DOCUMENT NUMBER
S12MSCANV3/D

MSCAN
Block Guide
V03.00

Original Release Date: 19 MAY 1998
Revised: 18 FEB 2003

Motorola, Inc.

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V03.00	18 FEB 2003	18 FEB 2003		- Added bus-off recovery by user request feature. - Redefined reset value of time stamp counter.

Table of Contents

Section 1 Introduction

1.1 Overview. 11
1.2 Features 12

Section 2 External Signal Description

2.1 Overview. 13
2.2 Detailed Signal Description 13
2.2.1 RXCAN — CAN Receiver Input Pin 13
2.2.2 TXCAN — CAN Transmitter Output Pin. 13
2.3 CAN System. 13

Section 3 Memory Map/Register Definition

3.1 Overview. 14
3.2 Module Memory Map 14
3.3 Register Descriptions 16
3.3.1 Programmer’s Model of Control Registers 16
3.3.2 Programmer’s Model of Message Storage. 45

Section 4 Functional Description

4.1 General. 51
4.2 Message Storage 51
4.2.1 Message Transmit Background 52
4.2.2 Transmit Structures 52
4.2.3 Receive Structures. 53
4.3 Identifier Acceptance Filter 54
4.3.1 Protocol Violation Protection 58
4.3.2 Clock System 58
4.4 Timer Link. 61
4.5 Modes of Operation 61
4.5.1 Normal Modes 61
4.5.2 Special Modes 61
4.5.3 Emulation Modes 61
4.5.4 Listen-Only Mode 61

4.5.5 Security Modes 62
4.6 Low Power Options 62
4.6.1 CPU Run Mode 62
4.6.2 CPU Wait Mode 63
4.6.3 CPU Stop Mode 63
4.6.4 MSCAN Sleep Mode 63
4.6.5 MSCAN Initialization Mode 65
4.6.6 MSCAN Power Down Mode 66
4.6.7 Programmable Wake-Up Function 67
4.7 Reset Initialization 67
4.8 Interrupts 67
4.8.1 Description of Interrupt Operation 67
4.8.2 Transmit Interrupt 67
4.8.3 Receive Interrupt 68
4.8.4 Wake-Up Interrupt 68
4.8.5 Error Interrupt 68
4.8.6 Interrupt Acknowledge 68
4.8.7 Recovery from STOP or WAIT 69

Section 5 Initialization/Application Information

5.1 MSCAN initialization 70
5.2 Bus-Off recovery. 70

List of Figures

Figure 1-1 MSCAN Block Diagram 11
Figure 2-1 The CAN System 13
Figure 3-1 MSCAN Control Register 0 (CANCTL0) 17
Figure 3-2 MSCAN Control Register 1 (CANCTL1) 20
Figure 3-3 MSCAN Bus Timing Register 0 (CANBTR0) 22
Figure 3-4 MSCAN Bus Timing Register 1 (CANBTR1) 23
Figure 3-5 MSCAN Receiver Flag Register (CANRFLG) 25
Figure 3-6 MSCAN Receiver Interrupt Enable Register (CANRIER) 27
Figure 3-7 MSCAN Transmitter Flag Register (CANTFLG) 29
Figure 3-8 MSCAN Transmitter Interrupt Enable Register (CANTIER) 30
Figure 3-9 MSCAN Transmitter Message Abort Request Register (CANTARQ) 31
Figure 3-10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK) . . 32
Figure 3-11 MSCAN Transmit Buffer Selection Register (CANTBSEL) 33
Figure 3-12 MSCAN Identifier Acceptance Control Register (CANIDAC). 35
Figure 3-13 MSCAN Reserved Register 37
Figure 3-14 MSCAN Miscellaneous Register (CANMISC) 38
Figure 3-15 MSCAN Receive Error Counter (CANRXERR) 39
Figure 3-16 MSCAN Transmit Error Counter (CANTXERR). 40
Figure 3-17 MSCAN Identifier Acceptance Registers (1st Bank) 41
Figure 3-18 MSCAN Identifier Acceptance Registers (2nd Bank) 41
Figure 3-19 MSCAN Identifier Mask Registers (1st Bank) 43
Figure 3-20 MSCAN Identifier Mask Registers (2nd Bank). 43
Figure 3-21 Receive / Transmit Message Buffer Extended Identifier 46
Figure 3-22 Standard Identifier Mapping. 47
Figure 3-23 Transmit Buffer Priority Register (TBPR). 49
Figure 3-24 Time Stamp Register (TSRH - High Byte). 49
Figure 3-25 Time Stamp Register (TSRL - Low Byte) 50
Figure 4-1 User Model for Message Buffer Organization 51
Figure 4-2 32-bit Maskable Identifier Acceptance Filter 56
Figure 4-3 16-bit Maskable Identifier Acceptance Filters 56
Figure 4-4 8-bit Maskable Identifier Acceptance Filters 57
Figure 4-5 MSCAN Clocking Scheme. 58
Figure 4-6 Segments within the Bit Time 60

Figure 4-7 Sleep Request / Acknowledge Cycle 63
Figure 4-8 Simplified State Transitions for Entering/Leaving Sleep Mode 65
Figure 4-9 Initialization Request/Acknowledge Cycle 66

List of Tables

Table 3-1 MSCAN Register Organization 14
Table 3-2 Module Memory Map 15
Table 3-3 Synchronization Jump Width 22
Table 3-4 Baud Rate Prescaler 22
Table 3-5 Time Segment 2 Values. 23
Table 3-6 Time Segment 1 Values. 24
Table 3-7 Identifier Acceptance Mode Settings 35
Table 3-8 Identifier Acceptance Hit Indication 36
Table 3-9 Message Buffer Organization 45
Table 3-10 Data Length Codes 48
Table 4-1 Time Segment Syntax 60
Table 4-2 CAN Standard Compliant Bit Time Segment Settings 61
Table 4-3 CPU vs. MSCAN Operating Modes 62
Table 4-4 Interrupt Vectors 67

Preface

Terminology

Acronyms and Abbreviations	
ACK	Acknowledge
CAN	Controller Area Network
CRC	Cyclic Redundancy Code
EOF	End of Frame
FIFO	First-In-First-Out Memory
IFS	Inter-Frame Sequence
MSCAN	Motorola Scalable CAN Module
SOF	Start of Frame
CPU bus	CPU related read/write data bus
CAN bus	CAN protocol related serial bus
Oscillator Clock	Direct clock from external oscillator
Bus Clock	Clock related to CPU bus
CAN Clock	Clock related to the MSCAN protocol

Document References

1. Bosch CAN 2.0A/B protocol specification dated September 1991

Section 1 Introduction

1.1 Overview

The Motorola Scalable Controller Area Network (MSCAN) definition is based on the MSCAN12 definition which is the specific implementation of the Motorola Scalable CAN concept targeted for the Motorola MC68HC12 Microcontroller Family.

The module is a communication controller implementing the CAN 2.0 A/B protocol as defined in the BOSCH specification dated September 1991. For users to fully understand the MSCAN specification, it is recommended that the Bosch specification be read first to familiarize the reader with the terms and concepts contained within this document.

The CAN protocol was primarily, but not only, designed to be used as a vehicle serial data bus, meeting the specific requirements of this field: real-time processing, reliable operation in the EMI environment of a vehicle, cost-effectiveness and required bandwidth.

MSCAN utilizes an advanced buffer arrangement resulting in a predictable real-time behavior and simplifies the application software.

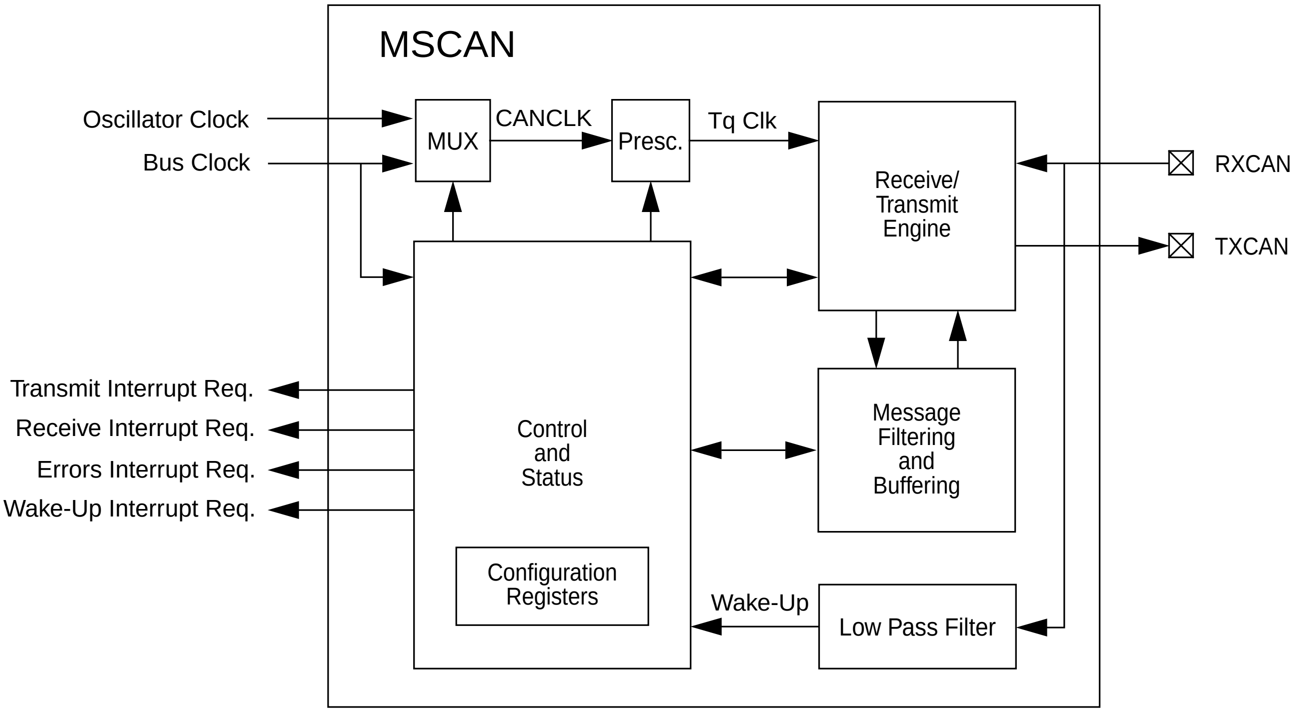

Figure 1-1 MSCAN Block Diagram

1.2 Features

The basic features of the MSCAN are as follows:

- Implementation of the CAN protocol - Version 2.0A/B
  - Standard and extended data frames
  - 0 - 8 bytes data length
  - Programmable bit rate up to 1 Mbps[1]
  - Support for remote frames
- 5 receive buffers with FIFO storage scheme
- 3 transmit buffers with internal prioritization using a “local priority” concept
- Flexible maskable identifier filter supports two full size extended identifier filters (two 32-bit) or four 16-bit filters or eight 8-bit filters
- Programmable wake-up functionality with integrated low-pass filter
- Programmable loop back mode supports self-test operation
- Programmable listen-only mode for monitoring of CAN bus
- Programmable Bus-Off recovery functionality
- Separate signalling and interrupt capabilities for all CAN receiver and transmitter error states (Warning, Error Passive, Bus-Off)
- Programmable MSCAN clock source either Bus Clock or Oscillator Clock
- Internal timer for time-stamping of received and transmitted messages
- Three low power modes: Sleep, Power Down and MSCAN Enable
- Global initialization of configuration registers

NOTES:

[1] Depending on the actual bit timing and the clock jitter of the PLL.

Section 2 External Signal Description

2.1 Overview

This section lists and describes the signals that connect off chip.

2.2 Detailed Signal Description

The MSCAN uses two external pins.

2.2.1 RXCAN — CAN Receiver Input Pin

RXCAN is the MSCAN receiver input pin.

2.2.2 TXCAN — CAN Transmitter Output Pin

TXCAN is the MSCAN transmitter output pin. The TXCAN output pin represents the logic level on the CAN bus:

0 = Dominant state
1 = Recessive state

2.3 CAN System

A typical CAN system with MSCAN is shown in **Figure 2-1**. Each CAN station is connected physically to the CAN bus lines through a transceiver chip. The transceiver is capable of driving the large current needed for the CAN bus and has current protection against defected CAN or defected stations.

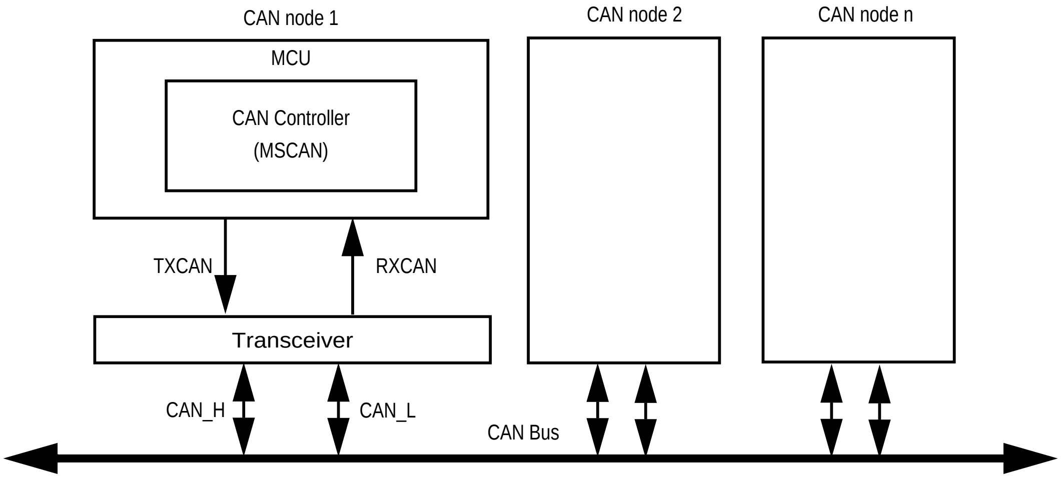

Figure 2-1 The CAN System

Section 3 Memory Map/Register Definition

3.1 Overview

This section provides a detailed description of all registers accessible in the MSCAN.

3.2 Module Memory Map

Table 3-1 and **Table 3-2** give an overview on all registers and their individual bits in the MSCAN memory map. The *register address* results from the addition of *base address* and *address offset*. The *base address* is determined at the MCU level. The *address offset* is defined at the module level.

The MSCAN occupies 64 bytes in the memory space. The base address of the MSCAN module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset.

