 296](#)).

TABLE 33-6: INTERRUPTS

Source	Description
PWM_WDT	PWM watchdog time out

## 33.5 Low Power Mode

The [Blinking/Breathing PWM](#) may be put into a [Low Power Mode](#) by the chip-level power, clocks, and reset (PCR) circuitry ([Table 33-7](#)). [Low Power Mode](#) is only prevented in the [General Purpose PWM](#) mode. When the [32.768 kHz clock](#) is selected, the [Blinking/Breathing PWM](#) function continues to operate, even when the [20.27 MHz clock](#) is stopped.

**Note:** The LED[2:0] bits in [EC Blocks Sleep Enables Register 2](#) and [EC Blocks Clock Required Status Register 2](#) are the sleep enable and clock required status bits for the three instances of the [Blinking/Breathing PWM](#) in the MEC1618/MEC1618i (see [Section 5.0, "Power, Clocks, and Resets," on page 74](#)).

TABLE 33-7: LOW POWER MODE BEHAVIOR

Clock Source ( <a href="#">Note 33-2</a> )	Control ( <a href="#">Note 33-1</a> )	Mode	Low Power Mode	Description
X	'00'b	PWM 'OFF'	Yes	<a href="#">32.768 kHz clock</a> is required.
X	'01'b	<a href="#">Breathing</a>	Yes	
1	'10'b	<a href="#">General Purpose PWM</a>	No	<a href="#">20.27 MHz clock</a> is required, even when a sleep command to the block is asserted.

**TABLE 33-7: LOW POWER MODE BEHAVIOR (CONTINUED)**

Clock Source (Note 33-2)	Control (Note 33-1)	Mode	Low Power Mode	Description
0	'10'b	Blinking	Yes	32.768 kHz clock is required.
X	'11'b	PWM 'ON'	Yes	

**Note 33-1** The CONTROL field is Bits[1:0] in the [LED CONFIGURATION \(LED\\_CFG\)](#) register.

**Note 33-2** CLOCK SOURCE is Bit2 in the [LED CONFIGURATION \(LED\\_CFG\)](#) register.

## 33.6 Description

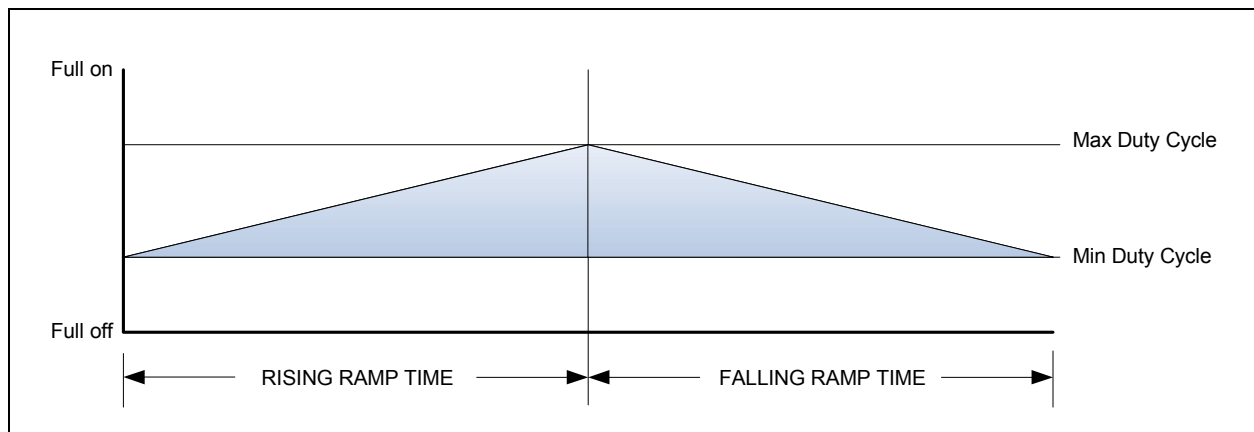
### 33.6.1 BREATHING

If an LED blinks rapidly enough, the eye will interpret the light as reduced brightness, rather than a blinking pattern. Therefore, if the blinking period is short enough, modifying the duty cycle will set the apparent brightness, rather than a blinking rate. At a blinking rate of 128Hz or greater, almost all people will perceive a continuous light source rather than an intermittent pattern.

Because making an LED appear to breathe is an esthetic effect, the breathing mechanism must be adjustable or customers may find the breathing effect unattractive. There are several variables that can affect breathing appearance, as described below.

The following figure illustrates some of the variables in breathing:

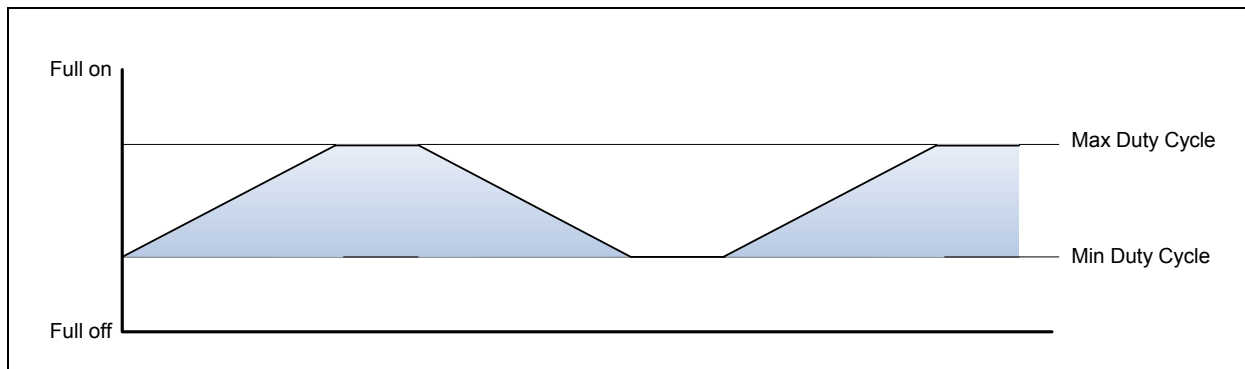
**FIGURE 33-2: BREATHING LED EXAMPLE**



The breathing range of and LED can range between full on and full off, or in a range that falls within the full-on/full-off range, as shown in this figure. The ramp time can be different in different applications. For example, if the ramp time was 1 second, the LED would appear to breathe quickly. A time of 2 seconds would make the LED appear to breathe more leisurely.

The breathing pattern can be clipped, as shown in the following figure, so that the breathing effect appears to pause at its maximum and minimum brightnesses:

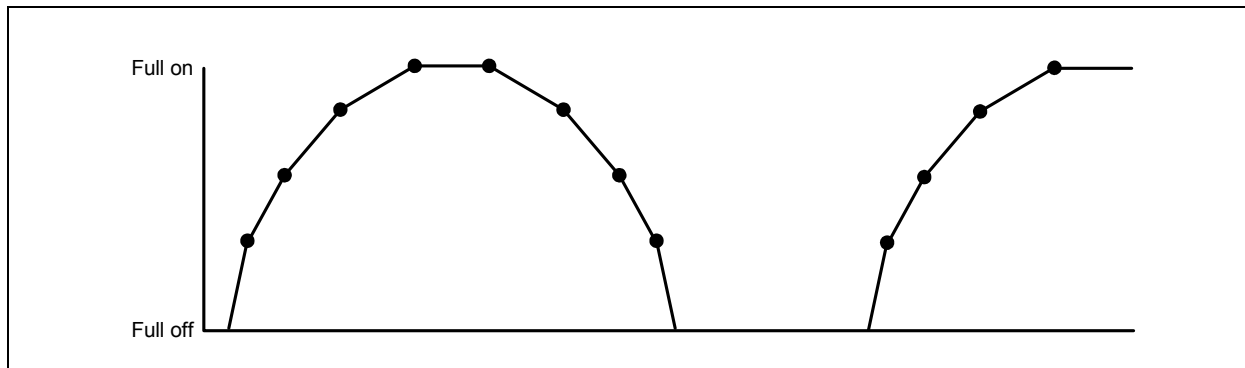
**FIGURE 33-3: CLIPPING EXAMPLE**



The clipping periods at the two extremes can be adjusted independently, so that for example an LED can appear to breathe (with a short delay at maximum brightness) followed by a longer “resting” period (with a long delay at minimum brightness).

The brightness can also be changed in a non-linear fashion, as shown in the following figure:

**FIGURE 33-4: EXAMPLE OF A SEGMENTED CURVE**



In this figure, the rise and fall curves are implemented in 4 linear segments and are the rise and fall periods are symmetric.

The breathing mode uses the [32.768 kHz clock](#) for its time base.

## 33.6.2 BLINKING

When configured for blinking, a subset of the hardware used in breathing is used to implement the blinking function. The PWM (an 8-bit accumulator plus an 8-bit duty cycle register) drives the LED directly. The Duty Cycle register is programmed directly by the user, and not modified further. The PWM accumulator is configured as a simple 8-bit up counter. The counter uses the [32.768 kHz clock](#), and is pre-scaled by the Delay counter, to slow the PWM down from the 128Hz provided by directly running the PWM on the [32.768 kHz clock](#).

With the pre-scaler, the blink rate of the LED could be as fast as 128Hz (which, because it is blinking faster than the eye can distinguish, would appear as a continuous level) to 0.03125Hz (that is, with a period of 7.8msec to 32 seconds). Any duty cycle from 0% (0h) to 100% (FFh) can be configured, with an 8-bit precision. An LED with a duty cycle value of 0h will be fully off, while an LED with a duty cycle value of FFh will be fully on.

In Blinking mode the PWM counter is always in 8-bit mode.

Table 33-8, "LED Blink Configuration Examples" shows some example blinking configurations:

**TABLE 33-8: LED BLINK CONFIGURATION EXAMPLES**

Prescale	Duty Cycle	Blink Frequency	Blink	
			'ON' Time	'OFF' Time
000h	00h	128 Hz	–	FULL 'OFF'
000h	FFh	128 Hz	FULL 'ON'	–
001h	40h	64 Hz	3.9 msec	11.7 msec
003h	80h	32 Hz	15.6 msec	15.6 msec
07Fh	20h	1 Hz	125 msec	875 msec
0BFh	16h	0.66 Hz	129 msec	1.371 sec
0FFh	10h	0.5 Hz	125 msec	1.875 sec
180h	0Bh	0.33 Hz	129 msec	2.878 sec
1FFh	40h	0.25 Hz	1 sec	3 sec

The Blinking and General Purpose PWM modes share the hardware used in the breathing mode. The Prescale value is derived from the LD field of the LED\_DELAY register and the Duty Cycle is derived from the MIN field of the LED LIMITS register.

**TABLE 33-9: BLINKING MODE CALCULATIONS**

Parameter	Unit	Equation (Note 33-3)
FREQUENCY	Hz	$X32K\_CLK / (PRESCALE + 1) / 256$
PERIOD	Seconds	$1 / FREQUENCY$
'H' Width	Seconds	$PERIOD \times (DUTY\_CYCLE / 256)$
'L' Width	Seconds	$PERIOD \times (1 - DUTY\_CYCLE / 256)$

**Note 33-3** See also Table 33-8 for programming examples.

# MEC1618/MEC1618i

## 33.6.3 GENERAL PURPOSE PWM

When used in the Blinking configuration with the [MCLK](#), the LED module can be used as a general-purpose programmable Pulse-Width Modulator with an 8-bit programmable pulse width. It can be used for fan speed control, sound volume, etc. With the [MCLK](#) source, the PWM frequency can be configured in the range shown in [Table 33-10](#).

**TABLE 33-10: PWM CONFIGURATION EXAMPLES**

Prescale	PWM Frequency
000h	<a href="#">MCLK</a> /256
001h	<a href="#">MCLK</a> /256/2
003h	<a href="#">MCLK</a> /256/4
006h	<a href="#">MCLK</a> /256/7
00Bh	<a href="#">MCLK</a> /256/12
07Fh	<a href="#">MCLK</a> /256/128
1FFh	<a href="#">MCLK</a> /256/512
FFFh	<a href="#">MCLK</a> /256/4096

**TABLE 33-11: GENERAL PURPOSE PWM MODE CALCULATIONS**

Parameter	Unit	Equation ( <a href="#">Note 33-4</a> )
FREQUENCY	Hz	<a href="#">MCLK</a> /(PRESCALE + 1)/256
PERIOD	Seconds	1/FREQUENCY
'H' Width	Seconds	PERIOD x (DUTY CYCLE/256)
'L' Width	Seconds	PERIOD x (1 - DUTY CYCLE/256)

**Note 33-4** When the Duty Cycle is 00h, the [General Purpose PWM](#) is fully 'OFF'; when the Duty Cycle is FFh, the [General Purpose PWM](#) is fully 'ON'.

### 33.6.3.1 PWM WDT

When the PWM is configured as a general-purpose PWM (in the Blinking configuration with the [20.27 MHz clock](#)), the PWM includes a Watch Dog Timer (WDT). The WDT consists of an internal 8-bit counter and an 8-bit reload value (the field WDTLD in [LED CONFIGURATION \(LED\\_CFG\)](#) register). The internal counter is loaded with the reset value of WDTLD (14h, or 4 seconds) on system [nSYS\\_RST](#) and loaded with the contents of WDTLD whenever either the [LED CONFIGURATION \(LED\\_CFG\)](#) register is written or the MIN byte in the [LED LIMITS \(LED\\_LIMIT\)](#) register is written (the MIN byte controls the duty cycle of the PWM).

Whenever the internal counter is non-zero, it is decremented by 1 for every tick of the [5 Hz clock](#). If the counter decrements from 1 to 0, a WDT Terminal Count causes an interrupt to be generated and reset sets the [Control \(CNTRL\)](#) bit in the [LED CONFIGURATION \(LED\\_CFG\)](#) to 3h, which forces the PWM to be full on. No other PWM registers or fields are affected.

If the [5 Hz clock](#) halts, the watchdog timer stops decrementing but retains its value, provided the device continues to be powered. When the [5 Hz clock](#) restarts, the watchdog counter will continue decrementing where it left off.

Setting the WDTLD bits to 0 disables the PWM WDT. Other sample values for WDTLD are:

01h = 200 ms

02h = 400 ms

03h = 600 ms  
 04h = 800 ms  
 ...  
 14h = 4seconds  
 FFh = 51 seconds

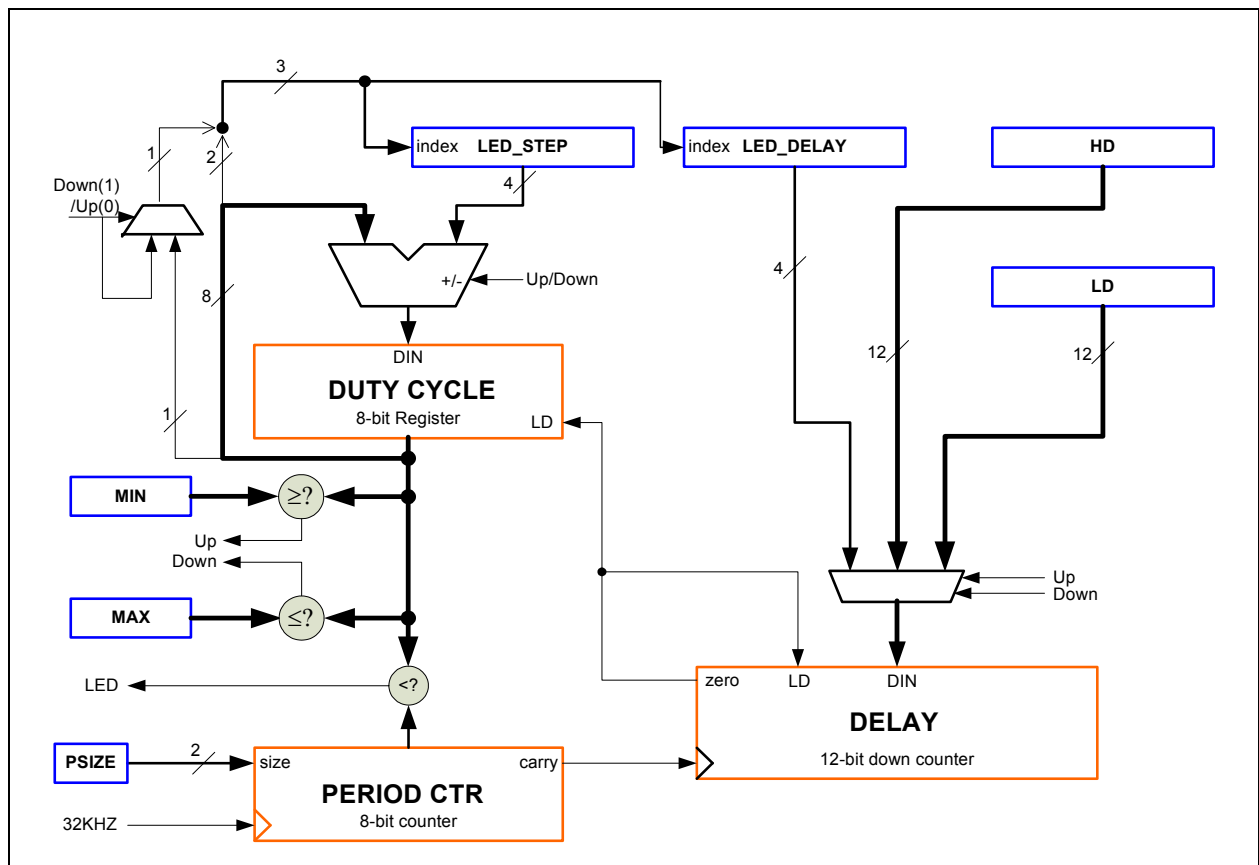
## 33.7 Implementation

In addition to the registers described in [Section 33.8, "EC-Only Registers"](#), the PWM is implemented using a number of components that are interconnected differently when configured for breathing operation and when configured for blinking/PWM operation.

### 33.7.1 BREATHING CONFIGURATION

When configured for breathing, components of the blinking/breathing hardware are interconnected as shown in the following figure:

**FIGURE 33-5: BREATHING LED HARDWARE CONFIGURATION**



In this figure, the boxes outlined in blue refer to registers, or fields in the registers, and boxes outlined in orange are internal counters and registers that are not directly accessible to firmware.

The **PSIZE** parameter can configure the PWM to one of three modes: 8-bit, 7-bit and 6-bit. The **PERIOD CTR** counts ticks of its input clock. In 8-bit mode, it counts from 0 to 255 (that is, 256 steps), then repeats continuously. In this mode, a full cycle takes 7.8msec (128Hz). In 7-bit mode it counts from 0 to 127 (128 steps), and a full cycle takes 3.9msec (256Hz). In 6-bit mode it counts from 0 to 63 (64 steps) and a full cycle takes 1.95msec (512Hz).

# MEC1618/MEC1618i

The output of the LED circuit is asserted whenever the **PERIOD CTR** is less than the contents of the **DUTY CYCLE** register. The appearance of breathing is created by modifying the contents of the **DUTY CYCLE** register in a continuous manner. When the LED control is off the internal counters and registers are all reset to 0 (i.e. after a write setting the **RESET** bit in the **LED CONFIGURATION (LED\_CFG)** Register.) After being enabled, the **DUTY CYCLE** register is increased by an amount determined by the **LED\_STEP** register and at a rate determined by the **DELAY** counter. After the duty cycle reaches its maximum value (determined by the field **MAX**), the duty cycle is held constant for a period determined by the field **HD**. After the hold time is complete, the **DUTY CYCLE** register is decreased, again by an amount determined by the **LED\_STEP** register and at a rate determined by the **DELAY** counter. When the duty cycle then falls at or below the minimum value (determined by the field **MIN**), the duty cycle is held constant for a period determined by the field **HD**. After the hold time is complete, the cycle repeats, with the duty cycle oscillating between **MIN** and **MAX**.

The rising and falling ramp times as shown in [FIGURE 33-2: Breathing LED Example on page 531](#) can be either symmetric or asymmetric depending on the setting of the **Symmetry (SYM)** bit in the **LED CONFIGURATION (LED\_CFG)** Register. In Symmetric mode the rising and falling ramp rates have mirror symmetry; both rising and falling ramp rates use the same (all) 8 segments fields in each of the following registers (see [Table 33-12](#)): the **LED UPDATE STEPSIZE (LED\_STEP)** register and the **LED UPDATE INTERVAL (LED\_INT)** register. In Asymmetric mode the rising ramp rate uses 4 of the 8 segments fields and the falling ramp rate uses the remaining 4 of the 8 segments fields (see [Table 33-12](#)).

The parameters **MIN**, **MAX**, **HD**, **LD** and the 8 fields in **LED\_STEP** and **LED\_INT** determine the brightness range of the LED and the rate at which its brightness changes. See the descriptions of the fields in [Section 33.8, "EC-Only Registers"](#), as well as the examples in [Section 33.7.3, "Breathing Examples"](#) for information on how to set these fields.

**TABLE 33-12: SYMMETRIC BREATHING MODE REGISTER USAGE**

Rising/ Falling Ramp Times in <a href="#">FIGURE 33-2: on page 531</a>	Duty Cycle	Segment Index	Symmetric Mode Register Fields Utilized	
X	000xxxxxb	000b	STEP[0]/INT[0]	Bits[3:0]
X	001xxxxxb	001b	STEP[1]/INT[1]	Bits[7:4]
X	010xxxxxb	010b	STEP[2]/INT[2]	Bits[11:8]
X	011xxxxxb	011b	STEP[3]/INT[3]	Bits[15:12]
X	100xxxxxb	100b	STEP[4]/INT[4]	Bits[19:16]
X	101xxxxxb	101b	STEP[5]/INT[5]	Bits[23:20]
X	110xxxxxb	110b	STEP[6]/INT[6]	Bits[27:24]
X	111xxxxxb	111b	STEP[7]/INT[7]	Bits[31:28]
<b>Note:</b> In Symmetric Mode the Segment_Index[2:0] = Duty Cycle Bits[7:5]				

**TABLE 33-13: ASYMMETRIC BREATHING MODE REGISTER USAGE**

Rising/ Falling Ramp Times in <a href="#">FIGURE 33-2: on page 531</a>	Duty Cycle	Segment Index	Asymmetric Mode Register Fields Utilized	
Rising	00xxxxxxb	000b	STEP[0]/INT[0]	Bits[3:0]
Rising	01xxxxxxb	001b	STEP[1]/INT[1]	Bits[7:4]
Rising	10xxxxxxb	010b	STEP[2]/INT[2]	Bits[11:8]



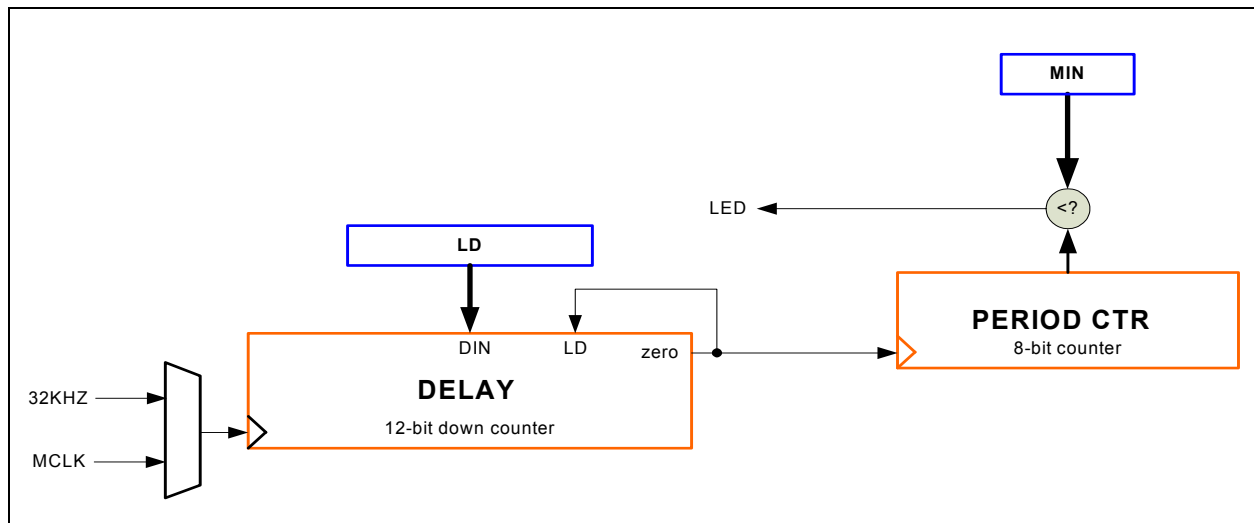
**TABLE 33-13: ASYMMETRIC BREATHING MODE REGISTER USAGE (CONTINUED)**

Rising/ Falling Ramp Times in <a href="#">FIGURE 33-2: on page 531</a>	Duty Cycle	Segment Index	Asymmetric Mode Register Fields Utilized	
Rising	11xxxxxb	011b	STEP[3]/INT[3]	Bits[15:12]
falling	00xxxxxb	100b	STEP[4]/INT[4]	Bits[19:16]
falling	01xxxxxb	101b	STEP[5]/INT[5]	Bits[23:20]
falling	10xxxxxb	110b	STEP[6]/INT[6]	Bits[27:24]
falling	11xxxxxb	111b	STEP[7]/INT[7]	Bits[31:28]
<b>Note:</b> In Asymmetric Mode the Segment_Index[2:0] is the bit concatenation of following: Segment_Index[2] = (FALLING RAMP TIME in <a href="#">FIGURE 33-2: on page 531</a> ) and Segment_Index[1:0] = Duty Cycle Bits[7:6].				

## 33.7.2 BLINKING CONFIGURATION

When configured for blinking or standard PWM operation, components of the blinking/breathing hardware are interconnected as shown in the following figure:

**FIGURE 33-6: BREATHING LED HARDWARE CONFIGURATION**



The Delay counter and the PWM counter are the same as in the breathing configuration, except in this configuration they are connected differently. The Delay counter is clocked on either the [32.768 kHz clock](#) or the [20.27 MHz clock](#), rather than the output of the PWM. The PWM counter is clocked by the zero output of the Delay counter, which functions as a prescaler for the input clocks to the PWM. The Delay counter is reloaded from the LD field of the LED\_DELAY register. When the LD field is 0 the input clock is passed directly to the PWM counter without prescaling. In Blinking/PWM mode the PWM counter is always 8-bit, and the PSIZE parameter has no effect.

The frequency of the PWM pulse waveform is determined by the formula:

$$f_{PWM} = \frac{f_{clock}}{(256 \times (LD + 1))}$$

where  $f_{PWM}$  is the frequency of the PWM,  $f_{clock}$  is the frequency of the input clock ([32.768 kHz clock](#) or [20.27 MHz clock](#)) and LD is the contents of the LD field.

# MEC1618/MEC1618i

**Note:** At a duty cycle value of 00h (in the MIN register), the LED output is fully off. At a duty cycle value of 255h, the LED output is fully on. This is equivalent to setting the Configuration Register CONTROL field to Always Off and Always On.

The other registers in the block do not affect the PWM or the LED output in Blinking/PWM mode.

## 33.7.3 BREATHING EXAMPLES

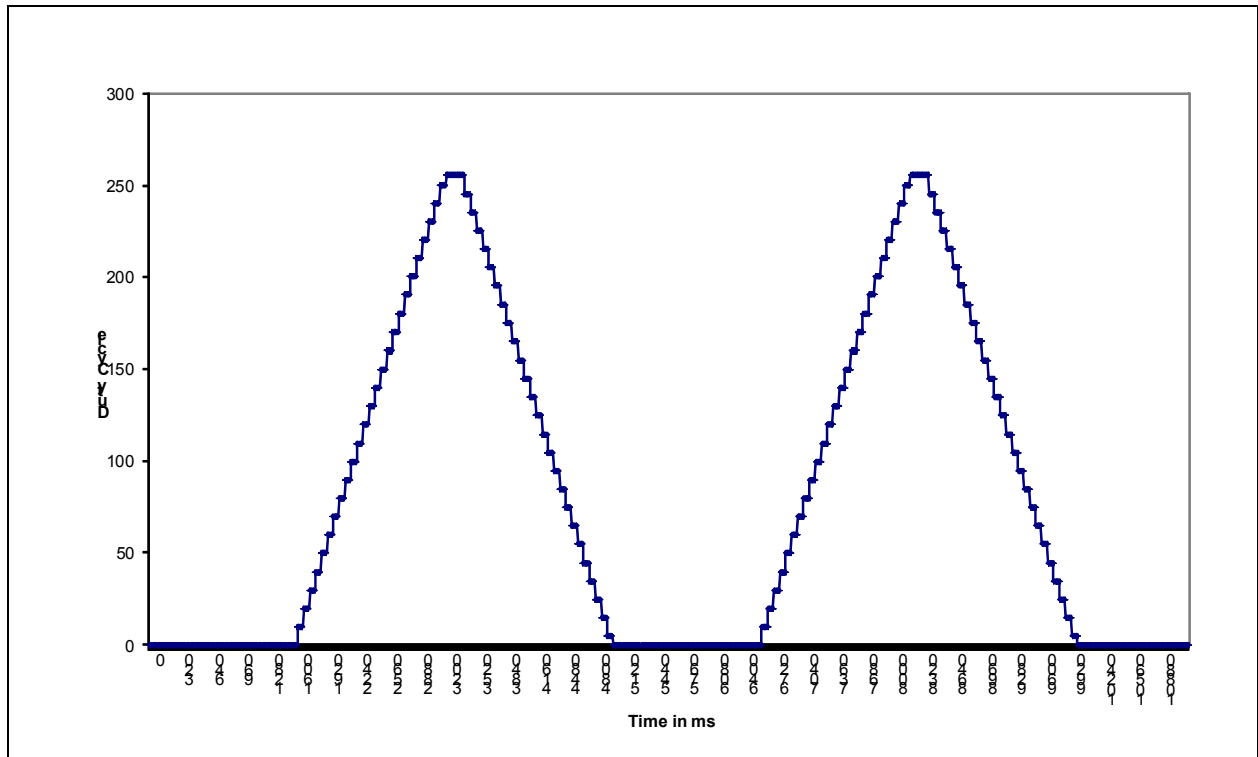
### 33.7.3.1 Linear LED Brightness Change

In this example, the brightness of the LED increases and diminishes in a linear fashion. The entire cycle takes 5 seconds. The rise time and fall time are 1.6 seconds, with a hold time at maximum brightness of 200msec and a hold time at minimum brightness of 1.6 seconds. The LED brightness varies between full off and full on. The PWM size is set to 8-bit, so the time unit for adjusting the PWM is approximately 8msec. The registers are configured as follows:

**TABLE 33-14: LINEAR EXAMPLE CONFIGURATION**

Field	Value							
PSIZE	8-bit							
MAX	255							
MIN	0							
HD	25 ticks (200msec)							
LD	200 ticks (1.6sec)							
Duty cycle most significant bits	000b	001b	010b	011b	100b	101b	110b	111b
LED_INT	8	8	8	8	8	8	8	8
LED_STEP	10	10	10	10	10	10	10	10

**FIGURE 33-7: LINEAR BRIGHTNESS CURVE EXAMPLE**



### 33.7.3.2 Non-linear LED Brightness Change

In this example, the brightness of the LED increases and diminishes in a non-linear fashion. The brightness forms a curve that is approximated by four piece wise-linear line segments. The entire cycle takes about 2.8 seconds. The rise time and fall time are about 1 second, with a hold time at maximum brightness of 320msec and a hold time at minimum brightness of 400msec. The LED brightness varies between full off and full on. The PWM size is set to 7-bit, so the time unit for adjusting the PWM is approximately 4ms. The registers are configured as follows:

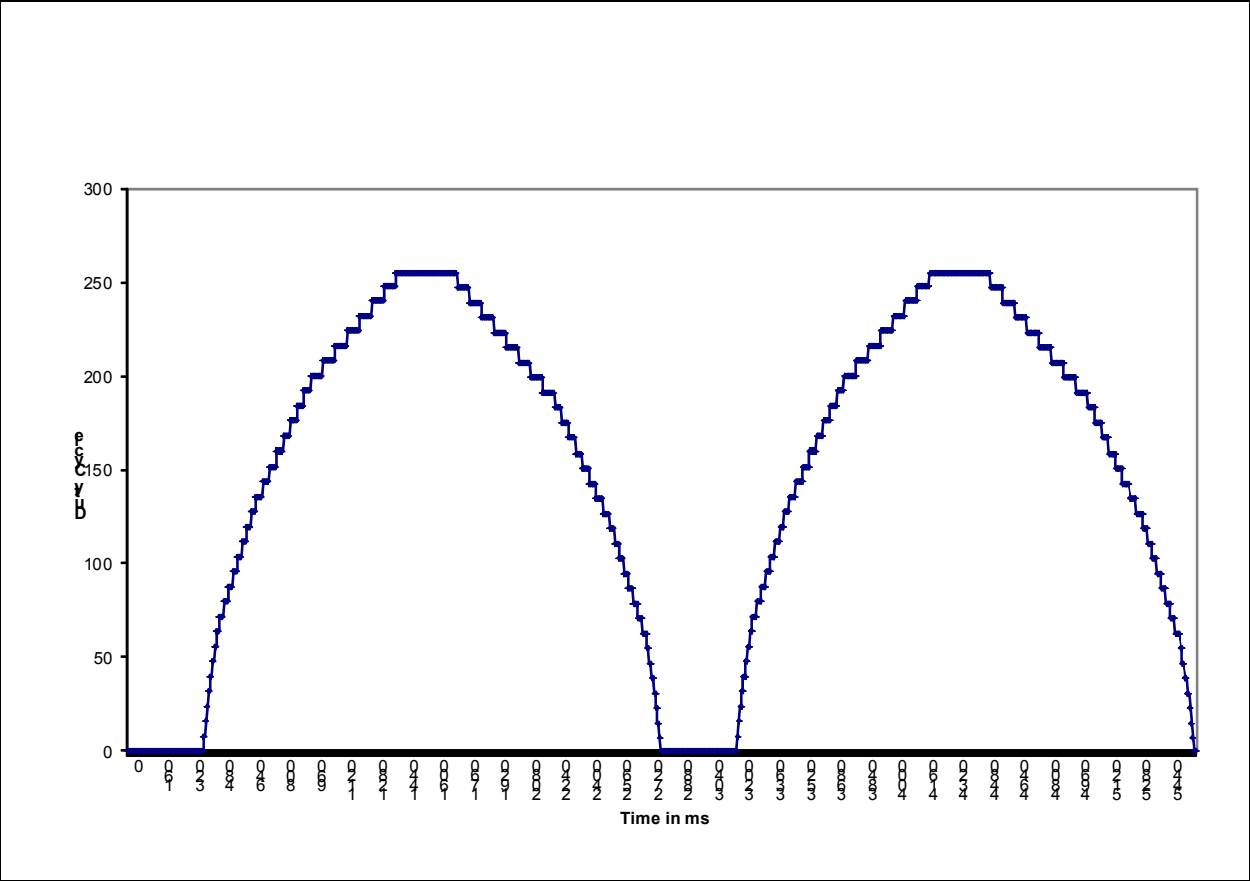
**TABLE 33-15: NON-LINEAR EXAMPLE CONFIGURATION**

Field	Value							
<b>PSIZE</b>	7-bit							
<b>MAX</b>	255 (effectively 127)							
<b>MIN</b>	0							
<b>HD</b>	80 ticks (320ms)							
<b>LD</b>	100 ticks (400ms)							
Duty cycle most significant bits	000b	001b	010b	011b	100b	101b	110b	111b
<b>LED_INT</b>	2	3	6	6	9	9	16	16
<b>LED_STEP</b>	4	4	4	4	4	4	4	4

# MEC1618/MEC1618i

The resulting curve is shown in the following figure:

**FIGURE 33-8: NON-LINEAR BRIGHTNESS CURVE EXAMPLE**



### 33.8 EC-Only Registers

The registers listed in the [EC-Only Registers Address Range Table](#) are for a single instance of the [Blinking/Breathing PWM](#). The addresses of each register listed in this table are defined as a relative offset to the host “Base Address” defined in [EC-Only Registers Address Range Table](#).

**TABLE 33-16: EC-Only Registers ADDRESS RANGE TABLE**

Block Instance	Instance Number	Host	Address Space	Base Address
LED	0	EC	24-bit internal address space	<a href="#">F0_8400h</a>
LED	1	EC	24-bit internal address space	<a href="#">F0_8400h + 80h</a>
LED	2	EC	24-bit internal address space	<a href="#">F0_8400h + 100h</a>

**TABLE 33-17: EC-ONLY REGISTER SUMMARY**

Offset	Register Name (Mnemonic)
00h	LED CONFIGURATION (LED_CFG)
04h	LED LIMITS (LED_LIMIT)
08h	LED DELAY (LED_DELAY)
0Ch	LED UPDATE STEPSIZE (LED_STEP)
10h	LED UPDATE INTERVAL (LED_INT)

In the following register definitions, a “PWM period” is defined by time the PWM counter goes from 000h to its maximum value (FFh in 8-bit mode, FEh in 7-bit mode and FCh in 6-bit mode, as defined by the PSCALE field in register LED\_CFG). The end of a PWM period occurs when the PWM counter wraps from its maximum value to 0.

The registers in this block can be written 32-bits, 16-bits or 8-bits at a time. Writes to LED CONFIGURATION (LED\_CFG) take effect immediately. Writes to LED LIMITS (LED\_LIMIT) are held in a holding register and only take effect only at the end of a PWM period. The update takes place at the end of every period, even if only one byte of the register was updated. This means that in blink/PWM mode, software can change the duty cycle with a single 8-bit write to the MIN field in the LED\_LIMIT register. Writes to LED DELAY (LED\_DELAY), LED UPDATE STEPSIZE (LED\_STEP) and LED UPDATE INTERVAL (LED\_INT) also go initially into a holding register. The holding registers are copied to the operating registers at the end of a PWM period only if the Enable Update bit in the LED CONFIGURATION (LED\_CFG) is set to 1. If LED\_CFG is 0, data in the holding registers is retained but not copied to the operating registers when the PWM period expires. To change an LED breathing configuration, software should write these three registers with the desired values and then set LED\_CFG to 1. This mechanism ensures that all parameters affecting LED breathing will be updated consistently, even if the registers are only written 8 bits at a time.

## 33.8.1 LED CONFIGURATION (LED\_CFG)

Offset	00h			
Bits	Description	Type	Default	Reset Event
31:16	RESERVED	RES	-	-
16	<p>Symmetry (SYM) When this bit is a 0, the rising and falling ramp times (as shown in <a href="#">FIGURE 33-2: Breathing LED Example on page 531</a>) are in Symmetric mode. <a href="#">Table 33-12</a> defines the Segment Index that are utilized by LED UPDATE STEPSIZE (LED_STEP) register and the LED UPDATE INTERVAL (LED_INT) register.</p> <p>When this bit is a 1, the rising and falling ramp times (as shown in <a href="#">FIGURE 33-2: Breathing LED Example on page 531</a>) are in Asymmetric mode. <a href="#">Table 33-13</a> defines the Segment Index that are utilized by LED UPDATE STEPSIZE (LED_STEP) register and the LED UPDATE INTERVAL (LED_INT) register.</p>	R/W	0b	nSYS_RST
15:8	<p>WDT Reload (WDTLD) The PWM Watchdog Timer counter reload value. On system reset, it defaults to 14h, which corresponds to a 4 second Watchdog timeout value.</p>	R/W	14h	nSYS_RST

# MEC1618/MEC1618i

Offset	00h			
Bits	Description	Type	Default	Reset Event
7	<b>RESET</b> Writes a '1' to this bit resets the <a href="#">Blinking/Breathing PWM</a> Registers their default values. This bit is self clearing. Writes a '0' to this bit has no effect.	WO	0b	<a href="#">nSYS_RST</a>
6	<b>Enable Update (ENUP; HW_CLR)</b> This bit is set to 1 when written with a 1. Writes of 0 have no effect. Hardware clears this bit to 0 when the breathing configuration registers are updated at the end of a PWM period. The current state of the bit is readable any time.  This bit is used to enable consistent configuration of LED_DELAY, LED_STEP and LED_INT. As long as this bit is 0, data written to those three registers is retained in a holding register. When this bit is 1, data in the holding register are copied to the operating registers at the end of a PWM period. When the copy completes, hardware clears this bit to 0.	R/WS	0b	<a href="#">nSYS_RST</a>
5:4	<b>PWM Size (PSIZE)</b> This bit controls the behavior of PWM:  0 = PWM is configured as an 8-bit PWM 1 = PWM is configured as a 7-bit PWM 2 = PWM is configured as a 6-bit PWM 3 = Reserved	R/W	0b	<a href="#">nSYS_RST</a>
3	<b>Synchronize (SYNC)</b> When this bit is 1, all counters for all LEDs are reset to their initial values. When this bit is 0 in the <a href="#">LED CONFIGURATION (LED_CFG)</a> Register for all LEDs, then all counters for LEDs that are configured to blink or breathe will increment or decrement, as required.  To synchronize blinking or breathing, the <a href="#">SYNC</a> bit should be set for at least one LED, the control registers for each LED should be set to their required values, then the <a href="#">SYNC</a> bits should all be cleared. If the all LEDs are set for the same blink period, they will all be synchronized.	R/W	0b	<a href="#">nSYS_RST</a>
2	<b>Clock Source (CLKSRC)</b> This bit controls the base clock for the PWM. It is only valid when CNTRL is set to blink (2).  0 = Clock source is the <a href="#">32.768 kHz clock</a> 1 = Clock source is the <a href="#">20.27 MHz clock</a>	R/W	0b	<a href="#">nSYS_RST</a>
1:0	<b>Control (CNTRL)</b> This bit controls the behavior of PWM:  0 = PWM is always off. All internal registers and counters are reset to 0. Clocks are gated 1 = LED breathing configuration 2 = LED blinking (standard PWM) 3 = PWM is always on	R/W	00b 11b	<a href="#">nSYS_RST</a> WDT TC

## 33.8.2 LED LIMITS (LED\_LIMIT)

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period. The two byte fields may be written independently. Reads of this register return the current contents and not the value of the holding register.

Offset	04h			
Bits	Description	Type	Default	Reset Event
31:16	RESERVED	RES	-	-
15:8	Maximum (MAX) In breathing mode, when the current duty cycle is greater than or equal to this value the breathing apparatus holds the current duty cycle for the period specified by the field HD in register LED_DELAY, then starts decrementing the current duty cycle	R/W	0h	nSYS_RST
7:0	Minimum (MIN) In breathing mode, when the current duty cycle is less than or equal to this value the breathing apparatus holds the current duty cycle for the period specified by the field LD in register LED_DELAY, then starts incrementing the current duty cycle  In blinking mode, this field defines the duty cycle of the blink function.	R/W	0h	nSYS_RST

# MEC1618/MEC1618i

## 33.8.3 LED DELAY (LED\_DELAY)

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period if the Enable Update bit in the LED Configuration register is set to 1. Reads of this register return the current contents and not the value of the holding register.

