Features

- Chipset Configuration
  - One master mXT1386E device
  - Three mXT154E devices
- maXTouch™ Touchscreen
  - True 12-bit multiple touch reporting and real-time XY tracking for up to 16 concurrent touches per touchscreen
- Number of Channels
  - Electrode grid configurations of up to 33 X and 42 Y lines supported
  - Touchscreens up to 1386 channels (subject to other configurations)
- Signal Processing
  - Advanced digital filtering using both hardware engine and firmware
  - Self-calibration
  - Auto drift compensation
  - Grip and face suppression algorithms to remove unintentional touches
  - Reports one-touch and two-touch gestures
  - Down-scaling and clipping support to match LCD resolution
  - Ultra-fast start-up and calibration for best user experience
  - Supports axis flipping and axis switch-over for portrait and landscape modes
- Scan Speed
  - Maximum single touch >200 Hz, subject to configuration
  - Configurable to allow power/speed optimization
  - Programmable timeout for automatic transition from active to idle states
- Response Times
  - Initial latency <25 ms for first touch from idle, subject to configuration
- Sensors
  - Works with PET or glass sensors, including curved profiles
  - Works with all proprietary sensor patterns recommended by Atmel®
  - Works with a passive stylus
- Panel Thickness
  - Glass up to 2.5 mm, screen size dependent
  - Plastic up to 1.2 mm, screen size dependent
- Interfaces
  - I^{2}C-compatible slave mode, 400 kHz
  - USB 2.0-compliant composite device, full speed (12 Mbps)
- Power
  - Digital 3.3 V nominal
  - Analog 3.3 V nominal
- Master Package
  - 64-pin QFN 9 × 9 × 1 mm, 0.5 mm pin pitch
- Slave Packages
  - 49-ball UFBGA 5 × 5 × 0.6 mm, 0.65 ball pitch
  - 48-pin QFN 6 × 6 × 0.6 mm, 0.4 mm pin pitch

maXTouch 1386-channel Touchscreen Controller

mXT1386E Revision 2.9

9767BX–AT42–09/12

1. Overview of the mXT1386E

1.1 Introduction

The Atmel® mXT1386E, together with its three associated mXT154E slave devices, is part of the maXTouch™ family of touchscreen controllers. This chipset builds on the success of the maXTouch family to provide a greatly improved user experience:

- **Patented capacitive sensing method** – The mXT1386E uses a unique charge-transfer acquisition engine to implement the QMatrix® capacitive sensing method patented by Atmel. This allows the measurement of up to 1386 mutual capacitance nodes. Coupled with a state-of-the-art CPU, the entire touchscreen sensing solution can measure, classify and track finger touches with a high degree of accuracy.
- **Capacitive Touch Engine (CTE)** – The acquisition engine uses an optimal measurement approach to ensure almost complete immunity from parasitic capacitance on the receiver inputs (Y lines). The engine includes sufficient dynamic range to cope with touchscreen mutual capacitances spanning 0.63 pF to 5 pF. This allows great flexibility for use with the Atmel proprietary ITO pattern designs. One and two layer ITO sensors are possible using glass or PET substrates.
- **Processing power** – The master mXT1386E combines with its slave mXT154E devices to allow the signal acquisition, preprocessing, postprocessing and housekeeping to be partitioned in an efficient and flexible way. This gives ample scope for sensing algorithms, touch tracking or advanced shape-based filtering.
- **Noise filtering** – The mXT1386E makes use of the noise filtering algorithms found on the maXTouch solution and copes well with LCD noise and RF noise, but operational enhancements allow the mXT1386E to cope even better with severe noise.
- **User experience** – The mXT1386E makes use of the Atmel mutual capacitance method to provide unambiguous multitouch performance and a responsive user experience. Hysteresis algorithms ensure that where a light touch is applied this is reported as a continuous touch, even when close to the touch threshold level, to prevent jitter on the screen. Algorithms also ensure that an on-screen cursor is stationary after the touch is removed, or remains on the edge of the visible area after a drag gesture.
- **Interpreting user intention** – The *mXT1386E Object Protocol* provides enhanced signal processing capabilities. Stylus support allows stylus touches to be detected and distinguished from other touches, such as finger touches. The suppression of unintentional touches from the user's gripping fingers, a resting palm, or a touching cheek or ear also help ensure that the user's intentions are correctly interpreted.

1.2 Chipset Architecture

The master mXT1386E device controls three slave mXT154E devices, as shown in Figure 1-1.

Each of the three mXT154E slave devices controls 11 X lines and 14 Y lines, which makes a total of 33 X lines and 42 Y lines available for use.