Address Offset		
$__00 $__0B	CONTROL REGISTERS	12 BYTES
$__0C	RESERVED	1 BYTE
$__0D	CONTROL REGISTER	1 BYTE
$__0E $__0F	ERROR COUNTERS	2 BYTES
$__10 $__1F	IDENTIFIER FILTER	16 BYTES
$__20 $__2F	RECEIVE BUFFER (Window)	16 BYTES
$__30 $__3F	TRANSMIT BUFFER (Window)	16 BYTES

Table 3-1 MSCAN Register Organization

Table 3-1 shows the individual registers associated with the MSCAN and their relative offset from the base address. The detailed register descriptions follow in the order they appear in the register map (see **Table 3-2**).

Table 3-2 Module Memory Map

Address	Use	Access
$__00	MSCAN Control Register 0 (CANCTL0)	R/W[1]
$__01	MSCAN Control Register 1 (CANCTL1)	R/W[1]
$__02	MSCAN Bus Timing Register 0 (CANBTR0)	R/W
$__03	MSCAN Bus Timing Register 1 (CANBTR1)	R/W
$__04	MSCAN Receiver Flag Register (CANRFLG)	R/W[1]
$__05	MSCAN Receiver Interrupt Enable Register (CANRIER)	R/W
$__06	MSCAN Transmitter Flag Register (CANTFLG)	R/W[1]
$__07	MSCAN Transmitter Interrupt Enable Register (CANTIER)	R/W[1]
$__08	MSCAN Transmitter Message Abort Control (CANTARQ)	R/W[1]
$__09	MSCAN Transmitter Message Abort Control (CANTAAK)	R
$__0A	MSCAN Transmit Buffer Selection (CANTBSEL)	R/W[1]
$__0B	MSCAN Identifier Acceptance Control Register (CANIDAC)	R/W[1]
$__0C	RESERVED	
$__0D	MSCAN Miscellaneous Register (CANMISC)	R/W[1]
$__0E	MSCAN Receive Error Counter Register (CANRXERR)	R
$__0F	MSCAN Transmit Error Counter Register (CANTXERR)	R
$__10	MSCAN Identifier Acceptance Register 0 (CANIDAR0)	R/W
$__11	MSCAN Identifier Acceptance Register 1 (CANIDAR1)	R/W
$__12	MSCAN Identifier Acceptance Register 2 (CANIDAR2)	R/W
$__13	MSCAN Identifier Acceptance Register 3 (CANIDAR3)	R/W
$__14	MSCAN Identifier Mask Register 0 (CANIDMR0)	R/W
$__15	MSCAN Identifier Mask Register 1 (CANIDMR1)	R/W
$__16	MSCAN Identifier Mask Register 2 (CANIDMR2)	R/W
$__17	MSCAN Identifier Mask Register 3 (CANIDMR3)	R/W
$__18	MSCAN Identifier Acceptance Register 4 (CANIDAR4)	R/W
$__19	MSCAN Identifier Acceptance Register 5 (CANIDAR5)	R/W
$__1A	MSCAN Identifier Acceptance Register 6 (CANIDAR6)	R/W
$__1B	MSCAN Identifier Acceptance Register 7 (CANIDAR7)	R/W
$__1C	MSCAN Identifier Mask Register 4 (CANIDMR4)	R/W
$__1D	MSCAN Identifier Mask Register 5 (CANIDMR5)	R/W
$__1E	MSCAN Identifier 6 Mask Register 6 (CANIDMR6)	R/W
$__1F	MSCAN Identifier Mask Register 7 (CANIDMR7)	R/W
$__20 -$__2F	Foreground Receive Buffer (CANRXFG)	R[2]
$__30 -$__3F	Foreground Transmit Buffer (CANTXFG)	R[2]/W

NOTES:

1. Refer to detailed register description for write access restrictions on per bit basis.
2. Reserved bits and unused bits within the TX- and RX-Buffers (CANTXFG, CANRXFG) will be read as “X”, because of RAM based implementation.

3.3 Register Descriptions

This section describes in detail all the registers and register bits in the MSCAN module. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order. All bits of all registers in this module are completely synchronous to internal clocks during a register read.

3.3.1 Programmer’s Model of Control Registers

The programmer’s model is laid out for maximum simplicity and efficiency. **Table 3-2** provides an overview of the control registers for the MSCAN.

3.3.1.1 MSCAN Control Register 0 (CANCTL0)

The CANCTL0 register provides various control bits of the MSCAN module as described below.

Address Offset: $__00

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
Write:		▨		▨				
Reset:	0	0	0	0	0	0	0	1

▨ = Unimplemented

Figure 3-1 MSCAN Control Register 0 (CANCTL0)

NOTE: *The CANCTL0 register, except the WUPE, INITRQ and SLPRQ bits, is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime when out of Initialization Mode; exceptions are read-only bits RXACT and SYNCH, bit RXFRM which is set by the module only and bit INITRQ which is also writable in Initialization Mode.

RXFRM — Received Frame Flag

This bit is read and clear only. It is set when a receiver has received a valid message correctly, independently of the filter configuration. Once set, it remains set until cleared by software or reset. Clearing is done by writing a ‘1’ to the bit. A write ‘0’ is ignored. This bit is not valid in loop back mode.

1 = A valid message was received since last clearing of this flag
0 = No valid message was received since last clearing this flag.

NOTE: *The MSCAN must be in Normal Mode for this bit to become set.*

RXACT — Receiver Active Status

This read-only flag indicates the MSCAN is receiving a message. The flag is controlled by the receiver front end. This bit is not valid in loop back mode.

1 = MSCAN is receiving a message (including when arbitration is lost)[1]
0 = MSCAN is transmitting or idle[1].

CSWAI — CAN Stops in Wait Mode

Enabling this bit allows for lower power consumption in Wait Mode by disabling all the clocks at the CPU bus interface to the MSCAN module.

NOTES:

[1] See [**1.**] for a detailed definition of transmitter and receiver states.

1 = The module ceases to be clocked during Wait Mode.
0 = The module is not affected during Wait Mode.

NOTE: *In order to protect from accidentally violating the CAN protocol the TXCAN pin is immediately forced to a recessive state when the CPU enters Wait (CSWAI=1) or Stop Mode (see* ***4.6.2 CPU Wait Mode*** *and* ***4.6.3 CPU Stop Mode****)*

SYNCH — Synchronized Status

This read-only flag indicates whether the MSCAN is synchronized to the CAN bus and, as such, can participate in the communication process. It is set and cleared by the MSCAN.
1 = MSCAN is synchronized to the CAN bus.
0 = MSCAN is not synchronized to the CAN bus.

TIME - Timer Enable

This bit activates an internal 16-bit wide free running timer which is clocked by the bit clock rate. If the timer is enabled, a 16-bit time stamp will be assigned to each transmitted/received message within the active TX/RX buffer. As soon as a message is acknowledged on the CAN bus, the time stamp will be written to the highest bytes ($_E, $_F) in the appropriate buffer **3.3.2 Programmer's Model of Message Storage**. The internal timer is reset (all bits set to "0") when disabled. This bit is held low in Initialization Mode.
1 = Enable internal MSCAN timer.
0 = Disable internal MSCAN timer.

WUPE — Wake-Up Enable

This configuration bit allows the MSCAN to restart from Sleep Mode when traffic on CAN is detected (see **4.6.4 MSCAN Sleep Mode**).
1 = Wake-Up enabled– The MSCAN is able to restart.
0 = Wake-Up disabled– The MSCAN ignores traffic on CAN.

NOTE: *The CPU has to make sure that the WUPE register and the WUPIE Wake-Up interrupt enable register* ***3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)*** *is enabled, if the recovery mechanism from STOP or WAIT is required.*

SLPRQ — Sleep Mode Request

This bit requests the MSCAN to enter Sleep Mode, which is an internal power saving mode (see **4.6.4 MSCAN Sleep Mode**). The Sleep Mode request is serviced when the CAN bus is idle, i.e. the module is not receiving a message and all transmit buffers are empty. The module indicates entry to Sleep Mode by setting SLPAK=1 (**3.3.1.2 MSCAN Control Register 1 (CANCTL1)**). Sleep Mode will be active until SLPRQ is cleared by the CPU or, depending on the setting of WUPE bit, the MSCAN detects activity on the CAN bus and clears the SLPRQ itself.
1 = Sleep Mode Request – The MSCAN enters Sleep Mode when CAN bus idle.
0 = Running – The MSCAN functions normally.

NOTE: *The CPU cannot clear the SLPRQ bit before the MSCAN has entered Sleep Mode (SLPRQ=1 and SLPAK=1)*

INITRQ — Initialization Mode Request

When this bit is set by the CPU, the MSCAN skips to Initialization Mode (see **4.6.5 MSCAN Initialization Mode**). Any ongoing transmission or reception is aborted and synchronization to the CAN bus is lost. The module indicates entry to Initialization Mode by setting INITAK=1 (**3.3.1.2 MSCAN Control Register 1 (CANCTL1)**).

The following registers enter their hard reset state and restore their default values: CANCTL0[1], CANRFLG[2], CANRIER[3], CANTFLG, CANTIER, CANTARQ, CANTAAK, CANTBSEL.

The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, CANIDMR0-7 can only be written by the CPU when the MSCAN is in Initialization Mode (INITRQ=1 and INITAK=1). The values of the error counters are not affected by Initialization Mode.

When this bit is cleared by the CPU, the MSCAN restarts and then tries to synchronize to the CAN bus. If the MSCAN is not in Bus-Off state, it synchronizes after 11 consecutive recessive bits on the CAN bus; if the MSCAN is in Bus-Off state it continues to wait for 128 occurrences of 11 consecutive recessive bits.

Writing to other bits in CANCTL0, CANRFLG, CANRIER, CANTFLG or CANTIER must only be done after Initialization Mode is left, which is INITRQ=0 and INITAK=0.
1 = MSCAN in Initialization Mode.
0 = Normal operation.

NOTE: *The CPU cannot clear the INITRQ bit before the MSCAN has entered Initialization Mode (INITRQ=1 and INITAK=1)*

NOTE: *In order to protect from accidentally violating the CAN protocol the TXCAN pin is immediately forced to a recessive state when the Initialization Mode is requested by the CPU. Thus the recommended procedure is to bring the MSCAN into Sleep Mode (SLPRQ=1 and SLPAK=1) before.*

NOTES:

[1] Except the WUPE, INITRQ and SLPRQ bits

[2] The TSTAT1, TSTAT0 bits are not affected by Initialization Mode

[3] The RSTAT1, RSTAT0 bits are not affected by Initialization Mode

3.3.1.2 MSCAN Control Register 1 (CANCTL1)

The CANCTL1 register provides various control bits and handshake status information of the MSCAN module as described below.

Address Offset: $__01

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
Write:							▨	▨
Reset:	0	0	0	1	0	0	0	1

▨ = Unimplemented

Figure 3-2 MSCAN Control Register 1 (CANCTL1)

Read: Anytime

Write: Anytime when INITRQ=1 <u>and</u> INITAK=1, except CANE which is write once in normal and anytime in special system operation modes when the MSCAN is in Initialization Mode (INITRQ=1 and INITAK=1).

CANE — MSCAN Enable
1 = The MSCAN module is enabled.
0 = The MSCAN module is disabled.

CLKSRC — MSCAN Clock Source

This bit defines the clock source for the MSCAN module (only for systems with a clock generation module; **4.3.2 Clock System** and **Figure 4-5**).
1 = The MSCAN clock source is the Bus Clock.
0 = The MSCAN clock source is the Oscillator Clock.

LOOPB — Loop Back Self Test Mode

When this bit is set, the MSCAN performs an internal loop back which can be used for self test operation. The bit stream output of the transmitter is fed back to the receiver internally. The RXCAN input pin is ignored and the TXCAN output goes to the recessive state (logic ‘1’). The MSCAN behaves as it does normally when transmitting and treats its own transmitted message as a message received from a remote node. In this state, the MSCAN ignores the bit sent during the ACK slot in the CAN frame Acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

1 = Loop Back Self Test enabled
0 = Loop Back Self Test disabled

LISTEN — Listen Only Mode

This bit configures the MSCAN as a CAN bus monitor. When the bit is set, all valid CAN messages with matching ID are received, but no acknowledgement or error frames are sent out **4.5.4 Listen-Only Mode**. In addition the error counters are frozen.
Listen Only Mode supports applications which require “hot plugging” or throughput analysis. The MSCAN is unable to transmit any messages, when Listen Only Mode is active.

1 = Listen Only Mode activated
0 = Normal operation

BORM — Bus-Off Recovery Mode

This bits configures the Bus-Off state recovery mode of the MSCAN. Refer to section **5.2 Bus-Off recovery** for details.

1 = Bus-Off recovery upon user request
0 = Automatic Bus-Off recovery (see [**1.**])

WUPM — Wake-Up Mode

This bit defines whether the integrated low-pass filter is applied to protect the MSCAN from spurious wake-up **4.6.4 MSCAN Sleep Mode**.

1 = MSCAN wakes-up the CPU only in case of a dominant pulse on the CAN bus which has a length of T_{wup} and WUPE=1 in CANCTL0 (see **3.3.1.1 MSCAN Control Register 0 (CANCTL0)**).
0 = MSCAN wakes-up the CPU after any recessive to dominant edge on the CAN bus and WUPE=1 in CANCTL0.

SLPAK — Sleep Mode Acknowledge

This flag indicates whether the MSCAN module has entered Sleep Mode **4.6.4 MSCAN Sleep Mode**. It is used as a handshake flag for the SLPRQ Sleep Mode request. Sleep Mode is active when SLPRQ=1 and SLPAK=1. Depending on the setting of the WUPE bit the MSCAN will clear the flag if it detects activity on the CAN bus while in Sleep Mode.

1 = Sleep Mode Active – The MSCAN has entered Sleep Mode.
0 = Running – The MSCAN operates normally.

INITAK — Initialization Mode Acknowledge

This flag indicates whether the MSCAN module is in Initialization Mode **4.6.5 MSCAN Initialization Mode**. It is used as a handshake flag for the INITRQ Initialization Mode request. Initialization Mode is active when INITRQ=1 and INITAK=1.

The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, CANIDMR0-7 can only be written by the CPU when the MSCAN is in Initialization Mode.

1 = Initialization Mode Active – The MSCAN has entered Initialization Mode.
0 = Running – The MSCAN operates normally.

3.3.1.3 MSCAN Bus Timing Register 0 (CANBTR0)

The CANBTR0 register configures various CAN bus timing parameters of the MSCAN module.