Offset	08h			
Bits	Description	Type	Default	Reset Event
31:24	RESERVED	RES	-	-
23:12	High Delay (HD) In breathing mode, the number of PWM periods to wait before updating the current duty cycle when the current duty cycle is greater than or equal to the value MAX in register LED_LIMIT. 0 = The delay counter is bypassed and the current duty cycle is decremented after one PWM period 1 = The delay counter is bypassed and the current duty cycle is decremented after two PWM period ... 4095: The current duty cycle is decremented after 4096 PWM periods	R/W	000h	nSYS_RST
11:0	Low Delay (LD) The number of PWM periods to wait before updating the current duty cycle when the current duty cycle is greater than or equal to the value MIN in register LED_LIMIT. 0 = The delay counter is bypassed and the current duty cycle is incremented after one PWM period 1 = The delay counter is bypassed and the current duty cycle is incremented after two PWM period ... 4095: The current duty cycle is incremented after 4096 PWM periods  In blinking mode, this field defines the prescaler for the PWM clock	R/W	000h	nSYS_RST

## 33.8.4 LED UPDATE STEPSIZE (LED\_STEP)

This register has eight segment fields which provide the amount the current duty cycle is adjusted at the end of every PWM period. Segment field selection is decoded based on the segment index. The segment index equation utilized depends on the [Symmetry \(SYM\)](#) bit in the [LED CONFIGURATION \(LED\\_CFG\)](#) Register (See [Table 33-12 on page 536](#) & [Table 33-13 on page 536](#) for detailed description of segment selection)

- In Symmetric Mode the [Segment\\_Index\[2:0\] = Duty Cycle Bits\[7:5\]](#)
- In Asymmetric Mode the [Segment\\_Index\[2:0\]](#) is the bit concatenation of following: [Segment\\_Index\[2\] = \(FALLING RAMP TIME in FIGURE 33-2: on page 531\)](#) and [Segment\\_Index\[1:0\] = Duty Cycle Bits\[7:6\]](#).

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period if the Enable Update bit in the LED Configuration register is set to 1. Reads of this register return the current contents and not the value of the holding register.

In 8-bit mode, each 4-bit STEPSIZE field represents 16 possible duty cycle modifications, from 1 to 16 as the duty cycle is modified between 0 and 255:

0: Modify the duty cycle by 1

1: Modify the duty cycle by 2

...

15: Modify the duty cycle by 16



In 7-bit mode, the least significant bit of the 4-bit field is ignored, so each field represents 8 possible duty cycle modifications, from 1 to 8, as the duty cycle is modified between 0 and 127:

0, 1: Modify the duty cycle by 1

2, 3: Modify the duty cycle by 2

...

14, 15: Modify the duty cycle by 8

In 6-bit mode, the two least significant bits of the 4-bit field is ignored, so each field represents 4 possible duty cycle modifications, from 1 to 4 as the duty cycle is modified between 0 and 63:

0, 1, 2, 3: Modify the duty cycle by 1

4, 5, 6, 7: Modify the duty cycle by 2

8, 9, 10, 11: Modify the duty cycle by 3

12, 13, 14, 15: Modify the duty cycle by 4

Offset	0Ch			
Bits	Description	Type	Default	Reset Event
31:28	Update Step7 (STEP7) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 111.	R/W	0h	nSYS_RST
27:24	Update Step6 (STEP6) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 110.	R/W	0h	nSYS_RST
23:20	Update Step5 (STEP5) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 101.	R/W	0h	nSYS_RST
19:16	Update Step4 (STEP4) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 100.	R/W	0h	nSYS_RST
15:12	Update Step3 (STEP3) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 011.	R/W	0h	nSYS_RST
11:8	Update Step2 (STEP2) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 010.	R/W	0h	nSYS_RST
7:4	Update Step1 (STEP1) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 001.	R/W	0h	nSYS_RST
3:0	Update Step0 (STEP0) Amount the current duty cycle is adjusted at the end of every PWM period when the segment index is equal to 000.	R/W	0h	nSYS_RST

# MEC1618/MEC1618i

## 33.8.5 LED UPDATE INTERVAL (LED\_INT)

This register has eight segment fields which provide the number of PWM periods between updates to current duty cycle. Segment field selection is decoded based on the segment index. The segment index equation utilized depends on the [Symmetry \(SYM\)](#) bit in the [LED CONFIGURATION \(LED\\_CFG\)](#) Register (See [Table 33-12 on page 536](#) & [Table 33-13 on page 536](#) for detailed description of segment selection)

- In Symmetric Mode the [Segment\\_Index\[2:0\]](#) = Duty Cycle Bits[7:5]
- In Asymmetric Mode the [Segment\\_Index\[2:0\]](#) is the bit concatenation of following: [Segment\\_Index\[2\]](#) = (FALLING RAMP TIME in [FIGURE 33-2: on page 531](#)) and [Segment\\_Index\[1:0\]](#) = Duty Cycle Bits[7:6].

This register may be written at any time. Values written into the register are held in an holding register, which is transferred into the actual register at the end of a PWM period if the Enable Update bit in the LED Configuration register is set to 1. Reads of this register return the current contents and not the value of the holding register.

Offset	10h			
Bits	Description	Type	Default	Reset Event
31:28	Update Interval7 (INT7) The number of PWM periods between updates to current duty cycle when the segment index is equal to 111. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST
27:24	Update Interval6 (INT6) The number of PWM periods between updates to current duty cycle when the segment index is equal to 110. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST
23:20	Update Interval5 (INT5) The number of PWM periods between updates to current duty cycle when the segment index is equal to 101. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST
19:16	Update Interval4 (INT4) The number of PWM periods between updates to current duty cycle when the segment index is equal to 100. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST
15:12	Update Interval3 (INT3) The number of PWM periods between updates to current duty cycle when the segment index is equal to 011. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST
11:8	Update Interval2 (INT2) The number of PWM periods between updates to current duty cycle when the segment index is equal to 010. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST

Offset	10h			
Bits	Description	Type	Default	Reset Event
7:4	Update Interval1 (INT1) The number of PWM periods between updates to current duty cycle when the segment index is equal to 001. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST
3:0	Update Interval0 (INT0) The number of PWM periods between updates to current duty cycle when the segment index is equal to 000. 0 = Wait 1 PWM period ... 15 = Wait 16 PWM periods	R/W	0h	nSYS_RST

# MEC1618/MEC1618i

## 34.0 PS/2 DEVICE INTERFACE

### 34.1 General Description

There are three PS/2 Ports in the MEC1618/MEC1618i independent EC PS/2 serial ports implemented in hardware which are directly controlled by the EC (see [FIGURE 34-1: on page 548](#)). The hardware implementation eliminates the need to bit bang I/O ports to generate PS/2 traffic, however bit banging is available via the associated GPIO pins.

Each EC PS/2 serial channels use a synchronous serial protocol to communicate with the auxiliary device. Each PS/2 channel has Clock and Data signal lines. The signal lines are bi-directional and employ open drain outputs capable of sinking 16mA. A pull-up resistor, typically 10K, is connected to both lines. This allows either the MEC1618/MEC1618i EC PS/2 logic or the auxiliary device to drive the lines. Regardless of the drive source, the auxiliary device always provides the clock for transmit and receive operations. The serial packet is made up of eleven bits, listed in the order they appear on the data line: start bit, eight data bits (least significant bit first), odd parity, and stop bit. Each bit cell is from 60μS to 100μS long.

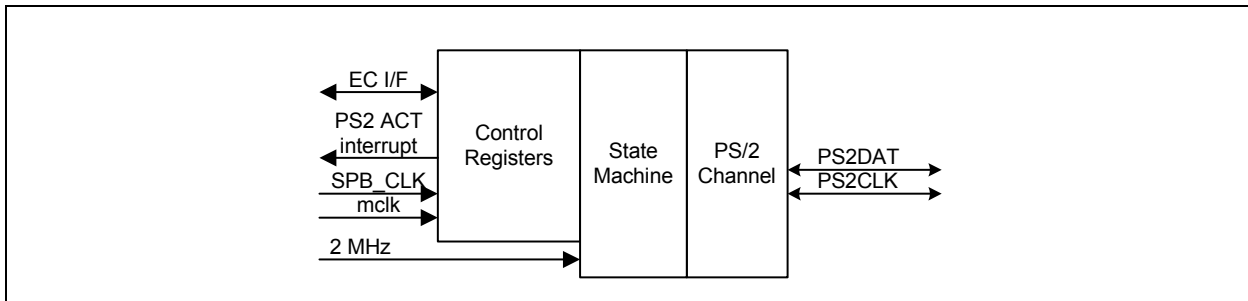
All PS/2 Serial Channel signals (CLK and DAT) are driven by open drain drivers which can be pulled to VTR or VCC (+3.3V nominal) through 10K-ohm resistors.

The MEC1618/MEC1618i supports a PS/2 Wake Interface that can wake the EC from the IDLE or SLEEP states. The PS/2 Wake Interface is powered by [VTR](#) and can generate wake interrupts without a clock.

**APPLICATION NOTE:** the PS/2 Wake Interface is only active when the KBC/Mouse and external pull-up resistors are powered by the VCC1supply. The external pull-up resistor must always be powered by the same source as the KBC/Mouse.

### 34.2 Block Diagram

**FIGURE 34-1: PORT PS/2 BLOCK DIAGRAM**



### 34.3 PS/2 Port Physical Layer Byte Transmission Protocol

The PS/2 physical layer transfers a byte of data via an eleven bit serial stream as shown in [Table 34-1](#). A logic 1 is sent at an active high level. Data sent from a Keyboard or mouse device to the host is read on the falling edge of the clock signal. The Keyboard or mouse device always generates the signal. The Host may inhibit communication by pulling the Clock line low. The Clock line must be continuously high for at least 50 microseconds before the Keyboard or mouse device can begin to transmit its data. See [Table 34-2, "PS/2 Port Physical Layer Bus States"](#).

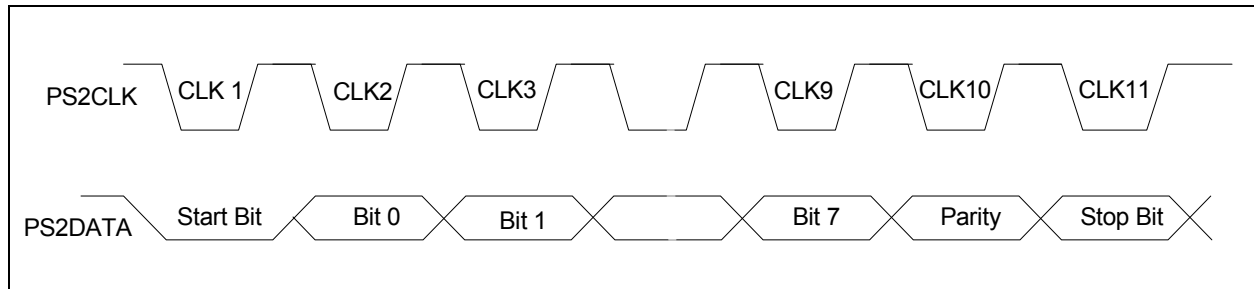
**TABLE 34-1: PS/2 PORT PHYSICAL LAYER BYTE TRANSMISSION PROTOCOL**

Bit	Function
1	Start bit (always 0)
2	Data bit 0 (least significant bit)
3	Data bit 1
4	Data bit 2

**TABLE 34-1: PS/2 PORT PHYSICAL LAYER BYTE TRANSMISSION PROTOCOL (CONTINUED)**

Bit	Function
5	Data bit 3
6	Data bit 4
7	Data bit 5
8	Data bit 6
9	Data bit 7 (most significant bit)
10	Parity bit (odd parity)
11	Stop Bit (always 1)

**FIGURE 34-2: PS/2 PORT PHYSICAL LAYER BYTE TRANSMISSION PROTOCOL**



**TABLE 34-2: PS/2 PORT PHYSICAL LAYER BUS STATES**

Data	Clock	State
high	high	Idle
high	low	Communication Inhibited
low	low	Request to Send

## 34.4 Block Diagram Signal List

**TABLE 34-3: PS/2 PORT LIST**

Signal Name	Direction	Description
PS2DAT	INPUT/OUTPUT	Data from the PS/2 device
PS2CLK	INPUT/OUTPUT	Clock from the PS/2 device
<a href="#">nSYS_RST</a>	INPUT	Chip Power on Reset (i.e., Suspend Well)
SPB_CLK	INPUT	Clock Source to EC microcontroller. Used for reading/writing registers on the EC memory i/f.
mclk	INPUT	<a href="#">MCLK</a>
2 MHz	INPUT	<a href="#">MCLK_DIV10_EN</a> , State machine clock

# MEC1618/MEC1618i

TABLE 34-3: PS/2 PORT LIST (CONTINUED)

Signal Name	Direction	Description
SLEEP_EN	INPUT	External <a href="#">Power Management</a> enable/disable input signal used to put the block in the lowest power consumption state. 0=No Sleep Requested. The block should operate as configured. 1=Sleep Requested. The block enters sleep mode.
CLK_REQ	OUTPUT	<a href="#">Power Management</a> output indicates when this block requires <a href="#">MCLK</a> input. 0= Clock can be turned 'off' when appropriate 1= Clock is required to be 'on.'
EC Interface	I/O BUS	EC-side SPB bus
PS2 ACT	OUTPUT	Asynchronous Interrupt

## 34.5 Power, Clocks and Reset

### 34.5.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

### 34.5.2 CLOCKS

This block uses the [EC Bus Clock](#), [MCLK](#) and the 2MHz [MCLK\\_DIV10\\_EN](#). [EC Bus Clock](#) is used when reading and writing the [PS/2 Device Interface](#) registers. The state machine is clocked with [MCLK\\_DIV10\\_EN](#).

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

## 34.6 Reset

This block is reset on a [nSYS\\_RST](#).

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 34.7 Instance Description

There are three block instances defined in this chapter: PS/2[0,1,2]. The pin signals are defined in [Table 2-16, "PS/2 Interface," on page 20](#). PS/2 ports ending with signal functions ending with "A" or "B" are MUXed to a single controller. Only one Port set of clock and data are intended to be used at a time (either "A" or "B" not both.) The unused port segment should have its associated [Pin Control Register](#)'s, MUX Control Field programmed away from the PS2 controller. See [Section 21.0, "GPIO Interface," on page 398](#).

## 34.8 Interrupts

Each EC PS/2 Channel has two interrupts: a PS/2 activity interrupt event and a START Bit detection Wake-up event. Each PS/2 Channel activity interrupt event is generated by changes in status bits in this block. The PS/2 Channel activity interrupt event are routed to the [PS2\\_ACT\\_0](#), [PS2\\_ACT\\_1](#), [PS2\\_ACT\\_2](#) bits in the [GIRQ19 Source Register on page 335](#). The START Bit detection Wake-up event is a PS/2 Channel/segment (see [Section 34.9](#)) Data pin signal edge detection interrupt and wake event. Each PS/2 Channel/segment START Bit detection Wake-up event is controlled by their associated Data pin's [Pin Control Register](#). (See [Section 21.0, "GPIO Interface," on page 398](#). The START Bit detection Wake-up events are routed to the [PS2\\_WK\\_0A](#), [PS2\\_WK\\_0B](#), [PS2\\_WK\\_1A](#), [PS2\\_WK\\_1B](#), [PS2\\_WK\\_2](#) bits in the [GIRQ19 Source Register on page 335](#).

**APPLICATION NOTE:** The pin control registers for a PS2 wakeup event should be programmed to Input, Falling Edge Triggered, non-inverted polarity detection.

### 34.9 Registers

There are three block instances defined in this chapter: PS/2[0,1,2]. The pin signals are defined in [Table 2-16, “PS/2 Interface,” on page 20](#). PS/2 ports ending with signal functions ending with “A” or “B” are MUXed to a single controller. Only one Port set of clock and data are intended to be used at a time (either “A” or “B” not both.) The unused port segment should have its associated [Pin Control Register](#)’s MUX Control Field programmed away from the PS2 controller. (See [Section 21.0, “GPIO Interface,” on page 398](#)).

Each instance of the [PS/2 Device Interface](#) has its Base Address as indicated in [Table 34-4](#).

**TABLE 34-4: PS/2 Device Interface BASE ADDRESS TABLE**

PS/2 Device Interface Instance	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
PS/2.0	22h	F0_8800h
PS/2.1		F0_8880h = F0_8800h + 80h
PS/2.2		F0_8900h = F0_8800h + 100h

[Table 34-5](#) is a register summary for each instance of the [PS/2 Device Interface](#).

**TABLE 34-5: PS/2 Device Interface REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	LC Type	
<a href="#">PS/2 Transmit Buffer Register</a>	00h	0	W	
<a href="#">PS/2 Receive Buffer Register</a>	00h	0	R	
<a href="#">PS/2 Control Register</a>	04h	0	R/W	
<a href="#">PS/2 Status Register</a>	08h	0	R/WC	

# MEC1618/MEC1618i

## 34.10 Detailed Description of Accessible Registers

### 34.10.1 PS/2 TRANSMIT BUFFER

TABLE 34-6: PS/2 TRANSMIT BUFFER REGISTER

HOST ADDRESS	NA						HOST SIZE	
EC OFFSET	00h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	--	-	-	-	-
EC TYPE	W	W	W	W	W	W	W	W
BIT NAME	Transmit Data[7:0]							

#### TRANSMIT DATA

The byte written to this register, when PS2\_T/R and PS2\_EN in the [PS/2 Control Register](#) and XMIT\_IDLE in the [PS/2 Status Register](#) are set, is transmitted automatically by the PS/2 channel control logic. If any of these three bits (PS2\_T/R, PS2\_EN, and XMIT\_IDLE) are not set, then writes to this register are ignored. On successful completion of this transmission or upon a Transmit Time-out condition, the PS2\_T/R bit is automatically cleared and the XMIT\_IDLE bit is automatically set. The PS2\_T/R bit must be written to a '1' before initiating another transmission to the remote device.

#### Note:

- Even if PS2\_T/R, PS2\_EN, and XMIT\_IDLE are all set, writing the Transmit Register will not start a transmission if RDATA\_RDY in the [PS/2 Status Register](#) is set. The automatic PS/2 logic forces data to be read from the Receive Register before allowing a transmission.
- An interrupt is generated on the low to high transition of XMIT\_IDLE.
- All bits of this register are write only.



## 34.10.2 PS/2 RECEIVE BUFFER

TABLE 34-7: PS/2 RECEIVE BUFFER REGISTER

HOST ADDRESS	NA							HOST SIZE	
EC OFFSET	00h				8-bit			EC SIZE	
POWER	VTR				FFh			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	--	-	-	-	-	
EC TYPE	R								
BIT NAME	Receive Data[7:0]								

## RECEIVE DATA

When PS2\_EN=1 and PS2\_T/R=0 in the [PS/2 Control Register](#), the PS2 Channel is configured to automatically receive data on that channel (both the CLK and DATA lines will float waiting for the peripheral to initiate a reception by sending a start bit followed by the data bits). After a successful reception, data is placed in this register and the RDATA\_RDY bit in the [PS/2 Status Register](#) is set and the CLK line is forced low by the PS2 channel logic. RDATA\_RDY is cleared and the CLK line is released to high-z following a read of this register. This automatically holds off further receive transfers until the EC has had a chance to get the data.

**Note:**

- The Receive Register is initialized to FFh after a read or after a Time-out has occurred.
- The channel can be enabled to automatically transmit data (PS2\_EN=1) by setting PS2\_T/R while RDATA\_RDY is set, however a transmission can not be kicked off until the data has been read from the Receive Register.
- An interrupt is generated on the low to high transition of RDATA\_RDY.
- If a receive time-out (REC\_TIMEOUT=1 in the [PS/2 Control Register](#)) or a transmit time-out (XMIT\_TIMEOUT=1 in the [PS/2 Control Register](#)) occurs the channel is busied (CLK held low) for 300us (Hold Time) to guarantee that the peripheral aborts. Writing to the Transmit Register will be allowed, however the data written will not be transmitted until the Hold Time expires.
- All bits in this register are read only

**Note 34-1** In receive mode the RX\_BUSY bit for a particular channel is set in the [PS/2 Status Register](#).

## 34.10.3 PS/2 CONTROL

There are three PS/2 Control Registers, one for each channel.

# MEC1618/MEC1618i

TABLE 34-8: PS/2 CONTROL REGISTER

HOST ADDRESS	NA						HOST SIZE	
EC OFFSET	04h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Reserved		STOP		PARITY		PS2_EN	PS2_T/R

## PS2\_T/R

PS/2 Channel Transmit/Receive (default = 0). Configures the PS/2 logic for automatic transmission when set or reception when cleared. This bit is only valid when PS2\_EN is set.

When set the PS/2 channel is enabled to transmit data. To properly initiate a transmit operation, this bit must be set prior to writing to the Transmit Register. Writes to the Transmit Register are blocked when this bit is cleared. Upon setting the PS2\_T/R bit, the channel will drive its CLK line low and then float the DATA line and hold this state until a write occurs to the Transmit Register or until the PS2\_T/R bit is cleared. Writing to the Transmit Register initiates the transmit operation. MEC1618/MEC1618i drives the data line low and, within 80ns, floats the clock line (externally pulled high by the pull-up resistor) to signal to the external PS/2 device that data is now available. The PS2\_T/R bit is cleared on the 11th clock edge of the transmission or if a Transmit Time-out error condition occurs.

**Note:** If the PS2\_T/R bit is set while the channel is actively receiving data prior to the leading edge of the 10th (parity bit) clock edge, the receive data is discarded. If this bit is not set prior to the 10th clock signal, then the receive data is saved in the Receive Register.

When the PS2\_T/R bit is cleared, the PS/2 channel is enabled to receive data. Upon clearing this bit, if RDATA\_RDY is also cleared, the channel's CLK and DATA will float waiting for the external PS/2 device to signal the start of a transmission. If the PS2\_T/R bit is set while RDATA\_RDY is set, then the channel's DATA line will float but its CLK line will be held low, holding off the peripheral, until the Receive Register is read.

## PS2\_EN

PS2 Channel ENable (default = 0). When PS2\_EN is set, the PS/2 State machine is enabled allowing the channel to perform automatic reception or transmission depending on the bit value of PS2\_T/R. When PS2\_EN is cleared, the channel's automatic PS/2 state machine is disabled and the PS/2 channel's CLK pin driven low and DATA pin not driven.

**Note:** If the PS2\_EN bit is cleared prior to the leading edge (falling edge) of the 10th (parity bit) clock edge the receive data is discarded (RDATA\_RDY remains low). If the PS2\_EN bit is cleared following the leading edge of the 10th clock signal, then the receive data is saved in the Receive Register (RDATA\_RDY goes high) assuming no parity error.

## PARITY

These bits are used to set the parity expected by the PS/2 channel state machine. These bits are therefore only valid when PS2\_EN is set.

00=Receiver expects Odd Parity (default).

01=Receiver expects Even Parity.

10=Receiver ignores level of the parity bit (10th bit is not interpreted as a parity bit).

11=Reserved

## STOP

These bits are used to set the level of the stop bit expected by the PS/2 channel state machine. These bits are therefore only valid when PS2\_EN is set.

00=Receiver expects an active high stop bit.

01=Receiver expects an active low stop bit.

10=Receiver ignores the level of the Stop bit (11th bit is not interpreted as a stop bit).

11=Reserved.

**APPLICATION NOTE:** Changing values in the control register at a rate faster than 2 MHz, may result in unpredictable behavior.

This register should be read to determine the status of PS2\_T/R and PS2\_EN prior to clearing by writing a 1 to that bit.

## 34.10.4 PS/2 STATUS

**TABLE 34-9: PS/2 STATUS REGISTER**

HOST ADDRESS	NA						HOST SIZE	
EC OFFSET	08h						8-bit	EC SIZE
POWER	VTR						10h	nSYS_RST DEFAULT
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R	R/WC	R	R/WC	R/WC	R/WC	R
BIT NAME	XMIT_ START_ TIMEOUT	RX_ BUSY	XMIT_ TIME- OUT	XMIT_ IDLE	FE	PE	REC_ TIME- OUT	RDATA_ RDY

**APPLICATION NOTE:** This register should be read to determine the status of R/WC bits prior to clearing by writing a 1 to that bit.

# MEC1618/MEC1618i

## RDATA\_RDY

Receive Data Ready: Under normal operating conditions, this bit is set following the falling edge of the 11th clock given successful reception of a data byte from the PS/2 peripheral (i.e., no parity, framing, or receive time-out errors) and indicates that the received data byte is available to be read from the Receive Register. This bit may also be set in the event that the PS2\_EN bit is cleared following the 10th CLK edge. Reading the Receive Register clears this bit.

**Note:** An Interrupt is generated on the low-to-high transition of the RDATA\_RDY bit.

## REC\_TIMEOUT

Following assertion of the REC\_TIMEOUT bit, the channel's CLK line is automatically pulled low for a minimum of 300us until the PS/2 status register is read. Under PS2 automatic operation, PS2\_EN is set, this bit is set on one of three receive error conditions:

- When the receiver bit time (time between falling edges) exceeds 300us.
  - If the time from the first bit (start) to the 10th bit (parity) exceeds 2ms.
  - On a receive parity error along with the Parity Error (PE) bit.
  - On a receive framing error due to an incorrect STOP bit along with the framing error (FE) bit.
- The REC\_TIMEOUT bit is cleared when the Status Register is read.

**Note:** An Interrupt is generated on the low-to-high transition of the REC\_TIMEOUT bit.

## PE

Parity Error: When receiving data, the parity bit is clocked in on the falling edge of the 10th CLK edge. If the channel is configured to expect either even or odd parity and the 10th bit is contrary to the expected parity, then the PE and REC\_TIMEOUT bits are set following the falling edge of the 10th CLK edge and an interrupt is generated.

## FE

Framing Error: When receiving data, the stop bit is clocked in on the falling edge of the 11th CLK edge. If the channel is configured to expect either a high or low stop bit and the 11th bit is contrary to the expected stop polarity, then the FE and REC\_TIMEOUT bits are set following the falling edge of the 11th CLK edge and an interrupt is generated.

## XMIT\_IDLE

Transmitter Idle: When low, the XMIT\_IDLE bit is a status bit indicating that the PS/2 channel is actively transmitting data to the PS2 peripheral device. Writing to the Transmit Register when the channel is ready to transmit will cause the XMIT\_IDLE bit to clear and remain clear until one of the following conditions occur: the falling edge of the 11th CLK, XMIT\_TIMEOUT is set; the PS2\_T/R bit is cleared or the PS2\_EN bit is cleared.

**Note 34-2** An interrupt is generated on the low-to-high transition of XMIT\_IDLE.

## XMIT\_TIMEOUT

When the XMIT\_TIMEOUT bit is set, the PS2\_T/R bit is held clear, the PS/2 channel's CLK line is pulled low for a minimum of 300us until the PS/2 Status register is read. The XMIT\_TIMEOUT bit is set on one of three transmit conditions: when the transmitter bit time (time between falling edges) exceeds 300us, when the transmitter start bit is not received within 25msec from signaling a transmit start event or if the time from the first bit (start) to the 10th bit (parity) exceeds 2ms.

## RX\_BUSY

When a RX\_BUSY bit is set, the associated channel is actively receiving PS/2 data; when a RX\_BUSY bit is clear, the channel is idle. See [Note 34-1 on page 553](#).

## XMIT\_START\_TIMEOUT

When the XMIT\_START\_TIMEOUT bit is set, a start bit was not received within 25msec following the transmit start event. Writing a '1' to the bit clears the XMIT\_START\_TIMEOUT bit. The XMIT\_START\_TIMEOUT bit is a 'sticky' bit and is intended to uniquely indicate the status of the transmit start bit time-out condition. These bit affect no other logic. Note that the transmit start bit time-out condition is also indicated by the XMIT\_TIMEOUT bit.

**PROGRAMMER'S NOTE:** Always check that an EC PS/2 channel is idle, i.e. the RX\_BUSY bit is clear, before attempting to transmit on that channel. Receive data may be lost by setting an EC PS/2 channel to transmit while the RX\_BUSY bit is set depending where in the message frame the transmit mode change occurs.

### 34.11 Power Management

**TABLE 34-10: PS/2 Device Interface Power Management**

PS2_EN Bit	SLEEP_EN	Block Idle Status (Note 34-3)	CLK_REQ	State	Description
0	X	X	0	SLEEPING	The block is disabled and the clock can be stopped.
1	0	NOT IDLE	1	NORMAL OPERATION	The block is not idle and neither disabled by firmware nor commanded to sleep.
	1	NOT IDLE	1	PREPARING TO SLEEP	The block is commanded to sleep, but the clock is required until the Block is idle.
	1	IDLE	0	SLEEPING	The block is commanded to sleep and idle. The clock can be stopped.

**Note 34-3** The PS/2 Device Interface 'idle' state is determined using the RX\_BUSY and XMIT\_IDLE bits as shown in Table 34-11.

**TABLE 34-11: PS/2 IDLE STATUS**

RX_BUSY	XMIT_IDLE	Status
0	1	IDLE
1	X	NOT IDLE
X	0	

# MEC1618/MEC1618i

## 35.0 KEYBOARD MATRIX SCAN SUPPORT

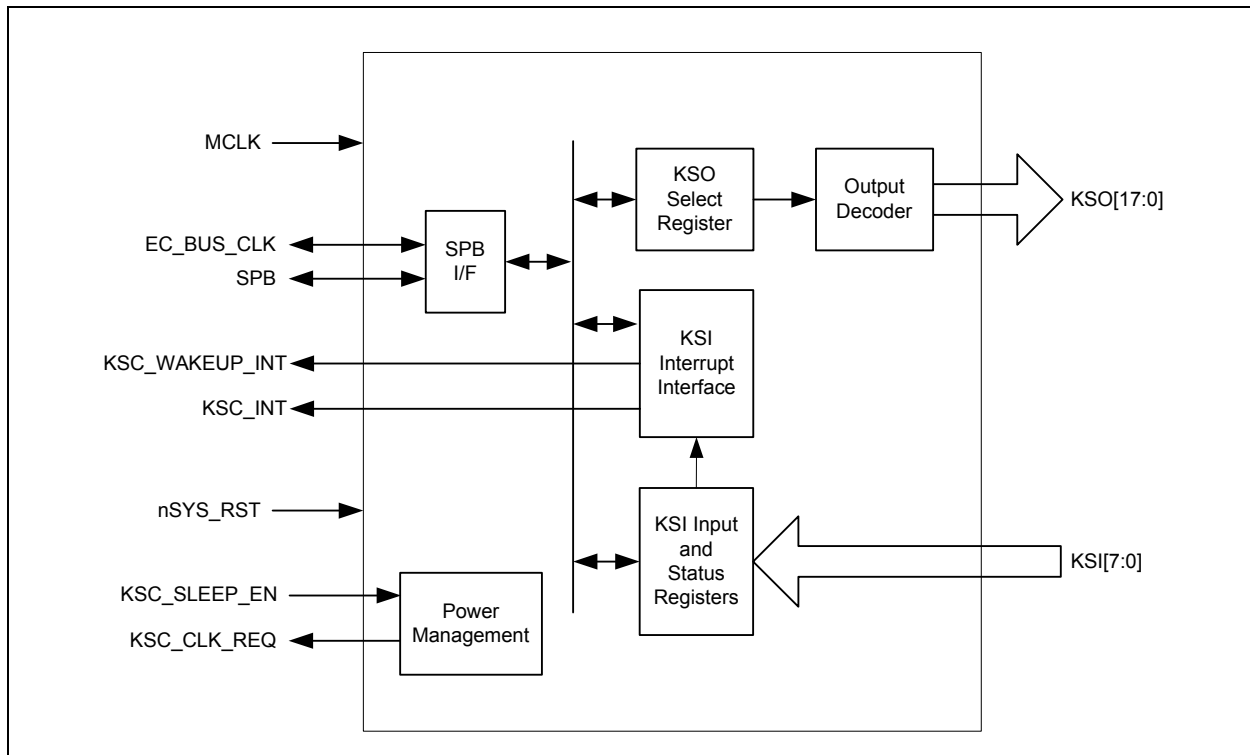
### 35.1 Overview

The Keyboard Scan Interface block provides a register interface to the EC to directly scan an external keyboard matrix of size up to 18x8. This block is attached to the EC SPB bus as EC Logical Device 8h.

**APPLICATION NOTE:** 18x8 is the maximum configuration. For smaller matrix size, firmware does not configure GPIO MUX control of unused KSOs and masks out unused KSIs and associated interrupts.

### 35.2 Block Diagram

**FIGURE 35-1:** Keyboard Matrix Scan Support Block Diagram



### 35.3 Port List

**TABLE 35-1: KEYBOARD SCAN INTERFACE SIGNAL LIST**

Signal Name	Direction	Description
SPB	I/O Bus	EC MEC1618/MEC1618i peripheral bus
MCLK	INPUT	Master MEC1618/MEC1618i clock
EC_BUS_CLK_EN	INPUT	SPB Bus Clock
nSYS_RST	INPUT	Block reset signal
KSI[7:0]	INPUT	Column inputs from external keyboard matrix. See <a href="#">Section 35.6</a>
KSO[17:0]	OUTPUT	Row outputs to external keyboard matrix. See <a href="#">Section 35.6</a>

TABLE 35-1: KEYBOARD SCAN INTERFACE SIGNAL LIST (CONTINUED)

Signal Name	Direction	Description
KSC_INT	OUTPUT	Interrupt request to the Interrupt Aggregator's interrupt interface
KSC_WAKEUP_INT	OUTPUT	Wake-up request to the Interrupt Aggregator's wake-up interface
KSC_SLEEP_EN	INPUT	Sleep enable. See <a href="#">Section 35.5</a> .
KSC_CLK_REQ	OUTPUT	Clock request. See <a href="#">Section 35.5</a> .

### 35.4 Power, Clocks, and Resets

#### 35.4.1 POWER DOMAIN

This block is powered by the VTR power supply.

#### 35.4.2 CLOCKS

The block uses the [EC\\_BUS\\_CLK\\_EN](#) and the [MCLK](#)

#### 35.4.3 RESET

The block is reset on assertion of [nSYS\\_RST](#).

### 35.5 Power Management

Keyboard Scan Interface power management operation is shown in [Table 35-2](#).

TABLE 35-2: KEY SCAN INTERFACE POWER MANAGEMENT

KSEN ( <a href="#">Note 35-1</a> )	External SLEEP_EN Input ( <a href="#">Note 35-2</a> )	Scan Active ( <a href="#">Note 35-3</a> )	Core Clock Required Status Output ( <a href="#">Note 35-2</a> )	Mode	Power	Description
1	X	X	0	DISABLED	MINIMUM	The Keyboard Scan Interface is disabled by the EC and the core clock is gated 'off' internally.
0	X	YES	1	FULL POWER	MAXIMUM	FULL POWER mode identifies that the Keyboard Scan Interface is enabled and clocks are required to complete a scan related operation.
		NO	0	SLEEPING	MINIMUM	In SLEEPING mode, there will be no clocks to the Keyboard Scan Interface except during access to the registers.

**Note 35-1** the KSEN bit is in the KSO Select Register in LPC Configuration Space. When the KSEN bit is '1', there will be no clocks active in the Keyboard Scan Interface except during access to the registers.

**Note 35-2** the external sleep enable input and clock required status output are part of the chip-level clocking interface. Note that the external SLEEP\_EN input has no affect on the clock required status output.

# MEC1618/MEC1618i

**Note 35-3** the SCAN ACTIVE mode in [Table 35-2](#) depends upon access to the Keyboard Scan Interface registers and scan-related operations involving transitions on the KSO pins.

## 35.6 Pins and I/O Buffers

26 pins are connected to the keyboard matrix. All are multiplexed with GPIOs and are configured using GPIO pin control registers.

Row outputs: KSO[17:0] - 18 tri-state open-drain outputs; output well powered by VTR. When the block is disabled (KSEN = 1), KSO output buffers are tri-stated.

Column inputs: KSI[7:0] - 8 Schmitt trigger inputs with internal pull up; input well powered by VTR.

### 35.6.1 CONFIGURATION OF I/O BUFFERS (INFORMATIONAL)

For KSO[17:0],

GPIO MUX control bits are set to select KSO function.

Buffer type = open drain (KSOs from the Key Scan block can be used as output buffer enables; buffers are disabled if the block is disabled, i.e., when KSEN = 1).

Pull-up (if no external pull ups) or none (if external pull ups)

For KSI[7:0],

I/O buffers can be configured either as signal function (KSI) or GPIO inputs. When the latter is selected, GPIO interrupts can be used in place of the block's interrupts.

### 35.6.2 PRE DRIVE MODE

There is an optional [Pre Drive Mode](#) that can be enabled to actively drive the KSO pins high before switching to open-drain operation. The active [Pre Drive Mode](#) timing is shown in [Table 35-3](#).

**TABLE 35-3: ACTIVE PRE DRIVE MODE TIMING**

	Parameter	Symbol	Value			Units	Notes
			Min	Typ	Max		
1.	Active Pre Drive Mode	$t_{\text{PREDRIVE}}$	96.2	98.7	101.1	ns	

The PREDRIVE ENABLE bit in the [Keyscan Extended Control Register](#) is used to support the PREDRIVE option.

There are specific Key Scan Interface [Pin Configuration Requirements](#) to support the PREDRIVE option.

#### 35.6.2.1 Pin Configuration Requirements

When the PREDRIVE ENABLE bit in the [Keyscan Extended Control Register](#) is not asserted ('0' default), the KSO pins must be configured as open-drain drivers using the GPIO Pin Control Registers.

When the PREDRIVE ENABLE bit in the [Keyscan Extended Control Register](#) is asserted ('1'), the KSO pins must be configured as push-pull drivers using the GPIO Pin Control Registers.

#### 35.6.2.2 Predrive Mode Programming

The following precautions should be taken to prevent output pad damage during [Predrive Mode Programming](#).

#### 35.6.2.3 Asserting PREDRIVE ENABLE

1. Disable Key Scan Interface (KSEN = '1')
2. Enable Predrive function (PREDRIVE ENABLE = '1')
3. Program buffer type for all KSO pins to "push-pull"
4. Enable Keyscan Interface (KSEN = '0')



## 35.6.2.4 De-asserting PREDRIVE ENABLE

1. Disable Key Scan Interface (KSEN = '1')
2. Program buffer type for all KSO pins to "open-drain"
3. Disable Predrive function (PREDRIVE ENABLE = '0')
4. Enable Keyscan Interface (KSEN = '0')

## 35.7 Operation (Informational)

During scanning the firmware sequentially drives low one of the rows (KSO[17:0]) and then reads the column data line (KSI[7:0]). A key press is detected as a zero in the corresponding position in the matrix. Keys that are pressed are debounced by firmware. After being confirmed, the corresponding keycode is loaded into host data read buffer in the 8042 Host Interface module. Firmware may need to buffer keycodes in memory in case this interface is stalled or the host requests a Resend.

### 35.7.1 INTERRUPT GENERATION

To support interrupt-based processing, interrupt can optionally be generated on the high-to-low transition on any of the KSI inputs. Interrupts are to be registered without a running clock.

#### 35.7.1.1 Runtime Interrupt

KSC\_INT output port is the block's runtime active-high level interrupt. It is connected to the interrupt interface of the Interrupt Aggregator, which then relays interrupts to the EC.

Associated with each KSI input are a status register bit and an interrupt enable register bit. A status bit is set when the associated KSI input goes from high to low. If the interrupt enable bit for that input is set, an interrupt is generated. Interrupt is de-asserted when the status bit and/or interrupt enable bit is clear. A status bit cleared by writing '1' to it.

Interrupts from individual KSIs are logically ORed together to drive the KSC\_INT output port. After being asserted, interrupt is not asserted again until either all KSI[7:0] have returned high or the [KSO Driver Select\[4:0\]](#) has changed.

#### 35.7.1.2 Wake-up Interrupt

KSC\_WAKEUP\_INT is the block's wakeup interrupt. It is routed to of the Interrupt Aggregator, [KEYSCAN](#) bit of the [GIRQ18 Source Register](#).

During sleep mode, i.e., when the bus clock is stopped, a high-to-low transition on any KSI whose interrupt enable bit is set causes the KSC\_WAKEUP\_INT to be asserted. Also set is the associated status bit in the [EC Blocks Clock Required Status Register 2](#). KSC\_WAKEUP\_INT remains active until the bus clock is started.

The aforementioned transition on KSI also sets corresponding status bit in the [KSI Status Register](#). If enabled, a runtime interrupt is also asserted on KSC\_INT when the bus clock resumes running.

### 35.7.2 WAKE PROGRAMMING

Using the Keyboard Scan Interface to 'wake' the MEC1618/MEC1618i can be accomplished using either the Keyboard Scan Interface wake interrupt, or using the wake capabilities of the GPIO Interface pins that are multiplexed with the Keyboard Scan Interface pins.

Using the GPIO Interface for this purpose is the most complicated of the two methods because a minimum of seven pin control registers, and an interrupt enable register must be programmed for 'wake' functionality before the system can sleep. Enabling the Keyboard Scan Interface wake interrupt requires only a single interrupt enable access and is recommended over using the GPIO Interface for this purpose.

# MEC1618/MEC1618i

## 35.8 Registers

### 35.8.1 REGISTERS SUMMARY

The Keyboard Scan Interface is assigned EC Logical Device Number 8h, which has AHB base address F0\_2000h.

**TABLE 35-4: KSC REGISTER SUMMARY**

Base Address: F0_2000h	EC Interface			Notes
Register Name	SPB Offset	Byte Lane	EC Type	
Reserved	0h	3-0	R	
<a href="#">KSO Select Register</a>	4h	0	R/W	
<a href="#">KSI Input Register</a>	8h	0	R	
<a href="#">KSI Status Register</a>	Ch	0	R/WC	
<a href="#">KSI Interrupt Enable Register</a>	10h	0	R/W	
<a href="#">Keyscan Extended Control Register</a>	14h	0	R/W	

### 35.8.2 KSO SELECT REGISTER

**TABLE 35-5: KSO SELECT REGISTER**

HOST OFFSET	N/A			N/A			HOST SIZE	
EC OFFSET	4h (R/W)				32-bit		EC SIZE	
POWER	VTR			0...040h			VTR POR DEFAULT	
	EC SPB							
BIT	D31	D30	D29	.....		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	-	-	-	-	-	-	-	-
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	KSO INVERT	KSEN	KSO ALL	KSO Driver Select[4:0]				

## KSO INVERT

**KSO INVERT** = 1 inverts KSO[17:0]. When **KSO INVERT** = 0 KSO[17:0] operate normally See Table 35-7, "Keyboard Scan Out Control Summary," on page 564.

## KSEN

**KSEN** = 1 disables keyboard drives. See first row in Table 35-7. **KSEN** = 0 enables keyboard scan

## KSO ALL

**KSO ALL** = 1, drives all KSO lines according to **KSO INVERT** bit. See Table 35-7, "Keyboard Scan Out Control Summary," on page 564.

## KSO DRIVER SELECT[4:0]

**KSO Driver Select[4:0]** controls the corresponding KSO line (00000b = KSO[0] etc.) according to **KSO INVERT**. See Table 35-6, "KSO Select Decode".

**TABLE 35-6: KSO SELECT DECODE**

KSO Select [4:0]	KSO Selected
00h	KSO00
01h	KSO01
02h	KSO02
03h	KSO03
04h	KSO04
05h	KSO05
06h	KSO06
07h	KSO07
08h	KSO08
09h	KSO09
0Ah	KSO10
0Bh	KSO11
0Ch	KSO12
0Dh	KSO13
0Eh	KSO14
0Fh	KSO15
10h	KSO16
11h	KSO17

# MEC1618/MEC1618i

**TABLE 35-7: KEYBOARD SCAN OUT CONTROL SUMMARY**

D7 KSO Invert	D6 KSEN	D5 KSO ALL	D[4:0] KSO Drivers Address	Description
X	1	x	x	Keyboard Scan disabled. KSO[17:0] output buffers disabled.
0	0	0	10001b-00000b	KSO[Drive Selected] driven low. All others driven high
1	0	0	10001b-00000b	KSO[Drive Selected] driven high. All others driven low
0	0	0	11111b-10010b	ALL KSO's driven high
1	0	0	11111b-10010b	All KSO's driven low
0	0	1	x	KSO[17:0] driven low
1	0	1	x	KSO[17:0] driven high

## 35.8.3 KSI INPUT REGISTER

**TABLE 35-8: KSI INPUT REGISTER**

BUS OFFSET	N/A			N/A			HOST SIZE	
EC OFFSET	8h (R)				32-bit		EC SIZE	
POWER	VTR			0...00h			VTR POR DEFAULT	
	EC SPB							
BIT	D31	D30	D29	.....		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	-	-	-	-	-	-	-	-
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	KS7	KS6	KS5	KS4	KS3	KS2	KS1	KS0

## 35.8.4 KSI STATUS REGISTER

**TABLE 35-9: KSI STATUS REGISTER**

HOST BUS OFFSET	N/A			N/A			HOST SIZE	
EC OFFSET	Ch (R/WC)				32-bit		HOST SIZE	
POWER	VTR			0...00h			VTR POR DEFAULT	
	EC SPB							
BIT	D31	D30	D29	.....		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	-	-	-	-	-	-	-	-
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC	R/WC
BIT NAME	KSI[7] STAT	KSI[6] STAT	KSI[5] STAT	KSI[4] STAT	KSI[3] STAT	KSI[2] STAT	KSI[1] STAT	KSI[0] STAT

### 35.8.4.1 Bit Definition

KSI[x] STAT bit is set on the falling edge of the corresponding KSI input. Writing a 1 to a bit will clear it.

A KSI interrupt is generated when its corresponding status bit and interrupt enable bit are both set. KSI interrupts are logically ORed together to produce KSC\_INT and KSC\_WAKEUP\_INT.

# MEC1618/MEC1618i

## 35.8.5 KSI INTERRUPT ENABLE REGISTER

**TABLE 35-10: KSI INTERRUPT ENABLE REGISTER**

HOST BUS OFFSET	N/A			N/A			HOST SIZE	
EC OFFSET	10h (R/W)				32-bit		HOST SIZE	
POWER	VTR			0...00h			VTR POR DEFAULT	
	EC SPB							
BIT	D31	D30	D29	.....		D10	D9	D8
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R	R	R	R	R	R	R	R
BIT NAME	-	-	-	-	-	-	-	-
BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	KSI[7] IntEn	KSI[6] IntEn	KSI[5] IntEn	KSI[4] IntEn	KSI[3] IntEn	KSI[2] IntEn	KSI[1] IntEn	KSI[0] IntEn

### 35.8.5.1 Bit Definition

KSI[x] IntEn enables interrupt generation due to high-to-low transition on KSI[x] input. An interrupt is generated only when both KSI[x] IntEn and KSI[x] STAT bits are set.

## 35.8.6 KEYSKAN EXTENDED CONTROL REGISTER

OFFSET	14h			
BITS	DESCRIPTION	TYPE	DEFAULT	RESET EVENT
32:1	RESERVED	RES	0	—
0	PREDRIVE ENABLE (KSO_PREDRIVE_EN)	RW	0	nSYS_RST

### 35.8.6.1 Bit Definition

KSO\_PREDRIVE\_EN bit enables the PREDRIVE mode to actively drive the KSO pins high before switching to open-drain operation (see [Table 35-3, “Active Pre Drive Mode Timing,” on page 560](#)). 0= disable, 1=enable.

## 36.0 BC-LINK MASTER

### 36.1 General Description

The function of this block is to provide BC-Link to a slave device. The BC-Link protocol includes a start bit to signal the beginning of a message and a turnaround (TAR) period for bus transfer between the Master and Companion devices.

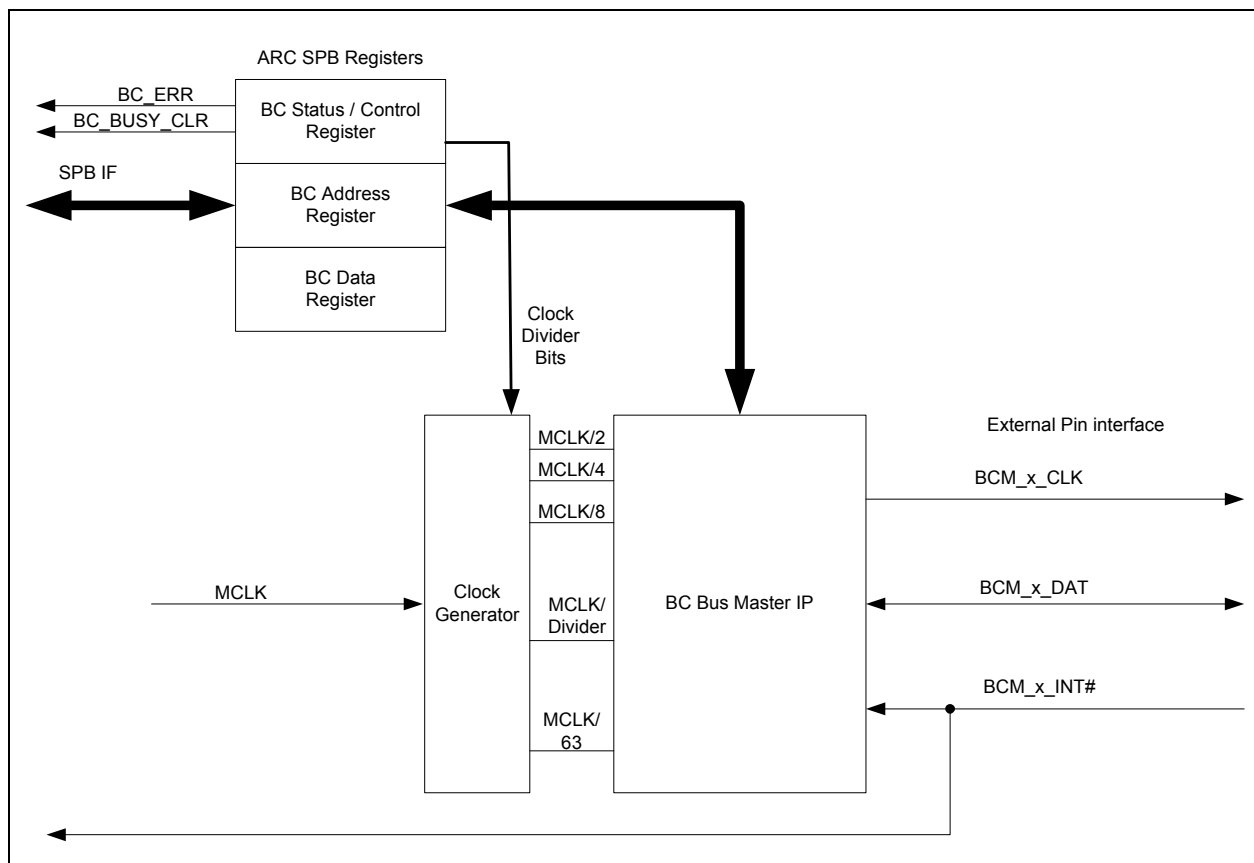
**Note:** A weak pull-up resistor is recommended on the data line (100K $\Omega$ ).

There are three instances of the BC-Link Master interface in the MEC1618/MEC1618i. Instance A and B are high speed BC-links with high speed buffers attached. Instance D is a low speed BC-Link which uses 8 mA buffer. Instance D consists of signals LSBCM\_D\_INT#, LSBCM\_D\_DAT, LSBCM\_D\_CLK.

The maximum usable clock frequencies are described in [Note 36-1 on page 569](#).

### 36.2 Block Diagram

**FIGURE 36-1: BC-LINK MASTER BLOCK DIAGRAM**



## 36.3 Signal List

TABLE 36-1: BC-LINK SIGNAL LIST

Signal Name	Direction	Description
BCM_x_CLK	OUTPUT	20.27MHz - 79.18kHz output clock, where x is A, B or D.
BCM_x_DAT	INPUT/OUTPUT	Bidirectional data line, where x is A, B or D.
BCM_x_INT#	INPUT	Input from the companion device, where x is A, B or D.
BC_x_ERR	OUTPUT	BC-Link master error interrupt
BC_x_BUSY_CLR	OUTPUT	BC-Link master Busy Clear interrupt
MCLK	INPUT	20.27MHz Master clock
SPB	I/O Bus	Bus used for register access
CLK_REQ	OUTPUT	<a href="#">Power Management</a> clock required output.

**Note:** The Low Speed BC-Link Master Block has an identical organization, but the external signals are LSBCM\_D\_CLK, LSBCM\_D\_DAT and LSBCM\_D\_INT#.

See [Table 2-8, "BC-Link Interface," on page 15](#) for the pin interface and associated [Note 3.1](#) & [Note 2-2 on page 15](#).

## 36.4 Power, Clocks and Reset

### 36.4.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.5, "Power Configuration," on page 101](#) for details on power domains.

### 36.4.2 CLOCKS

This block uses the [EC Bus Clock](#) and the [MCLK](#). The [EC Bus Clock](#) is used to access the [Registers](#) described in this block. [MCLK](#) is divided down to generate the external bus clock.

See [Section 5.4, "Clock Generator," on page 77](#) for details on clocks.

### 36.4.3 POWER ON RESET

This block is reset on a [nSYS\\_RST](#). On reset, the BC-Link state machine transitions to the Idle state and waits for the address and data registers to be written.

See [Section 5.6, "Reset Interface," on page 104](#) for details on reset.

## 36.5 Interrupts

Each [BC-Link Master](#) instance has three interrupts events: [BC\\_BUSY\\_CLR](#), [BC\\_ERR](#), & [BC\\_INT#](#). The [BC-Link Master](#) [BC\\_BUSY\\_CLR](#) and [BC\\_ERR](#) interrupts are generated by changes in the [BC-Link Status Register](#). The [BC\\_INT#](#) is an active low level interrupt generated by input pin signal function. The edge detection of the interrupt and wake events are controlled by their associated pin control registers in the [Section 21.0, "GPIO Interface," on page 398](#).

The [BC-Link Master](#) block Instance A are routed to the [BCM\\_BUSY\\_CLR\[A\]](#), [BCM\\_ERR\[A\]](#), & [BCM\\_INT#\[A\]](#) bits in the [GIRQ21 Source Register on page 339](#). The [BC-Link Master](#) block Instance B are routed to the [BCM\\_BUSY\\_CLR\[B\]](#), [BCM\\_ERR\[B\]](#), & [BCM\\_INT#\[B\]](#) bits in the [GIRQ21 Source Register on page 339](#). The [BC-Link Master](#) block Instance D are routed to the [BCM\\_BUSY\\_CLR\[D\]](#), [BCM\\_ERR\[D\]](#), & [BCM\\_INT#\[D\]](#) bits in the [GIRQ21 Source Register on page 339](#).



## 36.5.1 INSTANCE BUFFERS VS. CLOCK

There are three instances of the [BC-Link Master](#) block implemented in the MEC1618/MEC1618i enumerated as A, B and D. [Table 2-8, "BC-Link Interface," on page 15](#) lists the pin functions for all instances. The base addresses for all instances are listed in [Table 36-3, "BC-Link Master Base Address Table"](#).

Instance A and B are high speed BC-links with higher current buffers driving the external signals. Instance D is a low speed BC-Link, which is implemented lower current buffers. Signal names for the high speed interfaces are BCM[A,B]\_CLK, BCM[A,B]\_DAT and BCM[A,B]\_INT#. Signal names for the low speed instance are LSBCM\_D\_CLK, LSBCM\_D\_DAT and LSBCM\_D\_INT#.

**Note 36-1** For ribbon cable applications, the Low Speed BC-Link Master maximum clock frequency and the High Speed BC-Link Master maximum clock frequency are shown in [Table 36-2](#). The Clock frequency is set with the [BC-Link Clock Select Register](#). See also [Note 43-4 on page 648](#).

**TABLE 36-2: BC-Link Master PIN INTERFACE**

Pin	Description	Buffer	Max Freq	Min Value in BC-Link Clock Select Register
BCM[A,B]_CLK	High Speed Clock supplied by the MEC1618/MEC1618i Device	O16	20.27 Mhz	0
BCM[A,B]_DAT	High Speed Data Line	IO16 <a href="#">Note 36-2</a>		
BCM[A,B]_INT#	Interrupt signal	I		
LSBCM_D_CLK	Low Speed Clock supplied by the MEC1618/MEC1618i Device	O8	2.9 MHz	6
LSBCM_D_DAT	Data Line	IO8 <a href="#">Note 36-2</a>		
LSBCM_D_INT	Interrupt signal	I		

**Note 36-2** BCM[A,B]\_DAT & LSBCM\_D\_DAT pins requires a weak pull up (100K).

## 36.6 Registers

**TABLE 36-3: BC-Link Master BASE ADDRESS TABLE**

BC-Link Instances	LDN from ( <a href="#">Table 3-2 on page 51</a> )	AHB Base Address
BC-LINK.A	5h	<a href="#">F0_1400h</a>
BC-LINK.B		F0_1480h = <a href="#">F0_1400h</a> + 80h
BC-Link.D		F0_1580h = <a href="#">F0_1400h</a> + 180h

# MEC1618/MEC1618i

TABLE 36-4: BC-Link Master REGISTER SUMMARY

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
BC-Link Status Register	00h	0	R/W	
BC-Link Address Register	04h	0	R/W	
BC-Link Data Register	08h	0	R/W	
BC-Link Clock Select Register	0Ch	0	R/W	

## 36.6.1 BC-LINK STATUS

TABLE 36-5: BC-LINK STATUS REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	00h			8-bit			EC SIZE	
POWER	VTR			81h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/WC	R/W	R/W	R	R	R	R
BIT NAME	RESET	BC_ERR	BC_ERR_INT_EN	BC_Busy_CLR_INT_EN	Reserved			BUSY

### BUSY

This bit is asserted to '1' when the BC interface is transferring data and on reset. Otherwise it is cleared to '0'. When **BUSY** bit is cleared by hardware, an interrupt is generated if the **BC\_Busy\_CLR\_INT\_EN** bit is set to '1'.

### BC\_BUSY\_CLR\_INT\_EN

This bit is an enable for generating an interrupt when the **BUSY** bit is cleared by hardware. When the **BC\_Busy\_CLR\_INT\_EN** bit is set to '1', the interrupt signal is enabled. When the **BC\_Busy\_CLR\_INT\_EN** bit is cleared to '0', the interrupt is disabled. When enabled this interrupt occurs after a BC Bus read or write. [Figure 36-2](#) shows when the interrupt is generated.

### BC\_ERR\_INT\_EN

This bit is an enable for generating an interrupt when the **BC\_ERR** bit is set by hardware. When the **BC\_ERR\_INT\_EN** bit is '1', the interrupt signal is enabled. When the **BC\_ERR\_INT\_EN** bit is '0', the interrupt is disabled.

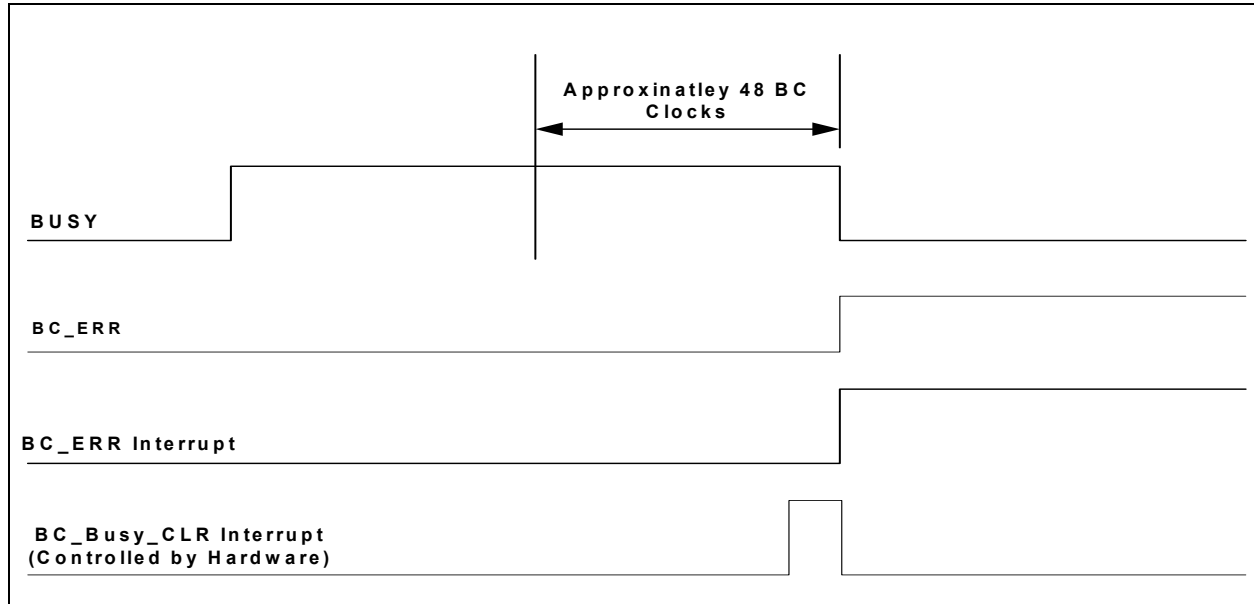
## BC\_ERR

This bit indicates that a BC Bus Error has occurred. If an error occurs the **BC\_ERR** bit is set by hardware when the **BUSY** bit is cleared. See [Figure 36-2](#) for relative timing. When **BC\_ERR** bit is set by hardware, an interrupt is generated if the **BC\_ERR\_INT\_EN** bit is set to '1'.

This **BC\_ERR** bit is cleared to '0', by software writing a '1' to this bit.

Errors that cause this interrupt are: Bad Data received by the BASE (CRC Error) or a time-out caused by the COMPANION not responding. All COMPANION errors cause the COMPANION to abort the operation and cause the BASE to time-out. [Figure 36-2](#) shows the timing of this interrupt.

**FIGURE 36-2: BC BUS BC\_ERR INTERRUPT TIMING**



## RESET

When set to '1', the **BC-Link Master** Interface will be placed in reset and be held in reset until this bit is de-asserted. Reset causes the **BUSY** bit to be set and will not be cleared until the reset operation of the BC Interface is completed (approximately 48 BC clocks).

**PROGRAMMER'S NOTE:** The de-assertion of the **BUSY** bit on reset will not generate an interrupt, even if the **BC\_Busy\_CLR\_INT\_EN** is asserted 1. The **BUSY** bit must be polled.

# MEC1618/MEC1618i

## 36.6.2 BC-LINK ADDRESS

TABLE 36-6: BC-LINK ADDRESS REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	04h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Address[7:0]							

## 36.6.3 BC-LINK DATA

TABLE 36-7: BC-LINK DATA REGISTER

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	08h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Data[7:0]							

## 36.6.4 BC CLOCK SELECT

**TABLE 36-8: BC-LINK CLOCK SELECT REGISTER**

HOST ADDRESS	N/A						HOST SIZE	
EC OFFSET	0Ch			8-bit			EC SIZE	
POWER	VTR			4			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE3 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	-	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Divider[7:0]							

### DIVIDER

The BC Clock is set to the Master Clock divided by this field, or  $MCLK / (Divider[7:0] + 1)$ , Where Divider[7:0] is 0 to 255. The clock divider bits can only be changed when the BC Bus is in soft RESET (when either the [Reset](#) bit is set by software or when the [BUSY](#) Bit is set by the interface).

Example settings are shown in [Table 36-9, "Example Frequency Settings"](#):

**TABLE 36-9: EXAMPLE FREQUENCY SETTINGS**

Divider	Frequency
0	20.27MHz
1	10.13MHz
2	6.756MHz
3	5.067MHz
4	4.054MHz
15	1.266MHz
21	0.921MHz
2A	0.471MHz
63	0.316MHz

## 36.7 BC-Link Master Operations

Descriptions of the BC-Link read and write operations follows:

### 36.7.1 READ

The BC-Link Read protocol requires two reads of the [BC-Link Data Register](#). The two reads drive a two state-state machine: the two states are Read#1 and Read#2. The Read#1 of the [BC-Link Data Register](#) starts the read protocol on the BC-Link pins and sets the Busy bit in the [BC-Link Status Register](#). The contents of the data read during Read#1 by the EC is stale and is not to be used. After the Busy bit in the BC-Link Status Register autonomously clears to '0', the Read#2 of the [BC-Link Data Register](#) transfers the data read from the peripheral/BC-Link companion chip to the EC.

1. Software starts by checking the status of the [BUSY](#) bit in the [BC-Link Status Register on page 570](#). If the Busy bit is 0, proceed, if Busy is a 1 Wait.
2. Software writes the address of the register to be read into the [BC-Link Address Register on page 572](#).
3. Software then reads the [BC-Link Data Register on page 572](#). This read returns random data. The read activates the [BC-Link Master](#) to transmit the read request packet to the BC-Link companion. When the transfer initiates, the hardware sets the [BUSY](#) bit to a 1.
4. The BC-Link companion reads the selected register and transmits the read response packet to the [BC-Link Master](#).

**Note 36-3** The companion will ignore the read request if there is a CRC error, this will cause the base to time-out and issue a [BC\\_ERR](#) Interrupt.

5. The [BC-Link Master](#) loads the [BC-Link Data Register on page 572](#) issues a BUSY Bit Clear interrupt and clears the busy bit to '0'.
6. Check the [BC\\_ERR](#) bit in the [BC-Link Status Register on page 570](#).
7. Software can now read the [BC-Link Data Register on page 572](#) which contains the valid data if there was no BC Bus error.
8. If a Bus Error occurs issue a soft reset by setting the [Reset](#) bit in the [BC-Link Status Register on page 570](#).
9. The read can be re-tried after [BUSY](#) is cleared.

**PROGRAMMER'S NOTE:** Steps 3 thorough 7 should be completed as a contiguous sequence. If not the LSBC interface could be presenting incorrect data when software thinks it is accessing a valid register read.

### 36.7.2 WRITE

1. Software starts by checking the status of the [BUSY](#) bit in the [BC-Link Status Register on page 570](#). If the [BUSY](#) bit is 0, proceed, if [BUSY](#) is a 1 Wait.
2. Software writes the address of the register to be written into the [BC-Link Address Register on page 572](#). Then writes the data to be written into the addressed register in to the [BC-Link Data Register on page 572](#).
3. The write to the [BC-Link Data Register on page 572](#) starts the BC\_Link write operation. The [BC-Link Master](#) sets the [BUSY](#) bit in the [BC-Link Status Register on page 570](#).
4. The [BC-Link Master](#) Interface transmits the write request packet.
5. When the write request packet is received by the BC-Link companion, the CRC is checked and data is written to the addressed companion register.
6. The companion sends an ACK if the write is completed.

**Note 36-4** A time-out will occur approximately 16 BC-Link clocks after the packet is sent by the [BC-Link Master](#). The [BC-Link Master](#) will issue a [BC\\_ERR](#) bit in the [BC-Link Status Register on page 570](#) approximately 48 clocks later. and clear the [BUSY](#) bit.

7. The [BC-Link Master](#) issues the Busy bit Clear interrupt and clears the [BUSY](#) bit after receiving the ACK from the companion
8. If a Bus Error occurs issue a soft reset by setting the [Reset](#) bit in the [BC-Link Status Register on page 570](#).
9. The write can be re-tried after [BUSY](#) is cleared.

### 36.8 Power Management

BC-Link Master Power Management is illustrated in Table 36-10. Note that the BC-Link Master does not include a sleep enable input.

**TABLE 36-10:** BC-Link Master Power Management

Block Enable (Note 36-5)	Block Idle Status (Note 36-6)	Clock Required Status Output (CLK_REQ)	State	Description
DISABLED	X	0	DISABLED	Block is disabled by firmware and the core clock is not needed.
ENABLED	NOT IDLE	1	NORMAL OPERATION	The block is enabled and performing a transaction.
	IDLE	0	SLEEPING	The block is enabled and not performing a transaction. The clock may be stopped.

**Note 36-5** The DISABLED state in Table 36-10 is defined as the Reset Bit being set to '1' and the & BUSY bit being cleared to '0' in the BC-Link Status Register.

**Note 36-6** The IDLE state in Table 36-10 is defined as both the Reset & BUSY Bit being cleared to '0' in the BC-Link Status Register.

## 37.0 BIOS DEBUG PORT

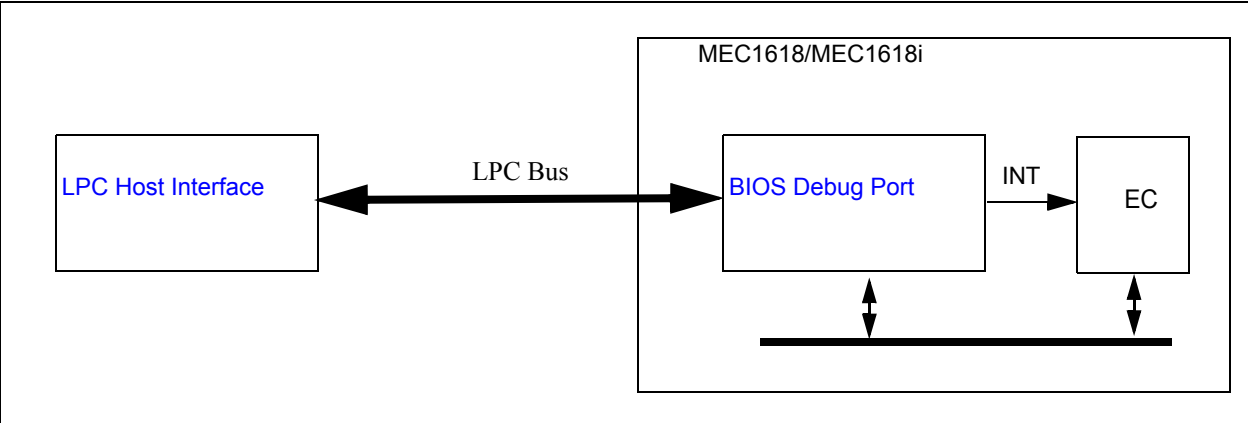
### 37.1 Overview

The [BIOS Debug Port](#) emulates the functionality of a “Port 80” ISA plug-in card. In addition, a timestamp for the debug data can be optionally added.

Diagnostic data is written by the [LPC Host Interface](#) to the [BIOS Debug Port](#), which is located in LPC I/O address space ([Figure 37-1](#)). The [BIOS Debug Port](#) generates [Interrupts](#) to the EC when host data is available. The EC reads this data along with the timestamp, if enabled.

The MEC1618/MEC1618i includes two instances of the [BIOS Debug Port](#).

**FIGURE 37-1: BIOS DEBUG PORT SYSTEM VIEW**



### 37.2 References

No references have been cited for this chapter

### 37.3 Terminology

**TABLE 37-1: TERMINOLOGY**

Term	Definition
–	–



## 37.4 Interface

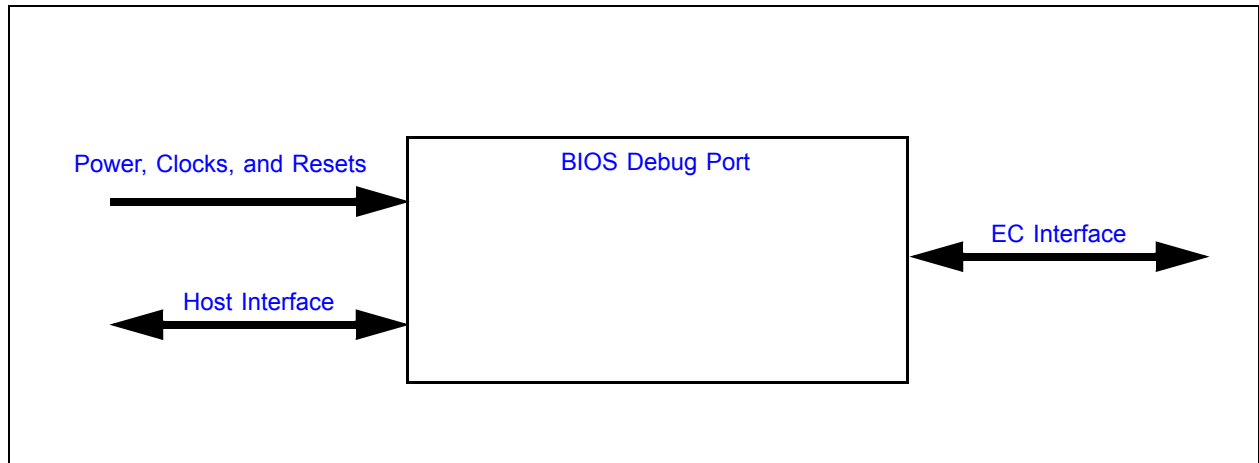
### 37.4.1 HOST INTERFACE

The host communicates to the [BIOS Debug Port](#) using the [LPC Host Interface](#) (see [Section 37.6, "LPC Host Interface," on page 578](#)).

### 37.4.2 EC INTERFACE

The EC communicates to the [BIOS Debug Port](#) using an internal bus interface and includes [Interrupts](#) as defined in [Section 37.7, "Interrupts," on page 578](#).

**FIGURE 37-2: EC INTERFACE TO BIOS DEBUG PORT**



### 37.4.3 POWER, CLOCKS, AND RESETS

**TABLE 37-2: POWER PLANES**

Name	Description
<a href="#">VTR</a>	The <a href="#">BIOS Debug Port</a> requires only a single power plane.

**TABLE 37-3: CLOCKS**

Name	Description
<a href="#">MCLK</a>	<a href="#">BIOS Debug Port</a> core clock.

**TABLE 37-4: RESETS**

Name	Description
<a href="#">nSYS_RST</a>	<a href="#">BIOS Debug Port</a> hardware reset.

# MEC1618/MEC1618i

## 37.5 Functional Description

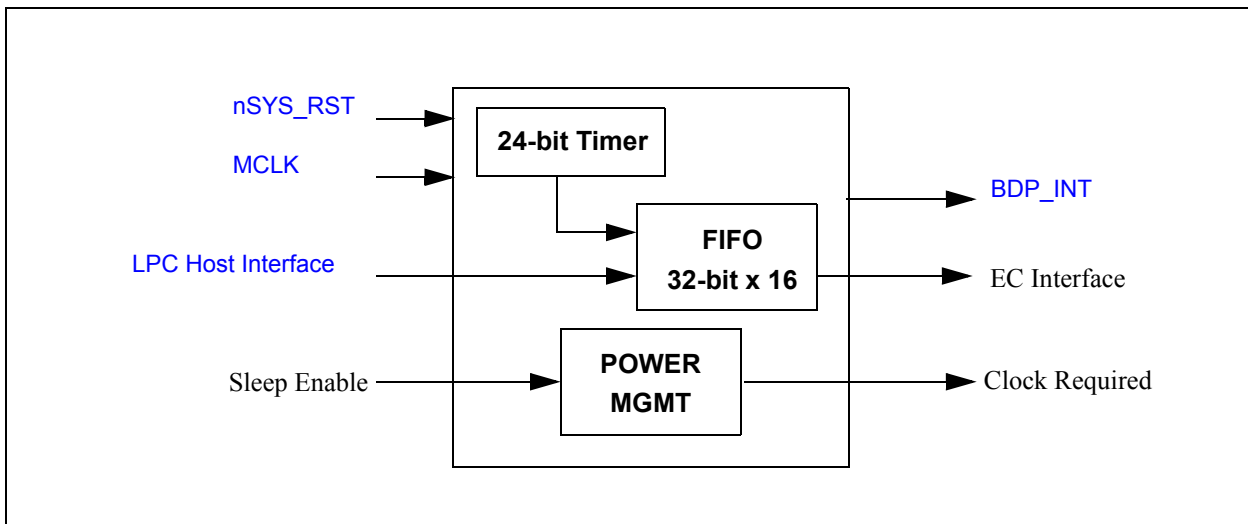
The BIOS Debug Port consists of a 32-bit wide x 16 deep FIFO, 24-bit free running timer, and Power Management Interface (Figure 37-3). Host and EC access to the BIOS Debug Port is through Registers as defined in Section 37.9, "Registers," on page 580.

The BIOS Debug Port Configuration Register and Count Register are used to control the FIFO and the 24-bit timer, as defined in Section 37.9.4, "Configuration Register," on page 582 and Section 37.9.6, "Count Register," on page 583.

Writes to the BIOS Debug Port Host Data Register are concatenated with the 24-bit timestamp and written to the FIFO. Reads of the Host Data Register return zero. If writes to the Host Data Register overrun the FIFO, the oldest data are discarded and the OVERRUN status bit in the Status Register is asserted.

Only the EC can read data from the FIFO, using the EC Data Register. The use of this data is determined by EC Firmware alone.

FIGURE 37-3: BIOS Debug Port BLOCK DIAGRAM



## 37.6 LPC Host Interface

Each BIOS Debug Port instance is an LPC logical device and occupies one byte in the LPC I/O Address Space (Host Data Register). The LPC Host Interface must initialize each BIOS Debug Port base address using the ISA Plug and Play Base Address Registers (BAR), and enable each instance using the Activate Register. The BIOS Debug Port can be mapped to any 16-bit LPC I/O address value.

## 37.7 Interrupts

**Note:** BDP0\_INT and BDP1\_INT in the GIRQ15 Source Register are the interrupt source bits for the two instances of the BIOS Debug Port in the MEC1618/MEC1618i (see Section 15.0, "EC Interrupt Aggregator," on page 296).

TABLE 37-5: INTERRUPTS

Name	Description
BDP_INT	The BIOS Debug Port generates an edge-sensitive interrupt when the amount of data in the FIFO equals or exceeds the FIFO THRESHOLD (see Section 37.9.4, "Configuration Register," on page 582). Note that there is no enable bit for this interrupt in the BIOS Debug Port Registers.

## 37.8 Power Management Interface

**Note:** BDP0 and BDP1 in the LPC Blocks Clock Required Status Register are the clock required status bits for the two instances of the BIOS Debug Port in the MEC1618/MEC1618i (see Section 5.0, "Power, Clocks, and Resets," on page 74).

**TABLE 37-6:** BIOS Debug Port Power Management Interface

Activate (Note 37-1)	External SLEEP_EN Input (Note 37-2)	Timer Active (Note 37-3)	Core Clock Required Status Output (Note 37-2)	Mode	Power	Description
0	X	X	0	DISABLED	MINIMUM	The BIOS Debug Port is disabled by the LPC Host and the core clock is gated 'off' internally.
1	X	YES	1	FULL POWER	MAXIMUM	FULL POWER mode identifies that the BIOS Debug Port is enabled by the LPC Host and the 24-bit Timer is active.
		NO	0	SLEEPING	MINIMUM	In SLEEPING mode, there will be no clocks to the BIOS Debug Port except during access to the Registers.

**Note 37-1** The ACTIVATE bit is in the Activate Register in LPC Configuration Space. When the ACTIVATE bit is '0', there will be no clocks active in the BIOS Debug Port except during access to the Registers.

**Note 37-2** The external sleep enable input and clock required status output are part of the chip-level clocking interface. Note that the external SLEEP\_EN input has no affect on the clock required status output.

**Note 37-3** The TIMER ACTIVE mode in Table 37-6 depends upon the state of the TIMER ENABLE bit in the Configuration Register.

# MEC1618/MEC1618i

## 37.9 Registers

### 37.9.1 SUMMARY

Each [BIOS Debug Port](#) instance includes an ACTIVATE bit and Base Address Register (BAR), accessible by the [LPC Host Interface](#) in LPC Configuration space.

When the ACTIVATE bit is asserted '1', the [BIOS Debug Port](#) is enabled. When the ACTIVATE bit is '0', writes by the [LPC Host Interface](#) to the [Host Data Register](#) are not claimed, the FIFO is flushed, the 24-bit Timer is reset, and the timer clock is stopped ([Note 37-5](#)). See also [Section 37.8, "Power Management Interface," on page 579](#).

**TABLE 37-7: BIOS Debug Port BASE ADDRESS TABLE**

BIOS Debug Port Instances	LDN (Table 3-2, "Host Logical Devices in the MEC1618/MEC1618i," on page 51)	AHB Base Address
BIOS Debug Port 0	20h	FF_8000h
BIOS Debug Port 1	21h	FF_8400h

**TABLE 37-8: BIOS Debug Port Registers Summary**

Host Offset	EC Interface SPB Offset	Register Name	Host Access	EC Access
00h	00h	<a href="#">Host Data Register</a>	YES	YES
–	100h	<a href="#">EC Data Register</a>	NO	YES
–	104h	<a href="#">Configuration Register</a>	NO	YES
–	108h	<a href="#">Status Register</a>	NO	YES
–	10Ch	<a href="#">Count Register</a>	NO	YES

**TABLE 37-9: REGISTER BIT ACCESS TYPES**

Register Bit Type	Description
R	<b>Read:</b> A register or bit with this attribute can be read.
W	<b>Write:</b> A register or bit with this attribute can be written.
RO	<b>Read Only:</b> A register or bit with this attribute is read only, writes have no effect.
RS	<b>Read to Set:</b> A register or bit with this attribute is set on read.
RC	<b>Read to Clear:</b> A register or bit with this attribute is cleared after the read, writes have no effect.
WO	<b>Write Only:</b> A register or bit with this attribute is write only, reads return zero.
WC	<b>Write One to Clear:</b> Writing a one to a bit with this attribute clears ('0') the value, writing a zero has no effect.

TABLE 37-9: REGISTER BIT ACCESS TYPES (CONTINUED)

Register Bit Type	Description
WS	<b>Write One to Set:</b> Writing a one to a bit with this attribute sets the value to '1', writing a zero has no effect.
WZS	<b>Write Zero to Set:</b> Writing a zero to a bit with this attribute sets the value to '1', writing a one has no effect.
RES	<b>Reserved:</b> Reads of a register or bit with this attribute return zero, writes are ignored.

## 37.9.2 HOST DATA REGISTER

The [LPC Host Interface](#) and the EC writes to the [BIOS Debug Port](#) FIFO using the [Host Data Register](#) ([Note 37-4](#)).

Offset	00h			
Bits	Description	Type ( <a href="#">Note 37-4</a> )	Default	Reset Event
31:8	RESERVED	RES	0	—
7:0	HOST_DATA	WO	0	nSYS_RST

**Note 37-4** Reads of the [Host Data Register](#) return 00h.

## 37.9.3 EC DATA REGISTER

The EC reads the [BIOS Debug Port](#) FIFO using the [EC Data Register](#).

Offset	100h			
Bits	Description	Type	Default	Reset Event
31:8	TIME_STAMP	RO	0	nSYS_RST
7:0	EC_DATA	RO	0	nSYS_RST

# MEC1618/MEC1618i

## 37.9.4 CONFIGURATION REGISTER

The EC configures and controls the [BIOS Debug Port](#) using the [Configuration Register](#).

**TABLE 37-10:** [Configuration Register](#)

Offset	104h			
Bits	Description	Type	Default	Reset Event
31:8	RESERVED	RES	0	–
7:6	FIFO THRESHOLD The <a href="#">FIFO THRESHOLD</a> bits determine the threshold for the <a href="#">BIOS Debug Port Interrupts</a> , as defined in <a href="#">Table 37-11</a> .	RW	0	nSYS_RST
5	TIMER ENABLE (RUN_TSTAMP) When the TIMER ENABLE bit is '1', the 24-bit Timer is actively counting at a rate determined by the TIMEBASE SELECT bits. When the TIMER ENABLE bit is '0', counting is stopped ( <a href="#">Note 37-5</a> ).	RW	0	nSYS_RST
4:3	TIMEBASE SELECT (TSTAMP_CLK) The TIMEBASE SELECT bits determine the clock for the 24-bit Timer as defined in <a href="#">Table 37-12</a> ( <a href="#">Note 37-5</a> ).	RW	0	nSYS_RST
2	RESET TIMESTAMP (RST_TSTAMP) When the RESET TIMESTAMP bit is '1', the 24-bit Timer is reset to '0'. The RESET TIMESTAMP bit is self-clearing. Writing zero to the <a href="#">Count Register</a> has the same effect.	WO	–	nSYS_RST
1	FLUSH (FLUSH_FIFO) When the FLUSH bit is '1', the FIFO is flushed. The FLUSH bit is self-clearing.	WO	–	nSYS_RST
0	RESERVED	RES	0	–

**Note 37-5** The ACTIVATE bit does not affect the TIMER ENABLE bit or the TIMEBASE SELECT bits in the [Configuration Register](#).

**TABLE 37-11:** [FIFO THRESHOLD BITS](#)

Bit 7	Bit 6	FIFO Depth Interrupt Threshold
0	0	1 (default)
0	1	4
1	0	8
1	1	14

**TABLE 37-12: TIMEBASE SELECT BITS**

Bit 4	Bit 3	24-bit Timer Clock
0	0	MCLK/8
0	1	MCLK/16
1	0	MCLK/32
1	1	MCLK/64

## 37.9.5 STATUS REGISTER

Offset	108h			
Bits	Description	Type	Default	Reset Event
31:2	RESERVED	RES	0	–
1	OVERRUN (FIFO_OVERRUN) The OVERRUN bit is '1' when the host writes the <a href="#">Host Data Register</a> when the FIFO is full.	WC	0	nSYS_RST
0	NOT EMPTY (FIFO_NEMPTY) The NOT EMPTY bit is '1' when there is data in the FIFO. The NOT EMPTY bit is '0' when the FIFO is empty.	RO	0	nSYS_RST

## 37.9.6 COUNT REGISTER

Offset	10Ch			
Bits	Description	Type	Default	Reset Event
32:8	COUNT Writes load data into the 24-bit Timer. Reads return the 24-bit Timer current value.	RW	–	–
7:0	RESERVED	RES	0	–

## 38.0 SERIAL DEBUG PORT

### 38.1 General Description

The Serial Debug Port serially transmits MCU-originated diagnostic vectors to an external debug trace system.

The Serial Debug Port consists of the [Debug Data Register](#), [Debug Control Register](#), a Parallel-to-Serial Converter, a Clock/Control Interface and a two-pin external interface (Debug\_CLK, Debug\_DAT). See [Figure 38-1](#).

### 38.2 Power, Clocks and Reset

#### 38.2.1 POWER DOMAIN

This block is powered by the VTR Power Supply.

See [Section 5.1.1, "Power Configuration," on page 74](#) for details on power domains.

#### 38.2.2 CLOCKS

This block has one clock input, the [EC Bus Clock](#).

See [Section 5.1.2, "Clock Generator," on page 74](#) for details on clocks.