Address Offset: $__02

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
Reset:	0	0	0	0	0	0	0	0

Figure 3-3 MSCAN Bus Timing Register 0 (CANBTR0)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

SJW1, SJW0 — Synchronization Jump Width

The synchronization jump width defines the maximum number of time quanta (Tq) clock cycles a bit can be shortened or lengthened to achieve resynchronization to data transitions on the CAN bus (see **Table 3-3**).

Table 3-3 Synchronization Jump Width

SJW1	SJW0	Synchronization jump width
0	0	1 Tq clock cycle
0	1	2 Tq clock cycles
1	0	3 Tq clock cycles
1	1	4 Tq clock cycles

BRP[5-0] — Baud Rate Prescaler

These bits determine the time quanta (Tq) clock which is used to build up the bit timing (see **Table 3-4**).

Table 3-4 Baud Rate Prescaler

BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	Prescaler value (P)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
0	0	0	0	1	1	4
:	:	:	:	:	:	:
1	1	1	1	1	1	64

3.3.1.4 MSCAN Bus Timing Register 1 (CANBTR1)

The CANBTR1 register configures various CAN bus timing parameters of the MSCAN module.

Address Offset: $__03

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
Reset:	0	0	0	0	0	0	0	0

Figure 3-4 MSCAN Bus Timing Register 1 (CANBTR1)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

SAMP — Sampling

This bit determines the number of samples of the CAN bus to be taken per bit time. If set, three samples per bit are taken; the regular one (sample point) and two preceding samples using a majority rule. For higher bit rates, it is recommended that SAMP be cleared which means that only one sample is taken per bit.

1 = Three samples per bit[1].
0 = One sample per bit.

TSEG22 – TSEG20 — Time Segment 2

Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see **Figure 4-6 Segments within the Bit Time**).

Time segment 2 (TSEG2) values are programmable as shown in **Table 3-5**.

Table 3-5 Time Segment 2 Values

TSEG22	TSEG21	TSEG20	Time segment 2
0	0	0	1 Tq clock cycle[1]
0	0	1	2 Tq clock cycles
:	:	:	:
1	1	0	7 Tq clock cycles
1	1	1	8 Tq clock cycles

NOTES:
1. This setting is not valid. Please refer to **Table 4-2 CAN Standard Compliant Bit Time Segment Settings** for valid settings.

NOTES:
[1] In this case, PHASE_SEG1 must be at least 2 Time Quanta.

TSEG13 – TSEG10 — Time Segment 1

Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see **Figure 4-6 Segments within the Bit Time**).

Time segment 1 (TSEG1) values are programmable as shown in **Table 3-6**.

Table 3-6 Time Segment 1 Values

TSEG13	TSEG12	TSEG11	TSEG10	Time segment 1
0	0	0	0	1 Tq clock cycle[1]
0	0	0	1	2 Tq clock cycles[1]
0	0	1	0	3 Tq clock cycles[1]
0	0	1	1	4 Tq clock cycles
:	:	:	:	:
1	1	1	0	15 Tq clock cycles
1	1	1	1	16 Tq clock cycles

NOTES:

1. This setting is not valid. Please refer to **Table 4-2 CAN Standard Compliant Bit Time Segment Settings** for valid settings.

The bit time is determined by the oscillator frequency, the baud rate prescaler, and the number of time quanta (Tq) clock cycles per bit (as shown in **Table 3-5** and **Table 3-6** above).

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{\text{CANCLK}}} \bullet (\text{number of Time Quanta})$$

3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)

A flag can only be cleared when the condition which caused the setting is no longer valid and can only be cleared by software (writing a ‘1’ to the corresponding bit position). Every flag has an associated interrupt enable bit in the CANRIER register.

Address Offset: $__04

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
Write:								
Reset:	0	0	0	0	0	0	0	0

Figure 3-5 MSCAN Receiver Flag Register (CANRFLG)

NOTE: *The CANRFLG register is held in the reset state*[1] *when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime when out of Initialization Mode, except RSTAT[1:0] and TSTAT[1:0] flags which are read-only; write of ‘1’ clears flag; write of ‘0’ ignored

WUPIF — Wake-Up Interrupt Flag

If the MSCAN detects CAN bus activity while in Sleep Mode **4.6.4 MSCAN Sleep Mode** and the WUPE=1 in CANTCTL0 (see **3.3.1.1 MSCAN Control Register 0 (CANCTL0)**), it will set the WUPIF flag. If not masked, a Wake-Up interrupt is pending while this flag is set.

1 = MSCAN detected activity on the CAN bus and requested wake-up.
0 = No wake-up activity observed while in Sleep Mode.

CSCIF — CAN Status Change Interrupt Flag

This flag is set when the MSCAN changes its current CAN bus status due to the actual value of the Transmit Error Counter (TEC) and the Receive Error Counter (REC). An additional 4-bit (RSTAT[1:0], TSTAT[1:0]) status register, which is split into separate sections for TEC/REC, informs the system on the actual CAN bus status **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)**. If not masked, an Error interrupt is pending while this flag is set. CSCIF provides a blocking interrupt. That guarantees that the Receiver / Transmitter status bits (RSTAT/TSTAT) are only updated when no CAN Status Change interrupt is pending. If the TECs/RECs change their current value after the CSCIF is asserted and therefore would cause an additional state change in the RSTAT/TSTAT bits, these bits keep their old state bits until the current CSCIF interrupt is cleared again.

1 = MSCAN changed current CAN bus status.
0 = No change in CAN bus status occurred since last interrupt.

NOTES:

[1] The RSTAT[1:0], TSTAT[1:0] bits are not affected by Initialization Mode

RSTAT1, RSTAT0 — Receiver Status Bits

The values of the error counters control the actual CAN bus status of the MSCAN. As soon as the Status Change Interrupt Flag (CSCIF) is set these bits indicate the appropriate receiver related CAN bus status of the MSCAN. The coding for the bits RSTAT1, RSTAT0 is:

00 = RxOK: 0 ≤ Receive Error Counter ≤ 96
01 = RxWRN: 96 < Receive Error Counter ≤ 127
10 = RxERR: 127 < Receive Error Counter
11 = Bus-Off[1]*: Transmit Error Counter > 255*

TSTAT1, TSTAT0 — Transmitter Status Bits

The values of the Error Counters control the actual CAN bus status of the MSCAN. As soon as the Status Change Interrupt Flag (CSCIF) is set these bits indicate the appropriate transmitter related CAN bus status of the MSCAN. The coding for the bits TSTAT1, TSTAT0 is:

00 = TxOK: 0 ≤ Transmit Error Counter ≤ 96
01 = TxWRN: 96 < Transmit Error Counter ≤ 127
10 = TxERR: 127 < Transmit Error Counter ≤ 255
11 = Bus-Off: Transmit Error Counter > 255

OVRIF — Overrun Interrupt Flag

This flag is set when a data overrun condition occurs. If not masked, an Error interrupt is pending while this flag is set.

1 = A data overrun detected.
0 = No data overrun condition.

RXF — Receive Buffer Full Flag

The RXF flag is set by the MSCAN when a new message is shifted in the receiver FIFO. This flag indicates whether the shifted buffer is loaded with a correctly received message (matching identifier, matching Cyclic Redundancy Code (CRC) and no other errors detected). After the CPU has read that message from the RxFG buffer in the receiver FIFO, the RXF flag must be cleared to release the buffer. A set RXF flag prohibits the shifting of the next FIFO entry into the foreground buffer (RxFG). If not masked, a Receive interrupt is pending while this flag is set.

1 = The receiver FIFO is not empty. A new message is available in the RxFG.
0 = No new message available within the RxFG.

NOTE: *To ensure data integrity, do not read the receive buffer registers while the RXF flag is cleared.*

For MCUs with dual CPUs, reading the receive buffer registers while the RXF flag is cleared may result in a CPU fault condition.

NOTES:

[1] Redundant Information for the most critical CAN bus status which is “Bus-Off”. This only occurs if the Tx Error Counter exceeds a number of 255 errors. Bus-Off affects the receiver state. As soon as the transmitter leaves its Bus-Off state the receiver state skips to RxOK too. Refer also to TSTAT[1:0] coding.

3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)

This register contains the interrupt enable bits for the interrupt flags described above.

Address Offset: $__05

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
Reset:	0	0	0	0	0	0	0	0

Figure 3-6 MSCAN Receiver Interrupt Enable Register (CANRIER)

> ***NOTE:*** *The CANRIER register is held in the reset state[1] when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime when out of Initialization Mode

WUPIE — Wake-Up Interrupt Enable
1 = A wake-up event causes a Wake-Up interrupt request.
0 = No interrupt request is generated from this event.

> ***NOTE:*** *The CPU has to make sure that the Wake-Up interrupt register and the WUPE register* ***3.3.1.1 MSCAN Control Register 0 (CANCTL0)*** *is enabled, if the recovery mechanism from STOP or WAIT is required.*

CSCIE — CAN Status Change Interrupt Enable
1 = A CAN Status Change event causes an error interrupt request.
0 = No interrupt request is generated from this event.

RSTATE1, RSTATE0— Receiver Status Change Enable

These RSTAT enable bits control the sensitivity level in which receiver state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the RSTAT flags still indicate the actual receiver state and are only updated if no CSCIF interrupt is pending.

11 = generate CSCIF interrupt on all state changes
10 = generate CSCIF interrupt only if the receiver enters or leaves “RxErr” or “Bus-Off”[2] state. Discard other receiver state changes for generating CSCIF interrupt.
01 = generate CSCIF interrupt only if the receiver enters or leaves “Bus-Off” state. Discard other receiver state changes for generating CSCIF interrupt.

NOTES:

[1] The RSTATE[1:0], TSTATE[1:0] bits are not affected by Initialization Mode

[2] Bus-Off state is only defined by the CAN standard [**1.**] for transmitters. Because the only possible state change for the transmitter from Bus-Off to TxOK also forces the receiver to skip its current state to RxOK, the coding of the RXSTAT[1:0] flags define an additional Bus-Off state for the receiver **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)**

00 = do not generate any CSCIF interrupt caused by receiver state changes.

TSTATE1, TSTATE0— Transmitter Status Change Enable

These TSTAT enable bits control the sensitivity level in which transmitter state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the TSTAT flags still indicate the actual transmitter state and are only updated if no CSCIF interrupt is pending.

11 = generate CSCIF interrupt on all state changes

10 = generate CSCIF interrupt only if the transmitter enters or leaves “TxErr” or “Bus-Off” state. Discard other transmitter state changes for generating CSCIF interrupt.

01 = generate CSCIF interrupt only if the transmitter enters or leaves “Bus-Off” state. Discard other transmitter state changes for generating CSCIF interrupt.

00 = do not generate any CSCIF interrupt caused by transmitter state changes.

OVRIE — Overrun Interrupt Enable

1 = An overrun event causes an error interrupt request.
0 = No interrupt request is generated from this event.

RXFIE — Receiver Full Interrupt Enable

1 = A receive buffer full (successful message reception) event causes a receiver interrupt request.
0 = No interrupt request is generated from this event.

3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)

The Transmit Buffer Empty flags each have an associated interrupt enable bit in the CANTIER register.

Address Offset: $__06

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	TXE2	TXE1	TXE0
Write:	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	1	1	1

▨ = Unimplemented

Figure 3-7 MSCAN Transmitter Flag Register (CANTFLG)

NOTE: *The CANTFLG register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime for TXEx flags when not in Initialization Mode; write of ‘1’ clears flag, write of ‘0’ ignored

TXE2 - TXE0 —Transmitter Buffer Empty

This flag indicates that the associated transmit message buffer is empty, and thus not scheduled for transmission. The CPU must clear the flag after a message is set up in the transmit buffer and is due for transmission. The MSCAN sets the flag after the message is sent successfully. The flag is also set by the MSCAN when the transmission request is successfully aborted due to a pending abort request (see **3.3.1.9 MSCAN Transmitter Message Abort Request Register (CANTARQ)**). If not masked, a Transmit interrupt is pending while this flag is set.

Clearing a TXEx flag also clears the corresponding ABTAKx (see **3.3.1.10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)**). When a TXEx flag is set, the corresponding ABTRQx bit is cleared (see **3.3.1.9 MSCAN Transmitter Message Abort Request Register (CANTARQ)**).

When Listen-Mode is active (see **3.3.1.2 MSCAN Control Register 1 (CANCTL1)**) the TXEx flags cannot be cleared and no transmission is started.

Read and write accesses to the transmit buffer will be blocked, if the corresponding TXEx bit is cleared (TXEx=’0’) and the buffer is scheduled for transmission.

1 = The associated message buffer is empty (not scheduled).
0 = The associated message buffer is full (loaded with a message due for transmission).

3.3.1.8 MSCAN Transmitter Interrupt Enable Register (CANTIER)

This register contains the interrupt enable bits for the Transmit Buffer Empty interrupt flags.

Address Offset: $__07

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
Write:	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-8 MSCAN Transmitter Interrupt Enable Register (CANTIER)

NOTE: *The CANTIER register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime when not in Initialization Mode

TXEIE2 - TXEIE0 — Transmitter Empty Interrupt Enable

1 = A transmitter empty (transmit buffer available for transmission) event causes a transmitter empty interrupt request.

0 = No interrupt request is generated from this event.

3.3.1.9 MSCAN Transmitter Message Abort Request Register (CANTARQ)

The CANTARQ register allows abort request of queued messages as described below.

Address Offset: $__08

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
Write:	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-9 MSCAN Transmitter Message Abort Request Register (CANTARQ)

NOTE: *The CANTARQ register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Anytime

Write: Anytime when not in Initialization Mode

ABTRQ2 - ABTRQ0 — Abort Request

The CPU sets the ABTRQx bit to request that a scheduled message buffer (TXEx=0) be aborted. The MSCAN grants the request if the message has not already started transmission, or if the transmission is not successful (lost arbitration or error). When a message is aborted, the associated TXE (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and Abort Acknowledge flags (ABTAK, see **3.3.1.10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)**) are set and a transmit interrupt occurs if enabled. The CPU cannot reset ABTRQx. ABTRQx is reset whenever the associated TXE flag is set.

1 = Abort request pending.
0 = No abort request.

3.3.1.10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)

The CANTAAK register indicates the successful abort of a queued message, if requested by the appropriate bits in the CANTARQ register

Address Offset: $__09

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)

NOTE: *The CANTAAK register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1).*

Read: Anytime

Write: Unimplemented for ABTAKx flags;

ABTAK2 - ABTAK0 — Abort Acknowledge

This flag acknowledges that a message was aborted due to a pending abort request from the CPU. After a particular message buffer is flagged empty, this flag can be used by the application software to identify whether the message was aborted successfully or was sent anyway. The ABTAKx flag is cleared whenever the corresponding TXE flag is cleared.