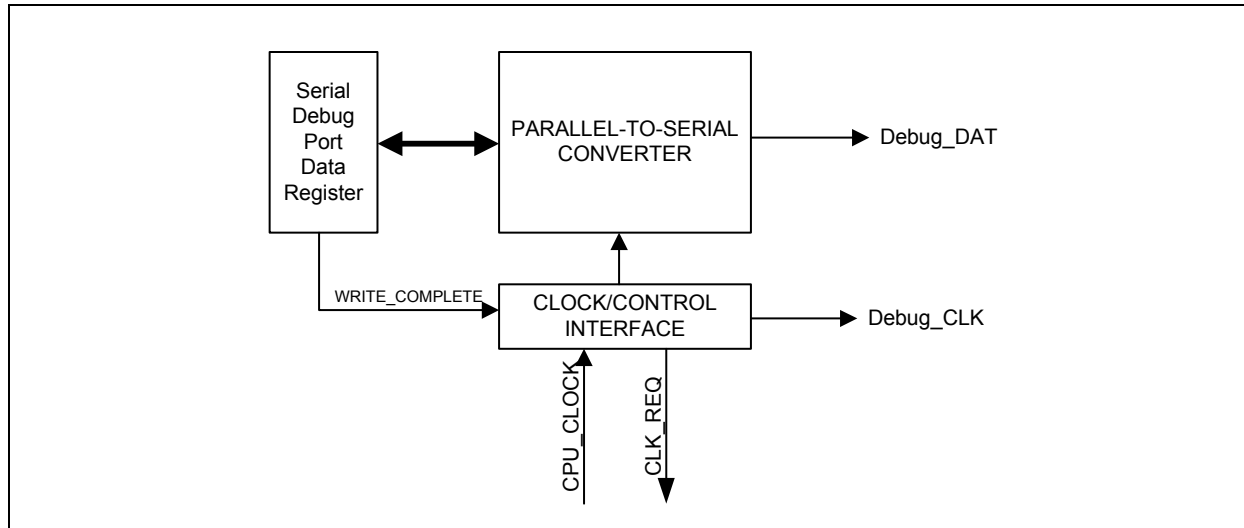
#### 38.2.3 RESET

This block is reset on a [nSYS\\_RST](#).

See [Section 5.1.3, "Reset Interface," on page 75](#) for details on reset.

### 38.3 Block Diagram

**FIGURE 38-1: SERIAL DEBUG PORT BLOCK DIAGRAM**





## 38.4 Block Diagram Port List

**TABLE 38-1:** Serial Debug Port LIST

Signal Name	Direction	Description
Debug Clock	OUTPUT	Derived from the <a href="#">EC Bus Clock</a>
Debug Data	OUTPUT	Serialized Data shifter out by the Debug Clock
<a href="#">EC Bus Clock</a>	INPUT	EC AHB Bus Clock
CLK_REQ	OUTPUT	<a href="#">Power Management</a> Clock Required output signal

## 38.5 Interrupts

There are no interrupts from this block.

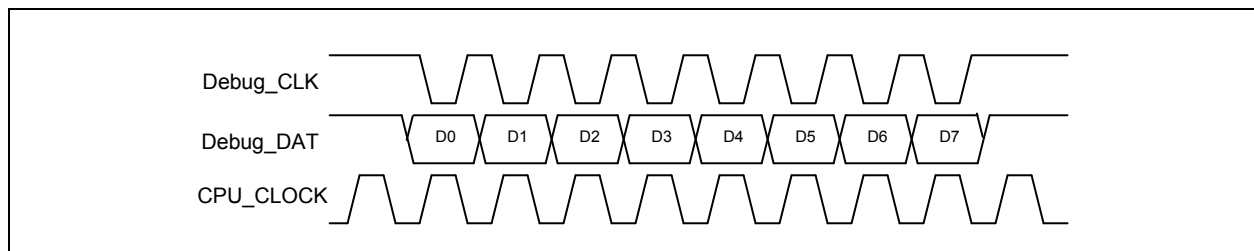
## 38.6 Functional Description

Writes to the [Debug Data Register](#) initiate an transfer cycle ([Figure 38-2](#)). Data from the [Debug Data Register](#) is shifted LSB first and is transmitted one byte per transfer cycle.

Data is transferred in one direction only from the [Debug Data Register](#) to the external interface. The data is shifted out on the clock edge selected by the [EDGE\\_SEL](#) bit in the [Debug Control Register on page 587](#). Valid data is guaranteed on opposite edge of Debug\_CLK after being shifted out. For example when the [EDGE\\_SEL](#) bit is '0' (default), Valid data is guaranteed on the falling edge of Debug\_CLK. The Setup Time to the falling edge of Debug\_CLK is 10ns minimum; the Hold Time is 1ns minimum.

The Debug\_CLK and Debug\_DAT outputs are '1' when the serial Debug Port is inactive. The transfer clock is [EC Bus Clock](#). See also [Section 43.12, "Serial Debug Port Timing," on page 655](#).

**FIGURE 38-2:** DATA TRANSFER



## 38.7 Instance Description

There is one [Serial Debug Port](#) instance defined in this chapter.

# MEC1618/MEC1618i

## 38.8 Registers

Each instance of the [Serial Debug Port](#) has its own Logical Device Number, and Base Address as indicated in [Table 38-2](#).

**TABLE 38-2: [Serial Debug Port](#) BASE ADDRESS TABLE**

Serial Debug Port Instance	LDN from ( <a href="#">Table 3-3</a> on page 52)	AHB Base Address
MCU Debug Port	23h	F0_8C00h

The following table summarizes the registers allocated for the [Serial Debug Port](#). The offset field in the following table is the offset from the [Serial Debug Port](#)'s EC Base Address.

**TABLE 38-3: [Serial Debug Port](#) REGISTER SUMMARY**

Register Name	EC Interface			Notes
	SPB Offset	Byte Lane	EC Type	
<a href="#">Debug Data Register</a>	00h	0	R/W	
<a href="#">Debug Control Register</a>	04h	0	R/W	

### 38.8.1 DETAILED REGISTER DESCRIPTIONS

The [Debug Data Register](#) is R/W. It always returns the last data written by the MCU or the power-on default '00h'.

**TABLE 38-4: DEBUG DATA REGISTER**

HOST ADDRESS	NA						HOST SIZE	
EC OFFSET	00h			8-bit			EC SIZE	
POWER	VTR			00h			nSYS_RST DEFAULT	
BUS	EC SPB							
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0
HOST TYPE	-	-	-	--	-	-	-	-
EC TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Data[7:0]							

## DATA[7:0]

Debug data to be shifted out on the MCU Debug Port. While data is being shifted out, the SPB interface will 'hold-off' additional writes to the data register until the transfer is complete.

**TABLE 38-5: DEBUG CONTROL REGISTER**

HOST ADDRESS	NA							HOST SIZE	
EC OFFSET	04h				8-bit			EC SIZE	
POWER	VTR				00h			nSYS_RST DEFAULT	
BUS	EC SPB								
BYTE0 BIT	D7	D6	D5	D4	D3	D2	D1	D0	
HOST TYPE	-	-	-	--	-	-	-	-	
EC TYPE	R						R/W	R/W	
BIT NAME	Reserved						EN	EDGE_SEL	

## EDGE\_SEL

0= Data is shifted out on the rising edge of the debug clock (Default)

1= Data is shifted out on the falling edge of the debug clock

## EN

0= Clock is disabled (Default)

1= Clock enabled

**Note:** The EN bit must not be de-asserted in the middle of a transfer.

## 38.9 Power Management

[Serial Debug Port Power Management](#) is illustrated in [Table 38-6](#). Note that the [Serial Debug Port](#) does not include a sleep enable input.

**TABLE 38-6:** [Serial Debug Port Power Management](#)

Block Enable ( <a href="#">Note 38-1</a> )	Block Idle Status ( <a href="#">Note 38-2</a> )	Clock Required Status (CLK_REQ)	State	Description
DISABLED	X	0	DISABLED	<a href="#">Serial Debug Port</a> is disabled by firmware and the clock is not needed.
ENABLED	NOT IDLE	1	NORMAL OPERATION	The <a href="#">Serial Debug Port</a> is enabled and performing a transaction.
	IDLE	0	SLEEPING	The <a href="#">Serial Debug Port</a> is enabled and not performing a transaction. The clock may be stopped.

**Note 38-1** The Block Enable in [Table 38-6](#) is defined as the [EN](#) Bit in the [Debug Control Register](#).

**Note 38-2** The [Serial Debug Port](#) is 'idle' when not involved in a data transfer. There is no registered bit in the [Serial Debug Port](#) interface to indicate the idle state.

## 39.0 BOOT ROM

### 39.1 Overview

The [Boot ROM](#) code can be used to program the internal Flash using the 16C550A UART (see [Section 39.2, "Description"](#)). The [Boot ROM](#) also includes a [EEPROM API](#) to simplify access to the MEC1618/MEC1618i [EEPROM Emulation](#) in the internal [Flash Memory Array](#).

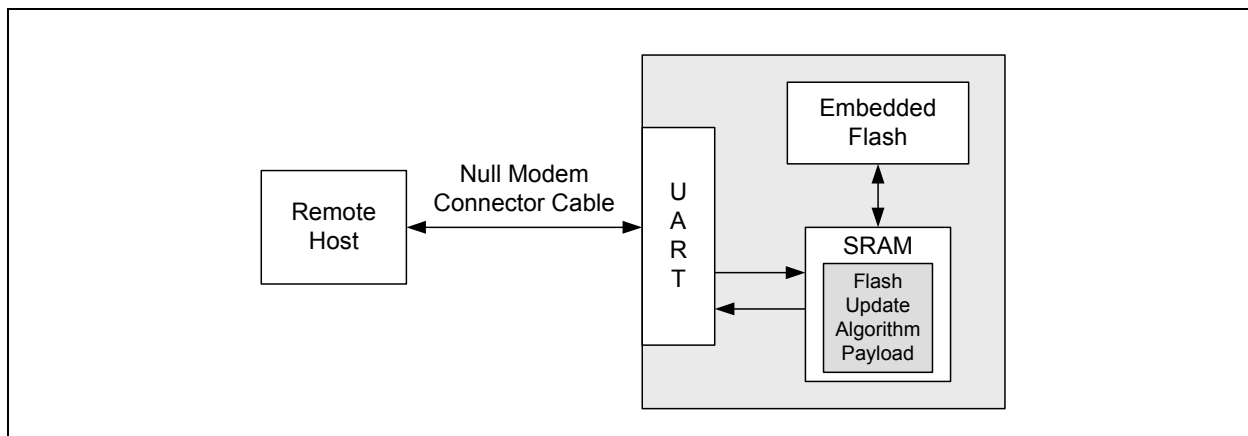
### 39.2 Description

The MEC1618/MEC1618i includes a 4 KB [Boot ROM](#) to download, in a factory environment, an SRAM-based EC application using the MEC1618/MEC1618i built-in serial UART. The SRAM application can then update the embedded Flash image. This behavior depends upon the state of [Control Flags](#), as described in [Section 39.3, on page 589](#).

Following a VTR power-on-reset, the [Boot ROM](#) configures the on-board 2-wire UART and attempts to establish communication with an external Host. If no communication is established within a configurable timeout period, the [Boot ROM](#) restores the UART to its default state and starts loading code from the first physical address of the embedded Flash. If proper Host communication is established, the [Boot ROM](#) downloads the application code into SRAM that performs the actual embedded Flash update. After the Flash application download is complete and the application payload is verified, the [Boot ROM](#) leaves the UART configured and transfers execution to the first physical address of SRAM.

[Figure 39-1](#) illustrates a system-level view of the connection of the remote host to the MEC1618/MEC1618i.

**FIGURE 39-1: SYSTEM-LEVEL VIEW**



### 39.3 Control Flags

The DWORD at address 00\_0FFCh in the [Flash Memory Array](#) contains [Boot ROM Control Flags](#), which are loaded by hardware into the read-only [Embedded Flash Initialization Register](#) following a VTR POR. The [Boot ROM](#) reads the state of the [Control Flags](#) using the [Embedded Flash Initialization Register](#). The [Control Flags](#) are defined in [Table 39-1](#).

**Note:** If the Flash is changed at address 00\_0FFCh, a VTR POR is required before the [Boot ROM](#) will see any change in the [Control Flags](#).

If the Flash is erased, the [Boot ROM](#) implements an infinite timeout waiting for the Host ACK following a VTR POR.

**TABLE 39-1:**    [Boot ROM Control Flags](#)

Bits ( <a href="#">Note 39-1</a> )	Value	Description
0	0	JTAG accesses blocked except for mass erase.
	1	All JTAG accesses allowed.
1	0	Bypass <a href="#">Boot ROM</a> and jump to 00_0000h.
	1	Enable <a href="#">Boot ROM</a> host communication using UART.
3:2	00	20 ms timeout waiting for Host ACK.
	01	100 ms timeout waiting for Host ACK.
	10	200 ms timeout waiting for Host ACK.
	11	No timeout waiting for Host ACK (i.e., wait forever).
4	0	Use external UART clock (GPIO025 = 1.8432 MHz).
	1	Use internal UART clock (MCLK/11).

**Note 39-1**    These bits apply to both the [Flash Memory Array](#) at address 00\_0FFCh, and the [Embedded Flash Initialization Register](#).

## 39.4    EEPROM API

### 39.4.1    PROTOTYPES

#### 39.4.1.1    `eeeprom_read()`

`uint32_t eeeprom_read (uint32_t address_offset)`

#### 39.4.1.2    `eeeprom_erase()`

`void eeeprom_erase (uint32_t address_offset)`

#### 39.4.1.3    `eeeprom_write()`

`void eeeprom_write (uint32_t address_offset, uint32_t data_lsw, uint32_t data_msw)`

### 39.4.2    PARAMETERS

**TABLE 39-2:**    [EEPROM API PARAMETERS](#)

Parameter	Type	Description
<code>address_offset</code>	32-bit unsigned integer	Offset into 1KB EEPROM flash array of aligned 32-bit value to read. EEPROM routines bit-wise AND this value with 03FCh to insure alignment and valid range.
<code>data_lsw</code>	32-bit unsigned integer	Bytes [3:0] of the 8-byte EEPROM block to write.
<code>data_msw</code>	32-bit unsigned integer	Bytes [7:4] of the 8-byte EEPROM block to write.

## 39.4.3 ENTRY POINTS

**TABLE 39-3: EEPROM API ENTRY POINTS**

Entry Point Name	Physical Address	Description
eeeprom_read	10_0FFCh	Parameters: uint32_t address_offset
eeeprom_erase	10_0FF8h	Parameters: uint32_t address_offset
eeeprom_write	10_0FF4h	Parameters: uint32_t address_offset uint32_t data_lsw uint32_t data_msw

## 40.0 GANG PROGRAMMER INTERFACE

### 40.1 Introduction

The [Gang Programmer Interface](#) enables off-chip driven programming of the on-chip Flash memory after device packaging, and using equipment other than an ATE.

The [Gang Programmer Interface](#) accesses the on-chip Flash using the existing on-chip Flash Controller interface, and along with the on-chip JTAG interface provides the functions shown in [Table 40-1](#). [Activation](#) of the [Gang Programmer Interface](#) requires the JTAG interface.

**TABLE 40-1: GANG PROGRAMMER INTERFACE FUNCTIONS**

	Function	Description
1.	<a href="#">Mass Erase</a>	The <a href="#">Mass Erase</a> function is <i>not</i> performed using the <a href="#">Gang Programmer Interface</a> and only affects the Flash Main Memory Array and EEPROM. The <a href="#">Mass Erase</a> function is implemented using the on-chip JTAG Interface and the Mass Erase (ME) bit.
2.	<a href="#">Mass Programming</a>	The <a href="#">Mass Programming</a> function is performed using the <a href="#">Gang Programmer Interface</a> and can program 192 Kbytes in ~1 sec.
3.	<a href="#">Mass Verify</a>	The <a href="#">Mass Verify</a> function is performed using the <a href="#">Gang Programmer Interface</a> and can validate 192 Kbytes in 0.5 sec., or less.

### 40.2 References

No references have been cited for this chapter

### 40.3 Terminology

**TABLE 40-2: TERMINOLOGY**

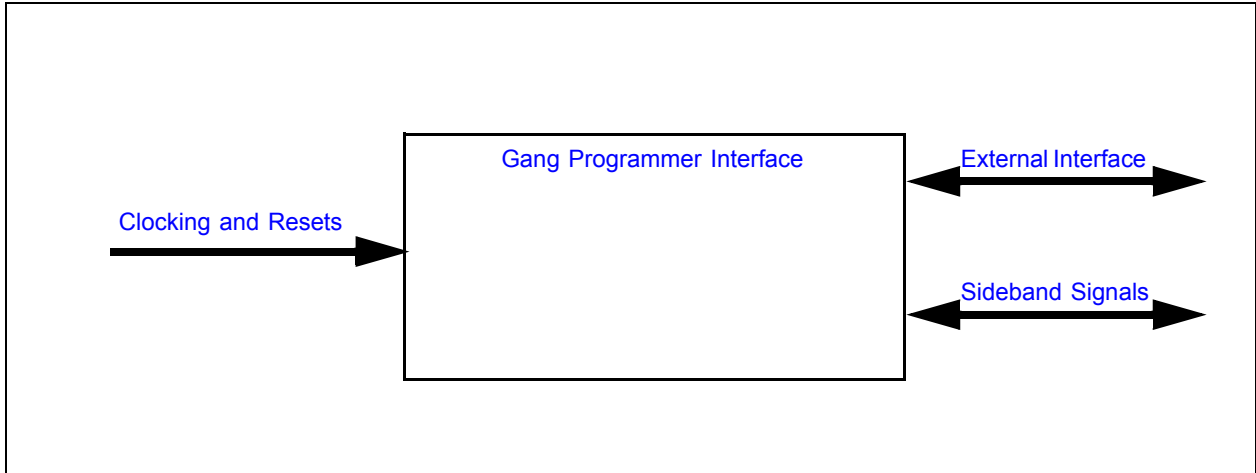
Term	Definition
—	—



## 40.4 Interface

This block is an IP block designed to be incorporated into a chip. The following diagram illustrates the the block interface.

**FIGURE 40-1: BLOCK DIAGRAM**



### 40.4.1 CLOCKING AND RESETS

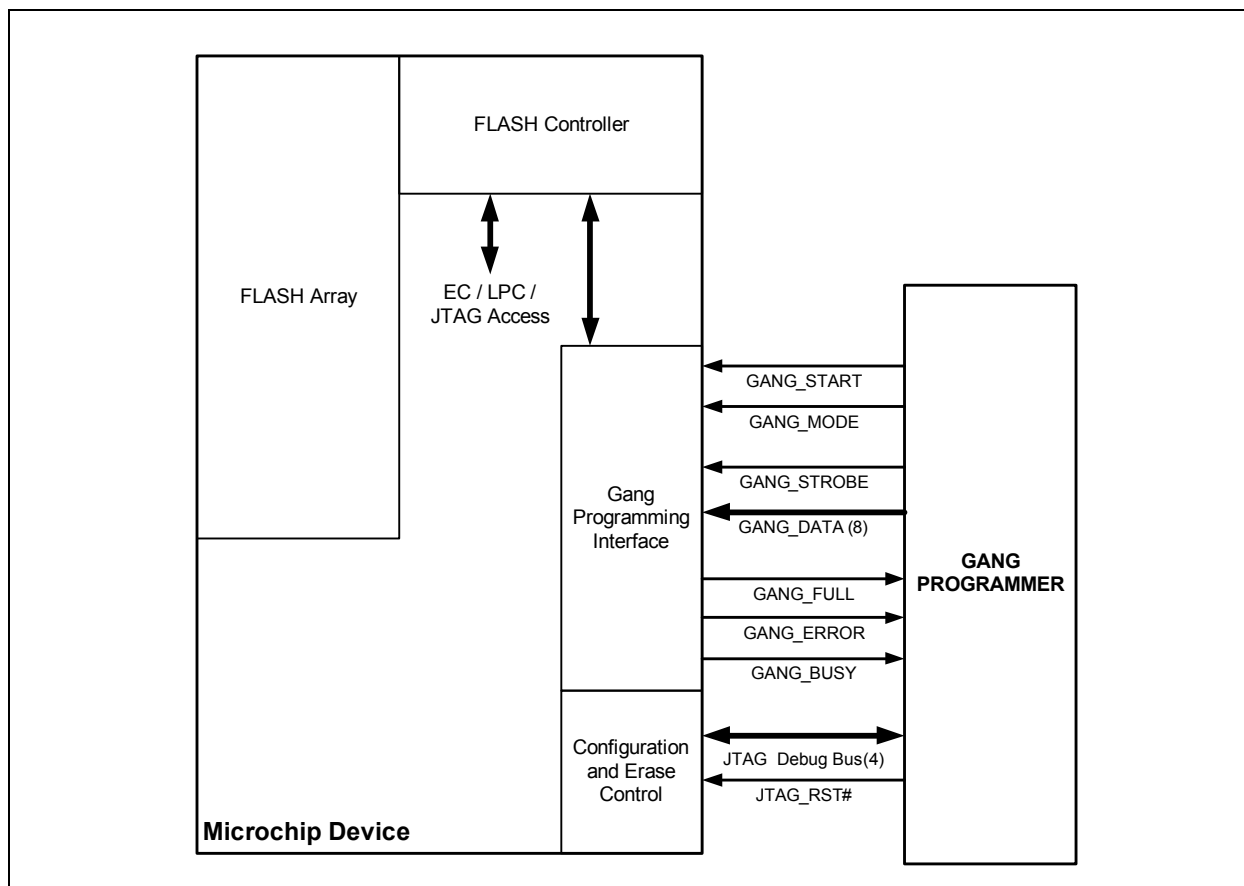
This IP block has the following clocks and reset ports. For a complete list of all the clocks and resets associated with this block see [Section 40.5, "Power, Clocks and Resets," on page 596](#).

**TABLE 40-3: CLOCKING AND RESETS SIGNAL DESCRIPTION TABLE**

Name	Direction	Description
<a href="#">nSYS_RST</a>	Input	reset asserted when power is cycled to the block
<a href="#">EC_BUS_CLK_EN</a>	Input	Clock source to the block. All block clocks are derived from this source.

# MEC1618/MEC1618i

## 40.4.2 EXTERNAL INTERFACE SIGNALS



**TABLE 40-4: EXTERNAL INTERFACE SIGNALS DESCRIPTION TABLE**

Name	DIR (Note 40-1)	Size	Type	Description
GANG_START	Input	1	Command	A low-to-high transition of the <a href="#">GANG_START</a> input starts a <a href="#">Gang Programmer Interface</a> operation. On the rising edge of the <a href="#">GANG_START</a> , the <a href="#">GANG_MODE</a> signal is latched internally and the internal state machines are reset to their standby state. <a href="#">GANG_MODE</a> determines what operation is to be performed.
GANG_MODE	Input	1		<a href="#">GANG_MODE</a> is latched on the rising edge of the <a href="#">GANG_START</a> . <a href="#">GANG_MODE</a> = 1 means <a href="#">Mass Programming</a> , <a href="#">GANG_MODE</a> = 0 means <a href="#">Mass Verify</a> .
GANG_STROBE	Input	1		<a href="#">GANG_STROBE</a> pulses high to latch each byte of data. The <a href="#">GANG_STROBE</a> pulse should be a minimum of 2 microseconds. <a href="#">GANG_START</a> has priority over <a href="#">GANG_STROBE</a> .

**TABLE 40-4: EXTERNAL INTERFACE SIGNALS DESCRIPTION TABLE (CONTINUED)**

Name	DIR (Note 40-1)	Size	Type	Description
GANG_DATA	Input	8	Data Transfer	<b>GANG_DATA</b> is the 8-bit Flash program data presented least-significant byte first from the lowest word address to the highest.
GANG_FULL	Output	1		<b>GANG_FULL</b> indicates whether there is space in the Flash Controller for the next byte of data. <b>GANG_FULL</b> = '1' means that there is no available space and the Gang Programmer must wait until <b>GANG_FULL</b> = '0' to continue.
GANG_ERROR	Output	1		<p><b>Mass Programming:</b> <b>GANG_ERROR</b> is set low when <b>GANG_START</b> is asserted. <b>GANG_ERROR</b> remains low unless the external system presents data while <b>GANG_FULL</b> = '1.' <b>GANG_ERROR</b> is asserted '1' when data has been lost as a result of this overrun. <b>GANG_ERROR</b> will retain its state until the next low-to-high transition of <b>GANG_START</b>.</p> <p><b>Mass Verify:</b> <b>GANG_ERROR</b> is set high when <b>GANG_START</b> is asserted. <b>GANG_ERROR</b> remains high as long as <b>GANG_START</b> remains high. When <b>GANG_START</b> is de-asserted (transitions high-to-low), <b>GANG_ERROR</b> will be set low if no errors occurred during verification, and all presented data matched the contents of the Flash. <b>GANG_ERROR</b> will be asserted '1' if any data byte presented by the Gang Programmer did not match the corresponding byte in the Flash, or the Gang Programmer did not present an integral number of 2048-byte sectors to the <a href="#">Gang Programmer Interface</a>.</p>
GANG_BUSY	Output	1		<p><b>GANG_BUSY</b> is held low initially and also while <b>GANG_START</b> is low. On the rising edge of <b>GANG_START</b>, <b>GANG_BUSY</b> goes high and remains high until either the entire array has been written or verified, or the <b>GANG_START</b> signal has again gone low, in which case the logic is reset.</p> <p>The <a href="#">Gang Programmer Interface</a> de-asserts <b>GANG_BUSY</b> when the <a href="#">Mass Programming</a> or <a href="#">Mass Verify</a> operation has completed and the Flash block is in standby state.</p> <p>If <b>GANG_START</b> is driven low during a <a href="#">Mass Programming</a> operation, <b>GANG_BUSY</b> is not de-asserted until the interface is again able to accept a new command. The external system must keep <b>GANG_START</b> low until <b>GANG_BUSY</b> is de-asserted.</p> <p>When the <b>GANG_BUSY</b> output goes back low to signal that the operation is complete the <b>GANG_ERROR</b> output indicates the summary status of the overall command, as appropriate to that command. This summary status remains presented until the <b>GANG_START</b> signal next goes high.</p>

**Note 40-1** All of the outputs in [Table 40-4](#) require push-pull drivers; that is, status output signals from multiple devices are not bussed in the [Gang Programmer Interface](#).

# MEC1618/MEC1618i

## 40.4.3 SIDEBAND SIGNALS

**TABLE 40-5: SIDEBAND SIGNALS DESCRIPTION TABLE**

Name	Direction	Description
HoldOff	Output	This output is used to hold the EC in reset until <a href="#">Gang Programmer Interface</a> operations have been completed ( <a href="#">Figure 40-2</a> ).
nStart	Input	The <a href="#">nStart</a> input is used to hold off the <a href="#">Gang Programmer Interface</a> until the system is fully operational.
GangEnable	Input	When the <a href="#">GangEnable</a> input is asserted '1,' the <a href="#">Gang Programmer Interface</a> has been enabled by the JTAG interface. <a href="#">GangEnable</a> represents the state of the Gang Enable bit in the JTAG Test register. See also <a href="#">Section 40.8.3.7, "Operational Flow," on page 602</a> for a description of the <a href="#">Gang Programmer Interface Operating Conditions</a> .
MassErase	Input	The <a href="#">MassErase</a> input is used to qualify the <a href="#">GangEnable</a> input as described in <a href="#">Section 40.8.3.7, "Operational Flow," on page 602</a> . <a href="#">MassErase</a> represents the state of the Mass Erase (ME) bit in the JTAG Test register.

## 40.5 Power, Clocks and Resets

### 40.5.1 POWER DOMAINS

**TABLE 40-6: POWER SOURCES**

Name	Description
<a href="#">VTR</a>	The <a href="#">Gang Programmer Interface</a> runs in a single power domain.

### 40.5.2 CLOCKS

**TABLE 40-7: CLOCKS**

Name	Description
<a href="#">EC_BUS_CLK_EN</a>	<a href="#">EC_BUS_CLK_EN</a> provides all clocking for the <a href="#">Gang Programmer Interface</a> .

### 40.5.3 RESETS

**TABLE 40-8: RESET SIGNALS**

Name	Description
<a href="#">nSYS_RST</a>	<a href="#">nSYS_RST</a> is the system reset (SysRst in <a href="#">Figure 40-2</a> ).

## 40.6 Interrupt Generation

None

## 40.7 Low Power Modes

The [Gang Programmer Interface](#) only consumes power when reading Flash Info block data and programming AHB registers immediately after a VTR POR (see [Section 40.8.3, "Functional Description," on page 598](#)). At all other times, clocking to the [Gang Programmer Interface](#) is gated 'off.'

## 40.8 Description

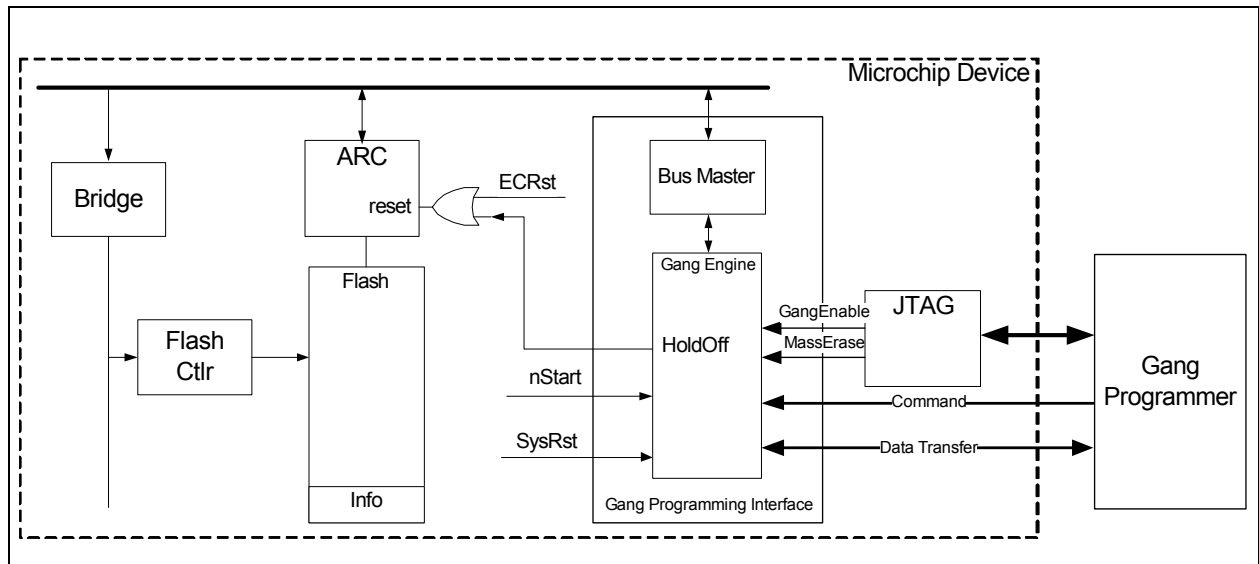
### 40.8.1 OVERVIEW

The [Gang Programmer Interface](#) consists of a state machine and a Bus Master Interface ([Figure 40-2](#)), and permits the [Flash Array](#) to be programmed in close to the minimum time permitted by the internal Flash hardware. The external Gang Programmer can program the devices using deterministic timing, or it can choose to optimize the programming time through the use of flow control signals that are output from the [Gang Programmer Interface](#).

The [Gang Programmer Interface](#) is used to program only the main internal [Flash Array](#). The emulated EEPROM region of the embedded Flash memory may be erased as part of the [Mass Erase](#) function but cannot be programmed, or read by the [Gang Programmer Interface](#).

The Command and Data Transfer signals in [Figure 40-2](#) refer to [Table 40-4, "External Interface Signals Description Table," on page 594](#).

**FIGURE 40-2: MEC1618/MEC1618i INTERFACE SYSTEM VIEW**



### 40.8.2 FLASH ARRAY

- 4 Byte Units (min.) for Programming
- 2048 Byte Units (min.) for Verification
- ~20 microseconds per Word Programming Time
- 64 Words per Row: 21 microseconds maintenance overhead per row = 20.33 microseconds/word total

## 40.8.3 FUNCTIONAL DESCRIPTION

### 40.8.3.1 Overview

The typical sequence for a gang programmer involves 3 steps:

1. Erase the main body of the Flash, using the [Mass Erase](#) function
2. Program the main body of the Flash, using the [Mass Programming](#) function
3. Verify that the program operation was successful using the [Mass Verify](#) function.

All of these functions are described in detail in the subsections that follow; including [Gang Programmer Interface Activation](#), a description of Gang Programmer [Data Transfers](#) independent of the operating mode, and an illustration of the Gang Programmer [Operational Flow](#).

### 40.8.3.2 Activation

[Activation](#) of the [Gang Programmer Interface](#), and initiating the [Mass Erase](#) function requires the on-chip JTAG interface. A description of the JTAG Interface is beyond the scope of this document.

[Activation](#) of the [Gang Programmer Interface](#) occurs following [nSYS\\_RST](#) while [GangEnable](#) is asserted, and after the [nStart](#) input is asserted ('0') ([Table 40-9](#)). Multiplexing for the [External Interface Signals](#) ([Table 40-4](#)) is enabled upon [Activation](#) of the [Gang Programmer Interface](#).

After the [Gang Programmer Interface](#) is operational, the EC is held in reset and the [Gang Programmer Interface](#) retains control of the system until the next [nSYS\\_RST](#) event. Following [Activation](#), the [Gang Programmer Interface](#) waits for a low-to-high transition on the [GANG\\_START](#) input to execute [Mass Programming](#) and [Mass Verify](#) commands (see [Section 40.8.3.7](#), "Operational Flow").

**TABLE 40-9: OPERATING CONDITIONS**

	<a href="#">nStart</a>	<a href="#">GangEnable</a>	<a href="#">MassErase</a>	Operational	Description ( <a href="#">Note 40-3</a> )
1.	X	0	X	No	<a href="#">Gang Programmer Interface</a> disabled.
2.	1	1	X	No	<a href="#">Gang Programmer Interface</a> enabled, but not operating until the on-chip system is fully ready.
3.	0	1	0	Yes ( <a href="#">Note 40-2</a> )	<a href="#">Gang Programmer Interface</a> is enabled and operating as shown in <a href="#">Figure 40-6</a> .
4.	X	1	1	No	<a href="#">Gang Programmer Interface</a> is disabled when <a href="#">Mass Erase</a> and <a href="#">Gang Enable</a> are asserted.

**Note 40-2** After the [Gang Programmer Interface](#) is operational, the EC is held in reset and the [Gang Programmer Interface](#) retains control of the system until the next [nSYS\\_RST](#) event.

**Note 40-3** [GangEnable](#) and [MassErase](#) are sampled on [nSYS\\_RST](#).

### 40.8.3.3 Mass Erase

The [Mass Erase](#) function is triggered via the JTAG Interface using the ME bit. After a Mass Erase the JTAG interface is used to force an internal system reset. This resets the internal state and sets up the device so that a JTAG write can set the Gang Enable bit.

### 40.8.3.4 Mass Programming

Following [Activation](#) of the [Gang Programmer Interface](#), [Mass Programming](#) is initiated using the [GANG\\_START](#) and [GANG\\_MODE](#) inputs as described in [Table 40-4](#), "External Interface Signals Description Table," on page 594. The [Gang Programmer Interface](#) acknowledges [Mass Programming](#) by asserting [GANG\\_BUSY](#), and de-asserting [GANG\\_ERROR](#), and waiting for [Data Transfers](#) to begin. See [Section 40.8.3.6](#) for a description of [Data Transfers](#).

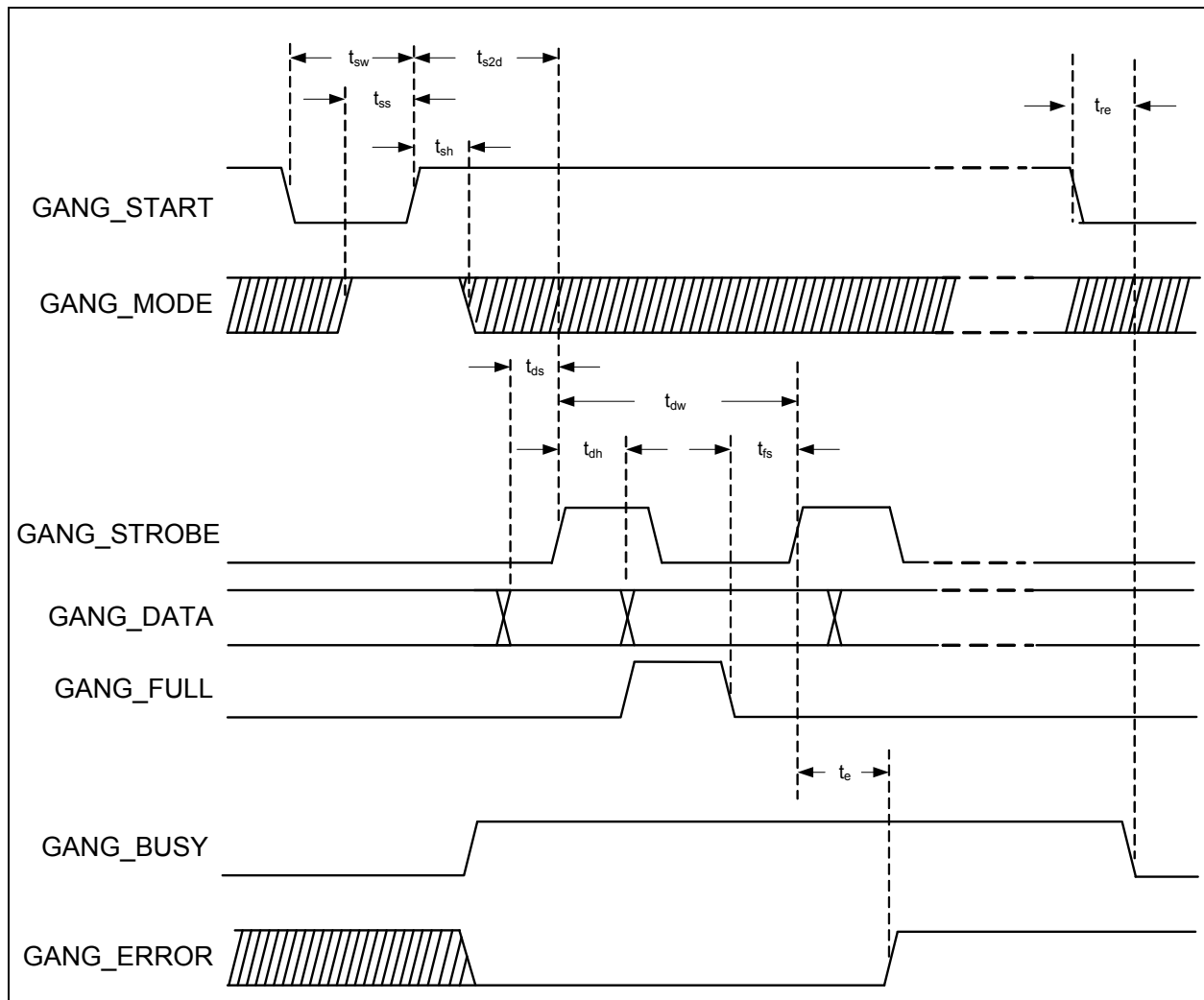
After being triggered, **Mass Programming** continues until the external Gang Programmer terminates it by de-asserting **GANG\_START** (see [FIGURE 40-3: Mass Programming Timing Diagram on page 599.](#)) When **GANG\_START** is brought low, further writing to the Flash is halted, even if fewer than 192 Kbytes of data have been written. **GANG\_BUSY** remains asserted until the **Gang Programmer Interface** completes the programming operation and the Flash controller is idle.

**Mass Programming** always starts at beginning of the **Flash Array**. Each 32-bit Word is provided as 4 one-byte transfers. The Gang Programmer is unaware of base address(es) of the data in the **Flash Array**, and only aware of the length of the array.

While the bytes are presented by the Gang Programmer during **Mass Programming Data Transfers**, the **GANG\_ERROR** signal will indicate whether there have been any data Overrun errors. After the last Byte has been written to the FLASH, and the Gang Programmer de-asserts the **GANG\_START** input, the **GANG\_BUSY** output signal is de-asserted to signal the end of the operation. Note that the Gang Programmer must keep **GANG\_START** low until **GANG\_BUSY** is de-asserted. After **GANG\_START** is de-asserted, the **GANG\_ERROR** error status will be held at its final value until **GANG\_START** is next asserted.

Programming more than the 192 KBytes of the internal Flash may cause errors, but the **Gang Programmer Interface** does not check if the Flash size was exceeded.

**FIGURE 40-3: MASS PROGRAMMING TIMING DIAGRAM**



**Note:** ALL TIMINGS ARE PRELIMINARY AND SUBJECT TO CHANGE

**TABLE 40-10: MASS PROGRAM TIMING PARAMETERS**

Name	Description	Min	Max	Units
$t_{ss}$	Gang_Mode setup time to rising edge of Gang_Start	50		nsec
$t_{sw}$	Gang_Start pulse width	200		nsec
$t_{sh}$	Gang_Mode hold time from rising edge of Gang_Start	200		nsec
$t_{ds}$	Gang_Data setup time to rising edge of Gang_Strobe	50		nsec
$t_{dw}$	Gang_Strobe pulse width	1		$\mu$ sec
	Gang_Strobe pulse width, slow enough to prevent Gang_Full from going high	5		$\mu$ sec
$t_{dh}$	Gang_Data hold time from rising edge of Gang_Strobe	400		nsec
$t_f$	Gang_Full low setup time before rising edge of Gang_Strobe	200		nsec
$t_{re}$	Gang_Start falling edge to Gang_Busy falling edge		20	$\mu$ sec
$t_e$	Gang_Strobe rising edge to Gang_Error rising edge		20	$\mu$ sec

## 40.8.3.5 Mass Verify

Following [Activation](#) of the [Gang Programmer Interface](#), [Mass Verify](#) is initiated using the [GANG\\_START](#) and [GANG\\_MODE](#) inputs as described in [Table 40-4, "External Interface Signals Description Table," on page 594](#). The [Gang Programmer Interface](#) acknowledges [Mass Verify](#) by asserting [GANG\\_BUSY](#), and asserting [GANG\\_ERROR](#), and waiting for [Data Transfers](#) to begin. See [Section 40.8.3.6](#) for a description of [Data Transfers](#).

[Mass Verify](#) always starts at beginning of the [Flash Array](#). Each 32-bit Word is provided as 4 one-byte transfers. The Gang Programmer must perform [Mass Verify](#) operations on sector boundaries (i.e., 2048 byte segments), or errors will occur. The Gang Programmer has no access to the actual Flash data.