1 = The message was aborted.
0 = The message was not aborted.

3.3.1.11 MSCAN Transmit Buffer Selection Register (CANTBSEL)

The CANTBSEL register allows the selection of the actual transmit message buffer, which will be then accessible in the CANTXFG register space (**3.3.1 Programmer’s Model of Control Registers**).

Address Offset: $__0A

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	TX2	TX1	TX0
Write:	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-11 MSCAN Transmit Buffer Selection Register (CANTBSEL)

NOTE: *The CANTBSEL register is held in the reset state when the Initialization Mode is active (INITRQ=1 and INITAK=1). This register is writable again as soon as the Initialization Mode is left (INITRQ=0 and INITAK=0).*

Read: Find the lowest ordered bit set to “1”, all other bits will be read as “0”

Write: Anytime when not in Initialization Mode

TX2 - TX0 — Transmit Buffer Select

The lowest numbered bit places the respective transmit buffer in the CANTXFG register space (e.g. TX1=1 and TX0=1 selects transmit buffer TX0, TX1=1 and TX0=0 selects transmit buffer TX1)

Read and write accesses to the selected transmit buffer will be blocked, if the corresponding TXEx bit is cleared and the buffer is scheduled for transmission **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**.

1 = The associated message Buffer is selected, if lowest numbered bit.
0 = The associated message buffer is deselected

NOTE: *The following gives a short programming example of the usage of the CANTBSEL register:*

The application software wants to get the next available transmit buffer. It reads the CANTFLG register and writes this value back into the CANTBSEL register. In this example Tx buffers TX1 and TX2 are available. The value read from CANTFLG is therefore 0b0000_0110. When writing this value back to CANTBSEL the Tx buffer TX1 is selected in the CANTXFG because the lowest numbered bit set to “1” is at bit position 1. Reading back this value out of CANTBSEL results in 0b0000_0010, because only the lowest numbered bit position set to “1” is presented. This mechanism eases the application software the selection of the next available Tx buffer.

LDD CANTFLG; value read is 0b0000_0110

STD CANTBSEL; value written is 0b0000_0110

LDD CANTBSEL; value read is 0b0000_0010

If all transmit message buffers are deselected no accesses are allowed to the CANTXFG registers.

3.3.1.12 MSCAN Identifier Acceptance Control Register (CANIDAC)

The CANIDAC register is used for identifier acceptance control as described below.

Address Offset: $__0B

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
Write:	▨	▨			▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-12 MSCAN Identifier Acceptance Control Register (CANIDAC)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1), except bits IDHITx which are read-only

IDAM1 - IDAM0 — Identifier Acceptance Mode

> The CPU sets these flags to define the identifier acceptance filter organization **4.3 Identifier Acceptance Filter**. **Table 3-7** summarizes the different settings. In Filter Closed mode, no message is accepted such that the foreground buffer is never reloaded.

Table 3-7 Identifier Acceptance Mode Settings

IDAM1	IDAM0	Identifier Acceptance Mode
0	0	Two 32 bit Acceptance Filters
0	1	Four 16 bit Acceptance Filters
1	0	Eight 8 bit Acceptance Filters
1	1	Filter Closed

IDHIT2 - IDHIT0 — Identifier Acceptance Hit Indicator

> The MSCAN sets these flags to indicate an identifier acceptance hit **4.3 Identifier Acceptance Filter**. **Table 3-8** summarizes the different settings.

Table 3-8 Identifier Acceptance Hit Indication

IDHIT2	IDHIT1	IDHIT0	Identifier Acceptance Hit
0	0	0	Filter 0 Hit
0	0	1	Filter 1 Hit
0	1	0	Filter 2 Hit
0	1	1	Filter 3 Hit
1	0	0	Filter 4 Hit
1	0	1	Filter 5 Hit
1	1	0	Filter 6 Hit
1	1	1	Filter 7 Hit

The IDHITx indicators are always related to the message in the foreground buffer (RxFG). When a message gets shifted into the foreground buffer of the receiver FIFO the indicators are updated as well.

3.3.1.13 MSCAN Reserved Register

This register is reserved for factory testing of the MSCAN module and is not available in normal system operation modes.

Address Offset: $__0C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-13 MSCAN Reserved Register

Read: Always read $00 in normal system operation modes

Write: Unimplemented in normal system operation modes

NOTE: *Writing to this register when in special modes can alter the MSCAN functionality.*

3.3.1.14 MSCAN Miscellaneous Register (CANMISC)

This register provides additional features.

Address Offset: $__0D

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	0	BOHOLD
Write:	▨	▨	▨	▨	▨	▨	▨	
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-14 MSCAN Miscellaneous Register (CANMISC)

Read: Anytime

Write: Anytime; write of ‘1’ clears flag; write of ‘0’ ignored

BOHOLD — Bus-Off state Hold until user request

If BORM is set in **3.3.1.2 MSCAN Control Register 1 (CANCTL1)** this bit indicates whether the module has entered the Bus-Off state. Clearing this bit requests the recovery from Bus-Off. Refer to section **5.2 Bus-Off recovery** for details.

1 = Module is Bus-Off and holds this state until user request
0 = Module is not Bus-Off or recovery has been requested by user in Bus-Off state

3.3.1.15 MSCAN Receive Error Counter (CANRXERR)

This register reflects the status of the MSCAN receive error counter.

Address Offset: $__0E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-15 MSCAN Receive Error Counter (CANRXERR)

Read: Only when in Sleep Mode (SLPRQ=1 and SLPAK=1) or Initialization Mode (INITRQ=1 and INITAK=1)

Write: Unimplemented

NOTE: *Reading this register when in any other mode other than Sleep or Initialization Mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.*

NOTE: *Writing to this register when in special modes can alter the MSCAN functionality.*

3.3.1.16 MSCAN Transmit Error Counter (CANTXERR)

This register reflects the status of the MSCAN transmit error counter.

Address Offset: $__0F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 3-16 MSCAN Transmit Error Counter (CANTXERR)

Read: Only when in Sleep Mode (SLPRQ=1 and SLPAK=1) or Initialization Mode (INITRQ=1 and INITAK=1)

Write: Unimplemented

NOTE: *Reading this register when in any other mode other than Sleep or Initialization Mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.*

NOTE: *Writing to this register when in special modes can alter the MSCAN functionality.*

3.3.1.17 MSCAN Identifier Acceptance Registers (CANIDAR0-7)

On reception, each message is written into the background receive buffer. The CPU is only signalled to read the message if it passes the criteria in the identifier acceptance and identifier mask registers (accepted); otherwise, the message is overwritten by the next message (dropped).

The acceptance registers of the MSCAN are applied on the IDR0 to IDR3 registers **3.3.2.1 Identifier Registers (IDR0-3)** of incoming messages in a bit by bit manner **4.3 Identifier Acceptance Filter**.

For extended identifiers, all four acceptance and mask registers are applied. For standard identifiers, only the first two (CANIDAR0/1, CANIDMR0/1) are applied.

	Bit 7	6	5	4	3	2	1	Bit 0
Address Offset:	$__10							CANIDAR0
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0
Address Offset:	$__11							CANIDAR1
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0
Address Offset:	$__12							CANIDAR2
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0
Address Offset:	$__13							CANIDAR3
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0

Figure 3-17 MSCAN Identifier Acceptance Registers (1st Bank)

	Bit 7	6	5	4	3	2	1	Bit 0
Address Offset:	$__18							CANIDAR4
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0

Figure 3-18 MSCAN Identifier Acceptance Registers (2nd Bank)

Address Offset: $__19 CANIDAR5

Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__1A CANIDAR6

Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__1B CANIDAR7

Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	0	0	0	0	0	0	0	0

Figure 3-18 MSCAN Identifier Acceptance Registers (2nd Bank)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

AC7 – AC0 — Acceptance Code Bits

AC7 – AC0 comprise a user defined sequence of bits with which the corresponding bits of the related identifier register (IDRn) of the receive message buffer are compared. The result of this comparison is then masked with the corresponding identifier mask register.

3.3.1.18 MSCAN Identifier Mask Registers (CANIDMR0-7)

The identifier mask register specifies which of the corresponding bits in the identifier acceptance register are relevant for acceptance filtering. To receive standard identifiers in 32 bit filter mode, it is required to program the last three bits (AM2 - AM0) in the mask registers CANIDMR1 and CANIDMR5 to “don’t care”. To receive standard identifiers in 16 bit filter mode, it is required to program the last three bits (AM2 - AM0) in the mask registers CANIDMR1, CANIDMR3, CANIDMR5 and CANIDMR7 to “don’t care”.

Address Offset: $__14 CANIDMR0

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__15 CANIDMR1

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__16 CANIDMR2

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__17 CANIDMR3

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Figure 3-19 MSCAN Identifier Mask Registers (1st Bank)

Address Offset: $__1C CANIDMR4

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__1D CANIDMR5

Figure 3-20 MSCAN Identifier Mask Registers (2nd Bank)

Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__1E CANIDMR6

Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Address Offset: $__1F CANIDMR7

Read: Write:	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset:	0	0	0	0	0	0	0	0

Figure 3-20 MSCAN Identifier Mask Registers (2nd Bank)

Read: Anytime

Write: Anytime in Initialization Mode (INITRQ=1 and INITAK=1)

AM7 – AM0 — Acceptance Mask Bits

If a particular bit in this register is cleared, this indicates that the corresponding bit in the identifier acceptance register must be the same as its identifier bit before a match is detected. The message is accepted if all such bits match. If a bit is set, it indicates that the state of the corresponding bit in the identifier acceptance register does not affect whether or not the message is accepted.

1 = Ignore corresponding acceptance code register bit.
0 = Match corresponding acceptance code register and identifier bits.

3.3.2 Programmer's Model of Message Storage

The following section details the organization of the receive and transmit message buffers and the associated control registers.

For reasons of programmer interface simplification, the receive and transmit message buffers have the same outline. Each message buffer allocates 16 bytes in the memory map containing a 13 byte data structure.

An additional Transmit Buffer Priority Register (TBPR) is defined for the transmit buffers. Within the last two bytes of this memory map the MSCAN stores a special 16-bit time stamp, which is sampled from an internal timer after successful transmission or reception of a message. This feature is only available for transmit and receiver buffers, if the TIME bit is set (**3.3.1.1 MSCAN Control Register 0 (CANCTL0)**).

The Time Stamp register is written by the MSCAN. The CPU can only read these registers.

Addr	Register Name
$__x0	Identifier Register 0
$__x1	Identifier Register 1
$__x2	Identifier Register 2
$__x3	Identifier Register 3
$__x4	Data Segment Register 0
$__x5	Data Segment Register 1
$__x6	Data Segment Register 2
$__x7	Data Segment Register 3
$__x8	Data Segment Register 4
$__x9	Data Segment Register 5
$__xA	Data Segment Register 6
$__xB	Data Segment Register 7
$__xC	Data Length Register
$__xD	Transmit Buffer Priority Register[1]
$__xE	Time Stamp Register (High Byte)[2]
$__xF	Time Stamp Register (Low Byte)[3]

Table 3-9 Message Buffer Organization

NOTES:
1. Not Applicable for Receive Buffers
2. Read-Only for CPU
3. Read-Only for CPU

Figure 3-21 shows the common 13 byte data structure of receive and transmit buffers for extended identifiers. The mapping of standard identifiers into the IDR registers is shown in **Figure 3-22**.

All bits of the receive and transmit buffers are 'x' out of reset because of RAM based implementation[1]. All reserved or unused bits of the receive and transmit buffers are always read 'x'.

NOTES:

[1] Exception: The Transmit Priority Registers are "0" out of reset

Register name		Bit 7	6	5	4	3	2	1	Bit 0	ADDR
IDR0	Read: Write:	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	$__x0
IDR1	Read: Write:	ID20	ID19	ID18	SRR (=1)	IDE (=1)	ID17	ID16	ID15	$__x1
IDR2	Read: Write:	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	$__x2
IDR3	Read: Write:	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR	$__x3
DSR0	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__x4
DSR1	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__x5
DSR2	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__x6
DSR3	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__x7
DSR4	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__x8
DSR5	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__x9
DSR6	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__xA
DSR7	Read: Write:	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	$__xB
DLR	Read: Write:	▨	▨	▨	▨	DLC3	DLC2	DLC1	DLC0	$__xC

▨ = Unused[1]

Figure 3-21 Receive / Transmit Message Buffer Extended Identifier

NOTES:

1. Unused bits are always read ‘x’

Read: Anytime for transmit buffers; only when RXF flag is set for receive buffers (see **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)**).

Write: Anytime for transmit buffers when TXEx flag is set (see **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**) and the corresponding transmit buffer is selected in CANTBSEL (see **3.3.1.11 MSCAN Transmit Buffer Selection Register (CANTBSEL)**); unimplemented for receive buffers

Reset: Undefined ($xx) because of RAM based implementation

Register name		Bit 7	6	5	4	3	2	1	Bit 0	ADDR
IDR0	Read: Write:	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	$__x0
IDR1	Read: Write:	ID2	ID1	ID0	RTR	IDE (=0)	▨	▨	▨	$__x1
IDR2	Read: Write:	▨	▨	▨	▨	▨	▨	▨	▨	$__x2
IDR3	Read: Write:	▨	▨	▨	▨	▨	▨	▨	▨	$__x3

▨ = Unused[1]

Figure 3-22 Standard Identifier Mapping

NOTES:

1. Unused bits are always read 'x'

3.3.2.1 Identifier Registers (IDR0-3)

The identifier registers for an extended format identifier consist of a total of 32 bits; ID28 - ID0, SRR, IDE, and RTR bits. The identifier registers for a standard format identifier consist of a total of 13 bits; ID10 - ID0, RTR, and IDE bits.

ID28 - ID0 — Extended format identifier

The identifiers consist of 29 bits (ID28 - ID0) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

ID10 - ID0 — Standard format identifier

The identifiers consist of 11 bits (ID10 – ID0) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

SRR — Substitute Remote Request

This fixed recessive bit is used only in extended format. It must be set to 1 by the user for transmission buffers and is stored as received on the CAN bus for receive buffers.

IDE — ID Extended

This flag indicates whether the extended or standard identifier format is applied in this buffer. In the case of a receive buffer, the flag is set as received and indicates to the CPU how to process the buffer identifier registers. In the case of a transmit buffer, the flag indicates to the MSCAN what type of identifier to send.