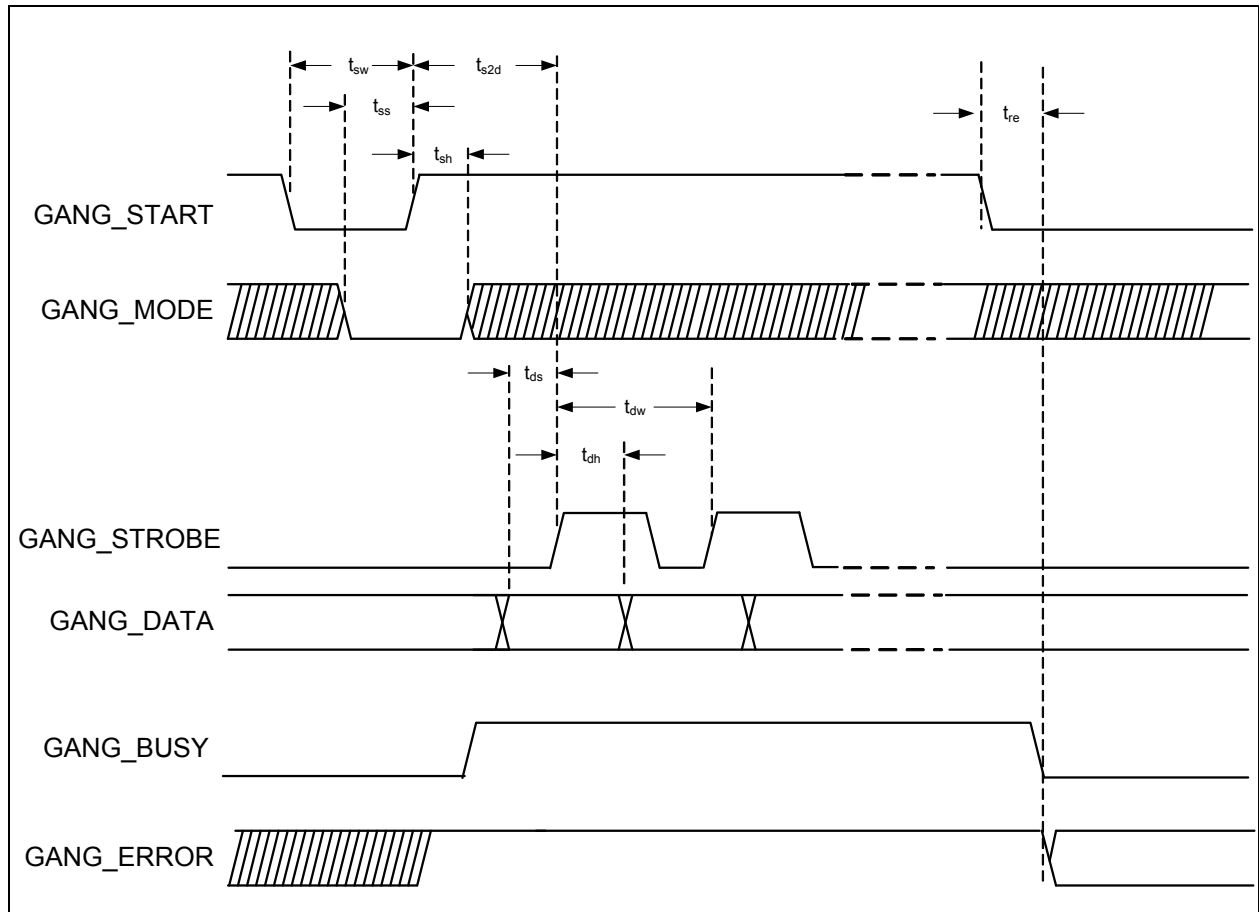
After being triggered, [Mass Verify](#) continues until the external Gang Programmer terminates it by de-asserting [GANG\\_START](#) (see [FIGURE 40-4: Mass Verify Timing Diagram on page 601](#).) When [GANG\\_START](#) is brought low, further comparison is halted, even if fewer than 192 Kbytes of data have been verified. [GANG\\_BUSY](#) remains asserted until the [Gang Programmer Interface](#) is idle.

The [Mass Verify](#) function compares the contents of the [Flash Array](#) against a pattern provided by the Gang Programmer during [Data Transfers](#). As each byte is input, the [Gang Programmer Interface](#) compare it against the corresponding [Flash Array](#) byte, and keeps a running 1-bit indication of whether any mismatches have occurred.

The end of the [Mass Verify](#) operation occurs when the Gang Programmer de-asserts [GANG\\_START](#). When the [GANG\\_BUSY](#) output goes low after the [Mass Verify](#) function terminates, the [GANG\\_ERROR](#) output reflects the state of the 1-bit mismatch indicator; i.e., there is no early indication of a mismatch, or error. [GANG\\_START](#) will be high when [GANG\\_BUSY](#) de-asserts if a mismatch occurred, or the number of [GANG\\_STROBE](#) assertions was not an exact multiple of 2048. [GANG\\_ERROR](#) retains its state until [GANG\\_START](#) is next asserted.



**FIGURE 40-4: MASS VERIFY TIMING DIAGRAM**



**Note:** ALL TIMINGS ARE PRELIMINARY AND SUBJECT TO CHANGE

**TABLE 40-11: MASS VERIFY TIMING PARAMETERS**

Name	Description	Min	Max	Units
$t_{ss}$	Gang_Mode setup time to rising edge of Gang_Start	50		nsec
$t_{sw}$	Gang_Start pulse width	200		nsec
$t_{sh}$	Gang_Mode hold time from rising edge of Gang_Start	200		nsec
$t_{ds}$	Gang_Data setup time to rising edge of Gang_Strobe	50		nsec
$t_{dw}$	Gang_Strobe pulse width	750		nsec
$t_{dh}$	Gang_Data hold time from rising edge of Gang_Strobe	500		nsec
$t_{re}$	Gang_Start fall to Gang_Busy and Gang_Error response		200	nsec

# MEC1618/MEC1618i

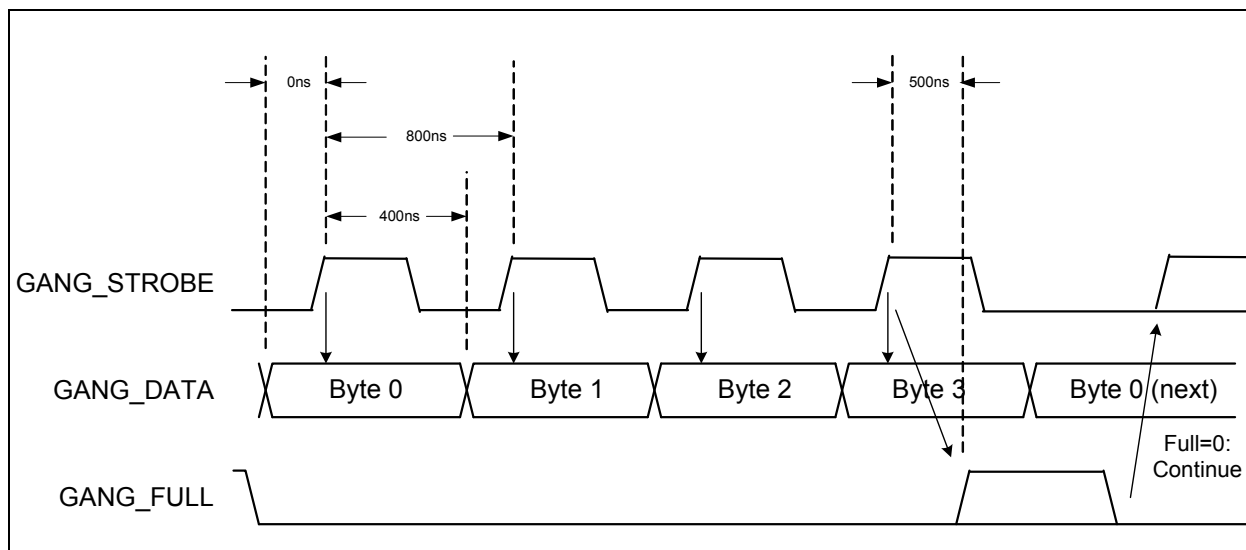
## 40.8.3.6 Data Transfers

The **GANG\_FULL** output always indicates when space is available for new data. After **GANG\_FULL** is not asserted ('0'), it will not be asserted except on an external write of more data, as shown in [Figure 40-5](#).

A low-to-high transition on **GANG\_STROBE** signals the availability of a new byte of data on **GANG\_DATA** pins. As shown in [Figure 40-5](#), no setup time is required. The Gang Programmer must hold **GANG\_DATA** valid for at least 400 ns after the low-to-high transition of **GANG\_STROBE**. The period between low-to-high transitions on **GANG\_STROBE** must be at least 800 ns. **GANG\_FULL** will be asserted, if required, a maximum of 500 ns after the low-to-high transition of **GANG\_STROBE**.

Because the on-chip Flash program time is approximately 5.2 microseconds/byte, if the Gang Programmer asserts **GANG\_STROBE** with a minimum of 6 microseconds of delay between bytes, the programmer can ignore the **GANG\_FULL** signal.

**FIGURE 40-5: DATA TRANSFER TIMING**

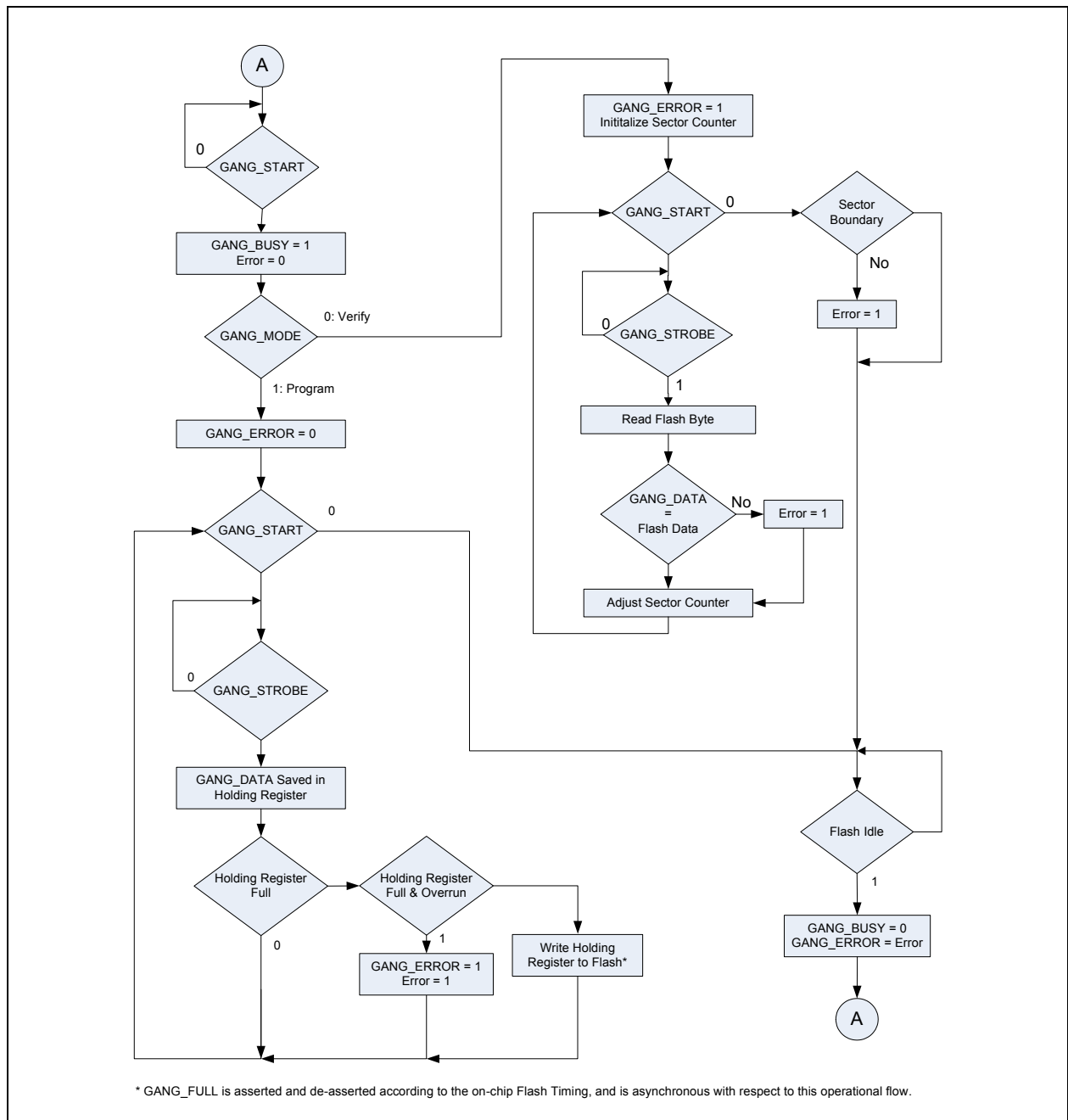


## 40.8.3.7 Operational Flow

[Figure 40-6](#) illustrates the [Operational Flow](#) for the [Mass Programming](#) and [Mass Verify](#) commands as described in the sections above. Point 'A' in this figure is entered following [Gang Programmer Interface Activation](#) as described in [Section 40.8.3.2, "Activation," on page 598](#), and again following command termination.

Note that **GANG\_FULL** is asserted and de-asserted according to the on-chip Flash Timing, and is asynchronous with respect to the operational flow shown in [Figure 40-6](#).

FIGURE 40-6: GANG PROGRAMMER OPERATIONAL FLOW



# MEC1618/MEC1618i

## 41.0 JTAG AND XNOR

### 41.1 General Description

The MEC1618/MEC1618i includes a [JTAG Slave for Debugging ARC Firmware](#), a [Boundary Scan](#) slave, and a [JTAG Master](#). All of these function share the MEC1618/MEC1618i [JTAG Interface](#) as defined in [Section 2.4.4](#).

The JTAG slaves are asynchronously reset and deactivated when the JTAG\_RST# input pin is asserted ('0'). When JTAG\_RST# is not asserted, only one slave is enabled. JTAG [Slave Selection](#) depends on the state of the [TAP Controller Select Strap Option](#). The [JTAG Master](#) is independent of the slave TAP controllers and cannot be used when the JTAG\_RST# pin is not asserted.

### 41.2 Slave Selection

#### 41.2.1 TAP CONTROLLER SELECT STRAP OPTION

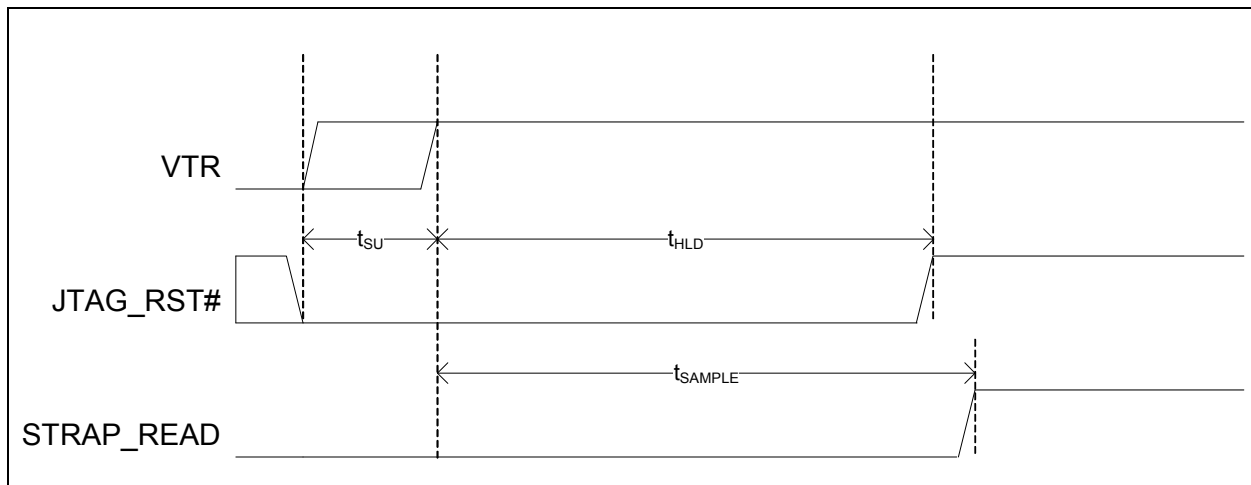
The [TAP Controller Select Strap Option](#) determines the JTAG slave that is selected when JTAG\_RST# is not asserted. The state of the [TAP Controller Select Strap Option](#) pin (see [Section 2.7, "Strapping Options," on page 45](#)) is sampled by hardware at VTR POR according to the [Slave Select Timing](#) as defined in [Section 41.2.2](#) and is registered internally to select between the debug and boundary scan TAP controllers.

If the [TAP Controller Select Strap Option](#) is sampled low, the debug TAP controller is selected; if the strap is sampled high, the boundary scan slave is selected. An internal pull-up resistor is enabled by default on the [TAP Controller Select Strap Option](#) pin and can be disabled by firmware, if necessary.

#### 41.2.2 SLAVE SELECT TIMING

The JTAG\_RST# input pin must be asserted at VTR power-up and follow the timing as defined in this section. The relationship between the [TAP Controller Select Strap Option](#) sample timing and the JTAG\_RST# pin de-assertion timing is illustrated below in [Figure 41-1](#) and [Table 41-1](#).

**FIGURE 41-1: ASYNCHRONOUS JTAG RESET AND TAP Controller Select Strap Option POWER-UP TIMING**



**TABLE 41-1:** Slave Select Timing PARAMETERS

Parameters	Symbol	Min	Units	Notes
JTAG_RST# Setup Time	$t_{SU}$	see <a href="#">Section 43.11</a> , "JTAG Interface Timing," on page 653		
JTAG_RST# Hold Time	$t_{HLD}$			
<a href="#">TAP Controller Select Strap Option</a> Sample Time	$t_{SAMPLE}$	ARC reset time	–	$t_{SAMPLE}$ is the same as $t_{STRETCH}$ in <a href="#">FIGURE 5-19: VTR Power-Up Timing</a> on page 107.

### 41.3 JTAG Slave for Debugging ARC Firmware

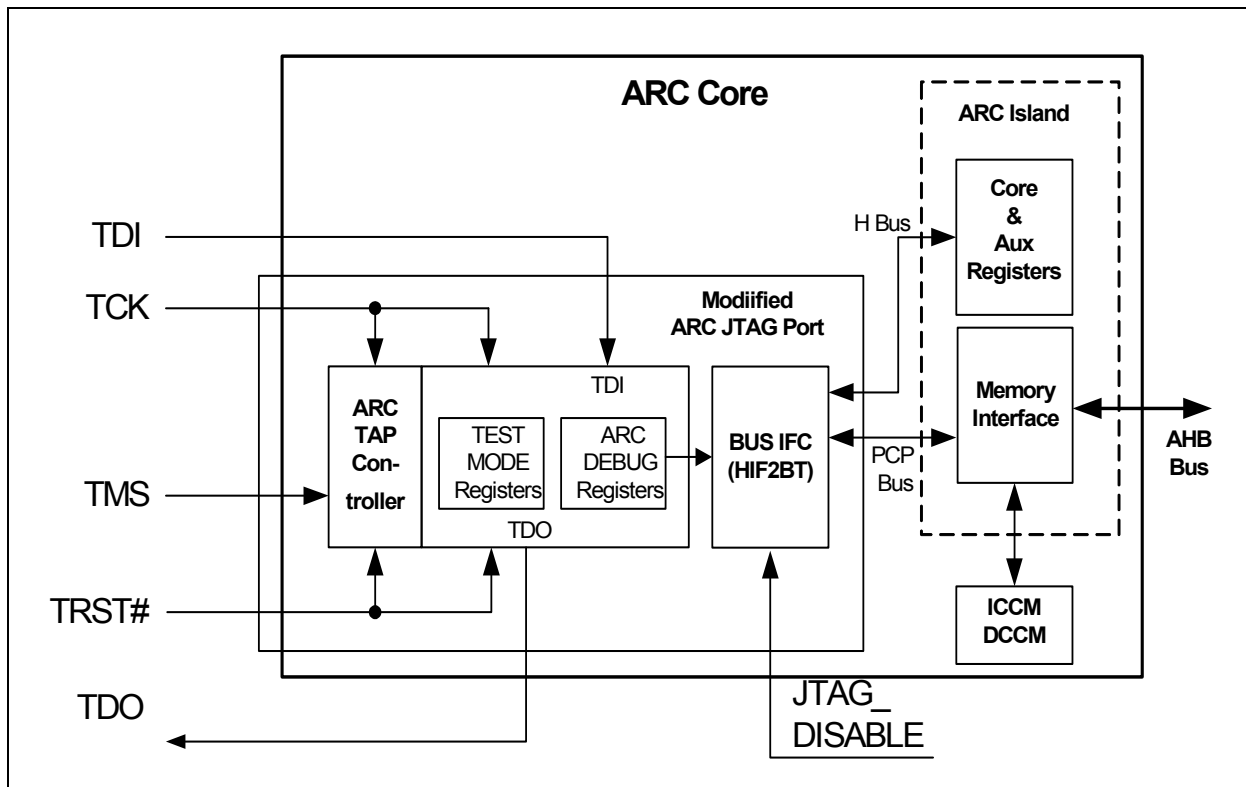
- The ARC JTAG Port is defined in the ARC 600 External Interfaces Reference Manual, Chapter 2. The Microchip modifications are described in this chapter. The ARC JTAG Port has been modified by Microchip to provide additional Data Registers (see [Note 41-3](#).) The TEST MODE Register provide additional on-chip support specific to the MEC1618/MEC1618i.

#### 41.3.1 ARC JTAG CAPABILITIES

- Fully compliant with IEEE1149.1 standard
- 4-bit Instruction Register
- Standard 1-bit BYPASS register
- Standard 32-bit IDCODE register
- Four JTAG registers give access to on-chip memory and register resources
- Can read or write a 32-bit quantity from or to any ARC Core Register, Aux Register or 32-bit aligned memory location. No other interfaces are provided or needed
- Accesses to Aux Registers and Memory do not require the ARC processor to be halted
- Memory accesses are always performed in units of 32 bits

# MEC1618/MEC1618i

FIGURE 41-2: BLOCK DIAGRAM OF ARC JTAG SYSTEM



As shown in Figure 41-2, the ARC JTAG Port sits within the ARC Core, between the JTAG signals and the ARC Island block, which contains the Core and Aux register sets and the Memory Interface to the Closely-Coupled Memories (ICCM / DCCM) and the AHB Bus (FLASH memory and Memory-Mapped I/O registers). It is not associated with any form of Boundary Scan, but instead is used by an external JTAG host to access memory and register resources on behalf of a debugger program. There are no other connections between the ARC JTAG Port and the rest of the ARC Core: register access is enough to halt and restart the processor, as well as to detect that the processor has halted (a continuous poll on the STATUS32 Aux Register).

Internally, the ARC JTAG port consists of the following sub-blocks:

- The ARC Test Access Port (TAP) Controller. This block is driven by the JTAG\_CLK, TMS and JTAG\_RST# inputs, and provides the control signals for the JTAG data transfers. It is a single state machine consisting of 16 states, whose transitions are controlled strictly by the state of TMS on each rising edge of JTAG\_CLK. The low-active JTAG\_RST# input provides an asynchronous reset, though JTAG\_CLK and TMS together can also bring the TAP Controller to the reset state (5 consecutive JTAG\_CLK rising edges with TMS=1).
- The ARC DEBUG & TEST MODE Registers blocks. This blocks handles the data transfers as directed by the TAP, and contains the registers and shift registers internal to the JTAG Port. The holding registers in this block consist of a 4-bit Instruction register, a 1-bit Bypass register, and a set of Data registers (see Note 41-3) of various lengths. There are three sets of Data Registers: **JTAG Standard Data Registers**, **JTAG Debug Data Registers**, & **JTAG Test Mode Data Registers**. The **JTAG Test Mode Data Registers** provide additional on-chip support specific to the MEC1618/MEC1618i.
- The Bus Interface block. This block accepts values for the **JTAG Debug Data Registers**: ADDRESS, DATA, STATUS and TRANSACTION COMMAND, and uses them to request data transfers from the ARC's inner core ("Island") sub-block. \
- As part of the Boot block protection the JTAG Port's **JTAG Debug Data Registers** can be disabled to prevent an external debugger from potentially halting the EC and then reading or re programming any word in the Boot Block.

## 41.4 Boundary Scan

[Boundary Scan](#) includes registers and functionality as defined in IEEE 1149.1 and the MEC1618/MEC1618i BSDL file. Functionality implemented beyond the standard definition is summarized in [Table 41-3](#). The MEC1618/MEC1618i [Boundary Scan](#) JTAG ID is shown in [Table 41-2](#).

**TABLE 41-2:** [Boundary Scan](#) JTAG ID

Part Number	JTAG ID
MEC1618	J2042445h

**TABLE 41-3:** EXTENDED [Boundary Scan](#) FUNCTIONALITY

Bits	Function	Description
12, 14	<a href="#">TAP Controller Select Strap Option</a> Override	When the <a href="#">TAP Controller Select Strap Option</a> Override is '1,' the <a href="#">TAP Controller Select Strap Option</a> is overridden to select the debug TAP Controller until the next time that the <a href="#">TAP Controller Select Strap Option</a> is sampled ( <a href="#">Note 41-1</a> ).

**Note 41-1** To set Strap Override Function, write 0X1FFFFD to the TAP controller instruction register, then write 0x5000 to the TAP controller data register. Note that the instruction register is 18 bits long; the data register is 16 bits long.

## 41.5 JTAG Master

### 41.5.1 OVERVIEW

The [JTAG Master](#) controller in the MEC1618/MEC1618i enables the embedded controller to perform full IEEE 1149.1 test functions as the master controller for test operations at assembly time or in the field.

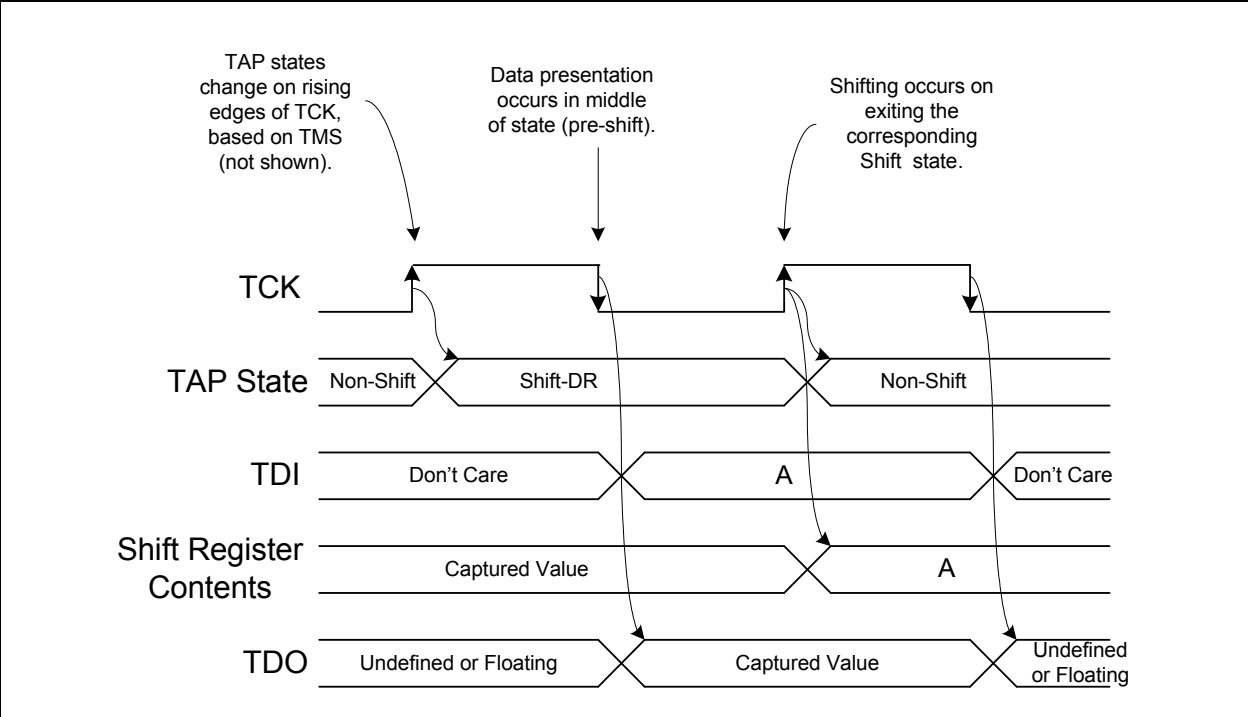
The [JTAG Master](#) interface shares the JTAG pin interface with the [Boundary Scan](#) and Debug TAP controllers; including, JTAG\_CLK, JTAG\_TDI, JTAG\_TDO and JTAG\_TMS. When the MEC1618/MEC1618i JTAG interface is configured as master, it is the responsibility of the master firmware to satisfy all requirements regarding JTAG port multiplexing. It is also it is the responsibility of the [JTAG Master](#) firmware to satisfy all requirements for external JTAG slave devices that require an external asynchronous reset (TRST#) input.

### 41.5.2 DESCRIPTION

When JTAG slave functions are not required and the [JTAG Master](#) is enabled, the JTAG Interface pins are turned around so that the pins JTAG\_CLK, JTAG\_TMS and JTAG\_TDI become outputs and the JTAG\_TDO becomes an input.

Figure 41-3, "JTAG Signal Clocking" shows the clocking behavior of JTAG in the TAP controller in a JTAG Slave device. The rows "TAP State" and "Shift Reg. Contents" refer to the state of the JTAG Slave device and are provided for reference. When configured as a Master, the JTAG interface drives JTAG\_CLK and will shift out data onto JTAG\_TMS and JTAG\_TDI in parallel, updating the pins on the falling edge of JTAG\_CLK. The Master will sample data on JTAG\_TDO on the rising edge of JTAG\_CLK.

FIGURE 41-3: JTAG SIGNAL CLOCKING



41.5.3 JTAG MASTER REGISTER INTERFACE

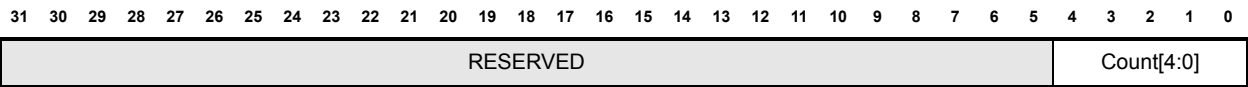
The JTAG Master interface uses the ARC Auxiliary Register interface; all JTAG Maser registers are Auxiliary registers, accessed by the ARC `lr` (load auxiliary register) and `sr` (store auxiliary register) instructions. [Table 41-4, "JTAG Auxiliary Registers"](#) lists these registers:

TABLE 41-4: JTAG AUXILIARY REGISTERS

Register Number	Auxiliary Name	LR/SR R/W	VTR Default	Description
FFFF_FFFh	JTAG_COM	W	0000_000h	JTAG Master Command Register
FFFF_FFFEh	JTAG_TMS	R/W	0000_000h	JTAG Master register source for TMS pin
FFFF_FFFDh	JTAG_TDI	R/W	0000_000h	JTAG Master register source for TDI pin
FFFF_FFFCh	JTAG_TDO	R/W	0000_000h	JTAG Master register destination for TDO pin
FFFF_FFFBh	JTAG_STATUS	R	0000_000h	JTAG Master Status register
FFFF_FFFAh	JTAG_CONFIG	R/W	0000_000h	JTAG Master Configuration register

41.5.3.1 JTAG Master Command, JTAG\_COM

FIGURE 41-4: JTAG\_COM





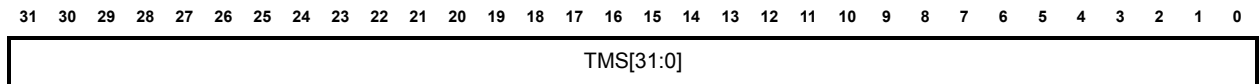
## COUNT

If **M** in **JTAG\_CONFIG** is 1, writing this field starts, clocking and shifting on the JTAG port. The JTAG Master port will shift count+1 times, so writing a 0 will shift 1 bit and writing 31 will shift 32 bits. The signal JTAG\_CLK will cycle count+1 times. **JTAG\_TMS** and **JTAG\_TDI** will be shifted out on the falling edge of JTAG\_CLK and **JTAG\_TDO** will get shifted in on the rising edge of JTAG\_CLK.

When **M** in **JTAG\_CONFIG** is 0 the JTAG port is configured as a Slave and writing this field has no effect.

### 41.5.3.2 JTAG Master TMS, JTAG\_TMS

**FIGURE 41-5: JTAG\_TMS**

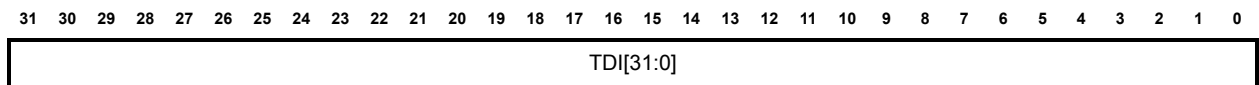


## TMS

When **JTAG\_COM** is written, from 1 to 32 bits are shifted out of TMS, starting with bit 0, onto the JTAG\_TMS pin. Shifting is at the rate determined by **CLK** in **JTAG\_CONFIG**.

### 41.5.3.3 JTAG Master TDI, JTAG\_TDI

**FIGURE 41-6: JTAG\_TDI**

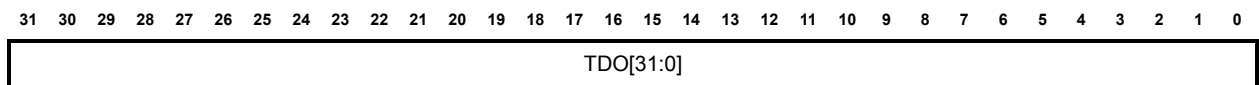


## TDI

When **JTAG\_COM** is written, from 1 to 32 bits are shifted out of TDI, starting with bit 0, onto the JTAG\_TDI pin. Shifting is at the rate determined by **CLK** in **JTAG\_CONFIG**.

### 41.5.3.4 JTAG Master TDO, JTAG\_TDO

**FIGURE 41-7: JTAG\_TDO**

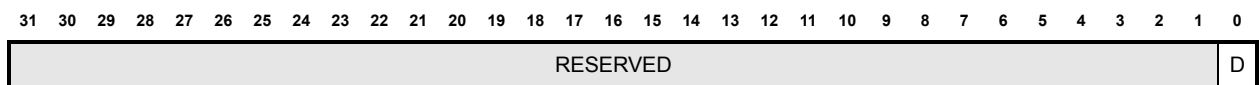


## TDO

When **JTAG\_COM** is written, from 1 to 32 bits are shifted into of TDO, starting with bit 0, onto the JTAG\_TDO pin. Shifting is at the rate determined by **CLK** in **JTAG\_CONFIG**.

### 41.5.3.5 JTAG Master Status, JTAG\_STATUS

**FIGURE 41-8: JTAG\_STATUS**



## D

This bit is read-only.

This bit is set to 1 when **JTAG\_COM** is written. It becomes 0 when the shifting has completed. Software can poll this bit to determine when a command has completed and it is therefore safe to retrieve the data in **JTAG\_TDO** and to load new data into **JTAG\_TMS** and **JTAG\_TDI**.

# MEC1618/MEC1618i

## 41.5.3.6 JTAG Master Configuration, JTAG\_CONFIG

**FIGURE 41-9: JTAG\_CONFIG**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																													M	CLK	

### CLK

This field determines the clock rate of the JTAG\_CLK signal. Options are shown in [Table 41-5, "JTAG Clock Options"](#):

**TABLE 41-5: JTAG CLOCK OPTIONS**

CLK Value	JTAG_CLK Clock Rate
0h	Reserved
1h	MCLK/2
2h	MCLK/4
3h	MCLK/8
4h	MCLK/16
5h	MCLK/32
6h	MCLK/64
7h	MCLK/128

**APPLICATION NOTE:** the ARC clock must be configured to be equal to or faster than the JTAG\_CLK.

### M

This bit controls Master/Slave JTAG multiplexing. When this bit is 0 (default), the JTAG port is configured as a slave. When this bit is 1, the JTAG port is configured as a Master.

## 41.6 JTAG Port Signal Interface Description

The signal pins are defined in [Section 2.4.4, "JTAG Interface," on page 16](#).

The JTAG\_CLK input is the clock that drives the JTAG interface. It is asynchronous to other clocks on-chip.

The TMS input is sampled on each rising edge of JTAG\_CLK, and governs the transitions among the 16 states of the state machine (TAP) that controls the transfer of data.

The TDI input is the serial data input, shifted in during the Shift-IR and Shift-DR states of the TAP. It is sampled on rising edges of JTAG\_CLK.

The TDO output is the serial data output. It is presented on falling edges of JTAG\_CLK, 1/2 clock before each input shift, to provide setup and hold time to the next JTAG controller in the chain. The final TDO output pin, after all on-chip chaining ([Figure 41-2](#)) is held in high-impedance mode (floating) except when valid data is being presented. The enabled/disabled state of the pin is also changed on falling edges of JTAG\_CLK.

The JTAG\_RST# input provides the [Async JTAG RESET](#). Note that the reset state of the JTAG port is only local to the JTAG port: its effect is to keep the JTAG port in an idle state and to disengage it from the rest of the system, so that it does not affect other on-chip logic in this state.

## 41.7 Power, Clocks and Reset

See [Section 43.11, "JTAG Interface Timing," on page 653](#) power on sequence and reset timing.

### 41.7.1 POWER DOMAINS

The JTAG block is powered by VTR.

### 41.7.2 CLOCKS

The JTAG port runs internally from the externally-provided JTAG\_CLK clock pulses only. There is no requirement for JTAG\_CLK to be constantly running.

The following JTAG Registers interface to the ARC Island block (as illustrated in [Figure 41-2](#)) for access to registers and memory: [STATUS Register \(8h\)](#), [TRANSACTION COMMAND Register \(9h\)](#), & [ADDRESS Register \(Ah\)](#), [DATA Register \(Bh\)](#). There is a clock relationship required between JTAG\_CLK and the ARC Core clock frequency. JTAG\_CLK may be asynchronous, but it must be slower than 1/2 the frequency of the ARC Core clock. In practical terms, then, JTAG\_CLK should be selected to be nominally 1/4 of the minimum Core clock frequency. See [APPLICATION NOTE: on page 113](#).

**APPLICATION NOTE:** The Ashling JTAG interface box is not documented to operate any slower than JTAG\_CLK = 1MHz, therefore the Core must be running at 4MHz in order to run the ARC debugger through the JTAG interface using the Ashling interface.

Stopping the Core clock disables the JTAG port for debugging purposes. It does not affect IDCODE or BYPASS operation.

See [Section 43.11, "JTAG Interface Timing," on page 653](#) for the maximum frequency  $f_{clk}$  on the JTAG\_CLK pin to access a JTAG Registers other than [STATUS Register \(8h\)](#), [TRANSACTION COMMAND Register \(9h\)](#), & [ADDRESS Register \(Ah\)](#), [DATA Register \(Bh\)](#).

### 41.7.3 RESET

The ARC JTAG block has two resets: [Async JTAG RESET](#) by its JTAG\_RST# input and [Sync JTAG RESET](#) by JTAG protocol.

#### 41.7.3.1 Async JTAG RESET

The JTAG\_RST# pin provides the [Async JTAG RESET](#) to the JTAG Registers. The JTAG\_RST# pin has an active low, asynchronous assertion and a synchronous de-assertion. The JTAG Registers will be reset asynchronously (and immediately) upon the active low JTAG\_RST# assertion. After the JTAG\_RST# pin has been de-asserted, a delay of three JTAG\_CLKs is required in order to access the JTAG Registers; i.e., the JTAG Registers will remain in reset for three clocks following the synchronous JTAG\_RST# pin de-assertion. See [Section 43.11, "JTAG Interface Timing," on page 653](#).

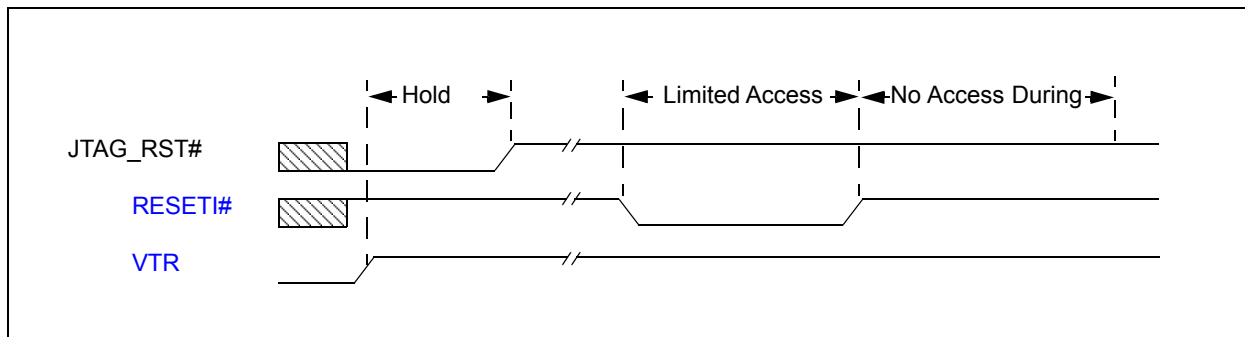
**APPLICATION NOTE:** After asserting and de-asserted the JTAG\_RST# pin, a [Sync JTAG RESET](#) can be applied before starting to access the JTAG Registers (to meet the JTAG\_RST# synchronous de-assertion requirement).

**Note 41-2** JTAG registers, in particular the [JTAG Test Mode Data Registers](#), are set to their initial values by the assertion of the JTAG\_RST# pin, not the VTR Power On Reset. JTAG\_RST# must be held low while the MEC1618/MEC1618i is powering up so the registers can be set to their proper default values. If JTAG\_RST# is high during power up, the [JTAG Test Mode Data Registers](#) may be set to unpredictable values, which may trigger unwanted test modes.

**APPLICATION NOTE:** Care should be taken during VTR power up to insure that JTAG\_RST# is asserted for a longer time then the VTR rise time due to capacitive loading. See [Section 41.2.2, "Slave Select Timing," on page 604](#) for timing requirement.

[Figure 41-10](#) illustrates an example of the [Async JTAG RESET](#) function. Refer to [Table 41-6, "Async JTAG RESET Functional Description," on page 612](#) for details.

**FIGURE 41-10: JTAG\_RST# FUNCTIONAL EXAMPLE**



**TABLE 41-6: Async JTAG RESET FUNCTIONAL DESCRIPTION**

RESETI#	VTR	JTAG_RST#	Description
X	0	X	System unpowered
	0->1	1	Undefined
0		0	System powered, device held in reset
1->0	1	1	Limited JTAG access; e.g., JTAG register access only, JTAG Master held in reset
1	0->1	0 (Hold)->1	JTAG_RST# hold time is required as defined in <a href="#">Section 43.11, "JTAG Interface Timing," on page 653</a>
0->1	1	1	No JTAG access allowed for the equivalent of the JTAG_RST# hold time ( <a href="#">Section 43.11</a> ).
1		0	Normal Operation, JTAG disabled
		1	Normal Operation, JTAG enabled

## 41.7.3.2 Sync JTAG RESET

It can also be reset synchronously by a JTAG\_CLK / TMS sequence, in accordance with the JTAG standard. A series of 5 successive JTAG\_CLK rising edges, with TMS held high throughout, will accomplish this from any state.

The ARC JTAG port, upon entering its Reset state, will be prepared to accept an Instruction or Data transfer. It will also be disengaged from external circuitry, allowing it to operate normally.

The initial contents of the Instruction Register are the IDCODE command (Ch). If a Data transfer is performed first after Reset, without an preceding Instruction transfer, then the IDCODE value will be loaded into its 32-bit shift register and presented serially, after which will appear the bits shifted in from TDI.

The initial contents of the Data registers are as listed in [Table on page 617](#).

## 41.8 Interrupts

There are no interrupts assigned to the ARC JTAG block. Control of the processor is performed by monitoring, setting and clearing the H bit (Halt) in the Aux register STATUS32, and by register manipulations while the processor is halted.

However, any interrupt or other event that can be triggered by accessing registers (Core, Aux or Memory-Mapped) can be triggered through the JTAG port.

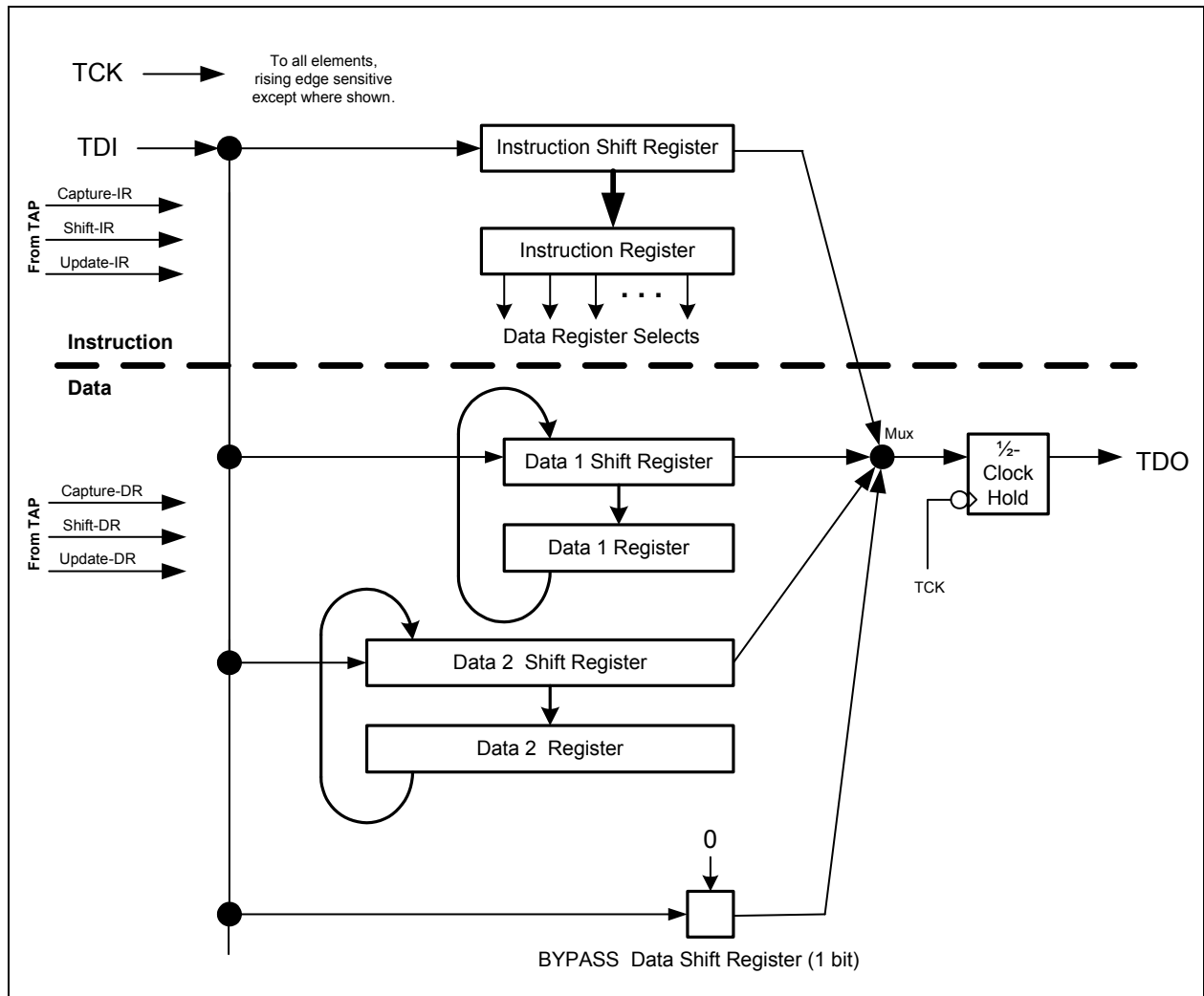
## 41.9 JTAG Background

The following is a simplified description, intended to provide background for the ARC JTAG port. For full details, see the JTAG specification (IEEE Standards 1149.1 and 1149.1b).

### 41.9.1 INTERNAL STRUCTURE

A JTAG port operates by transferring information serially into and out of an Instruction register and one or more Data registers. These registers are connected in parallel with each other, and can be of arbitrary length. See [Figure 41-11](#).

**FIGURE 41-11: STRUCTURE OF A JTAG PORT (SIMPLIFIED)**



The protocol for shifting information makes a distinction between an Instruction transfer (to/from a single Instruction register) and a Data transfer (to/from one of several Data registers). The Instruction register is handled separately because it selects which specific Data register is accessed by subsequent Data transfers.

In daisy-chained JTAG controllers, the Instruction registers form one chain, and the currently-selected set of Data registers in each JTAG controller combine to form a second chain. To shorten the Data chain when not all JTAG controllers are of interest, a mandatory one-bit Data register called **BYPASS** is provided. There is no bypassing for the Instruction chain, so its full length must be shifted as each new instruction is transferred anywhere. Selecting the **BYPASS** Data register is the equivalent of a No-Operation instruction for a JTAG controller, and this instruction is always defined as a '1' in all Instruction register bits.

# MEC1618/MEC1618i

---

Each entity called a “Register” actually consists of two parts: the Register itself, and an associated Shift Register which connects to TDI and TDO. The Register may load from, and/or source information in parallel to, the Shift Register. These two parts are the same length, meaning that (for example) a 5-bit Register will be associated with a 5-bit Shift Register.

The Instruction register and the Data registers respond to decoded state signals from the TAP Controller sub-block (Section 41.9.2), which represent sub-steps of a transfer. The sub-steps they perform are **Capture**, which loads the shift register in parallel, **Shift**, which shifts information in from TDI and out on TDO, and **Update**, which writes information from the Shift Register in parallel. The **Capture-IR**, **Shift-IR** and **Update-IR** controls affect only the Instruction register. The **Capture-DR**, **Shift-DR** and **Update-DR** controls affect only the Data register that is currently selected by the contents of the Instruction register.

## 41.9.2 TAP CONTROLLER AND PROTOCOL

The JTAG protocol is driven by the level of the TMS (Test Mode Select) input pin at each rising edge of the JTAG\_CLK clock. This is the responsibility of the TAP Controller section of the JTAG controller, which performs state transitions as illustrated in the state diagram in Figure 41-12. States whose names end with “IR” affect the Instruction register (the rightmost column of states in Figure 41-12), and those ending with “DR” affect a Data register (the middle column in Figure 41-12). Note that the TMS signal goes in parallel to all JTAG ports in a chain, so they are always in the same protocol state. The sequence of accessing any register is as follows:

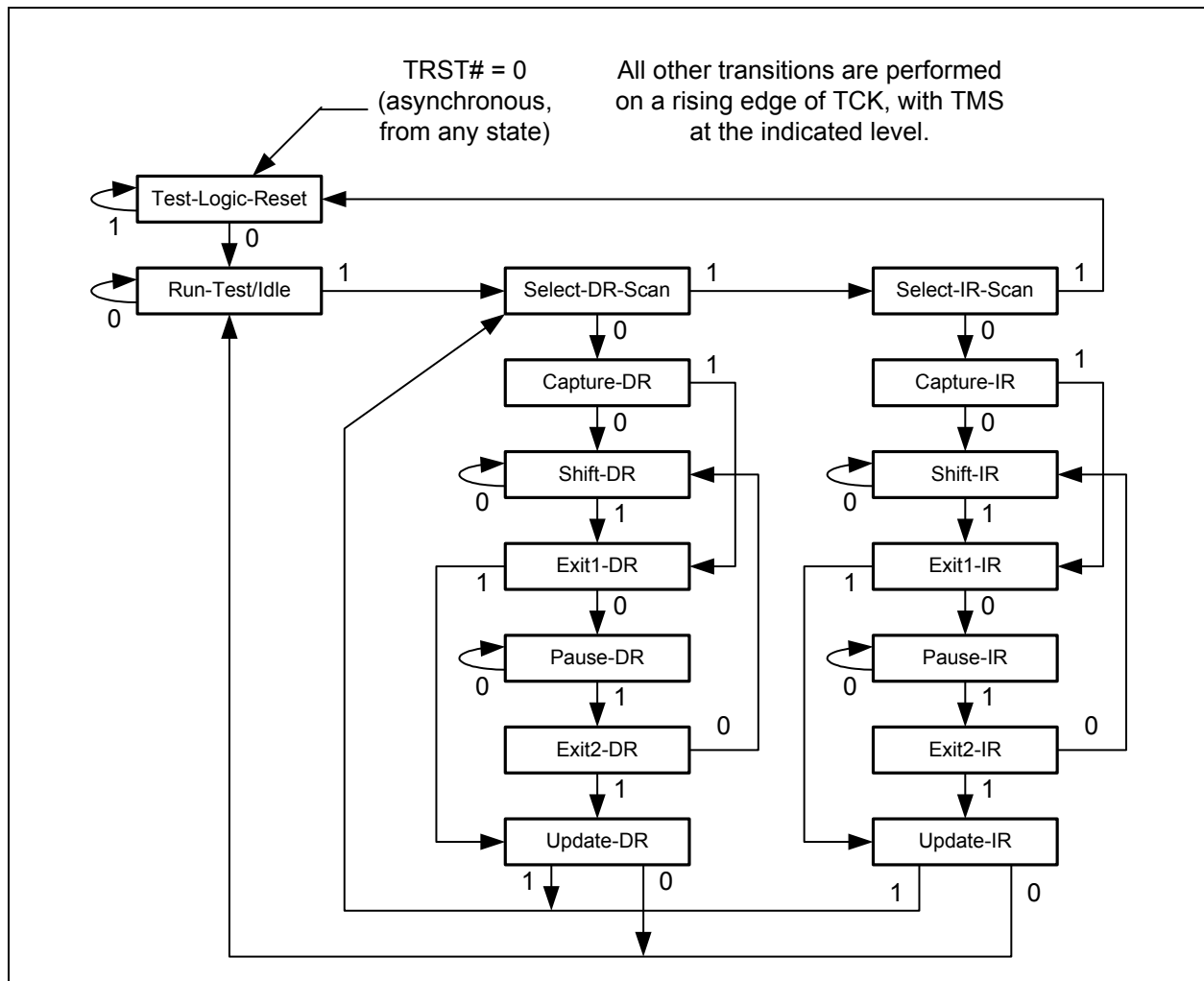
- Capture (IR or DR), which loads a shift register from its source in preparation for shifting it out. In the case of the Instruction register, this is a fixed value, and not the previous contents of the Instruction register. In the case of the BYPASS Data register, this is a fixed ‘0’ value. The Capture state is transitory, being present for only one JTAG\_CLK cycle, once per transfer.
- Shift (IR or DR), which shifts the Captured information in the Shift Register out on the TDO pin while also shifting information in from the TDI pin. The registers (by convention) shift from left to right, so the least-significant bit of a value is transferred first. This state may be held arbitrarily (holding TMS=0) to shift as many bits as desired.
- Update (IR or DR), which loads a Register from its Shift Register after the shifting has completed. The Update state is transitory, being present for only one JTAG\_CLK cycle, once per transfer.

There is also a Pause state (IR or DR) which may be used to exit and re-enter the Shift state without terminating the transfer in progress. This state may be held (TMS=0) in order to delay for any desired number of JTAG\_CLK cycles.

Outside of Instruction or Data transfers, there are two states which may be entered and held. These are shown in the leftmost column in Figure 41-12.

- The Test-Logic-Reset state holds the JTAG logic in its reset state. This re-initializes the registers that are internal to the JTAG logic. This state is entered asynchronously by assertion of JTAG\_RST# low, and it can be seen in Figure 41-12 that, from any other state, this state will be entered by 5 successive JTAG\_CLK cycles with TMS held to ‘1’.
- Run-Test/Idle holds JTAG logic idle, but not reset, between transfers.

FIGURE 41-12: TAP CONTROLLER STATE DIAGRAM



### 41.9.3 INTERFACE TIMING EXAMPLE

Figure 41-13 illustrates the timing relationship between data shifting and the TAP Controller's Shift states, using a 1-bit Data register as an example. (This is in fact the exact situation when the BYPASS Data register is selected: refer to FIGURE 41-11: on page 613.)

The TAP Controller changes states on each rising edge of JTAG\_CLK, traversing the state table in Figure 41-12 as directed by the TMS input signal from the external interface.

Previous to the waveform in Figure 41-13, the TAP Controller has already passed through a Capture-DR state, so the 1-bit Shift Register has been pre-loaded with a "Capture Value", either from its associated parallel Register or from another source. (For the BYPASS register, this would be a fixed '0'.)

At the first rising edge of JTAG\_CLK in Figure 41-13, the Shift-DR state is being entered. As yet, no valid data needs to be present on TDI or TDO.

At the first falling edge of JTAG\_CLK, while the Shift-DR state is active, the TDO pin begins presenting the least-significant bit of the Shift Register (the only bit, in this example), which is holding the Captured Value. At about this time also, the external interface will drive TDI to the desired new state for this Data register.

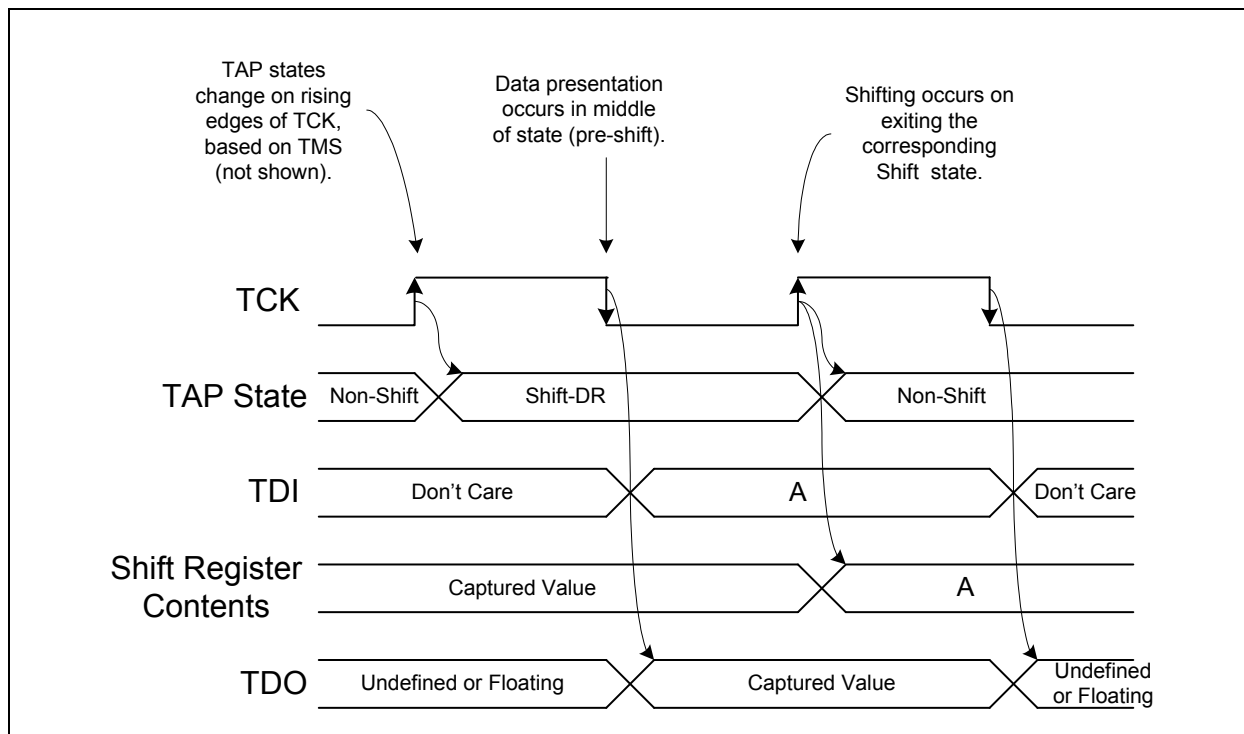
At the next rising edge of JTAG\_CLK, the Shift-DR state is exited, and that same clock edge is used to actually perform the commanded shift. The TDI value "A" is shifted into the Shift Register. This same rising edge of JTAG\_CLK is used by the external interface to shift in the Captured Value from TDO. The TDO output does not change yet, because it is held by a 1/2 clock delay stage (see FIGURE 41-11: on page 613), providing hold time for the external interface.

# MEC1618/MEC1618i

On the next falling edge, the TDO output changes. Since the Shift state is no longer present, TDO is not required at this time to present valid data, and in fact for an off-chip connection it is required to float at this time.

After this timing diagram completes, the TAP machine will continue to an Update-DR state, at which time the value A, now present in the Shift Register, will be written to its destination. (In the specific case of the BYPASS register, there is no destination, and that step will do nothing.)

**FIGURE 41-13: TIMING ILLUSTRATION: 1-BIT DATA REGISTER**



## 41.10 Registers

The ARC JTAG Port is defined in the ARC 600 External Interfaces Reference Manual, Chapter 2.

There are no JTAG registers accessible in any EC or AHB memory space. JTAG registers are accessible only through the JTAG pins themselves ([Section 41.10.1, "Instruction Register," on page 617](#)).



## 41.10.1 INSTRUCTION REGISTER

The Instruction Register is four bits wide. It selects among the implemented Data Registers as listed in [Table](#) . When the Tap Controller is placed into the Test-Logic-Reset state, the Instruction register is initialized to Ch, selecting the IDCODE Data Register.

Registers marked as MCHP reserved must not be modified. Modifications may lead to unpredictable and unwanted behavior.

**TABLE 41-7: ARC JTAG INSTRUCTION REGISTER ENCODINGS**

Instruction Register Contents	Data Register Selected	Function of Data Register	Width (Bits)	State on JTAG Reset (Hex)
0h	(Reserved: EXTEST)	Not implemented, but reserved as required by JTAG standard.	32	0000_0000h
1h	(Reserved: SAMPLE/PRELOAD)	Not implemented, but reserved as required by JTAG standard.	32	0000_0000h
2h	RESET TEST	<a href="#">RESET TEST Register (2h)</a>	32	0000_0000h
3h	TEST - MCHP Reserved		32	0000_0000h
4h	MCHP Reserved	Reserved for future use.)	32	0000_0000h
5h	MCHP Reserved	MCHP Reserved	32	0000_0000h
6h	(Reserved)	(Reserved for future use.)	32	0000_0000h
7h	(Reserved)	(Reserved for future use.)	32	0000_0000h
8h	STATUS	<a href="#">STATUS Register (8h)</a> Status of Current Debugger Transaction (Read-Only)	4	undefined (based on bus status)
9h	TRANSACTION COMMAND	<a href="#">TRANSACTION COMMAND Register (9h)</a> Initiates / Specifies a Debugger Transaction	4	3
Ah	ADDRESS	<a href="#">ADDRESS Register (Ah)</a> Address of a Debugger Transaction	32	0000_0000h
Bh	DATA	<a href="#">DATA Register (Bh)</a> Data In / Data Out for Debugger Transactions	32	Out = 0000_0000h In = undefined
Ch	IDCODE	<a href="#">IDCODE Register (Ch)</a> JTAG Standard IDCODE Register (Capture = Read-Only fixed value)	32	2000_24B1h
Dh	MCHP Reserved		32	0000_0000h
Eh	(Reserved)	Reserved for future use.	32	0000_0000h
Fh	BYPASS	<a href="#">BYPASS Register (Fh)</a> JTAG Standard BYPASS Register (Capture = Read-Only '0')	1	0

# MEC1618/MEC1618i

## 41.10.2 JTAG DEBUG DATA REGISTERS

**Note 41-3** Unfortunately, ARC names one of its JTAG Debug Data registers “DATA”. To avoid confusion, while maintaining the terminology in both ARC and JTAG documentation, the term “Data register” will refer to any of the JTAG Data registers, and the term “DATA register” (all upper-case) will refer to the specific JTAG Data register that is selected by Instruction Register = B. See [Section 41.10.2.2, “DATA Register \(Bh\),” on page 619](#).

The Debug Data Register set of the ARC JTAG Port provide the means for an external JTAG-connected debugger system to monitor and control the execution of a program. Using the JTAG Data registers ADDRESS, DATA, TRANSACTION COMMAND and STATUS, the debugger can perform “transactions” to read or write:

- Any Aux Register, giving it the ability to start, halt or step a program, and alter the PC and/or program status
- Any addressable memory or I/O location, as an aligned 32-bit value
- Any Core Register, if the processor is in a halted state

To write to a specific register or a memory location, the debugger will place the desired register number or memory address into the ADDRESS register, place the value to be written into the DATA register, and then trigger the transfer by placing the direction and addressing space (Core register / Aux register / Memory) into the TRANSACTION COMMAND register. It will then read the STATUS register until it indicates that the transaction is finished.

To read from a specific register or memory location, the debugger will place the desired register number or memory address into the ADDRESS register, and trigger the transfer by placing the direction and addressing space (Core register / Aux register / Memory) into the TRANSACTION COMMAND register. It will then read the STATUS register until it indicates that the transaction is finished, and read the DATA register to access the value.

Optimizations are possible in repeated accesses, because of the actions of the ADDRESS and DATA registers, as described in [Section 41.10.2.1](#) and [Section 41.10.2.2](#).

### 41.10.2.1 ADDRESS Register (Ah)

The ADDRESS register is a 32-bit register which receives from the debugger either a Core Register number, an Aux Register number or an address in the Memory space (memory or I/O).

**Note:** As a memory address, the low-order 2 bits of the ADDRESS register are ignored (assumed by hardware to be 00), and a full 32-bit value is referenced at that location. There is no way for the debugger to specify a smaller width of data, and so a write to a single byte (for example) is performed using a read transaction followed by a write transaction, preserving the values of the unaffected bytes.

After use, the ADDRESS register automatically increments, by 1 if a register was accessed, and by 4 if a memory location was accessed. Therefore, as long as the JTAG TAP Controller is not brought to the Test-Logic-Reset state between accesses, it is not necessary to provide a new ADDRESS register value between transactions involving successive registers or memory locations. (The Test-Logic-Reset state must be avoided because it resets the value of the ADDRESS register.)

**TABLE 41-8: ADDRESS REGISTER**

INSTRUCTION REGISTER CONTENTS	Ah			32 bits			REGISTER SIZE	
POWER	VTR			0000_0000h			Async JTAG RESET OR Sync JTAG RESET DEFAULT	
BIT	BIT31	BIT30	BIT29	...		BIT2	BIT1	BIT0
JTAG TYPE	-	-	-	-	-	-	-	-
BIT NAME	Address[31:0]							

## 41.10.2.2 DATA Register (Bh)

The DATA register is a 32-bit register which is the ARC JTAG Port's portal for data values that are being read or written by a transaction. When writing to a register or memory, the DATA register will be set up by the debugger before the transaction is triggered. When reading from a register or memory, the DATA register will be read by the debugger as the last step of the transaction. See [Note 41-2 on page 611](#).

The DATA register is not affected at the end of a write transaction, so (for example) to fill successive locations with the same value it is not necessary to provide it again, as long as the Test-Logic-Reset of the JTAG TAP Controller is not entered (which would clear it).

**TABLE 41-9: DATA REGISTER**

INSTRUCTION REGISTER CONTENTS	Bh			32 bits			REGISTER SIZE	
POWER	VTR			0000_0000h			Async JTAG RESET OR Sync JTAG RESET DEFAULT	
BIT	BIT31	BIT30	BIT29	...	BIT2	BIT1	BIT0	
JTAG TYPE	-	-	-	-	-	-	-	-
BIT NAME	Data[31:0]							

## 41.10.2.3 TRANSACTION COMMAND Register (9h)

The TRANSACTION COMMAND register is written by the debugger to trigger a transaction. It is a 4-bit register, which is written with one of the values in [Table 41-11](#) to specify the direction and addressing space of the transaction.

**TABLE 41-10: DATA REGISTER**

INSTRUCTION REGISTER CONTENTS	9h		4 bits		REGISTER SIZE
POWER	VTR		0h		Async JTAG RESET OR Sync JTAG RESET DEFAULT
BIT	BIT3	BIT2		BIT1	BIT0
JTAG TYPE	-	-		-	-
BIT NAME	Command[3:0]				

**TABLE 41-11: TRANSACTION COMMAND REGISTER ENCODINGS**

Encoding (Binary)	Transaction Type
0000	Write to Memory space
0001	Write to a Core register
0010	Write to an Aux register
0011	No Operation
0100	Read from Memory space
0101	Read from a Core register
0110	Read from an Aux register
0111	(obsolete Write form)
1000	(obsolete Read form)
(other)	Reserved

#### 41.10.2.4 STATUS Register (8h)

The STATUS register is a 4-bit read-only register. It is read by the debugger to determine when a transaction has completed internally, and when the next transaction may be started. It also provides additional status information useful to the debugger.

**TABLE 41-12: STATUS REGISTER**

INSTRUCTION REGISTER CONTENTS	8h	4 bits			REGISTER SIZE
POWER	VTR	0h			Async JTAG RESET OR Sync JTAG RESET DEFAULT
BIT	BIT3	BIT2	BIT1	BIT0	
JTAG TYPE	R	R	R	R	
BIT NAME	-PC	-RD	FL-	ST-	

(ST): STALLED

- 1 = The current transaction is stalled (busy)
- 0 = The current transaction is not stalled (not busy)

(FL): FAILURE

- 1 = The transaction has failed
- 0 = The transaction has not failed

A transaction will fail if it attempts to access a Core register while the processor is running. Bus errors should also set this bit.

(RD): READY

1 = The transaction is finished (ready)

0 = The transaction is not finished

(PC): PC\_SEL

This bit has no direct hardware effect. It displays the state of the PC\_SEL signal, which is bit 0 of the write-only Aux register PCPORT (Aux Register #24h). This bit is initialized to '1' on a processor reset, and is used internally by the debugger system as a means to communicate configuration information.

## 41.10.3 JTAG STANDARD DATA REGISTERS

### 41.10.3.1 IDCODE Register (Ch)

This is a 32-bit read-only register containing the ID value that serves to identify the ARC JTAG Port as belonging to an ARC600 core in a component containing one processor ([Table 41-13](#)).

IDCODE registers are required to conform to the JTAG standard and they include an 11-bit Manufacturer ID number.

**TABLE 41-13: IDCODE REGISTER**

INSTRUCTION REGISTER CONTENTS	Ch			32 bits			REGISTER SIZE	
POWER	VTR			2000_24B1h			Async JTAG RESET OR Sync JTAG RESET DEFAULT	
BIT	BIT31	BIT30	BIT29	...		BIT2	BIT1	BIT0
JTAG TYPE	R	R	R	R	R	R	R	R
BIT NAME	IDCODE[31:0]							

### 41.10.3.2 BYPASS Register (Fh)

The BYPASS register consists only of a 1-bit shift register cell. The Capture-DR state clears it to '0' when selected. The Update-DR state does nothing.

The function of this register is to provide the minimum amount of delay (one bit of '0') when other JTAG ports on the chain are being exercised.

# MEC1618/MEC1618i

TABLE 41-14: BYPASS REGISTER

INSTRUCTION REGISTER CONTENTS	Fh	1 bit	REGISTER SIZE
POWER	VTR	1000_24B1h	Async JTAG RESET OR Sync JTAG RESET DEFAULT
BIT	BIT0		
JTAG TYPE			
BIT NAME	BYPASS		

## 41.10.4 JTAG TEST MODE DATA REGISTERS

JTAG Test Registers are 32-bit read/write registers that are used for test functions. These registers are always available to the JTAG port, even if access to the other JTAG registers is blocked.

### 41.10.4.1 RESET TEST Register (2h)

The RESET TEST Register is a 32-bit register used to explicitly control reset functions inside the MEC1618/MEC1618i. The default for this register is 0000\_0000h.

**TABLE 41-15: RESET TEST REGISTER**

INSTRUCTION REGISTER CONTENTS	2h			32 bits			REGISTER SIZE	
POWER	VTR			0000_0000h			Async JTAG RESET DEFAULT	
BIT	Bits 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24
JTAG TYPE	R/W	R	R	R	R	R	R	R
BIT NAME	GANG_EN	Test	Reserved					
BIT	Bits 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16
JTAG TYPE	R	R	R	R	R	R	R	R
BIT NAME	Reserved							
BIT	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
JTAG TYPE	R	R/W	R	R	R	R	R/W	R/W
BIT NAME	Reserved	Test	Test	Test	Test	Test	Test	Test
BIT	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
JTAG TYPE	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
BIT NAME	Test	Test	Test	Test	POR EN	VTR POR	VCC POR	Test

## ME

Mass Erase. If this bit is '1' when the internal VTR Power On Reset signal transitions from '0' to '1', the Embedded Flash Subsystem will enter the [Emergency Mass Erase](#) mode, which will erase the entire Flash whether or not the [Boot\\_JTAG\\_Block](#) bit in the [Embedded Flash Initialization Register](#) is set.

## VCC POR

Asserts VCC Power On Reset: When the [VCC POR](#) active low bit is asserted '0' while the field [POR EN](#) in this register is '1', forces a VCC Power On Reset. When the [VCC POR](#) active low bit de-asserted '1', the VCC POR circuitry returns to its normal state.

## VTR POR

Asserts VTR Power On Reset: When the [VTR POR](#) active low bit is asserted '0' while the field [POR EN](#) in this register is '1', forces a VTR Power On Reset. When the [VTR POR](#) active low bit de-asserted '1', the VCC POR circuitry returns to its normal state.

## POR EN

Power On Reset Enable. When '1', the reset functions controlled by [VCC POR](#) and [VTR POR](#) are enabled. When '0', the [VCC POR](#) and [VTR POR](#) fields in this register have no effect on the POR circuitry.

## TEST

All TEST bits should be set to '0' when writing this register.

# MEC1618/MEC1618i

## GANG\_EN

When the **GANG\_EN** bit is asserted '1' and the ME bit (also in the [RESET TEST Register](#)) is '0' when the internal Power On Reset transitions from '0' to '1', the [Gang Programmer Interface](#) is enabled. In this mode, the [Embedded Flash Sub-system](#) is permitted to complete its initialization sequence, and the EC is held in Reset. See [Section 40.0, "Gang Programmer Interface," on page 592](#) for details regarding the operation of, and the multiplexing of pins in the [Gang Programmer Interface](#).

### 41.10.4.2 TEST REGISTER 4/Reset Register (Dh)

The RESET TEST Register is a 32-bit register used to explicitly control reset functions inside the MEC1618/MEC1618i. The default for this register is 0000\_0000h.

**TABLE 41-16: TEST REGISTER 4/RESET REGISTER**

INSTRUCTION REGISTER CONTENTS	Dh			32 bits			REGISTER SIZE	
POWER	VTR			0000_0000h			Async JTAG RESET DEFAULT	
BIT	BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24
JTAG TYPE	R/W	R/W	R	R	R	R/W	R/W	R
BIT NAME	Test		Reserved			Test	Test	Test
BIT	BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
JTAG TYPE	R	R	R/W	R/W	R/W	R/W	R/W	R
BIT NAME	Test	Test	Test	Test	Test	Test	Test	Rsrvd
BIT	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
JTAG TYPE	R/W	R	R	R	R/W	R/W	R/W	R/W
BIT NAME	Test	Reserved		Test	Test	Test		
BIT	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
JTAG TYPE	R	R	R	R/W	R/W	R/W	R/W	R/W
BIT NAME	Test	Test	Test	Test	Test	Test	Test_XNOR_En	ARC_Fast_Reset

## ARC\_FAST\_RESET

If this bit is '1b', the reset going to the ARC processor and select peripherals is reduced from its nominal 20ms duration. If this bit is '0b', the ARC reset is stretched by the nominal delay.

## TEST\_XNOR\_EN

If this bit is '1b', the Device-Under-Test XNOR chain test mode is enabled. If this bit is '0b', the XNOR mode is disabled. See [Section 41.11, "XNOR Chain," on page 625](#).



**Note:** After the XNOR chain is enabled, a power cycle is required to re-establish JTAG operation or.

## TEST

All TEST bits should be set to '0' when writing this register.

### 41.10.5 JTAG STANDARD PORT DISCOVERY

This section provides information that is not unique to ARC, but is part of the JTAG standard, and is provided for information.

The Discovery process will identify each JTAG controller that has an IDCODE register. Part of what needs to be derived is the length of the Instruction register in each of the JTAG ports. If this cannot be derived from the IDCODE values, or if some JTAG ports do not have an IDCODE register, then the missing lengths must be provided by other means.

In the Test-Logic-Reset state, a JTAG port is required to initialize its Instruction register to select the IDCODE Data register if present, or if it is not present, then to select the BYPASS Data register.

The IDCODE Data register:

- Must be exactly 32 bits in length
- Must have '1' in its first (least-significant) bit
- Must not have the pattern 000011111111 (FFh) in its first (least-significant) 12 bits.
- Will contain a completely definitive port identification, because 11 bits of it are a Manufacturer ID number assigned by the JEDEC standards organization.

A BYPASS Data register access will initialize its 1-bit shift register to '0' at the Capture-DR state, effectively making the BYPASS register appear to be 1-bit read-only '0'.

Discovery, therefore, consists of the external JTAG host doing the following:

- Place the chain of JTAG controllers into the Test-Logic-Reset state.
- Do a Data register access, without an Instruction register access first.
- This data access shifts in 8 bits of ones, followed by all zeroes for the duration of the discovery phase.
- While shifting, examine the data appearing on TDO for IDCODE values.
- A '0' indicates a JTAG port that has no IDCODE register. Collect only this bit, and note that the JTAG port exists. Start looking for an IDCODE value at the next bit.
- A '1' indicates that an IDCODE register is coming. Collect this bit and the next 31 bits to identify the JTAG port. If, however, the value seen is 00h0000FF, then this is guaranteed to be the value provided originally on TDI, and indicates the end of the chain.

## 41.11 XNOR Chain

### 41.11.1 OVERVIEW

The [XNOR Chain](#) test mode allows users to confirm that all MEC1618/MEC1618i pins are in contact with the motherboard during assembly and test operations. The [XNOR Chain](#) test mode is enabled and disabled through the JTAG interface, using bit [Test\\_XNOR\\_En](#) in JTAG [TEST REGISTER 4/Reset Register \(Dh\)](#).

An example of an [XNOR Chain](#) test structure is illustrated below in [Figure 41-14](#). When the [XNOR Chain](#) test mode is enabled all pins except for the [Excluded Pins](#) shown in [Section 41.11.2](#) are disconnected from their internal functions and forced as inputs to the [XNOR Chain](#). This allows a single input pin to toggle the [XNOR Chain](#) output if all other input pins are held high or low. The [XNOR Chain](#) output is the nRESET\_OUT pin.

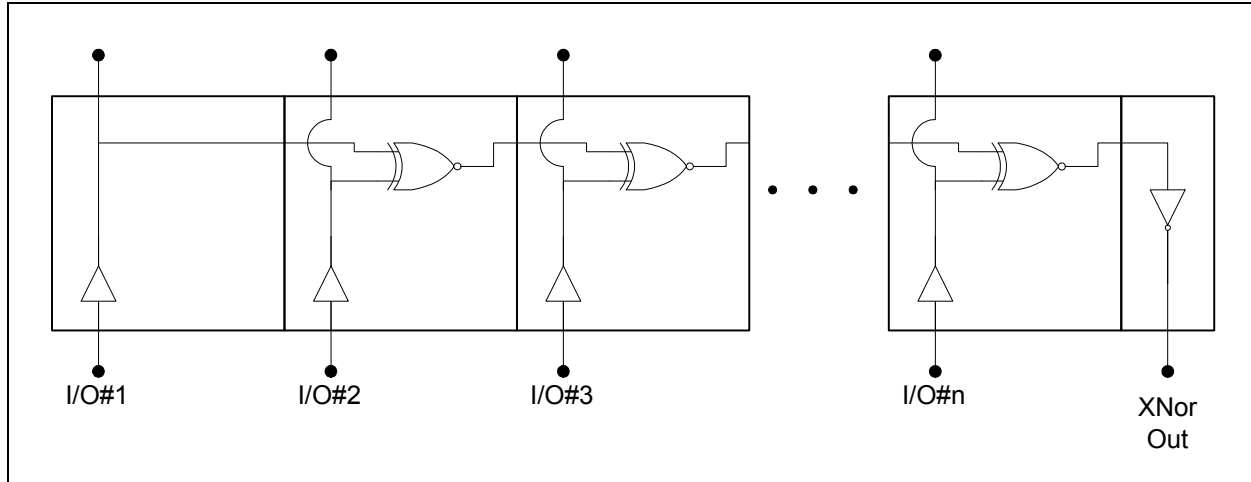
The tests that are performed when the [XNOR Chain](#) test mode is enabled require the board-level test hardware to control the device pins and observe the results at the [XNOR Chain](#) output pin; e.g., as described in [Section 41.11.3, "Test Procedure," on page 626](#).

# MEC1618/MEC1618i

## 41.11.2 EXCLUDED PINS

The following pins are **XNOR Chain Excluded Pins**: POWER PLANE pins, VR\_CAP, nRESET\_OUT, and JTAG\_RST# (see [Section 2.4.4, "JTAG Interface,"](#) on page 16).

**FIGURE 41-14: XNOR CHAIN TEST STRUCTURE**



## 41.11.3 TEST PROCEDURE

### 41.11.3.1 Setup

1. Connect the VSS and AGND pins to ground.
2. Connect the **VBAT** and **VTR** pins to an unpowered 3.3V power source.
3. Connect an oscilloscope or voltmeter to the nRESET\_OUT pin.
4. All other pins should be tied to ground.

**Warning:** Ensure power supply is off during Setup.

### 41.11.3.2 Testing

1. Turn on the 3.3V power source.
2. Enable the **XNOR Chain** through the JTAG interface (**Test\_XNOR\_En** in JTAG **TEST REGISTER 4/Reset Register (Dh)**). Note that at this point all inputs to the **XNOR Chain** are low and the output on the nRESET\_OUT pin is high (refer to the **Initial Configuration** row in [Table 41-17, "Toggling Inputs in Descending Pin Order"](#)).
3. Bring the highest numbered pin (N) high, where N is the number of pins to be tested as described in [Note 41-4](#). The output on the nRESET\_OUT pin should toggle (refer to [Step 1](#) in [Table 41-17](#)).
4. In descending pin order successively bring each input high. As shown in [Table 41-17](#) the nRESET\_OUT pin toggles after each step. Continue until all inputs are high. The output on the nRESET\_OUT pin is high (refer to the **Final Configuration** in [Table 41-17](#)).
5. The current state of the chip is now represented by the **Initial Configuration** row in [Table 41-18, "Toggling Inputs in Ascending Pin Order"](#).
6. Each input should now be brought low, starting at pin one ([Step N+1](#)) and continuing in ascending pin order until all inputs are low. The output on the nRESET\_OUT pin is high (refer to the **Final Configuration** in [Table 41-18](#)).
7. Exit the **XNOR Chain** Test Mode by cycling VTR power.

**TABLE 41-17: TOGGLING INPUTS IN DESCENDING PIN ORDER**

	Pin Number (Note 41-4)							nRESET_OUT
	N	N - 1	N - 2	N - 3	N - 4	...	1	
Initial Configuration	L	L	L	L	L	L	L	H
Step 1	H	L	L	L	L	L	L	L
Step 2	H	H	L	L	L	L	L	H
Step 3	H	H	H	L	L	L	L	L
Step 4	H	H	H	H	L	L	L	H
Step 5	H	H	H	H	H	L	L	L
...	H	H	H	H	H	...	L	...
Step N-1	H	H	H	H	H	H	L	L
Final Configuration	H	H	H	H	H	H	H	H

**TABLE 41-18: TOGGLING INPUTS IN ASCENDING PIN ORDER**

	Pin Number (Note 41-4)							nRESET_OUT
	1	2	3	4	5	...	N	
Initial Configuration	H	H	H	H	H	H	H	H
Step N+1	L	H	H	H	H	H	H	L
Step N+2	L	L	H	H	H	H	H	H
Step N+3	L	L	L	H	H	H	H	L
Step N+4	L	L	L	L	H	H	H	H
Step N+5	L	L	L	L	L	H	H	L
...	L	L	L	L	L	...	H	...
Step N+(N-1)	L	L	L	L	L	L	H	L
Final Configuration	L	L	L	L	L	L	L	H

**Note 41-4** Pin numbers in these tables represent the number of pins to be tested and do not include the pins listed in [Section 41.11.2, "Excluded Pins,"](#) on page 626.

# MEC1618/MEC1618i

## 42.0 ELECTRICAL SPECIFICATIONS

### 42.1 Maximum Ratings\*

\*Stresses exceeding those listed could cause permanent damage to the device. This is a stress rating only and functional operation of the device at any other condition above those indicated in the operation sections of this specification is not implied.

**Note:** When powering this device from laboratory or system power supplies, it is important that the Maximum Ratings not be exceeded or device failure can result. Some power supplies exhibit voltage spikes on their outputs when the AC power is switched on or off. In addition, voltage transients on the AC power line may appear on the DC output. If this possibility exists, it is suggested that a clamp circuit be used.

#### 42.1.1 MAXIMUM THERMAL RATINGS

**TABLE 42-1: MAXIMUM THERMAL RATINGS**

Parameter	Maximum Limits
Operating Temperature Range (Commercial)	0° C to +70° C
Operating Temperature Range (Industrial)	-40° C to +85° C
Storage Temperature Range	-55° C to +150° C
Lead Temperature Range	Refer to JEDEC Spec J-STD-020B

#### 42.1.2 MAXIMUM SUPPLY VOLTAGE RATINGS

**TABLE 42-2: POWER SUPPLY RATINGS**

Symbol	Parameter	Maximum Limits
VBAT	Battery Backup Power Supply	4V
VTR	VTR Power Supply	4V
AVTR_ADC	Analog VTR Supply	4V
VREF_ADC	ADC Voltage Reference Pin	4V

#### 42.1.3 MAXIMUM I/O VOLTAGE RATINGS

Parameter	Maximum Limits
Voltage with respect to ground on any signal pin without backdrive protection	-Determined by Power Supply I/O Buffer ( <a href="#">Note 42-1</a> )
Voltage with respect to ground on any signal pin without 5.0V Tolerance	-0.3 to 3.63V
Voltage with respect to ground on any signal pin with 5.0V Tolerance	-0.3 to 5.5V

**Note 42-1** On any signal pin without backdrive protection, the voltage level on the pin must never exceed the voltage level of the power supply used to power the buffer.