1 = Extended format (29 bit)
0 = Standard format (11 bit)

RTR — Remote Transmission Request

This flag reflects the status of the Remote Transmission Request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent.

1 = Remote frame
0 = Data frame

3.3.2.2 Data Segment Registers (DSR0-7)

The eight data segment registers, each with bits DB7-DB0, contain the data to be transmitted or received. The number of bytes to be transmitted or received is determined by the data length code in the corresponding DLR register.

DB7 - DB0 — Data bits 7-0

3.3.2.3 Data Length Register (DLR)

This register keeps the data length field of the CAN frame.

DLC3 - DLC0 — Data Length Code Bits

The data length code contains the number of bytes (data byte count) of the respective message. During the transmission of a remote frame, the data length code is transmitted as programmed while the number of transmitted data bytes is always 0. The data byte count ranges from 0 to 8 for a data frame. **Table 3-10** shows the effect of setting the DLC bits.

Table 3-10 Data Length Codes

Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8

3.3.2.4 Transmit Buffer Priority Register (TBPR)

This register defines the local priority of the associated message buffer. The local priority is used for the internal prioritization process of the MSCAN and is defined to be highest for the smallest binary number. The MSCAN implements the following internal prioritization mechanisms:

- All transmission buffers with a cleared TXEx flag participate in the prioritization immediately before the SOF (Start of Frame) is sent.
- The transmission buffer with the lowest local priority field wins the prioritization.

In cases of more than one buffer having the same lowest priority, the message buffer with the lower index number wins.

Address Offset: $xxxD

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PRIO7	PRIO6	PRIO5	PRIO4	PRIO3	PRIO2	PRIO1	PRIO0
Reset:	0	0	0	0	0	0	0	0

Figure 3-23 Transmit Buffer Priority Register (TBPR)

Read: Anytime

Write: Anytime

3.3.2.5 Time Stamp Register (TSRH, TSRL)

If the TIME bit is enabled, the MSCAN will write a special time stamp to the respective registers in the active transmit or receive buffer as soon as a message has been acknowledged on the CAN bus (**3.3.1.1 MSCAN Control Register 0 (CANCTL0)**). The time stamp is written on the bit sample point for the recessive bit of the ACK delimiter in the CAN frame. In case of a transmission, the CPU can only read the time stamp after the respective transmit buffer has been flagged empty.

The timer value, which is used for stamping, is taken from a free running internal CAN bit clock. A timer overrun is not indicated by the MSCAN. The timer is reset (all bits set to “0”) during Initialization Mode. The CPU can only read the Time Stamp registers.

Address Offset: $xxxE

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	X	X	X	X	X	X	X	X

Figure 3-24 Time Stamp Register (TSRH - High Byte)

Address Offset: $xxxF

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	X	X	X	X	X	X	X	X

Figure 3-25 Time Stamp Register (TSRL - Low Byte)

Read: Anytime

Write: Unimplemented

Section 4 Functional Description

4.1 General

This section provides a complete functional description of the MSCAN. It describes each of the features and modes listed in the introduction.

4.2 Message Storage

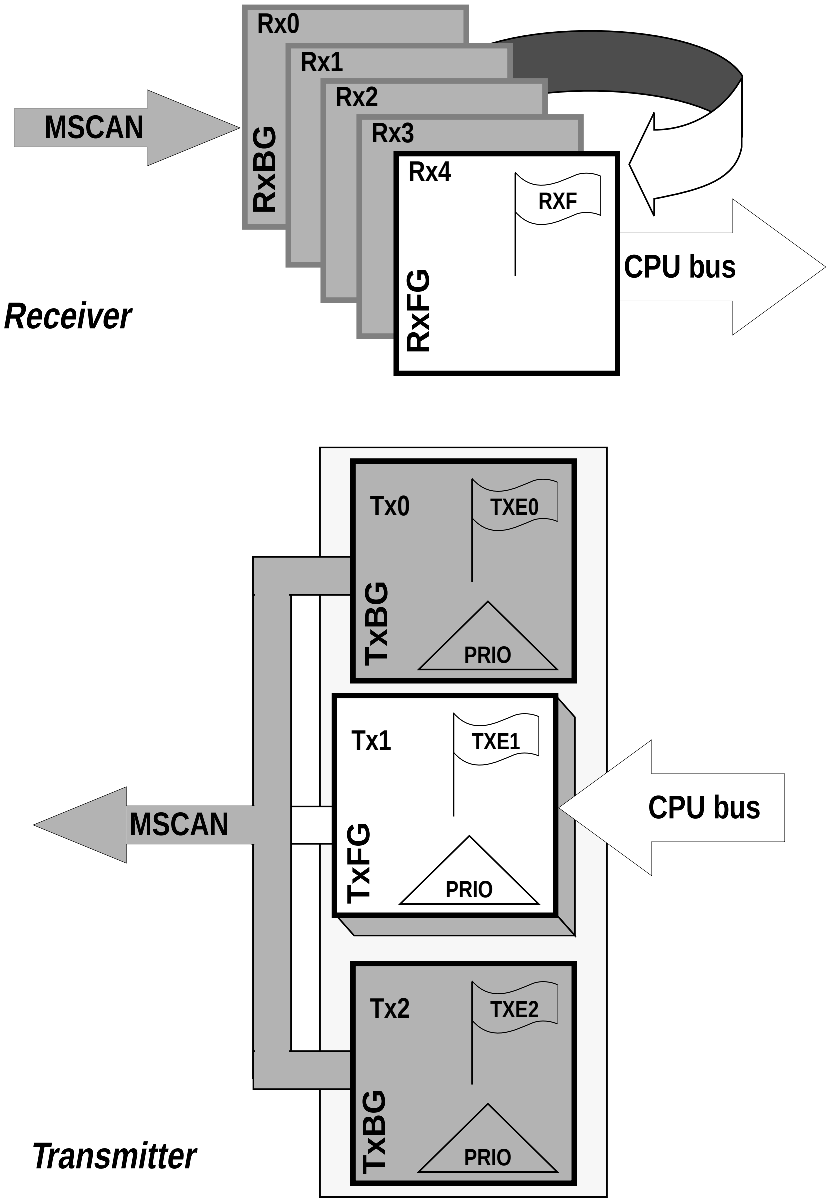

Figure 4-1 User Model for Message Buffer Organization

MSCAN facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

4.2.1 Message Transmit Background

Modern application layer software is built upon two fundamental assumptions:

- Any CAN node is able to send out a stream of scheduled messages without releasing the CAN bus between the two messages. Such nodes arbitrate for the CAN bus immediately after sending the previous message and only release the CAN bus in case of lost arbitration.
- The internal message queue within any CAN node is organized such that the highest priority message is sent out first, if more than one message is ready to be sent.

The above behavior cannot be achieved with a single transmit buffer. That buffer must be reloaded right after the previous message is sent. This loading process lasts a finite amount of time and has to be completed within the Inter-Frame Sequence (IFS)[**1.**] to be able to send an uninterrupted stream of messages. Even if this is feasible for limited CAN bus speeds, it requires that the CPU react with short latencies to the transmit interrupt.

A double buffer scheme de-couples the reloading of the transmit buffer from the actual message sending and, as such, reduces the reactiveness requirements on the CPU. Problems can arise if the sending of a message is finished while the CPU re-loads the second buffer. No buffer would then be ready for transmission and the CAN bus would be released.

At least three transmit buffers are required to meet the first of the above requirements under all circumstances. The MSCAN has three transmit buffers.

The second requirement calls for some sort of internal prioritization which the MSCAN implements with the “local priority” concept described in **4.2.2 Transmit Structures**.

4.2.2 Transmit Structures

The MSCAN has a triple transmit buffer scheme which allows multiple messages to be set up in advance and achieve an optimized real-time performance. The three buffers are arranged as shown in **Figure 4-1 User Model for Message Buffer Organization**.

All three buffers have a 13 byte data structure similar to the outline of the receive buffers **3.3.2 Programmer’s Model of Message Storage**. An additional **3.3.2.4 Transmit Buffer Priority Register (TBPR)** contains an 8-bit “Local Priority” field (PRIO) (see **3.3.2.4 Transmit Buffer Priority Register (TBPR)**). The remaining two bytes are used for time stamping of a message, if required (see **3.3.2.5 Time Stamp Register (TSRH, TSRL)**).

To transmit a message, the CPU has to identify an available transmit buffer which is indicated by a set Transmitter Buffer Empty (TXEx) flag **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**. If a transmit buffer is available, the CPU has to set a pointer to this buffer by writing to the CANTBSEL register (**3.3.1.11 MSCAN Transmit Buffer Selection Register (CANTBSEL)**). This makes the respective buffer accessible within the CANTXFG address space **3.3.2 Programmer’s Model of Message Storage**. The algorithmic feature associated with the CANTBSEL register simplifies the transmit buffer selection. In addition this scheme makes the handler software simpler as only one address area is applicable for the transmit process. In addition the required address space is minimized.

The CPU then stores the identifier, the control bits and the data content into one of the transmit buffers. Finally, the buffer is flagged as ready for transmission by clearing the associated TXE flag.

The MSCAN then schedules the message for transmission and signals the successful transmission of the buffer by setting the associated TXE flag. A transmit interrupt **4.8.2 Transmit Interrupt** is generated[1] when TXEx is set and can be used to drive the application software to re-load the buffer.

In case more than one buffer is scheduled for transmission when the CAN bus becomes available for arbitration, the MSCAN uses the “local priority” setting of the three buffers to determine the prioritization. For this purpose, every transmit buffer has an 8-bit local priority field (PRIO). The application software programs this field when the message is set up. The local priority reflects the priority of this particular message relative to the set of messages being transmitted from this node. The lowest binary value of the PRIO field is defined to be the highest priority. The internal scheduling process takes place whenever the MSCAN arbitrates for the CAN bus. This is also the case after the occurrence of a transmission error.

When a high priority message is scheduled by the application software, it may become necessary to abort a lower priority message in one of the three transmit buffers. As messages that are already in transmission cannot be aborted, the user has to request the abort by setting the corresponding Abort Request bit (ABTRQ) **3.3.1.9 MSCAN Transmitter Message Abort Request Register (CANTARQ)**. The MSCAN then grants the request, if possible, by: 1) setting the corresponding Abort Acknowledge flag (ABTAK) in the CANTAAK register, 2) setting the associated TXE flag to release the buffer, and 3) generating a transmit interrupt. The transmit interrupt handler software can tell from the setting of the ABTAK flag whether the message was aborted (ABTAK=1) or sent (ABTAK=0).

4.2.3 Receive Structures

The received messages are stored in a five stage input FIFO. The five message buffers are alternately mapped into a single memory area **Figure 4-1 User Model for Message Buffer Organization**. While the background receive buffer (RxBG) is exclusively associated with the MSCAN, the foreground receive buffer (RxFG) is addressable by the CPU **Figure 4-1 User Model for Message Buffer Organization**. This scheme simplifies the handler software as only one address area is applicable for the receive process.

All receive buffers have a size of 15 bytes to store the CAN control bits, the identifier (standard or extended), the data contents and a time stamp, if enabled (for details **3.3.2 Programmer’s Model of Message Storage**)[**1.**].

The Receiver Full flag (RXF) **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** signals the status of the foreground receive buffer. When the buffer contains a correctly received message with a matching identifier, this flag is set.

On reception, each message is checked to see if it passes the filter (**4.3 Identifier Acceptance Filter**) and in parallel, is written into the active RxBG. After successful reception of a valid message the MSCAN shifts the content of RxBG into the receiver FIFO[2], sets the RXF flag, and generates a receive interrupt

NOTES:

[1] The transmit interrupt occurs only if not masked. A polling scheme can be applied on TXEx also.

4.8.3 Receive Interrupt to the CPU[1]. The user's receive handler has to read the received message from the RxFG and then reset the RXF flag to acknowledge the interrupt and to release the foreground buffer. A new message, which can follow immediately after the IFS field of the CAN frame, is received into the next available RxBG. If the MSCAN receives an invalid message in its RxBG (wrong identifier, transmission errors etc.) the actual contents of the buffer will be over-written by the next message. The buffer will then not be shifted into the FIFO.

When the MSCAN module is transmitting, the MSCAN receives its own transmitted messages into the background receive buffer, RxBG, but does not shift it into the receiver FIFO, generate a receive interrupt, or acknowledge its own messages on the CAN bus. The exception to this rule is in loop back mode **3.3.1.2 MSCAN Control Register 1 (CANCTL1)** where the MSCAN treats its own messages exactly like all other incoming messages. The MSCAN receives its own transmitted messages in the event that it loses arbitration [**1.**]. If arbitration is lost, the MSCAN must be prepared to become a receiver.

An overrun condition occurs when all receive message buffers in the FIFO are filled with correctly received messages with accepted identifiers and another message is correctly received from the CAN bus with an accepted identifier. The latter message is discarded and an error interrupt with overrun indication is generated if enabled **4.8.5 Error Interrupt**. The MSCAN is still able to transmit messages while the receiver FIFO being filled, but all incoming messages are discarded. As soon as a receive buffer in the FIFO is available again, new valid messages will be accepted.

4.3 Identifier Acceptance Filter

The MSCAN Identifier Acceptance Registers (**3.3.1.12 MSCAN Identifier Acceptance Control Register (CANIDAC)**) define the acceptable patterns of the standard or extended identifier (ID10 - ID0 or ID28 - ID0). Any of these bits can be marked 'don't care' in the MSCAN Identifier Mask Registers **3.3.1.18 MSCAN Identifier Mask Registers (CANIDMR0-7)**.

A filter hit is indicated to the application software by a set Receive Buffer Full flag (RXF=1) and three bits in the CANIDAC register **3.3.1.12 MSCAN Identifier Acceptance Control Register (CANIDAC)**. These Identifier Hit flags (IDHIT2-0) clearly identify the filter section that caused the acceptance. They simplify the application software's task to identify the cause of the receiver interrupt. In case more than one hit occurs (two or more filters match), the lower hit has priority.

A very flexible programmable generic identifier acceptance filter has been introduced to reduce the CPU interrupt loading. The filter is programmable to operate in four different modes [**1.**]:

- Two identifier acceptance filters, each to be applied to a) the full 29 bits of the extended identifier and to the following bits of the CAN 2.0B frame: Remote Transmission Request (RTR), Identifier Extension (IDE), and Substitute Remote Request (SRR) or b)[2] the 11 bits of the standard identifier plus the RTR and IDE bits of the CAN 2.0A/B messages. This mode implements two filters for a

---

NOTES:

2. Only if the RXF flag is not set.
1. The receive interrupt occurs only if not masked. A polling scheme can be applied on RXF also.
2. Although this mode can be used for standard identifiers, it is recommended to use the four or eight identifier acceptance filters for standard identifiers

full length CAN 2.0B compliant extended identifier. **Figure 4-2** shows how the first 32-bit filter bank (CANIDAR0-3, CANIDMR0-3) produces a filter 0 hit. Similarly, the second filter bank (CANIDAR4-7, CANIDMR4-7) produces a filter 1 hit.