## 42.2 Operational Specifications

### 42.2.1 POWER SUPPLY OPERATIONAL CHARACTERISTICS

**TABLE 42-3: POWER SUPPLY OPERATING CONDITIONS**

Symbol	Parameter	Min	Typ	Max	Units
VBAT	Battery Backup Power Supply	2.0	3.0	3.6	V
VTR	VTR Power Supply	2.97	3.3	3.63	V
AVTR_ADC	Analog VTR Supply	2.97	3.3	3.63	V
VREF_ADC	ADC Voltage Reference Pin	2.97	3.3	3.63	V
AVTR_TMI	Thermal Monitoring Interface	2.97	3.3	3.63	V

### 42.2.2 CAPACITIVE LOADING SPECIFICATIONS

The following table defines the maximum capacitive load validated for the buffer characteristics listed in [Table 42-4, "DC Electrical Characteristics," on page 630](#)

CAPACITANCE  $T_A = 25^\circ\text{C}$ ;  $f_c = 1\text{MHz}$ ;  $V_{CC} = 3.3\text{VDC}$

**Note:** All output pins, except pin under test, tied to AC ground.

Parameter	Symbol	Limits			Units	Notes
		Min	Typ	Max		
Input Capacitance of PCI_I and PCI_IO pins	$C_{IN}$			<a href="#">Note 42-2</a>	pF	
Input Capacitance of PCI_CLK pin	$C_{IN}$			<a href="#">Note 42-2</a>	pF	
Output Load Capacitance supported by PCI_IO, PCI_O, and PCI_OD	$C_{OUT}$			<a href="#">Note 42-2</a>	pF	
Input Capacitance of Peci_IO	$C_{IN}$			10	pF	
Output Load Capacitance supported by Peci_IO	$C_{OUT}$			10	pF	
Input Capacitance (all other input pins)	$C_{IN}$			10	pF	<a href="#">Note 42-3</a>
Output Capacitance (all other output pins)	$C_{OUT}$			20	pF	<a href="#">Note 42-4</a>

**Note 42-2** The PCI buffers are designed to meet the defined PCI Local Bus Specification, Rev. 3.0, electrical requirements.

**Note 42-3** All input buffers can be characterized by this capacitance unless otherwise specified.

**Note 42-4** All output buffers can be characterized by this capacitance unless otherwise specified.

# MEC1618/MEC1618i

## 42.2.3 DC ELECTRICAL CHARACTERISTICS FOR I/O BUFFERS

**TABLE 42-4: DC ELECTRICAL CHARACTERISTICS**

Parameter	Symbol	Min	Typ	Max	Units	Comments
I <sub>AN</sub>	R <sub>IN</sub>			75	m Ohm	Maximum DC current carrying capability = 84mA
I Type Input Buffer						TTL Levels
Low Input Level	V <sub>ILI</sub>			0.8	V	
High Input Level	V <sub>IH</sub>	2.0			V	
Input Leakage	I <sub>IL</sub>	-10		+10	uA	V <sub>IN</sub> = 0 to Signal Power Well
IS Type Input Buffer						
Low Input Level	V <sub>ILIS</sub>			0.8	V	Schmitt Trigger
High Input Level	V <sub>IHIS</sub>	2.2			V	Schmitt Trigger
Schmitt Trigger Hysteresis	V <sub>HYS</sub>		250		mV	
Input Leakage	I <sub>I-Leak</sub>	-10		+10	uA	V <sub>IN</sub> = 0 to Signal Power Well
O-4mA Type Buffer						
Low Output Level	V <sub>OL</sub>			0.4	V	I <sub>OL</sub> = 4mA
High Output Level	V <sub>OH</sub>	2.4			V	I <sub>OH</sub> = -4mA
Output Leakage	I <sub>OL</sub>	-10		+10	μA	V <sub>IN</sub> = 0 to Signal Power Well
IO-4mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an O-4mA.
OD-4mA Type Buffer						
Low Output Level	V <sub>OL</sub>			0.4	V	V <sub>OL</sub> = 4mA
Output Leakage	I <sub>OH</sub>	-10		+10	μA	I <sub>OH</sub> = 0 to Signal Power Well
IOD-4mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an OD-4mA.
O-8mA Type Buffer						
Low Output Level	V <sub>OL</sub>			0.4	V	I <sub>OL</sub> = 8mA
High Output Level	V <sub>OH</sub>	2.4			V	I <sub>OH</sub> = -8mA
Output Leakage	I <sub>OL</sub>	-10		+10	μA	V <sub>IN</sub> = 0 to Signal Power Well
IO-8mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an O-8mA.
OD-8mA Type Buffer						
Low Output Level	V <sub>OL</sub>			0.4	V	V <sub>OL</sub> = 8mA
Output Leakage	I <sub>OH</sub>	-10		+10	μA	I <sub>OH</sub> = 0 to Signal Power Well

**TABLE 42-4: DC ELECTRICAL CHARACTERISTICS (CONTINUED)**

Parameter	Symbol	Min	Typ	Max	Units	Comments
IOD-8mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an OD-8mA.
O-12mA Type Buffer						
Low Output Level	$V_{OL}$			0.4	V	$I_{OL} = 12\text{mA}$
High Output Level	$V_{OH}$	2.4			V	$I_{OH} = -12\text{mA}$
Output Leakage	$I_{OL}$	-10		+10	$\mu\text{A}$	$V_{IN} = 0$ to Signal Power Well
IO-12mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an O-12mA.
OD-12mA Type Buffer						
Low Output Level	$V_{OL}$			0.4	V	$I_{OL} = 12\text{mA}$
Output Leakage	$I_{OL}$	-10		+10	$\mu\text{A}$	$V_{IN} = 0$ to Signal Power Well
IOD-12mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an OD-12mA.
O-16mA Type Buffer						
Low Output Level	$V_{OL}$			0.4	V	$I_{OL} = 16\text{mA}$
High Output Level	$V_{OH}$	2.4			V	$I_{OH} = -16\text{mA}$
Output Leakage	$I_{OL}$	-10		+10	$\mu\text{A}$	$V_{IN} = 0$ to Signal Power Well
IO-16mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an O-16mA.
OD-16mA Type Buffer						
Low Output Level	$V_{OL}$			0.4	V	$I_{OL} = 16\text{mA}$
Output Leakage	$I_{OL}$	-10		+10	$\mu\text{A}$	$V_{IN} = 0$ to Signal Power Well
IOD-16mA Type Buffer	—	—	—	—	—	Same characteristics as an I and an OD-16mA.

# MEC1618/MEC1618i

**TABLE 42-4: DC ELECTRICAL CHARACTERISTICS (CONTINUED)**

Parameter	Symbol	Min	Typ	Max	Units	Comments
PCI Buffers (PCI_ICLK, PCI_IO, PCI_I, PCI_O, PCI_OD)	$V_{IH}$	$0.5V_{TR}$		$V_{TR} + 0.5$	V	See <i>PCI Local Bus Specification</i> Rev. 2.2  See <a href="#">Note 42-6</a>
	$V_{IL}$	-0.5		$0.3V_{TR}$	V	
	$V_{TR}$	3.0		3.6	V	LPC Supply Voltage
	$I_{IL}$	-10		+10	$\mu A$	$0 < V_{IN} < V_{CC}$
	$V_{OH}$	$0.9V_{TR}$			V	$I_{OUT} = -500 \mu A$
	$V_{OL}$			$0.1V_{TR}$	V	$I_{OUT} = 1500 \mu A$
	$C_{IN}$			10	pF	
$V_{REF}$ Buffer						
PECI Bus Voltage	$V_{BUS}$	0.95		1.26	V	
SB-TSI Bus Voltage	$V_{BUS}$	1.425		1.8	V	
Input current	IDC			100	$\mu A$	
Input Low Current	ILEAK	-10		+10	$\mu A$	



**TABLE 42-4: DC ELECTRICAL CHARACTERISTICS (CONTINUED)**

Parameter	Symbol	Min	Typ	Max	Units	Comments
IO-PECI						
Input voltage range	VIn	-0.3		$V_{REF} + 0.3$	V	All input and output voltages are a function of VREF_VTT ( $V_{REF}$ ) buffer input. See Peci Specification.
Hysteresis	VHYS	$0.1 \times V_{REF}$	$0.2 \times V_{REF}$		V	
Low Input VLevel	VIL			$0.275 \times V_{REF}$	V	
High Input Level	VIH	$0.725 \times V_{REF}$			V	
Low Output Level	VOL			$0.25 \times V_{REF}$	V	$0.5mA < IOL < 1mA$
High Output Level	VOH	$0.75 \times V_{REF}$			V	$IOH = -6mA$
SB-TSI						
Input voltage range	VIn	-0.3		$V_{REF} + 0.3$	V	All input and output voltages are a function of VREF_VTT ( $V_{REF}$ ) buffer input.
Hysteresis	VHYS	$0.1 \times V_{REF}$	$0.2 \times V_{REF}$		V	
Low Input VLevel	VIL			$0.275 \times V_{REF}$	V	
High Input Level	VIH	$0.725 \times V_{REF}$			V	
Low Output Level	VOL			$0.25 \times V_{REF}$	V	$0.5mA < IOL < 1mA$
Pull-Down Impedance	PD	65	91	136	K Ohms	
Pull-Up Impedance	PU	53	74	110	K Ohms	

**Note 42-5** Non 5V Tolerant pins are listed in [Table 2-4, "Non 5 Volt Tolerant Pins," on page 12.](#)

**Note 42-6** All Signal functions in [Section 2.5, "Pin Multiplexing," on page 23](#) listed as "VCC" emulated power are tri-stated when the input [VCC\\_PWRGD](#) is inactive.

**Note 42-7** All 5V Tolerant I-type & I/O-type input buffers can be pulled to 5 volts.

**Note 42-8** All 5V Tolerant OD-type output buffers can be pulled to 5 volts.

**Note 42-9** All 5V Tolerant O-type and I/O-type output buffers will only drive to 3.3 volts, even if pulled-up externally to 5 volts.

# MEC1618/MEC1618i

## 42.2.3.1 Backdrive Protection

All MEC1618/MEC1618i signal pins are Backdrive Protected except those listed in [Section 2.3.5, "Non Backdrive Protected Pins,"](#) on page 14.

**TABLE 42-5: Backdrive Protection**

(T<sub>A</sub> = 0° C – 70° C Commercial, -40° C – 85° C Industrial)

Parameter	Symbol	Min	Typ	Max	Units	Comments
Input Leakage	I <sub>IL</sub>	-10		+10	μA	V <sub>IN</sub> = 5.0 V (5.5 V) @ V <sub>TR</sub> = 0 V (Note 42-10)

**Note 42-10** for V<sub>BAT</sub> powered pins, V<sub>TR</sub> = 0 V, V<sub>BAT</sub> = 0 V.

## 42.3 Power Consumption

**TABLE 42-6: MEC1618/MEC1618I POWER CONSUMPTION**

VCC2 (V <sub>CC</sub> )	VCC1 (V <sub>TR</sub> )	System "S" State	ARC State	Clock State	Supply Current (mA, unless noted otherwise)				Comments
						Typ (25°C)	Max (70°C)	Max (85°C)	
3.3V	3.3V	S0–S2	Run	20 MHz	ICC1	–	23	24	ABSOLUTE MAXIMUM
3.3V	3.3V	S0–S2	Run	20 MHz	ICC1	11.25	13.50	14.50	FULL POWER (Note 42-13), Note 42-11)
				10 MHz		8.25	10.10	11.00	
				2 MHz		5.50	7.25	8.00	
3.3V	3.3V	S0–S2	Sleep	2 MHz	ICC1	4.85	6.75	7.50	EC SLEEP All Master Clock Trees except EC Clock are 'on.'
						3.95	5.60	6.35	SYSTEM LIGHT SLEEP One Master Clock Trees is 'on.'
						3.30	5.30	6.05	SYSTEM HEAVY SLEEP 1
						3.20	5.20	5.95	SYSTEM HEAVY SLEEP 2 (Clock Tree Gating in Heavy Sleep)
						3.10	5.10	5.85	SYSTEM HEAVY SLEEP 3 (Clock Tree Gating in Heavy Sleep)

**TABLE 42-6: MEC1618/MEC1618I POWER CONSUMPTION (CONTINUED)**

VCC2 (V <sub>CC</sub> )	VCC1 (V <sub>TR</sub> )	System “S” State	ARC State	Clock State	Supply Current (mA, unless noted otherwise)				Comments
						Typ (25 <sup>0</sup> C)	Max (70 <sup>0</sup> C)	Max (85 <sup>0</sup> C)	
0V	3.3V	S3	Run	20 MHz	ICC1	10.50	12.25	13.25	FULL POWER (Note 42-14, Note 42-11)
				10 MHz		7.50	9.00	10.00	
				2 MHz		4.80	6.25	7.25	
0V	3.3V	S3*	Sleep	2 MHz	ICC1	3.10	4.00	5.00	SYSTEM LIGHT SLEEP
						2.30	3.50	4.50	SYSTEM HEAVY SLEEP 1
						0.60	1.30	2.50	SYSTEM HEAVY SLEEP 2 (Clock Tree Gating in Heavy Sleep)
						0.25	1.00	2.10	SYSTEM HEAVY SLEEP 3 (Clock Tree Gating in Heavy Sleep)
0V	3.3V	S3*	Sleep	Stop	ICC1	0.20	0.90	2.00	SYSTEM DEEPEST SLEEP Note 42-16, Note 42-17
0V	3.3V	S3*	Sleep	Stop (External 32 kHz Clock Input)	ICC0	1.00 μA	4.00 μA	6.00 μA	2.0V < V <sub>bat</sub> < 3.0V Note 42-16, Note 42-17
0V	0V	S5	Off	Local 32kHz Oscillator	ICC0	9.00 μA	17.00 μA @ 25°C	–	2.0V < V <sub>bat</sub> < 3.0V
				32kHz Disabled	ICC0	2.20 μA	10.00 μA @ 25°C	–	2.0V < V <sub>bat</sub> < 3.0V
* On AC power, System can enter the S3-S5 states when EC is in sleep mode. ICC0 is V <sub>BAT</sub> current; ICC1 is V <sub>TR</sub> current.									

**Note:** All inputs not being tested are pulled up to power rails; all outputs are floating.

**Note 42-11** ADC enabled

**Note 42-12** All Master Clock Trees are running; one block is enabled in each clock tree as follows: EC Clock: GPIO; EC Clock: 16-Bit Counter/Timer; EC Clock: PWM0; EC Clock: PECI; Host Clock Tree 0: LPC; Host Clock Tree 1: UART.

**Note 42-13** All Master Clock Trees are running; one block is enabled in each clock tree as follows: EC Clock: GPIO; EC Clock: 16-Bit Counter/Timer; EC Clock: PWM0; EC Clock: PECI; Host Clock Tree 0: LPC; Host Clock Tree 1: UART.

# MEC1618/MEC1618i

---

- Note 42-14** All EC [Master Clock Trees](#) are running as described in [Note 42-13](#); [Host Clock Tree 0](#) and [Host Clock Tree 1](#) are 'off.'
- Note 42-15** the MEC1618/MEC1618i Power Management (PM) States are described in [Section 5.4.8.2, "EC Controlled Dynamic Power States,"](#) on page 90.
- Note 42-16** To achieve the lowest possible suspend current the VREF\_VTT pin must be grounded even if the VREF\_VTT function is not selected using the [GPIO Interface Pin Control Register](#).
- Note 42-17** to achieve the lowest possible suspend current the [VREG SUS](#) bit must be asserted (see [Section 5.7.8, "VREG Control Register,"](#) on page 130).

## 43.0 TIMING DIAGRAMS

### 43.1 VTR/VBAT Power-Up and Power-Down Timing

FIGURE 43-1: VTR/VBAT POWER-UP

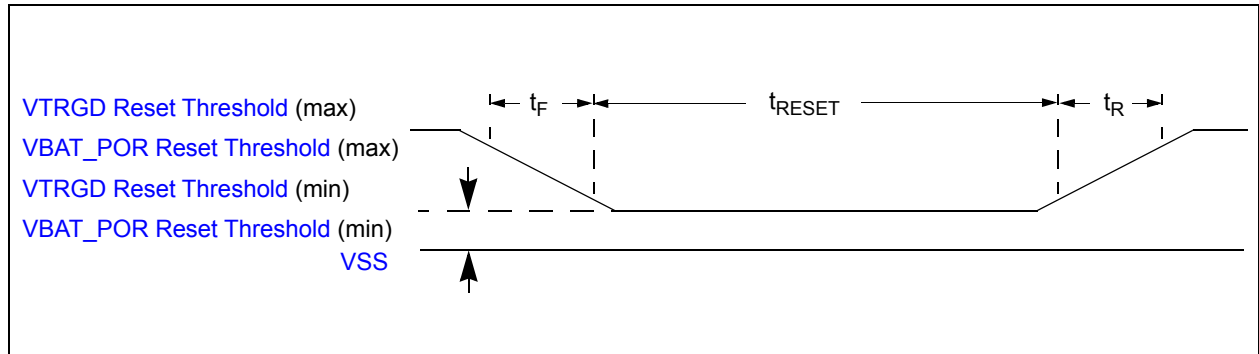


TABLE 43-1: VTR/VBAT POWER-UP TIMING PARAMETERS

Symbol	Parameter	Limits		Units	Comments
		Min	Max		
$t_F$	VTR/VBAT Fall time	30		$\mu\text{s}$	
$t_R$	VTR/VBAT Rise time	0.150	30	ms	
$t_{\text{RESET}}$	Minimum Reset Time	1		$\mu\text{s}$	

### 43.2 Clock and Reset Timing

FIGURE 43-2: PCI CLOCK TIMING

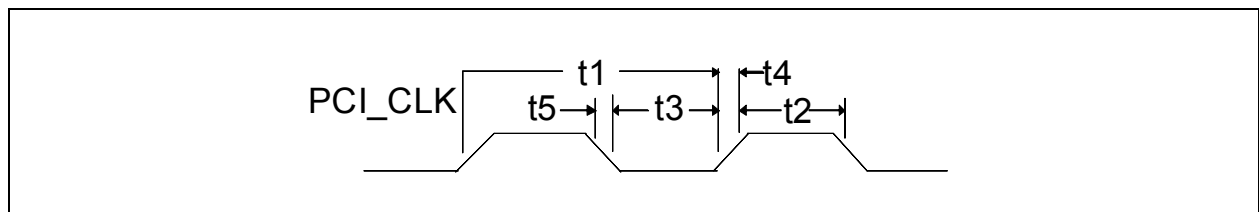


TABLE 43-2: PCI CLOCK TIMING PARAMETERS

Name	Description	Min	Typ	Max	Units
t1	Period	30		33.3	nsec
t2	High Time	11			
t3	Low Time				
t4	Rise Time			3	
t5	Fall Time				

FIGURE 43-3: RESET TIMING

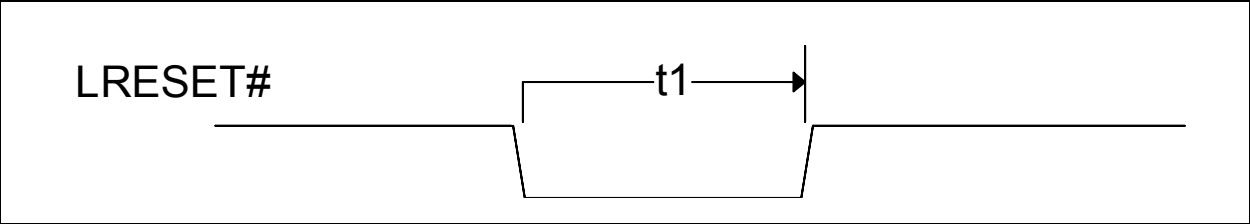


TABLE 43-3: RESET TIMING PARAMETERS

Name	Description	Min	Typ	Max	Units
t1	LRESET# width	1			ms

43.3 LPC Timing

FIGURE 43-4: OUTPUT TIMING MEASUREMENT CONDITIONS, LPC SIGNALS

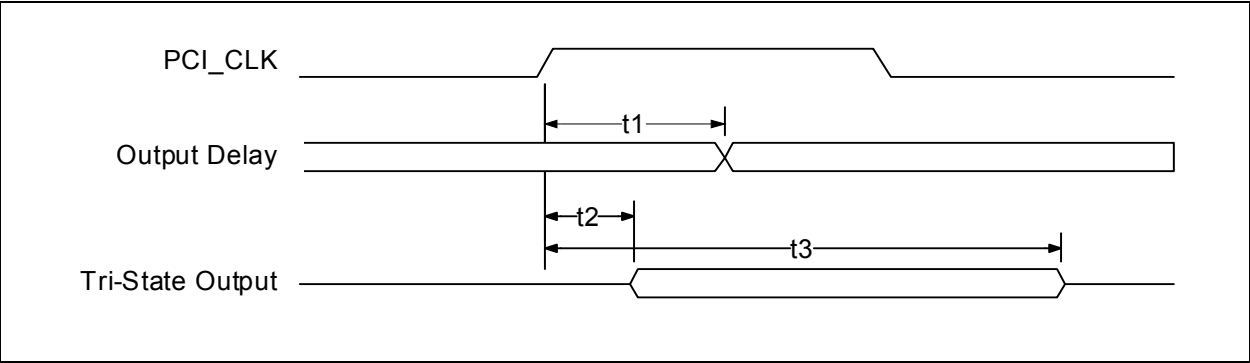
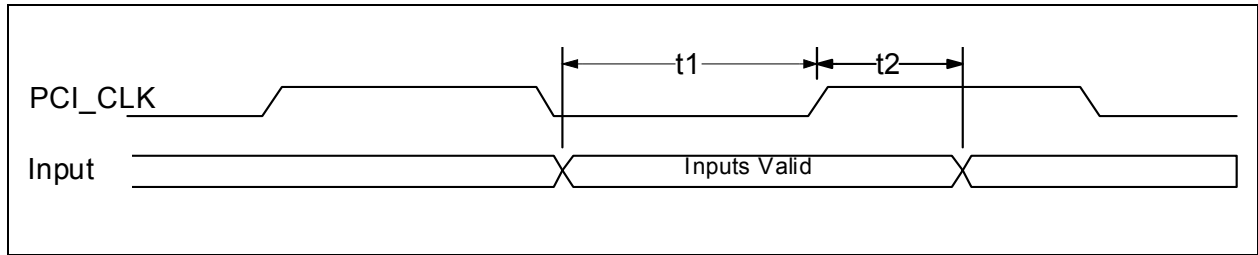


TABLE 43-4: OUTPUT TIMING MEASUREMENT CONDITIONS, LPC SIGNALS PARAMETERS

Name	Description	Min	Typ	Max	Units
t1	PCI_CLK to Signal Valid Delay – Bused Signals	2		11	ns
t2	Float to Active Delay				
t3	Active to Float Delay			28	

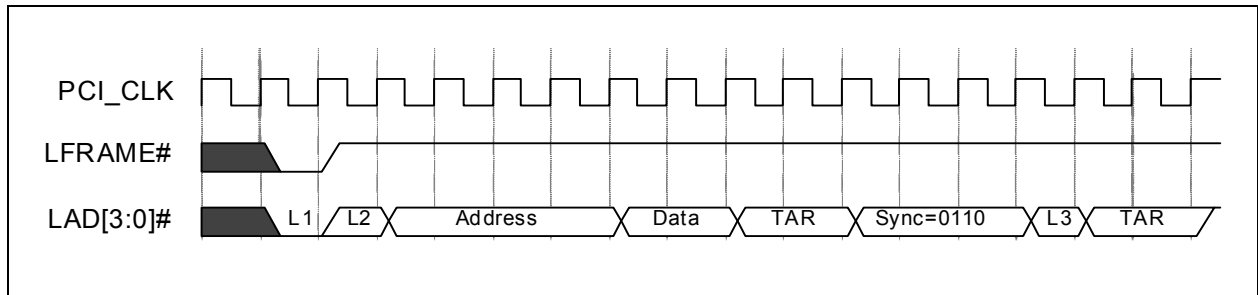
**FIGURE 43-5: INPUT TIMING MEASUREMENT CONDITIONS, LPC SIGNALS**



**TABLE 43-5: INPUT TIMING MEASUREMENT CONDITIONS, LPC SIGNALS PARAMETERS**

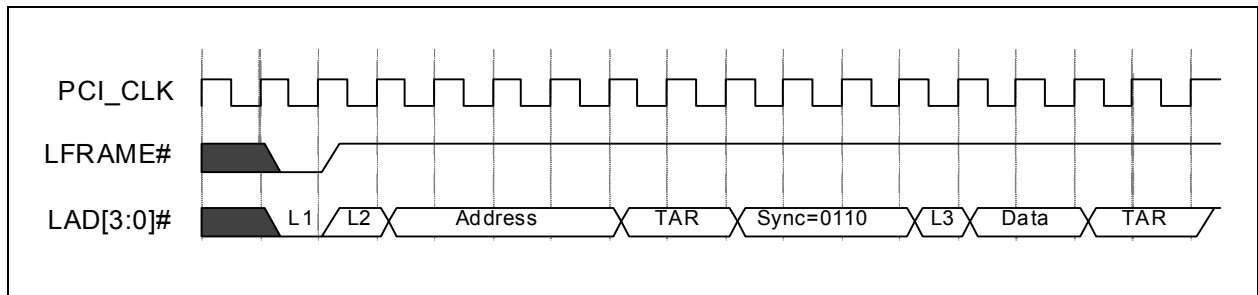
Name	Description	Min	Typ	Max	Units
t1	Input Set Up Time to PCI_CLK – Bused Signals	7			ns
t2	Input Hold Time from PCI_CLK	0			

**FIGURE 43-6: I/O WRITE**



**Note:** L1=Start; L2=CYCTYP+DIR; L3=Sync of 0000

**FIGURE 43-7: I/O READ**



**Note:** L1=Start; L2=CYCTYP+DIR; L3=Sync of 0000

# MEC1618/MEC1618i

FIGURE 43-8: DMA REQUEST ASSERTION THROUGH LDRQ#

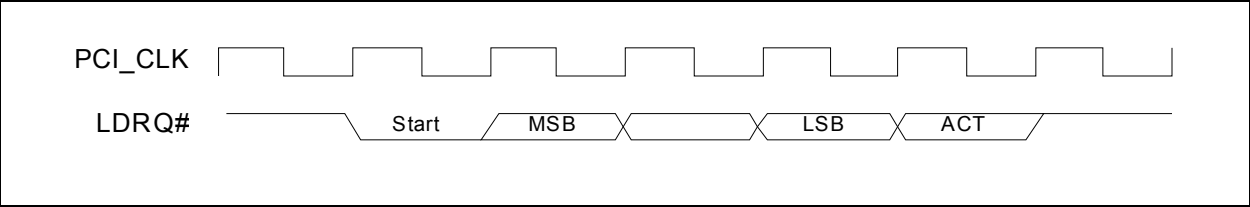
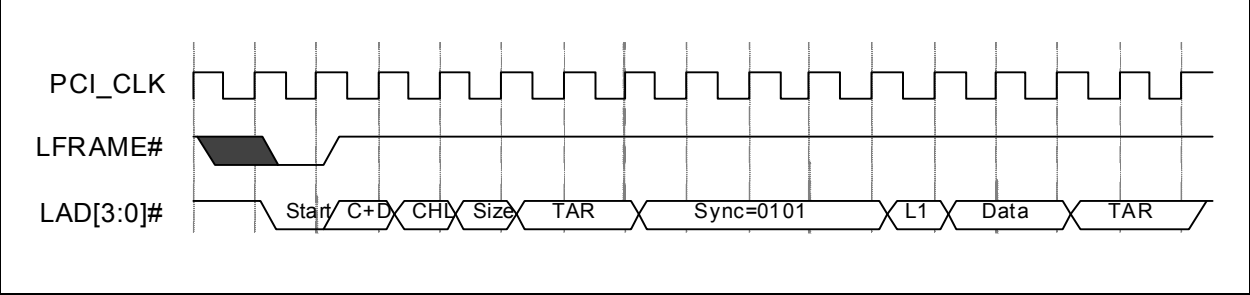
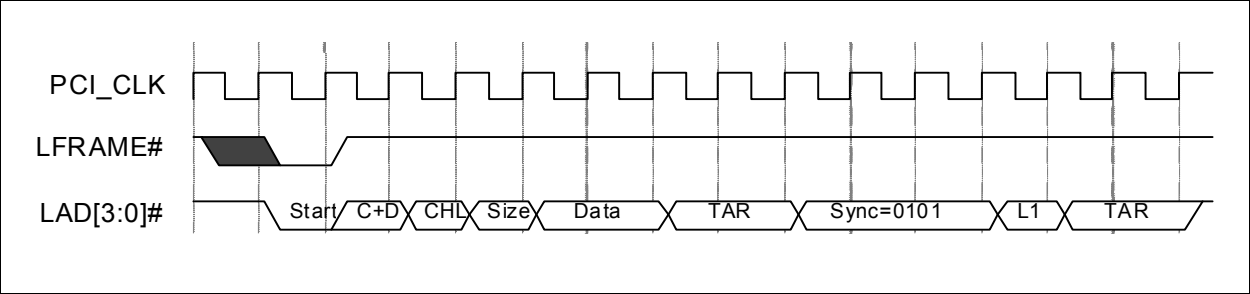


FIGURE 43-9: DMA WRITE (FIRST BYTE)



**Note:** L1=Sync of 0000

FIGURE 43-10: DMA READ (FIRST BYTE)

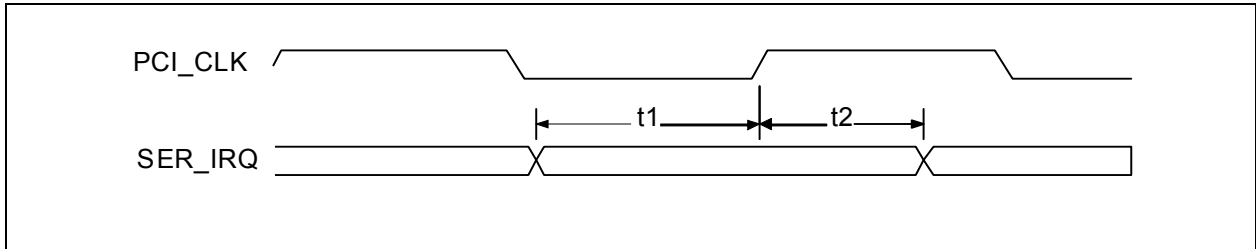


**Note:** L1=Sync of 0000



## 43.4 Serial IRQ Timing

**FIGURE 43-11: SETUP AND HOLD TIME**

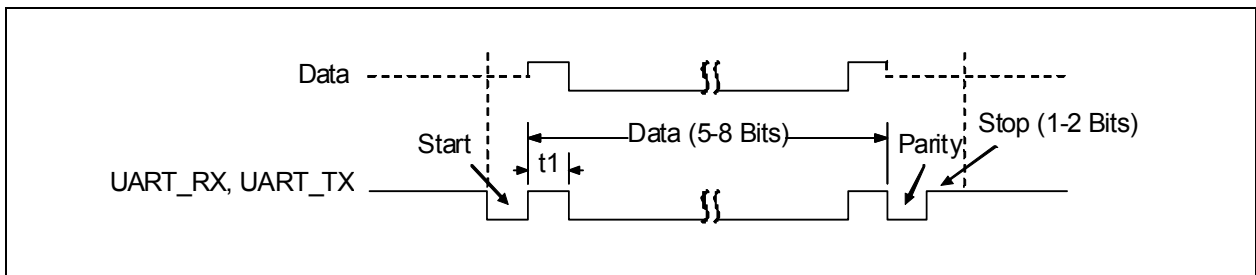


**TABLE 43-6: SETUP AND HOLD TIME**

Name	Description	Min	Typ	Max	Units
t1	SER_IRQ Setup Time to PCI_CLK Rising	7			nsec
t2	SER_IRQ Hold Time to PCI_CLK Rising	0			

## 43.5 Serial Port Data Timing

**FIGURE 43-12: SERIAL PORT DATA**



**TABLE 43-7: SERIAL PORT DATA PARAMETERS**

Name	Description	Min	Typ	Max	Units
t1	Serial Port Data Bit Time		$t_{BR}$ (Note 43-1)		nsec

**Note 43-1**  $t_{BR}$  is 1/Baud Rate. The Baud Rate is programmed through the divisor latch registers. Baud Rates have percentage errors indicated in [Table 12-21, “UART Baud Rates \(1.8432MHz source\),” on page 252.](#)

FIGURE 43-13: UART\_CLK EXTERNAL CLOCK TIMING

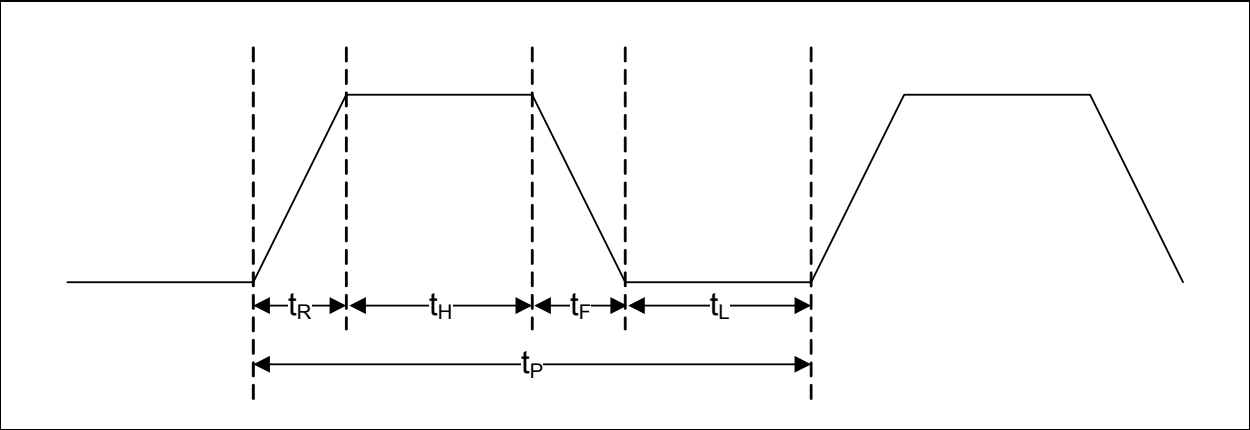


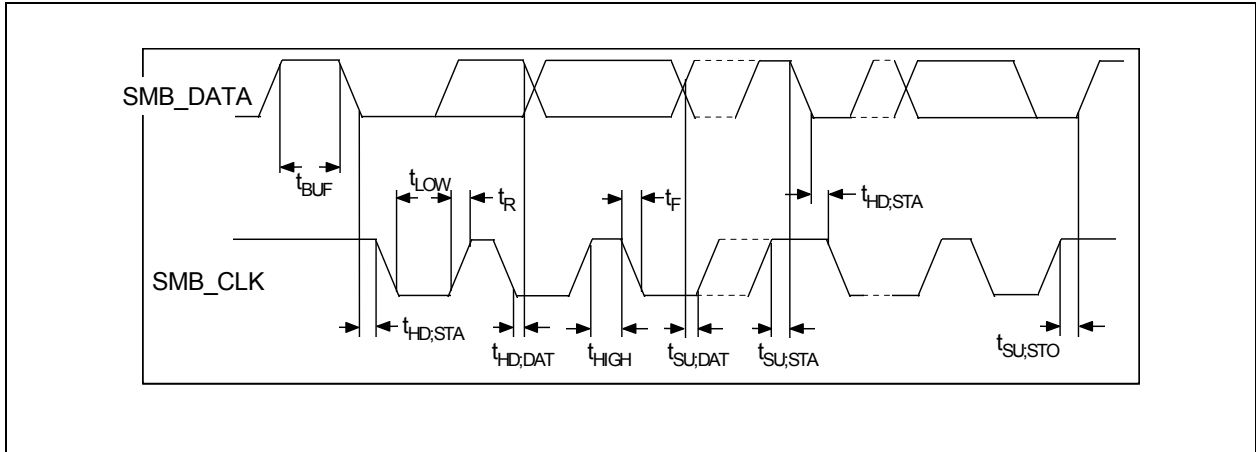
TABLE 43-8: UART\_CLK EXTERNAL CLOCK TIMING PARAMETERS

Name	Description	Min	Typ	Max	Units
t <sub>P</sub>	Period	553.6	542.5	553.6	nsec
t <sub>H</sub>	High Time	200			
t <sub>L</sub>	Low Time				
t <sub>R</sub>	Rise Time			10	
t <sub>F</sub>	Fall Time				

## 43.6 I2C/SMBus Timing

Note that the following timing applies to all of the MEC1618/MEC1618i I2C/SMBus and SB-TSI functions.

**FIGURE 43-14: I2C/SMBUS TIMING**



**TABLE 43-9: I2C/SMBUS TIMING PARAMETERS**

Symbol	Parameter	Standard-Mode		Fast-Mode		Units
		Min	Max	Min	Max	
$f_{SCL}$	SMB Clock Frequency		100		400	kHz
$t_{BUF}$	Bus Free Time	4.7		1.3		$\mu s$
$t_{SU,STA}$	START Condition Set-Up Time	4.7		0.6		$\mu s$
$t_{HD,STA}$	START Condition Hold Time	4.0		0.6		$\mu s$
$t_{LOW}$	SMB_CLK LOW Time	4.7		1.3		$\mu s$
$t_{HIGH}$	SMB_CLK HIGH Time	4.0		0.6		$\mu s$
$t_{R}$	SMB_CLK and SMB_DATA Rise Time		1.0		0.3	$\mu s$
$t_{F}$	SMB_CLK and SMB_DATA Fall Time		0.3		0.3	$\mu s$
$t_{SU,DAT}$	Data Set-Up Time	0.25		0.1		$\mu s$
$t_{HD,DAT}$	Data Hold Time	0		0		$\mu s$
$t_{SU,STO}$	STOP Condition Set-Up Time	4.0		0.6		$\mu s$

## 43.7 Fan Tachometer Timing

FIGURE 43-15: FAN TACHOMETER INPUT TIMING

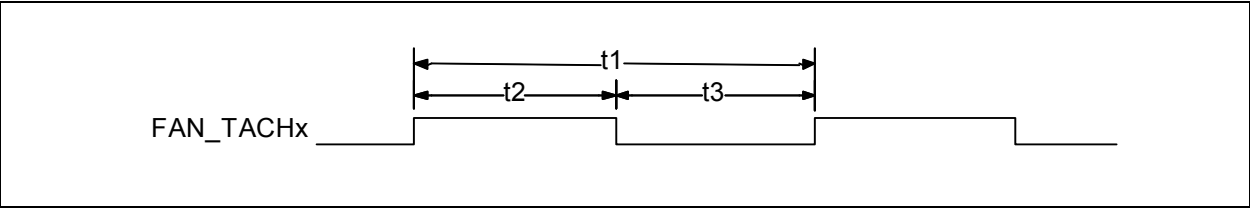


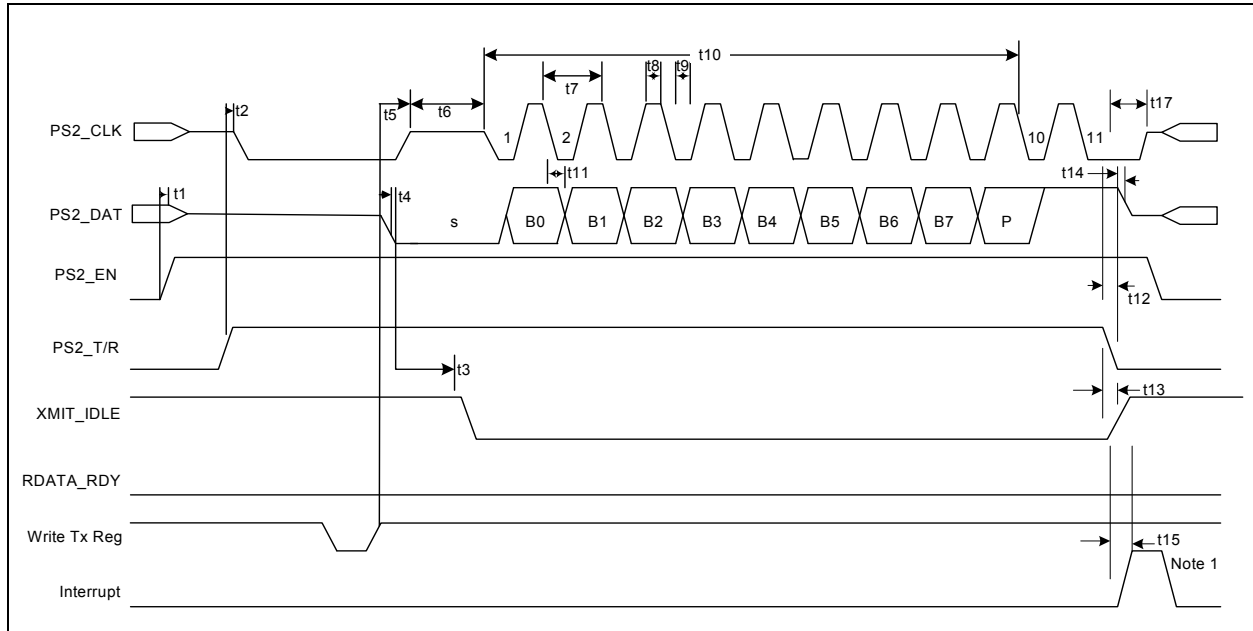
TABLE 43-10: FAN TACHOMETER INPUT TIMING PARAMETERS

Name	Description	Min	Typ	Max	Units
t1	Pulse Time	100			μsec
t2	Pulse High Time				
t3	Pulse Low Time	10			

**Note 43-2**  $t_{TACH}$  is the clock used for the tachometer counter. It is 30.52 \* prescaler, where the prescaler is programmed in the Fan Tachometer Timebase Prescaler register.

## 43.8 PS/2 Timing

**FIGURE 43-16: PS/2 TRANSMIT TIMING**



**TABLE 43-11: PS/2 CHANNEL TRANSMISSION TIMING PARAMETERS**

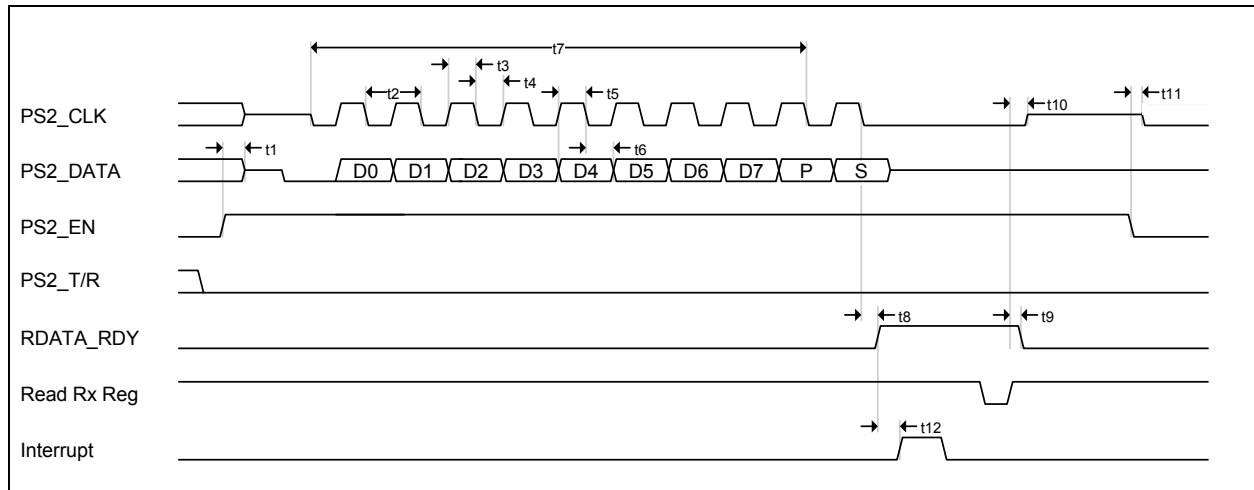
Name	Description	Min	Typ	Max	Units
t1	The PS/2 Channel's CLK and DATA lines are floated following PS2_EN=1 and PS2_T/R=0.			1000	ns
t2	PS2_T/R bit set to CLK driven low preparing the PS/2 Channel for data transmission.				
t3	CLK line floated to XMIT_IDLE bit de-asserted.				
t4	Trailing edge of WR to Transmit Register to DATA line driven low.	45		90	
t5	Trailing edge of EC WR of Transmit Register to CLK line floated.	90		130	ns
t6	Initiation of Start of Transmit cycle by the PS/2 channel controller to the auxiliary peripheral's responding by latching the Start bit and driving the CLK line low.	0.002		25.003	ms
t7	Period of CLK	60		302	μs
t8	Duration of CLK high (active)	30		151	
t9	Duration of CLK low (inactive)				

# MEC1618/MEC1618i

TABLE 43-11: PS/2 CHANNEL TRANSMISSION TIMING PARAMETERS (CONTINUED)

Name	Description	Min	Typ	Max	Units
t10	Duration of Data Frame. Falling edge of Start bit CLK (1st clk) to falling edge of Parity bit CLK (10th clk).			2.002	ms
t11	DATA output by MEC1618/MEC1618i following the falling edge of CLK. The auxiliary peripheral device samples DATA following the rising edge of CLK.			1.0	μs
t12	Rising edge following the 11th falling clock edge to PS_T/R bit driven low.	3.5		7.1	μs
t13	Trailing edge of PS_T/R to XMIT_IDLE bit asserted.			500	ns
t14	DATA released to high-Z following the PS2_T/R bit going low.				
t15	XMIT_IDLE bit driven high to interrupt generated. Note1- Interrupt is cleared by writing a 1 to the status bit in <a href="#">Table 15-53, "GIRQ19 Source Register,"</a> on page 335.				
t17	Trailing edge of CLK is held low prior to going high-Z				

**FIGURE 43-17: PS/2 RECEIVE TIMING**



**TABLE 43-12: PS/2 CHANNEL RECEIVE TIMING DIAGRAM PARAMETERS**

Name	Description	Min	Typ	Max	Units
t1	The PS/2 Channel's CLK and DATA lines are floated following PS2_EN=1 and PS2_T/R=0.			1000	ns
t2	Period of CLK	60		302	μs
t3	Duration of CLK high (active)	30		151	
t4	Duration of CLK low (inactive)				
t5	DATA setup time to falling edge of CLK. MEC1618/MEC1618i samples the data line on the falling CLK edge.	1			
t6	DATA hold time from falling edge of CLK. MEC1618/MEC1618i samples the data line on the falling CLK edge.	2			
t7	Duration of Data Frame. Falling edge of Start bit CLK (1st clk) to falling edge of Parity bit CLK (10th clk).			2.002	ms
t8	Falling edge of 11th CLK to RDATA_RDY asserted.			1.6	μs
t9	Trailing edge of the EC's RD signal of the Receive Register to RDATA_RDY bit de-asserted.			500	ns
t10	Trailing edge of the EC's RD signal of the Receive Register to the CLK line released to high-Z.				
t11	PS2_CLK is "Low" and PS2_DATA is "Hi-Z" when PS2_EN is de-asserted.				
t12	RDATA_RDY asserted an interrupt is generated.				

# MEC1618/MEC1618i

## 43.9 BC-Link Master Timing

FIGURE 43-18: BC-LINK TIMING

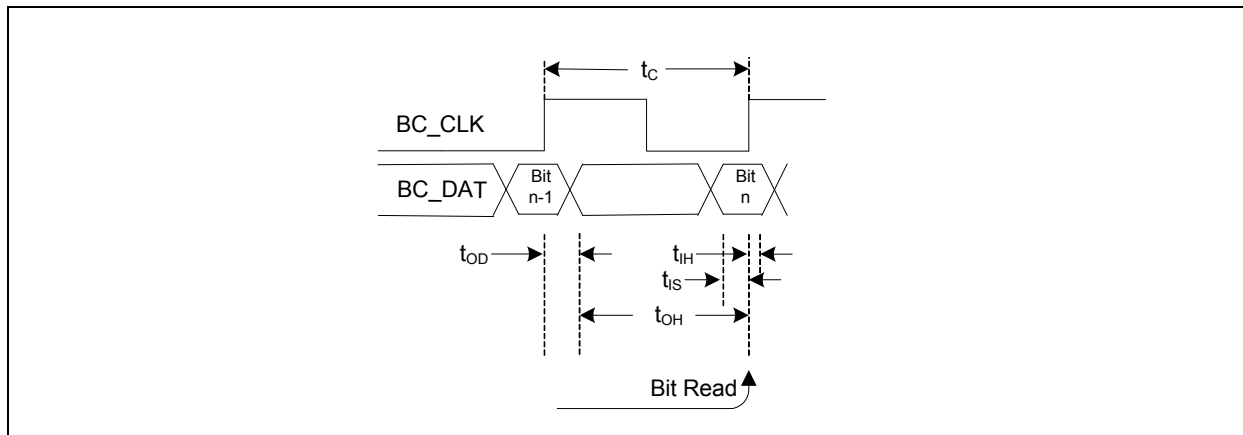


TABLE 43-13: BC-LINK MASTER TIMING DIAGRAM PARAMETERS

Name	Description	Min	Typ	Max	Units
<b><math>t_c</math>(High Speed)</b>	High Spec BC Clock Frequency		21.5	22.6	MHz
	High Spec BC Clock Period	44.3	46.7		ns
<b><math>t_c</math>(Low Speed)</b>	High Spec BC Clock Frequency		2.93	3.08	Mhz
	High Spec BC Clock Period	324.7	341.0		ns
$t_{OD}$	BC-Link Master DATA output delay after rising edge of CLK.			10	nsec
$t_{OH}$	Data hold time after falling edge of CLK	1 CLK - $t_{OD-max}$			nsec
$t_{IS}$	BC-Link Master DATA input setup time before rising edge of CLK.	15			nsec
$t_{IH}$	BC-Link Master DATA input hold time after rising edge of CLK.	0			nsec

**Note 43-3** The ( $t_{IH}$  in [Table 43-13](#)) BC-Link Master DATA input must be stable before next rising edge of CLK.

**Note 43-4** The BC-Link Clock frequency is limited by the application usage model (see [Note 36-1 on page 569](#) & [Table 36-2 on page 569](#).) The BC-Link Clock frequency is controlled by the [BC-Link Clock Select Register on page 573](#). The [tc\(High Speed\)](#) parameter implies both BC-link master and companion devices are located on the same circuit board and a high speed clock setting is possible. The [tc\(Low Speed\)](#) parameter implies the BC-link master and companion devices are located on separate circuit boards connected by 12 inch ribbon cable and a low speed clock setting is required.

**APPLICATION NOTE:** The timing budget equation is as follows for data from BC-Link slave to master:

$$T_c > T_{OD}(\text{master-clk}) + T_{prop}(\text{clk}) + T_{OD}(\text{slave}) + T_{prop}(\text{slave data}) + T_{IS}(\text{master}).$$



## 43.10 Serial Peripheral Interface (SPI) Timings

Note that the following timing applies to all of the MEC1618/MEC1618i Serial Peripheral Interface functions.

### 43.10.1 SPI CLOCK TIMING

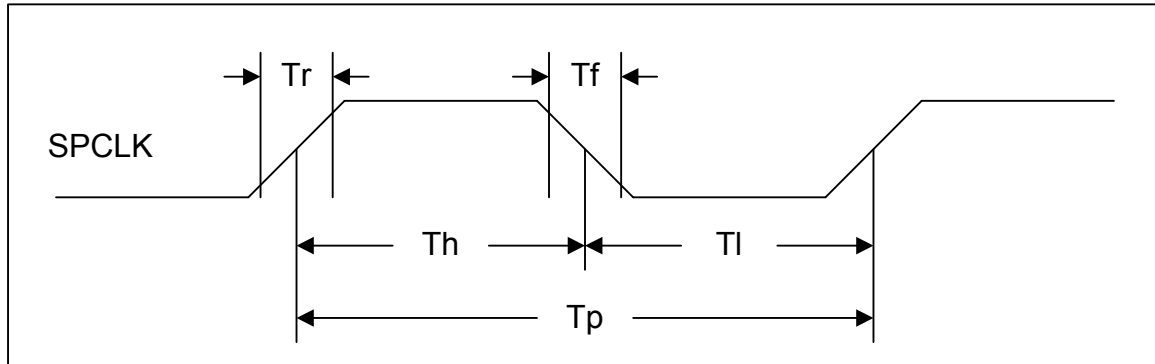


FIGURE 43-19: SPI CLOCK TIMING

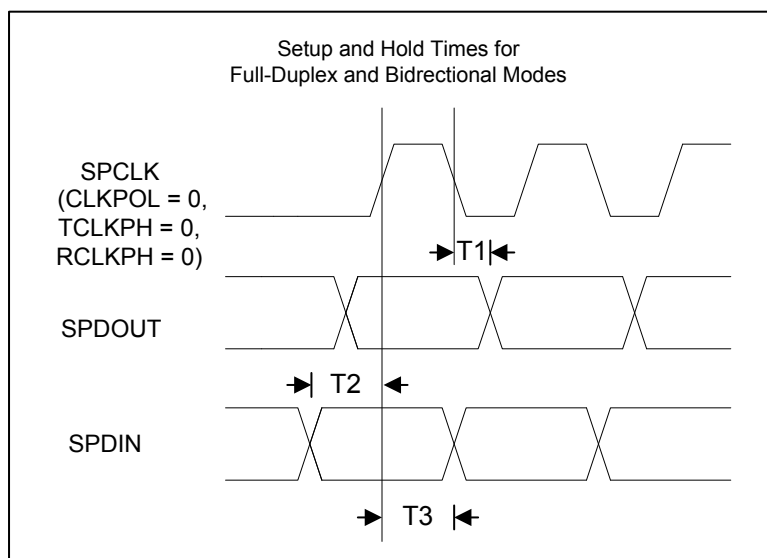
TABLE 43-14: SPI CLOCK TIMING PARAMETERS

Name	Description	Min	Typ	Max	Units
Tr	SPI Clock Rise Time. Measured from 10% to 90%.			10% of SPCLK Period	ns
Tf	SPI Clock Fall Time. Measured from 90% to 10%.			10% of SPCLK Period	ns
Th/Tl	SPI Clock High Time/SPI Clock Low Time	40% of SPCLK Period	50% of SPCLK Period ( <a href="#">Note 43-5</a> )	60% of SPCLK Period	ns
Tp	SPI Clock Period – As selected by <a href="#">SPI Clock Generator Register (SPICG)</a> on page 512	15.50		62492.25	ns

**Note 43-5** In the MEC1618/MEC1618i, the General Purpose Serial Peripheral Interface (GP-SPI) pins are 8 mA buffers. The maximum SPCLK pin clock frequency is [MCLK/2](#) for all modes. Limited functionality is available when SPCLK pin clock frequency is [MCLK](#) although performance is not guaranteed. See [TABLE 30-14: on page 513](#) and [Section 30.9.5.5, "Limits of SPI configurations," on page 506](#).

## 43.10.2 SPI SETUP AND HOLD TIMES

**FIGURE 43-20: SPI SETUP AND HOLD TIMES**



**TABLE 43-15: SPI SETUP AND HOLD TIMES PARAMETERS**

Name	Description	Min	Typ	Max	Units
T1	Data Output Delay			20	ns
T2	Data IN Setup Time	20			ns
T3	Data IN Hold Time	0			ns

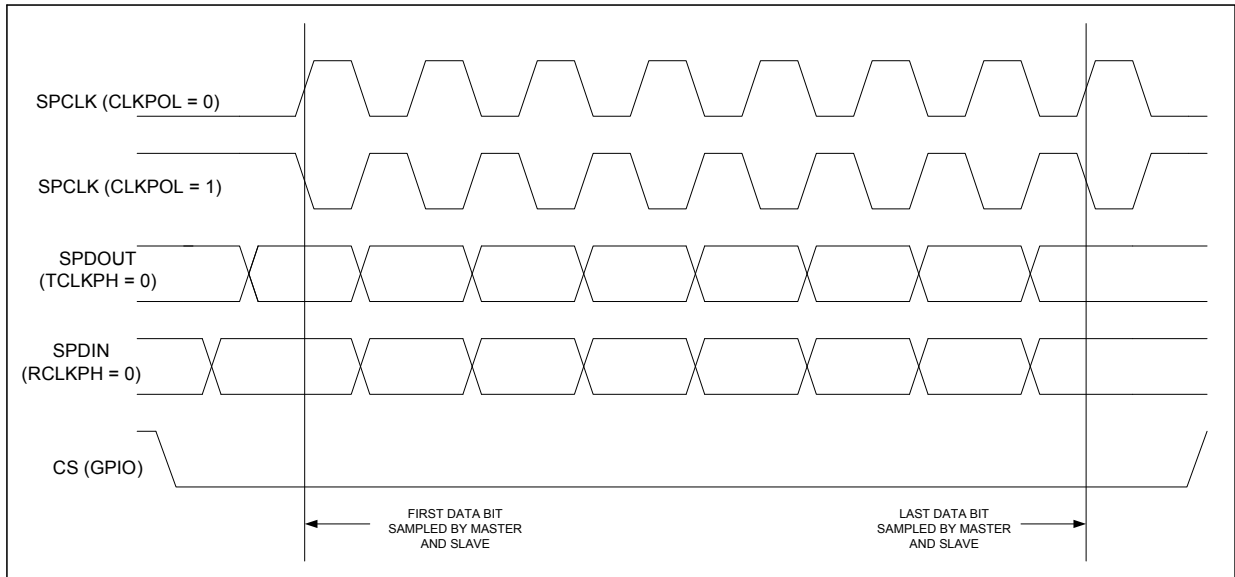
## 43.10.3 SPI INTERFACE TIMINGS

The following timing diagrams represent a single-byte transfer over the SPI interface using different SPCLK phase settings. Data bits are transmitted in bit order starting with the MSB (LSBF='0') or the LSB (LSBF='1'). See the [SPICR - SPI Control Register on page 508](#) for information on the LSBF bit. The CS signal in each diagram is a generic bit-controlled chip select signal required by most peripheral devices. This signal and additional chip selects can be GPIO controlled. Note that these timings for Full Duplex Mode are also applicable to Half Duplex (or Bi-directional) mode.

### 43.10.3.1 SPI Interface Timing – Full Duplex Mode (TCLKPH = 0, RCLKPH = 0)

In this mode, data is available immediately when a device is selected and is sampled on the first and following odd SPCLK edges by the master and slave.

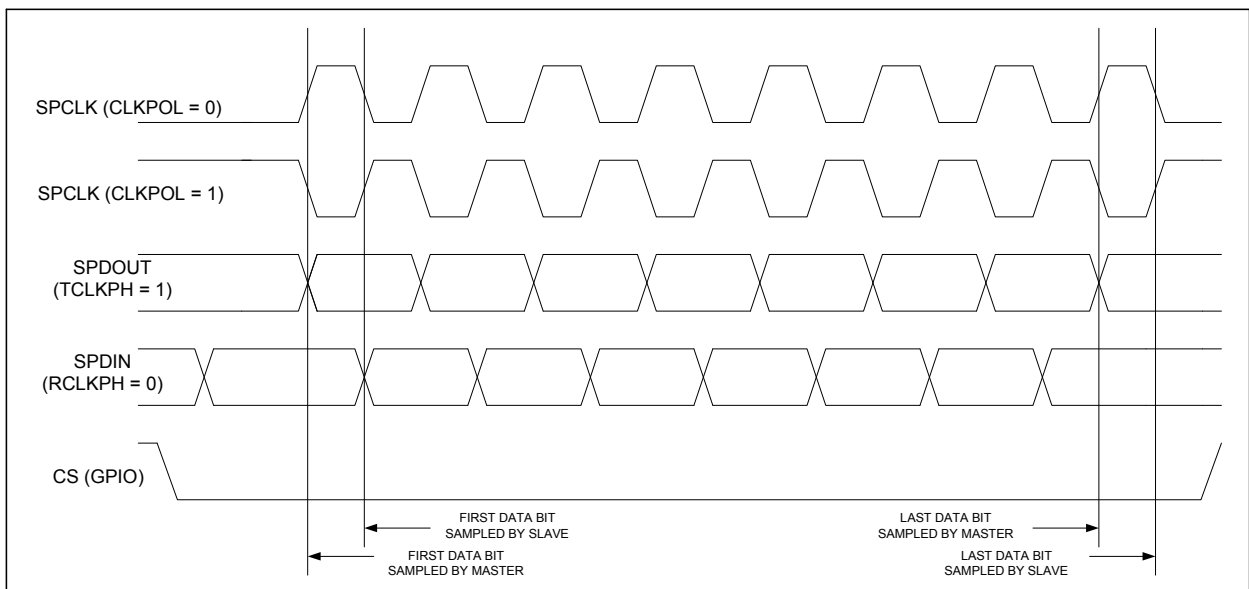
**FIGURE 43-21: INTERFACE TIMING, FULL DUPLEX MODE (TCLKPH = 0, RCLKPH = 0)**



## 43.10.3.2 SPI Interface Timing - Full Duplex Mode (TCLKPH = 1, RCLKPH = 0)

In this mode, the master requires an initial SPCLK edge before data is available. The data from slave is available immediately when the slave device is selected. The data is sampled on the first and following odd edges by the master. The data is sampled on the second and following even SPCLK edges by the slave.

**FIGURE 43-22: SPI INTERFACE TIMING, FULL DUPLEX MODE (TCLKPH = 1, RCLKPH = 0)**

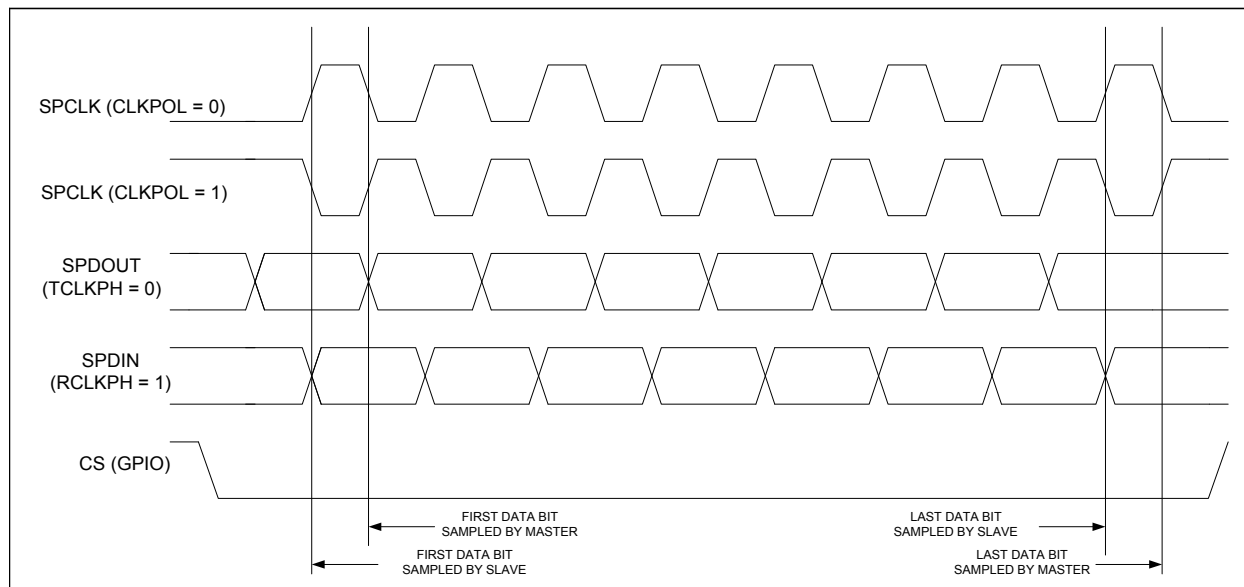


# MEC1618/MEC1618i

## 43.10.3.3 SPI Interface Timing - Full Duplex Mode (TCLKPH = 0, RCLKPH = 1)

In this mode, the data from slave is available immediately when the slave device is selected. The slave device requires an initial SPCLK edge before data is available. The data is sampled on the second and following even SPCLK edges by the master. The data is sampled on the first and following odd edges by the slave.

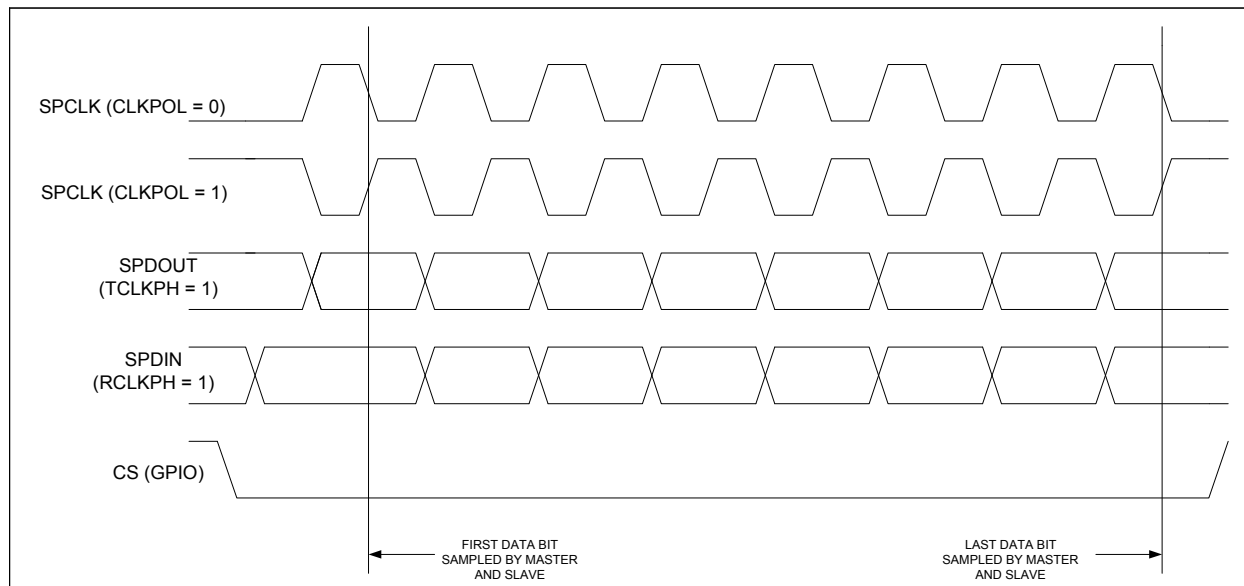
**FIGURE 43-23: SPI INTERFACE TIMING, FULL DUPLEX MODE (TCLKPH = 0, RCLKPH = 1)**



## 43.10.3.4 SPI Interface Timing - Full Duplex Mode (TCLKPH = 1, RCLKPH = 1)

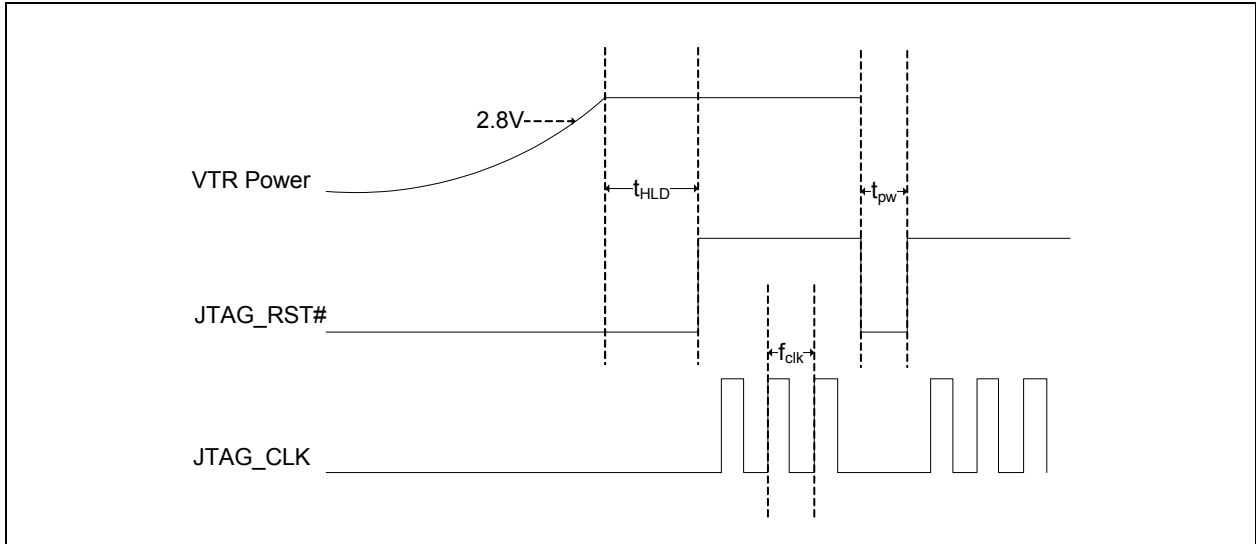
In this mode, the master and slave require an initial SPCLK edge before data is available. Data is sampled on the second and following even SPCLK edges by the master and slave.

**FIGURE 43-24: SPI INTERFACE TIMING - FULL DUPLEX MODE (TCLKPH = 1, RCLKPH = 1)**

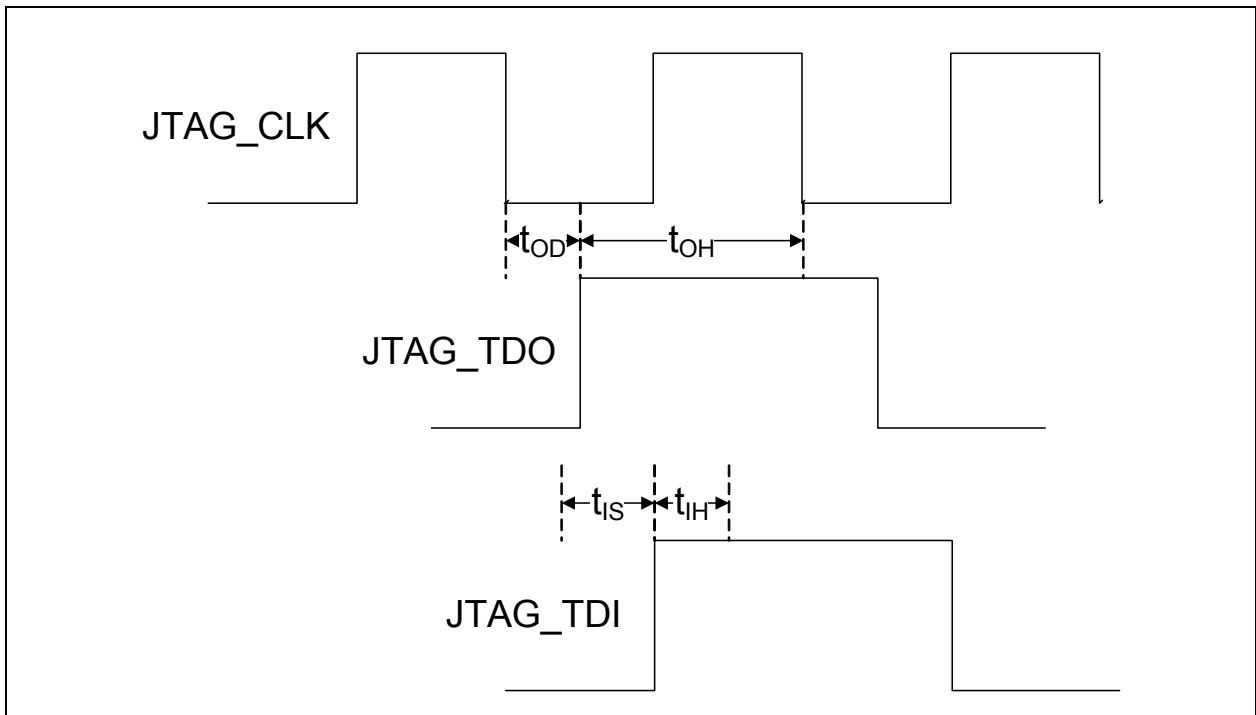


## 43.11 JTAG Interface Timing

**FIGURE 43-25: JTAG POWER-UP & ASYNCHRONOUS RESET TIMING**



**FIGURE 43-26: JTAG SETUP & HOLD PARAMETERS**



# MEC1618/MEC1618i

**TABLE 43-16: JTAG INTERFACE TIMING PARAMETERS**

Name	Description	Min	Typ	Max	Units
$t_{\text{HLD}}$	JTAG_RST# de-assertion after VTR power is applied	5			ms
$t_{\text{pw}}$	JTAG_RST# assertion pulse width	500			nsec
$f_{\text{clk}}$	JTAG_CLK frequency (see note)			8	MHz
$t_{\text{OD}}$	TDO output delay after falling edge of TCLK.	5		18	nsec
$t_{\text{OH}}$	TDO hold time after falling edge of TCLK	1 TCLK - $t_{\text{OD}}$			nsec
$t_{\text{IS}}$	TDI setup time before rising edge of TCLK.	5			nsec
$t_{\text{IH}}$	TDI hold time after rising edge of TCLK.	7			nsec

**Note 43-6**  $f_{\text{clk}}$  is the maximum frequency to access a JTAG Register. Additional JTAG\_CLK frequency constraints are described in [Section 41.7.2, "Clocks"](#), [Section 41.7.2, "Clocks,"](#) on page 611.

## 43.12 Serial Debug Port Timing

FIGURE 43-27: SERIAL DEBUG PORT TIMING PARAMETERS

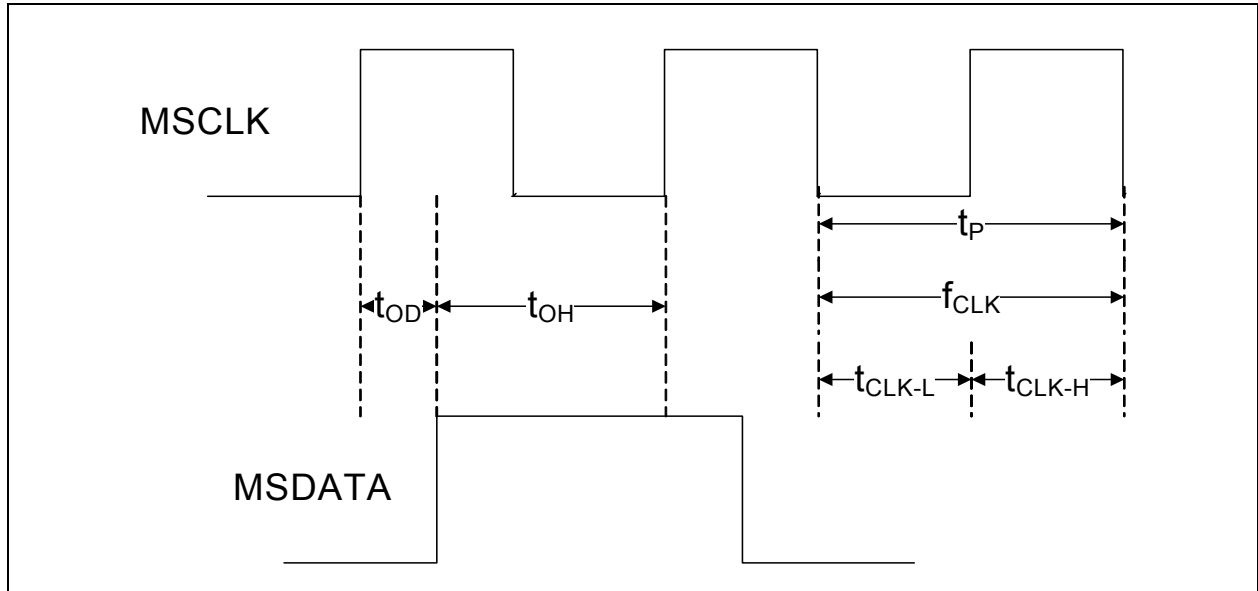


TABLE 43-17: SERIAL DEBUG PORT INTERFACE TIMING PARAMETERS

Name	Description	Min	Typ	Max	Units
$f_{clk}$	MSCLK frequency (see note)		32	48	MHz
$t_{OD}$	MSDATA output delay after falling edge of MSCLK.			5	nsec
$t_{OH}$	MSDATA hold time after falling edge of MSCLK	1 MSCLK - $t_{OD}$			nsec
$t_P$	MSCLK Period	$1/f_{clk}$			nsec
$t_{CLK-L}$	MSCLK Low Time		$t_P - t_{CLK-H}$		sec
$t_{CLK-H}$	MSCLK high Time		$t_P/2$		sec

## 43.13 CLKRUN# Timing

(see [Section 6.7.2, "Using CLKRUN#,"](#) on page 140)

## 43.14 RESET Input Timing

(see [Section 5.6.3, "RESET Pin Interface,"](#) on page 105)

## 43.15 VCI Input Timing

(see [Section 31.8, "Input Filtering,"](#) on page 522)

## 43.16 GPIO Pulse Width Timing

(see [Section 21.7, "Interrupts,"](#) on page 404)

## **43.17 CPU Reset Timing**

(see [Section 9.14, "CPU\\_RESET Hardware Speed-Up,"](#) on page 214)

## **43.18 Keyboard Scan Matrix Timing**

(see [Section 35.6.2, "Pre Drive Mode,"](#) on page 560)

## **43.19 HDMI-CEC Timing**

(see [HDMI-CEC Interface Controller References](#)[1])

## **43.20 Gang Programmer Timing**

(see [Gang Programmer Interface Section 40.8.3, "Functional Description,"](#) on page 598)



## 44.0 REFERENCE DOCUMENTS

This document was created using the following parent documents:

1. Intel Low Pin Count Specification, Revision 1.0, September 29, 1997
2. PCI Local Bus Specification, Revision 2.2, December 18, 1998
3. Advanced Configuration and Power Interface Specification, Revision 1.0b, February 2, 1999
4. System Management Bus Specification, Revision 1.1, December 11, 1998.
5. Plug and Play ISA Specification, Version 1.0a, Intel Corp. and Microsoft Corp., May 5, 1994
6. I2C-BUS Specification, Version 2.1, January 2000.
7. SMBus Controller Core Interface, Revision 2.0 (10 MHz), v3.20, Core-Level Architecture Specification, SMSC, 4/22/11
8. ECE1077 MEC-04 Keyboard Scan Extension, Product Architecture Specification, Rev 0.23, January 12, 2006, SMSC Confidential
9. Intel® 82801DBM I/O Controller Hub 4 Mobile (ICH4-M), Datasheet, Order Number: 252337-001, Intel Corp., January 2003
10. SMSC BC-Link Specification, Revision 1.02, dated September 05, 2007
11. IEEE Std 1149.1
12. [PECI Interface Core, Rev. 1.1, Core-Level Architecture Specification, SMSC Confidential.](#)
13. [PCI Mobile Design Guide, Version 1.1, PCI-SIG, December 18, 1998.](#)
14. *ARCompact™ Instruction Set Architecture Programmer's Reference*, ARC International, April 2009

# MEC1618/MEC1618i

---

## APPENDIX A: DATA SHEET REVISION HISTORY

TABLE A-1: REVISION HISTORY

Revision	Section/Figure/Entry	Correction
DS00002339A (01-17-17)	Replaces previous SMSC version Rev. 1.3 (01-23-13)	

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at [www.microchip.com](http://www.microchip.com). This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at [www.microchip.com](http://www.microchip.com). Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the web site at: <http://www.microchip.com/support>**

# MEC1618/MEC1618i

## PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

<u>PART NO.</u>		<u>[X]</u>	-	<u>XXX</u>	-	<u>[X]<sup>(1)</sup></u>
Device		Temperature Range		Package		Tape and Reel Option
Device:		MEC1618, MEC1618i				
Temperature Range:		Blank = 0°C to +85°C (Extended Commercial) i = -40°C to +85°C (Industrial)				
Package:		AJZP = 156-pin LFBGA				
Tape and Reel Option:		Blank = Standard packaging (tray) TR = Tape and Reel <sup>(1)</sup>				

**Examples:**  
a) MEC1618-AJZP  
156-pin LFBGA (11mm x 11mm, 0.8mm pitch)  
RoHS Compliant package  
b) MEC1618i-AJZP  
156-pin LFBGA (11mm x 11mm, 0.8mm pitch)  
RoHS Compliant package  
with Industrial Temperature rating

**Note 1:** Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option. Reel size is 4,000.

---

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

**Trademarks**

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KEELOQ, KEELOQ logo, Klear, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KlearNet, KlearNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2011-2017, Microchip Technology Incorporated, All Rights Reserved.

ISBN:9781522412489

**QUALITY MANAGEMENT SYSTEM**  
**CERTIFIED BY DNV**  
**== ISO/TS 16949 ==**

*Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*

## Worldwide Sales and Service

### AMERICAS

**Corporate Office**  
 2355 West Chandler Blvd.  
 Chandler, AZ 85224-6199  
 Tel: 480-792-7200  
 Fax: 480-792-7277  
 Technical Support:  
<http://www.microchip.com/support>  
 Web Address:  
[www.microchip.com](http://www.microchip.com)

**Atlanta**  
 Duluth, GA  
 Tel: 678-957-9614  
 Fax: 678-957-1455

**Austin, TX**  
 Tel: 512-257-3370

**Boston**  
 Westborough, MA  
 Tel: 774-760-0087  
 Fax: 774-760-0088

**Chicago**  
 Itasca, IL  
 Tel: 630-285-0071  
 Fax: 630-285-0075

**Dallas**  
 Addison, TX  
 Tel: 972-818-7423  
 Fax: 972-818-2924

**Detroit**  
 Novi, MI  
 Tel: 248-848-4000

**Houston, TX**  
 Tel: 281-894-5983

**Indianapolis**  
 Noblesville, IN  
 Tel: 317-773-8323  
 Fax: 317-773-5453  
 Tel: 317-536-2380

**Los Angeles**  
 Mission Viejo, CA  
 Tel: 949-462-9523  
 Fax: 949-462-9608  
 Tel: 951-273-7800

**Raleigh, NC**  
 Tel: 919-844-7510

**New York, NY**  
 Tel: 631-435-6000

**San Jose, CA**  
 Tel: 408-735-9110  
 Tel: 408-436-4270

**Canada - Toronto**  
 Tel: 905-695-1980  
 Fax: 905-695-2078

### ASIA/PACIFIC

**Asia Pacific Office**  
 Suites 3707-14, 37th Floor  
 Tower 6, The Gateway  
 Harbour City, Kowloon

**Hong Kong**  
 Tel: 852-2943-5100  
 Fax: 852-2401-3431

**Australia - Sydney**  
 Tel: 61-2-9868-6733  
 Fax: 61-2-9868-6755

**China - Beijing**  
 Tel: 86-10-8569-7000  
 Fax: 86-10-8528-2104

**China - Chengdu**  
 Tel: 86-28-8665-5511  
 Fax: 86-28-8665-7889

**China - Chongqing**  
 Tel: 86-23-8980-9588  
 Fax: 86-23-8980-9500

**China - Dongguan**  
 Tel: 86-769-8702-9880

**China - Guangzhou**  
 Tel: 86-20-8755-8029

**China - Hangzhou**  
 Tel: 86-571-8792-8115  
 Fax: 86-571-8792-8116

**China - Hong Kong SAR**  
 Tel: 852-2943-5100  
 Fax: 852-2401-3431

**China - Nanjing**  
 Tel: 86-25-8473-2460  
 Fax: 86-25-8473-2470

**China - Qingdao**  
 Tel: 86-532-8502-7355  
 Fax: 86-532-8502-7205

**China - Shanghai**  
 Tel: 86-21-3326-8000  
 Fax: 86-21-3326-8021

**China - Shenyang**  
 Tel: 86-24-2334-2829  
 Fax: 86-24-2334-2393

**China - Shenzhen**  
 Tel: 86-755-8864-2200  
 Fax: 86-755-8203-1760

**China - Wuhan**  
 Tel: 86-27-5980-5300  
 Fax: 86-27-5980-5118

**China - Xian**  
 Tel: 86-29-8833-7252  
 Fax: 86-29-8833-7256

### ASIA/PACIFIC

**China - Xiamen**  
 Tel: 86-592-2388138  
 Fax: 86-592-2388130

**China - Zhuhai**  
 Tel: 86-756-3210040  
 Fax: 86-756-3210049

**India - Bangalore**  
 Tel: 91-80-3090-4444  
 Fax: 91-80-3090-4123

**India - New Delhi**  
 Tel: 91-11-4160-8631  
 Fax: 91-11-4160-8632

**India - Pune**  
 Tel: 91-20-3019-1500

**Japan - Osaka**  
 Tel: 81-6-6152-7160  
 Fax: 81-6-6152-9310

**Japan - Tokyo**  
 Tel: 81-3-6880-3770  
 Fax: 81-3-6880-3771

**Korea - Daegu**  
 Tel: 82-53-744-4301  
 Fax: 82-53-744-4302

**Korea - Seoul**  
 Tel: 82-2-554-7200  
 Fax: 82-2-558-5932 or  
 82-2-558-5934

**Malaysia - Kuala Lumpur**  
 Tel: 60-3-6201-9857  
 Fax: 60-3-6201-9859

**Malaysia - Penang**  
 Tel: 60-4-227-8870  
 Fax: 60-4-227-4068

**Philippines - Manila**  
 Tel: 63-2-634-9065  
 Fax: 63-2-634-9069

**Singapore**  
 Tel: 65-6334-8870  
 Fax: 65-6334-8850

**Taiwan - Hsin Chu**  
 Tel: 886-3-5778-366  
 Fax: 886-3-5770-955

**Taiwan - Kaohsiung**  
 Tel: 886-7-213-7830

**Taiwan - Taipei**  
 Tel: 886-2-2508-8600  
 Fax: 886-2-2508-0102

**Thailand - Bangkok**  
 Tel: 66-2-694-1351  
 Fax: 66-2-694-1350

### EUROPE

**Austria - Wels**  
 Tel: 43-7242-2244-39  
 Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
 Tel: 45-4450-2828  
 Fax: 45-4485-2829

**Finland - Espoo**  
 Tel: 358-9-4520-820

**France - Paris**  
 Tel: 33-1-69-53-63-20  
 Fax: 33-1-69-30-90-79

**France - Saint Cloud**  
 Tel: 33-1-30-60-70-00

**Germany - Garching**  
 Tel: 49-8931-9700

**Germany - Haan**  
 Tel: 49-2129-3766400

**Germany - Heilbronn**  
 Tel: 49-7131-67-3636

**Germany - Karlsruhe**  
 Tel: 49-721-625370

**Germany - Munich**  
 Tel: 49-89-627-144-0  
 Fax: 49-89-627-144-44

**Germany - Rosenheim**  
 Tel: 49-8031-354-560

**Israel - Ra'anana**  
 Tel: 972-9-744-7705

**Italy - Milan**  
 Tel: 39-0331-742611  
 Fax: 39-0331-466781

**Italy - Padova**  
 Tel: 39-049-7625286

**Netherlands - Drunen**  
 Tel: 31-416-690399  
 Fax: 31-416-690340

**Norway - Trondheim**  
 Tel: 47-7289-7561

**Poland - Warsaw**  
 Tel: 48-22-3325737

**Romania - Bucharest**  
 Tel: 40-21-407-87-50

**Spain - Madrid**  
 Tel: 34-91-708-08-90  
 Fax: 34-91-708-08-91

**Sweden - Gothenberg**  
 Tel: 46-31-704-60-40

**Sweden - Stockholm**  
 Tel: 46-8-5090-4654

**UK - Wokingham**  
 Tel: 44-118-921-5800  
 Fax: 44-118-921-5820