- Four identifier acceptance filters, each to be applied to a) the 14 most significant bits of the extended identifier plus the SRR and IDE bits of CAN 2.0B messages or b) the 11 bits of the standard identifier, the RTR and IDE bits of CAN 2.0A/B messages. **Figure 4-3** shows how the first 32-bit filter bank (CANIDAR0-3, CANIDMR0-3) produces filter 0 and 1 hits. Similarly, the second filter bank (CANIDAR4-7, CANIDMR4-7) produces filter 2 and 3 hits.
- Eight identifier acceptance filters, each to be applied to the first 8 bits of the identifier. This mode implements eight independent filters for the first 8 bits of a CAN 2.0A/B compliant standard identifier or a CAN 2.0B compliant extended identifier. **Figure 4-4** shows how the first 32-bit filter bank (CANIDAR0-3, CANIDMR0-3) produces filter 0 to 3 hits. Similarly, the second filter bank (CANIDAR4-7, CANIDMR4-7) produces filter 4 to 7 hits.
- Closed filter. No CAN message is copied into the foreground buffer RxFG, and the RXF flag is never set.

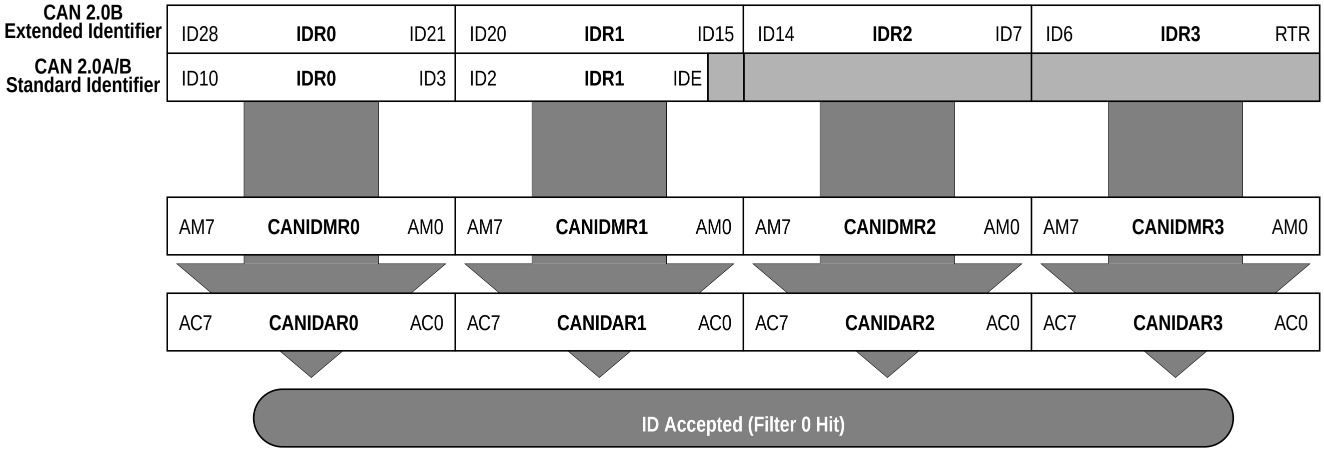

Figure 4-2 32-bit Maskable Identifier Acceptance Filter

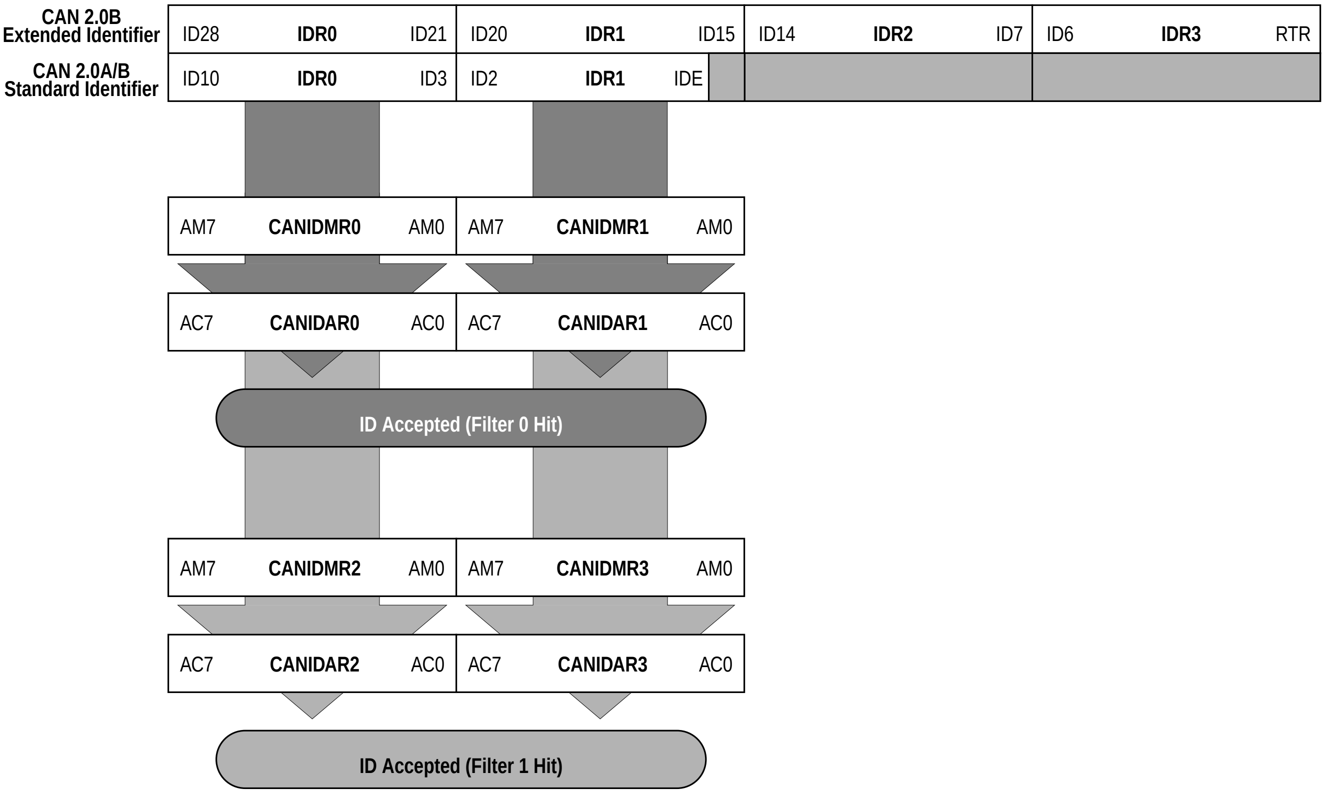

Figure 4-3 16-bit Maskable Identifier Acceptance Filters

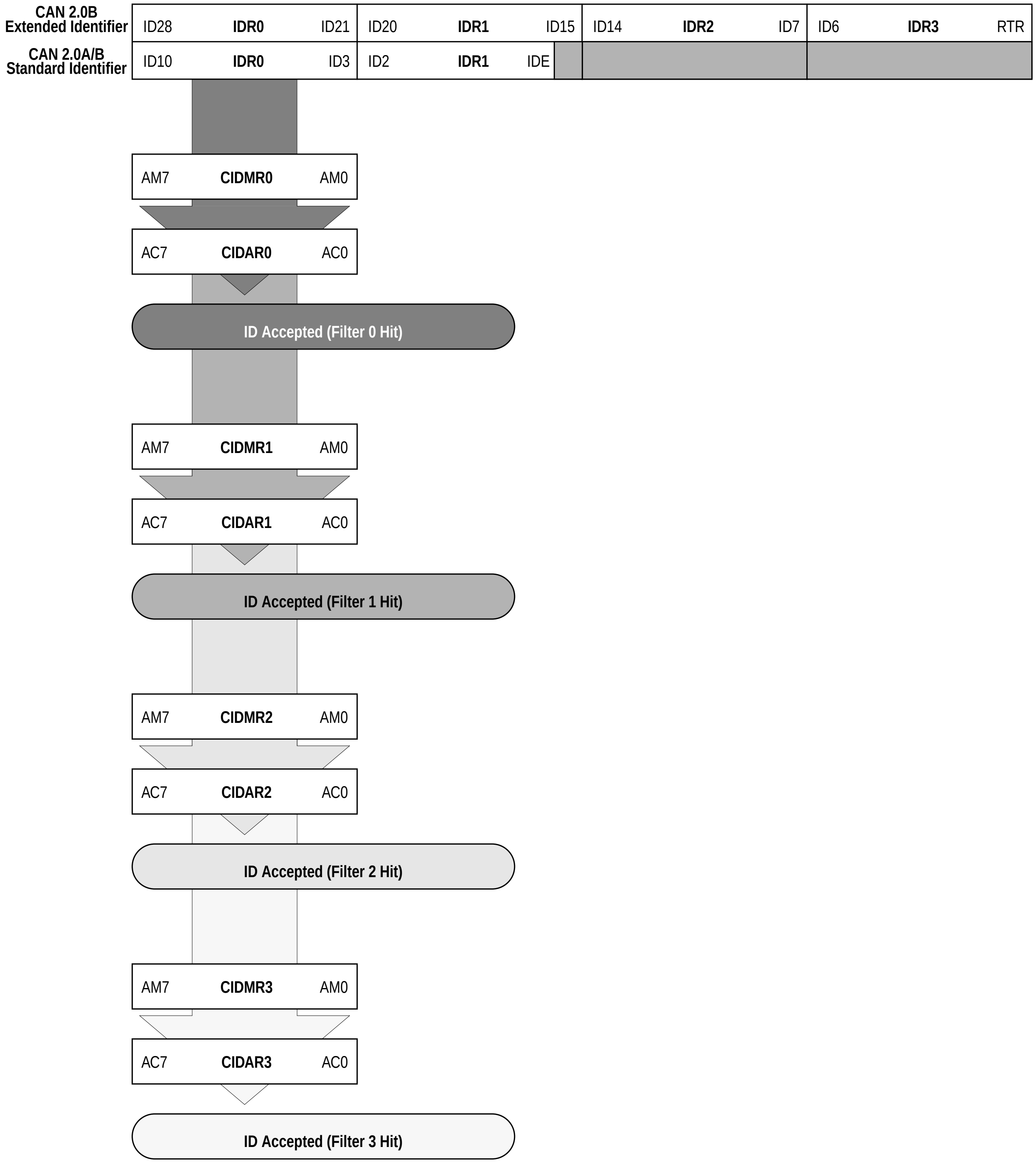

Figure 4-4 8-bit Maskable Identifier Acceptance Filters

4.3.1 Protocol Violation Protection

The MSCAN protects the user from accidentally violating the CAN protocol through programming errors. The protection logic implements the following features:

- The receive and transmit error counters cannot be written or otherwise manipulated.
- All registers which control the configuration of the MSCAN cannot be modified while the MSCAN is on-line. The MSCAN has to be in Initialization Mode. The corresponding INITRQ/INITAK handshake bits in the CANCTL0/CANCTL1 registers **3.3.1.1 MSCAN Control Register 0 (CANCTL0)** serve as a lock to protect the following registers:
  - MSCAN Control 1 Register (CANCTL1)
  - MSCAN Bus Timing Registers 0 and 1 (CANBTR0, CANBTR1)
  - MSCAN Identifier Acceptance Control Register (CANIDAC)
  - MSCAN Identifier Acceptance Registers (CANIDAR0-7)
  - MSCAN Identifier Mask Registers (CANIDMR0-7)
- The TXCAN pin is immediately forced to a recessive state when the MSCAN goes into the Power Down Mode or Initialization Mode (see **4.6.6 MSCAN Power Down Mode** and **4.6.5 MSCAN Initialization Mode**).
- The MSCAN enable bit (CANE) is only writable once in normal system operation modes as further protection against inadvertently disabling the MSCAN.

4.3.2 Clock System

Figure 4-5 shows the structure of the MSCAN clock generation circuitry.

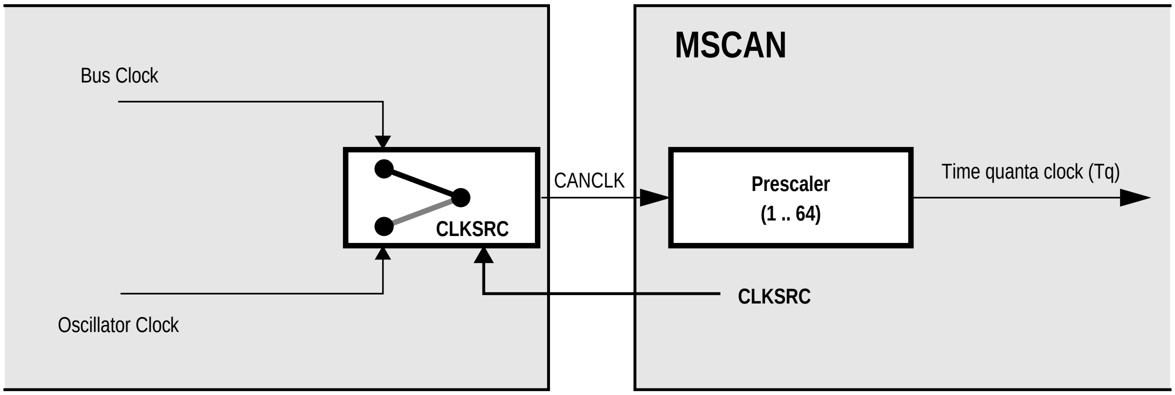

Figure 4-5 MSCAN Clocking Scheme

The clock source bit (CLKSRC) in the CANCTL1 register **3.3.1.2 MSCAN Control Register 1 (CANCTL1)** defines whether the internal CANCLK is connected to the output of a crystal oscillator (Oscillator Clock) or to the Bus Clock.

The clock source has to be chosen such that the tight oscillator tolerance requirements (up to 0.4%) of the CAN protocol are met. Additionally, for high CAN bus rates (1 Mbps), a 45%-55% duty cycle of the clock is required.

If the Bus Clock is generated from a PLL, it is recommended to select the Oscillator Clock rather than the Bus Clock due to jitter considerations, especially at the faster CAN bus rates.

For microcontrollers without a clock and reset generator (CRG), CANCLK is driven from the crystal oscillator (Oscillator Clock).

A programmable prescaler generates the time quanta (Tq) clock from CANCLK. A time quantum is the atomic unit of time handled by the MSCAN.

$$f_{Tq} = \frac{f_{CANCLK}}{(\text{Prescaler value})}$$

A bit time is subdivided into three segments [**1.**] (reference **Figure 4-6**):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section.
- Time Segment 1: This segment includes the PROP_SEG and the PHASE_SEG1 of the CAN standard. It can be programmed by setting the parameter TSEG1 to consist of 4 to 16 time quanta.
- Time Segment 2: This segment represents the PHASE_SEG2 of the CAN standard. It can be programmed by setting the TSEG2 parameter to be 2 to 8 time quanta long.

$$\text{Bit Rate} = \frac{f_{Tq}}{(\text{number of Time Quanta})}$$

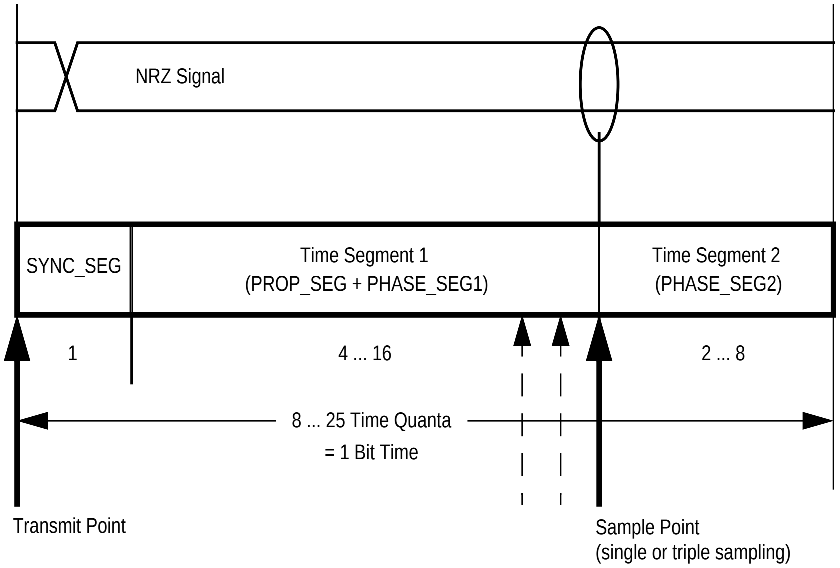

Figure 4-6 Segments within the Bit Time

Table 4-1 Time Segment Syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the CAN bus during this period.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node in receive mode samples the CAN bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The Synchronization Jump Width [**1.**] can be programmed in a range of 1 to 4 time quanta by setting the SJW parameter.

The above parameters are set by programming the MSCAN Bus Timing Registers (CANBTR0, CANBTR1) (see **3.3.1.3 MSCAN Bus Timing Register 0 (CANBTR0)** and **3.3.1.4 MSCAN Bus Timing Register 1 (CANBTR1)**).

Table 4-2 gives an overview of the CAN compliant segment settings and the related parameter values.

NOTE: *It is the user’s responsibility to ensure the bit time settings are in compliance with the CAN standard.*

Time Segment 1	TSEG1	Time Segment 2	TSEG2	Synchronization Jump Width	SJW
5 .. 10	4 .. 9	2	1	1 .. 2	0 .. 1
4 .. 11	3 .. 10	3	2	1 .. 3	0 .. 2
5 .. 12	4 .. 11	4	3	1 .. 4	0 .. 3
6 .. 13	5 .. 12	5	4	1 .. 4	0 .. 3
7 .. 14	6 .. 13	6	5	1 .. 4	0 .. 3
8 .. 15	7 .. 14	7	6	1 .. 4	0 .. 3
9 .. 16	8 .. 15	8	7	1 .. 4	0 .. 3

Table 4-2 CAN Standard Compliant Bit Time Segment Settings

4.4 Timer Link

The MSCAN generates an internal time stamp whenever a valid frame is received or transmitted and the TIME bit is enabled. Because the CAN specification defines a frame to be valid if no errors occur before the End of Frame (EOF) field is transmitted successfully, the actual value of an internal timer is written at EOF to the appropriate time stamp position within the transmit buffer. For receive frames the time stamp is written to the receive buffer.

4.5 Modes of Operation

4.5.1 Normal Modes

The MSCAN module behaves as described within this specification in all normal system operation modes.

4.5.2 Special Modes

The MSCAN module behaves as described within this specification in all special system operation modes.

4.5.3 Emulation Modes

In all emulation modes, the MSCAN module behaves just like normal system operation modes as described within this specification.

4.5.4 Listen-Only Mode

In an optional CAN bus monitoring mode (Listen-Only), the CAN node is able to receive valid data frames and valid remote frames, but it sends only “recessive” bits on the CAN bus. In addition it cannot start a transmision. If the MAC sub-layer is required to send a “dominant” bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MAC sub-layer monitors this “dominant” bit, although the CAN bus may remain in recessive state externally.

4.5.5 Security Modes

The MSCAN module has no security features.

4.6 Low Power Options

If the MSCAN is disabled (CANE=0), the MSCAN clocks are stopped for power savings.

If the MSCAN is enabled (CANE=1), the MSCAN has two additional modes with reduced power consumption, compared to Normal Mode: Sleep and Power Down Mode. In Sleep Mode power consumption is reduced by stopping all clocks except those to access the registers from the CPU side. In Power Down Mode, all clocks are stopped and no power is consumed.

Table 4-3 summarizes the combinations of MSCAN and CPU modes. A particular combination of modes is entered by the given settings on the CSWAI and SLPRQ/SLPAK bits.

For all modes, an MSCAN Wake-Up interrupt can only occur if the MSCAN is in Sleep Mode (SLPRQ=1 and SLPAK=1), wake-up functionality is enabled (WUPE=1) and the Wake-Up interrupt is enabled (WUPIE=1).

Table 4-3 CPU vs. MSCAN Operating Modes

CPU Mode	MSCAN Mode			
	Normal	Reduced Power Consumption		
		Sleep	Power Down	Disabled (CANE=0)

RUN	CSWAI = X[1]
	SLPRQ = 0
	SLPAK = 0
		CSWAI = X
		SLPRQ = 1
		SLPAK = 1
				CSWAI = X
				SLPRQ = X
				SLPAK = X

WAIT	CSWAI = 0
	SLPRQ = 0
	SLPAK = 0
		CSWAI = 0
		SLPRQ = 1
		SLPAK = 1
			CSWAI = 1
			SLPRQ = X
			SLPAK = X
				CSWAI = X
				SLPRQ = X
				SLPAK = X

STOP			CSWAI = X
			SLPRQ = X
			SLPAK = X
				CSWAI = X
				SLPRQ = X
				SLPAK = X

NOTES:

[1] ‘X’ means don’t care.

4.6.1 CPU Run Mode

As can be seen in **Table 4-3 CPU vs. MSCAN Operating Modes**, only MSCAN Sleep Mode is available as low power option, when CPU is in Run Mode.

4.6.2 CPU Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode. If the CSWAI bit is set, then additional power can be saved in Power Down Mode since the CPU clocks are stopped. After leaving this Power Down Mode the MSCAN restarts its internal controllers and enters Normal Mode again.

While the CPU is in Wait Mode, the MSCAN can be operated in Normal Mode and generate interrupts (registers can be accessed via background debug mode). The MSCAN can also operate in any of the low power modes depending on the values of the SLPRQ/SLPAK and CSWAI bits as seen in **Table 4-3 CPU vs. MSCAN Operating Modes**.

4.6.3 CPU Stop Mode

The STOP instruction puts the MCU in a low power consumption stand-by mode. In Stop Mode, the MSCAN set in Power Down mode regardless of the value of the SLPRQ/SLPAK and CSWAI bits **Table 4-3**.

4.6.4 MSCAN Sleep Mode

The CPU can request the MSCAN to enter this low power mode by asserting the SLPRQ bit in the CANCTL0 register. The time when the MSCAN enters Sleep Mode depends on a fixed synchronization delay and its current activity:

- If there are one or more message buffers scheduled for transmission (TXEx = 0), the MSCAN will continue to transmit until all transmit message buffers are empty (TXEx = 1, transmitted successfully or aborted) and then goes into Sleep Mode.
- If it is receiving, it continues to receive and goes into Sleep Mode as soon as the CAN bus next becomes idle.
- If it is neither transmitting nor receiving, it immediately goes into Sleep Mode.

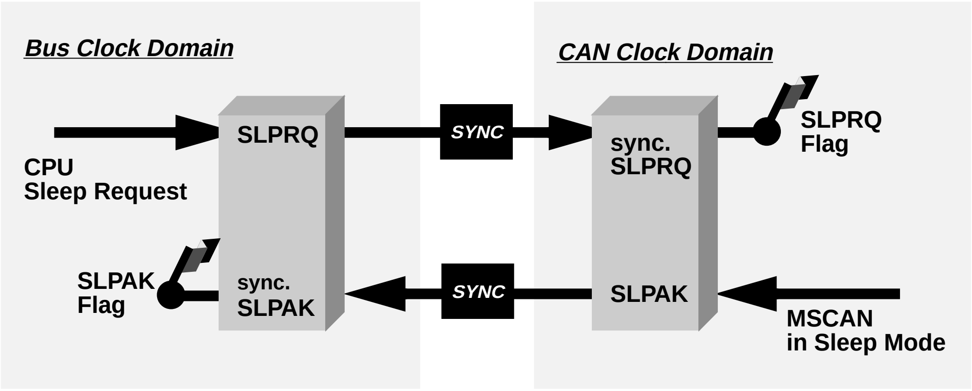

Figure 4-7 Sleep Request / Acknowledge Cycle

NOTE: *The application software must avoid setting up a transmission (by clearing one or more TXEx flag(s)) and immediately request Sleep Mode (by setting SLPRQ). It depends on the exact sequence of operations whether the MSCAN starts transmitting or goes into Sleep Mode directly.*

If Sleep Mode is active, the SLPRQ and SLPAK bits are set (**Figure 4-7**). The application software must use SLPAK as a handshake indication for the request (SLPRQ) to go into Sleep Mode.

When in Sleep Mode (SLPRQ=1 and SLPAK=1), the MSCAN stops its internal clocks. However, clocks to allow register accesses from the CPU side still run.

If the MSCAN is in Bus-Off state, it stops counting the 128*11 consecutive recessive bits due to the stopped clocks. The TXCAN pin remains in a recessive state. If RXF=1, the message can be read and RXF can be cleared. Shifting a new message into the foreground buffer of the receiver FIFO (RxFG) does not take place while in Sleep Mode.

It is possible to access the transmit buffers and to clear the associated TXE flags. No message abort takes place while in Sleep Mode.

If the WUPE bit in CANCLT0 is not asserted, the MSCAN will mask any activity it detects on CAN. The RXCAN pin is therefore held internally in a recessive state. This locks the MSCAN in Sleep Mode (**Figure 4-8 Simplified State Transitions for Entering/Leaving Sleep Mode**).

The MSCAN is only able to leave Sleep Mode (wake up) when

- CAN bus activity occurs and WUPE=1 or
- the CPU clears the SLPRQ bit

NOTE: *The CPU cannot clear the SLPRQ bit before Sleep Mode (SLPRQ=1 and SLPAK=1) is active.*

After wake-up, the MSCAN waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, if the MSCAN is woken-up by a CAN frame, this frame is not received.

The receive message buffers (RxFG and RxBG) contain messages if they were received before Sleep Mode was entered. All pending actions will be executed upon wake-up; copying of RxBG into RxFG, message aborts and message transmissions. If the MSCAN is still in Bus-Off state after Sleep Mode was left, it continues counting the 128*11 consecutive recessive bits.

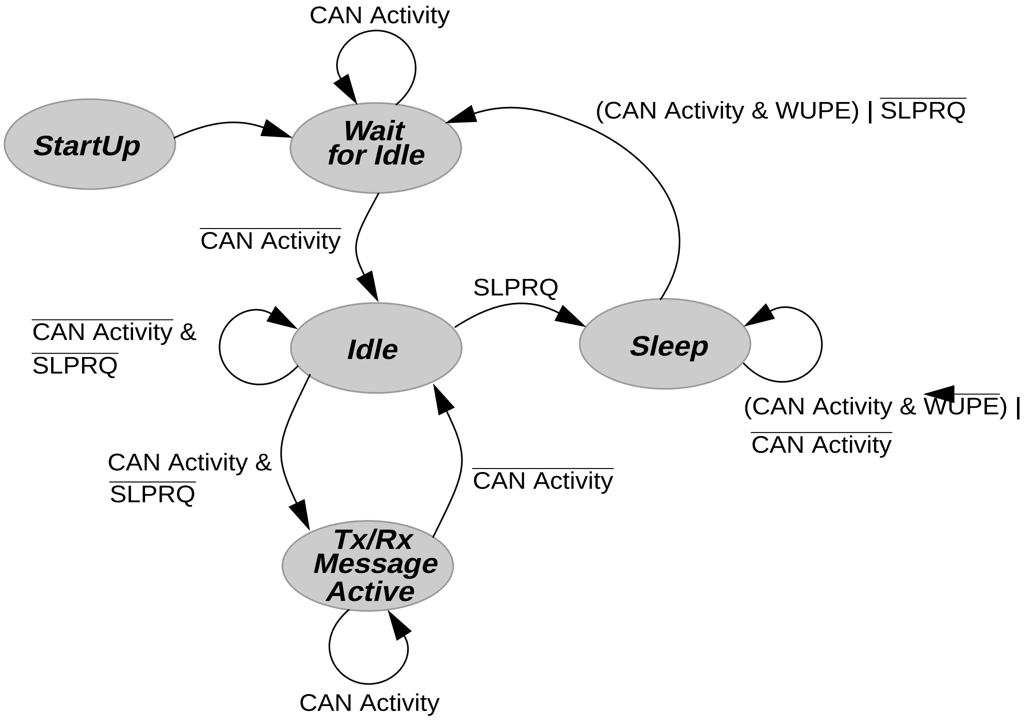
CAN Activity
StartUp
Wait for Idle
(CAN Activity & WUPE) | $\overline{\text{SLPRQ}}$
$\overline{\text{CAN Activity}}$
SLPRQ
$\overline{\text{CAN Activity}}$ &
$\overline{\text{SLPRQ}}$
Idle
Sleep
(CAN Activity & $\overline{\text{WUPE}}$) |
$\overline{\text{CAN Activity}}$
CAN Activity &
$\overline{\text{SLPRQ}}$
$\overline{\text{CAN Activity}}$
Tx/Rx Message Active
CAN Activity

Figure 4-8 Simplified State Transitions for Entering/Leaving Sleep Mode

4.6.5 MSCAN Initialization Mode

In Initialization Mode, any ongoing transmission or reception is immediately aborted and synchronization to the CAN bus is lost potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations, the MSCAN immediately drives the TXCAN pin into a recessive state.

NOTE: *The user is responsible for ensuring that the MSCAN is not active when Initialization Mode is entered. The recommended procedure is to bring the MSCAN into Sleep Mode (SLPRQ=1 and SLPAK=1) before setting the INITRQ bit in the CANCTL0 register. Otherwise the abort of an ongoing message can cause an error condition and can have an impact on the other CAN bus devices.*

In Initialization Mode, the MSCAN is stopped. However, interface registers can still be accessed. This mode is used to reset the CANCTL0, CANRFLG, CANRIER, CANTFLG, CANTIER, CANTARQ, CANTAAK, CANTBSEL registers to their default values. In addition it enables the configuration of the CANBTR0, CANBTR1 bit timing registers, CANIDAC and the CANIDAR, CANIDMR message filters. **3.3.1.1 MSCAN Control Register 0 (CANCTL0)** for a detailed description of the Initialization Mode.

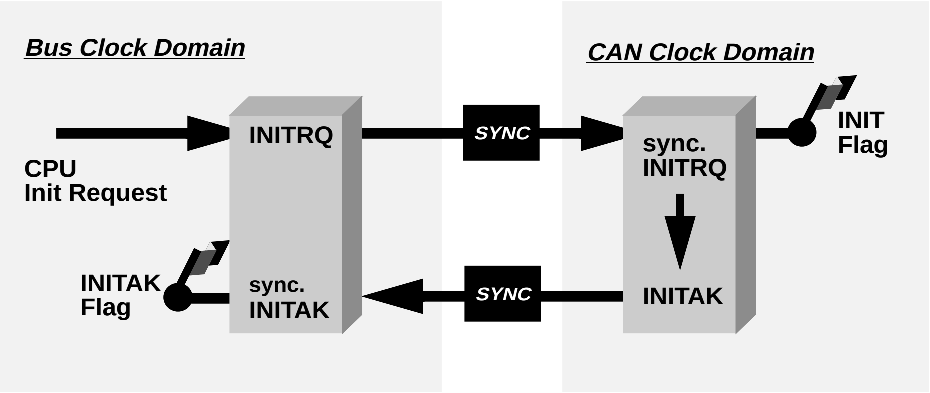

Figure 4-9 Initialization Request/Acknowledge Cycle

Due to independent clock domains within the MSCAN the INITRQ has to be synchronized to all domains by using a special handshake mechanism. This handshake causes additional synchronization delay (**Figure 4-9 Initialization Request/Acknowledge Cycle**).

If there is no message transfer ongoing on the CAN bus, the minimum delay will be two additional Bus Clocks and three additional CAN Clocks. When all parts of the MSCAN are in Initialization Mode the INITAK flag is set. The application software must use INITAK as a handshake indication for the request (INITRQ) to go into Initialization Mode.

NOTE: *The CPU cannot clear the INITRQ bit before Initialization Mode (INITRQ=1 and INITAK=1) is active.*

4.6.6 MSCAN Power Down Mode

The MSCAN is in Power Down Mode (**Table 4-3**) when

- the CPU is in Stop Mode or
- the CPU is in Wait Mode and the CSWAI bit is set.

When entering the Power Down Mode, the MSCAN immediately stops all ongoing transmissions and receptions, potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations to the above rule, the MSCAN immediately drives the TXCAN pin into a recessive state.

NOTE: *The user is responsible for ensuring that the MSCAN is not active when Power Down Mode is entered. The recommended procedure is to bring the MSCAN into Sleep Mode before the STOP or WAI instruction (if CSWAI is set) is executed. Otherwise the abort of an ongoing message can cause an error condition and can have an impact on the other CAN bus devices.*

In Power Down Mode, all clocks are stopped and no registers can be accessed. If the MSCAN was not in Sleep Mode before Power Down Mode became active, the module would perform an internal recovery cycle after powering up. This causes some fixed delay before the module enters Normal Mode again.

4.6.7 Programmable Wake-Up Function

The MSCAN can be programmed to wake up the MSCAN as soon as CAN bus activity is detected (see control bit WUPE in **3.3.1.1 MSCAN Control Register 0 (CANCTL0)**). The sensitivity to existing CAN bus action can be modified by applying a low-pass filter function to the RXCAN input line while in Sleep Mode (see control bit WUPM in **3.3.1.2 MSCAN Control Register 1 (CANCTL1)**).

This feature can be used to protect the MSCAN from wake-up due to short glitches on the CAN bus lines. Such glitches can result e.g. from electromagnetic interference within noisy environments.

4.7 Reset Initialization

The reset state of each individual bit is listed within the Register Description section **3.3 Register Descriptions** which details all the registers and their bit-fields.

4.8 Interrupts

This section describes all interrupts originated by the MSCAN. It documents the enable bits and generated flags. Each interrupt is listed and described separately.

4.8.1 Description of Interrupt Operation

The MSCAN supports four interrupt vectors(**Table 4-4 Interrupt Vectors**), any of which can be individually masked (for details see sections **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)** to **3.3.1.8 MSCAN Transmitter Interrupt Enable Register (CANTIER)**).

NOTE: *The dedicated interrupt vector addresses are defined at the MCU level. Please refer to the respective Device Guide.*

Table 4-4 Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Wake-Up Interrupt (WUPIF)	I bit	CANRIER (WUPIE)
Error Interrupts Interrupt (CSCIF, OVRIF)	I bit	CANRIER (CSCIE, OVRIE)
Receive Interrupt (RXF)	I bit	CANRIER (RXFIE)
Transmit Interrupts (TXE2 - TXE0)	I bit	CANTIER (TXEIE2 - TXEIE0)

4.8.2 Transmit Interrupt

At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. The TXEx flag of the empty message buffer is set.

4.8.3 Receive Interrupt

A message is successfully received and shifted into the foreground buffer (RxFG) of the receiver FIFO. This interrupt is generated immediately after receiving the EOF symbol. The RXF flag is set. If there are multiple messages in the receiver FIFO, the RXF flag is set as soon as the next message is shifted to the foreground buffer.

4.8.4 Wake-Up Interrupt

Activity on the CAN bus occurred during MSCAN internal Sleep Mode and WUPE **3.3.1.1 MSCAN Control Register 0 (CANCTL0)** enabled.

4.8.5 Error Interrupt

An overrun of the receiver FIFO, error, warning or Bus-Off condition occurred. The **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** indicates one of the following conditions:

- Overrun

  An overrun condition of the receiver FIFO as described in **4.2.3 Receive Structures** occurred.

- CAN Status Change

  The actual value of the Transmit and Receive Error Counters control the CAN bus state of the MSCAN.
  As soon as the error counters skip into a critical range (Tx/Rx-Warning, Tx/Rx-Error, Bus-Off) the MSCAN flags an error condition. The status change, which caused the error condition, is indicated by the TSTAT and RSTAT flags (see section **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** and **3.3.1.6 MSCAN Receiver Interrupt Enable Register (CANRIER)**).

4.8.6 Interrupt Acknowledge

Interrupts are directly associated with one or more status flags in either the **3.3.1.5 MSCAN Receiver Flag Register (CANRFLG)** or the **3.3.1.7 MSCAN Transmitter Flag Register (CANTFLG)**. Interrupts are pending as long as one of the corresponding flags is set. The flags in the above registers must be reset within the interrupt handler to handshake the interrupt. The flags are reset by writing a “1” to the corresponding bit position. A flag cannot be cleared if the respective condition still prevails.

NOTE: *It must be guaranteed that the CPU only clears the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.*

4.8.7 Recovery from STOP or WAIT

The MSCAN can recover from STOP or WAIT via the Wake-Up interrupt. This interrupt can only occur if the MSCAN was in Sleep Mode (SLPRQ=1 and SLPAK=1) before entering Power Down Mode, the wake-up option is enabled (WUPE=1) and the Wake-Up interrupt is enabled (WUPIE=1).

Section 5 Initialization/Application Information

5.1 MSCAN initialization

The procedure to initially start up the MSCAN module out of reset is as follows:

1. Assert CANE
2. Write to the configuration registers in Initialization Mode
3. Clear INITRQ to leave Initialization Mode and enter Normal Mode

If the configuration of registers which are writable in Initialization Mode only needs to be changed when the MSCAN module is in Normal Mode:

1. Make sure that the MSCAN transmission queue gets empty and bring the module into Sleep Mode by asserting SLPRQ and awaiting SLPAK
2. Enter Initialization Mode: Assert INITRQ and await INITAK
3. Write to the configuration registers in Initialization Mode
4. Clear INITRQ to leave Initialization Mode and continue in Normal Mode

5.2 Bus-Off recovery

The Bus-Off recovery is user configurable: The Bus-Off state can either be left automatically or on user request.

For reasons of backwards compatibility the MSCAN defaults to automatic recovery after reset. In this case the MSCAN will become Error Active again after counting 128 occurrences of 11 consecutive recessive bits on the CAN bus [**1.**].

If the MSCAN is configured for user request (BORM set in **3.3.1.2 MSCAN Control Register 1 (CANCTL1)**), the recovery from Bus-Off starts after both independent events have become true:

1. 128 occurrences of 11 consecutive recessive bits on the CAN bus have been monitored and
2. BOHOLD in **3.3.1.14 MSCAN Miscellaneous Register (CANMISC)** has been cleared by the user.

These two events may occur in any order.

Index

–A–

Abort Acknowledge 32
Abort Request 31, 38
ABTAK2 - ABTAK0 32
ABTRQ2 - ABTRQ0 31
AC7 – AC0 42
Acceptance Code Bits 42
Acceptance Mask Bits 44
address offset 14
AM7 – AM0 44

–B–

base address 14
Baud Rate Prescaler 22
Block Diagram 11
BOHOLD 38
BORM 21
BOSCH specification 11
BRP 22
bus monitoring mode 61
Bus-Off 19, 26, 27, 28, 38, 64, 68
Bus-Off Recovery 21
Bus-Off recovery 70

–C–

CAN protocol 11
CAN Status Change 68
CAN Status Change Interrupt Enable 27
CAN Status Change Interrupt Flag 25
CAN Stops in Wait Mode 17
CAN system 13
CANBTR0 19, 21, 22
CANBTR1 21, 23
CANCTL0 17, 19
CANCTL1 19, 20, 21
CANE 20
CANIDAC 21, 35
CANIDAR0-7 21, 41
CANIDMR0-7 21
CANRFLG 19, 25
CANRIER 19, 27
CANRXERR 39
CANTAAK 19, 32
CANTARQ 19, 31, 38
CANTBSEL 19, 33
CANTFLG 19, 29
CANTIER 19, 30
CANTXERR 40
CLKSRC 20
clock source 20
CRC 26
crystal 59
CSCIE 27
CSCIF 25
CSWAI 17

–D–

Data Length Code bits 48
data segment register 48
DB7 - DB0 48
DLC3 - DLC0 48
DLR 48

–E–

emulation modes 61
End of Frame 61
EOF 61
error counter 19, 21, 26, 58
Extended format identifier 47
extended identifiers 45

–F–

FIFO 12, 26, 36, 53, 54, 64, 68

–H–

handshake 20
hard reset 19

–I–

ID Extended 47
ID10 - ID0 47
ID28 - ID0 47
IDAM1 - IDAM0 35
IDE 47
Identifier Acceptance Mode 35
identifier register 47
IDHIT2 - IDHIT0 35
IDR0-3 47
IFS 52
INITAK 17, 19, 21
Initialization Mode 19, 21, 65
Initialization Mode Acknowledge 21
Initialization Mode Request 19
INITRQ 17, 19
interrupt enable 27
Interrupt Operation 67

–L–

LISTEN 20
Listen Only Mode 20
Listen-Only 61
local priority 52
Loop Back Self Test Mode 20
LOOPB 20

–M–

Memory Map 14
MSCAN Clock Source 20

–N–

normal modes 61

–O–

Overrun 68
overrun 26
Overrun Interrupt Flag 26
OVRIF 26

–P–

PHASE_SEG1 59
PHASE_SEG2 59
PLL 59
Power Down Mode 66
prescaler 59
PROP_SEG 59

–R–

REC 25
Receive Buffer Full Flag 26
Receive Error Counter 25, 39
Received Frame Flag 17
Receiver Active Status 17
Receiver Input Pin 13
Receiver Status Bits 26
register map 15
Remote Transmission Request 48
Reserved Registers 37
reset 67
RSTAT1, RSTAT0 26
RSTATE1, RSTATE0 27
RTR 48
run mode 17, 62
RXACT 17
RXCAN 13
RXFRM 17

–S–

SAMP 23
sample point 23
Sampling 23
security 62
SJW1, SJW0 22
Sleep Mode 18, 21, 63, 64, 65, 66, 67, 68, 69
Sleep Mode Acknowledge 21
Sleep Mode Request 18
SLPRQ 17, 18
SOF 49
special modes 61
SRR 47
Standard format identifie 47
Start of Frame 49
STOP 18, 69
Stop Mode 63
Substitute Remote Request 47
SYNC_SEG 59
SYNCH 18
Synchronization Jump Width 22, 60
Synchronized Status 18

–T–

TBPR 45, 49
TEC 25
TIME 18, 49
Time Segment 1 24, 59
Time Segment 2 23, 59
Time Stamp register 45
Timer Enable 18
transceiver 13
Transmit Buffer Priority Register 45
Transmit Buffer Select 33
Transmit Error Counter 25, 40
Transmitter Output Pin 13
TSEG1 59
TSEG13 – TSEG10 24
TSEG2 59
TSEG22 – TSEG20 23
TSRH, TSRL 49
TSTAT1, TSTAT0 26
TXCAN 13

–W–

WAIT 18, 69
Wait Mode 63
wake-up 27
Wake-Up Enable 18
Wake-Up Function 67
Wake-up Interrupt Enable 27
Wake-up Interrupt Flag 25

warning condition 68
WUPE 17, 18
WUPIE 18
WUPIF 25
WUPM 21

Block Guide End Sheet

FINAL PAGE OF
76
PAGES