The X lines are distributed across the three slave devices in three sequential blocks. The Y lines, however, are distributed across the three slave devices in an interleaved manner, such that Y0 is controlled by Slave A, Y1 is controlled by Slave B, Y2 is controlled by Slave C, and so on. See Section 5.4 on page 21 for more information.

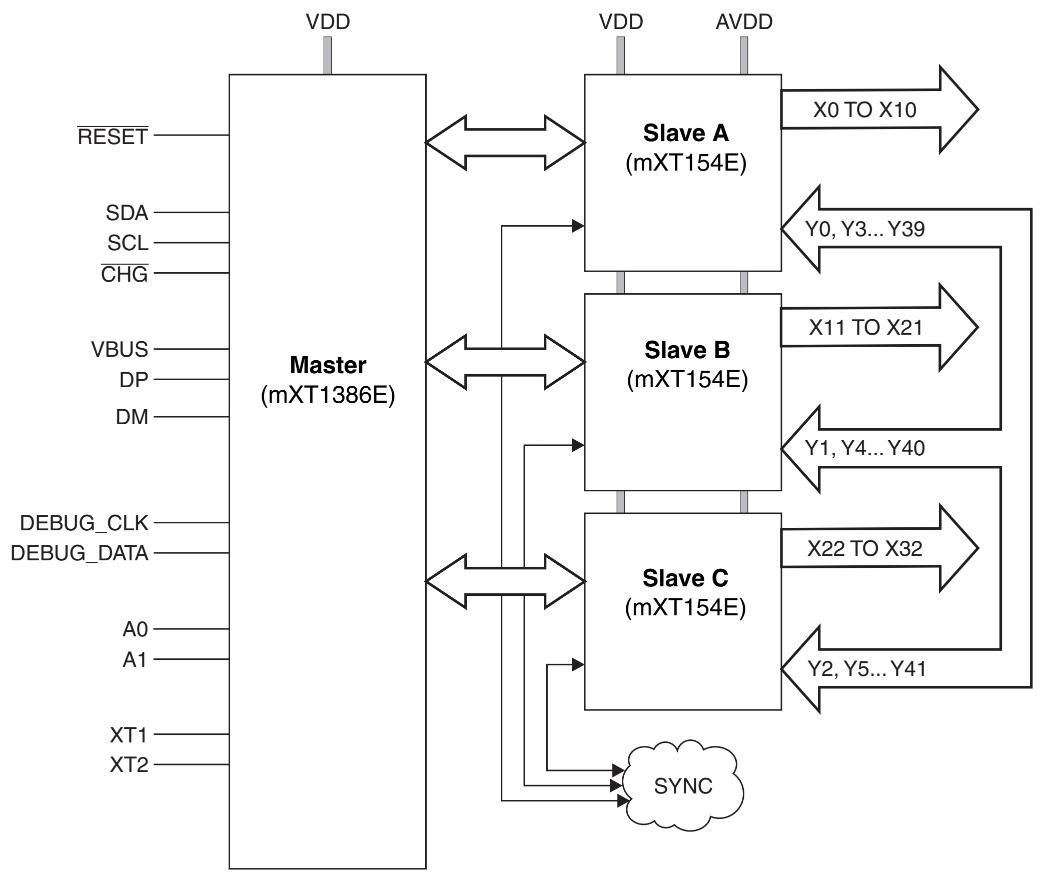
Table 1-1. Sense Lines

	Controls Sense Lines...	
Slave Device	**X**	**Y**
Slave A	X0 to X-10	Y0, Y3, Y6... Y39
Slave B	X11 to X-21	Y1, Y4, Y7... Y40
Slave C	X22 to X32	Y2, Y5 Y8... Y41

The host interfaces with the single master device only; it never needs to deal with the slave devices. It is the responsibility of the master chip to ensure that the configuration and use of the slaves is carried out in a uniform and consistent manner.

Communication with the host is achieved using either the I^{2}C-compatible interface or the USB interface. Either interface can be used, depending on the needs of the user's project, but only one interface should be used in any one design. See Table 2-1 for details of what to do with unused pins.

Figure 1-1. System Block Diagram



1.3 Understanding Unfamiliar Concepts

If some of the concepts mentioned in this datasheet are unfamiliar, see the following sections for more information:

- Appendix C on page 82 for a glossary of terms
- Appendix D on page 84 for QMatrix technology

1.4 Resources

The following datasheet provides essential information on configuring the chipset:

- *mXT1386E Protocol Guide*

The following documents may also be useful (available by contacting the Atmel Touch Technology division):

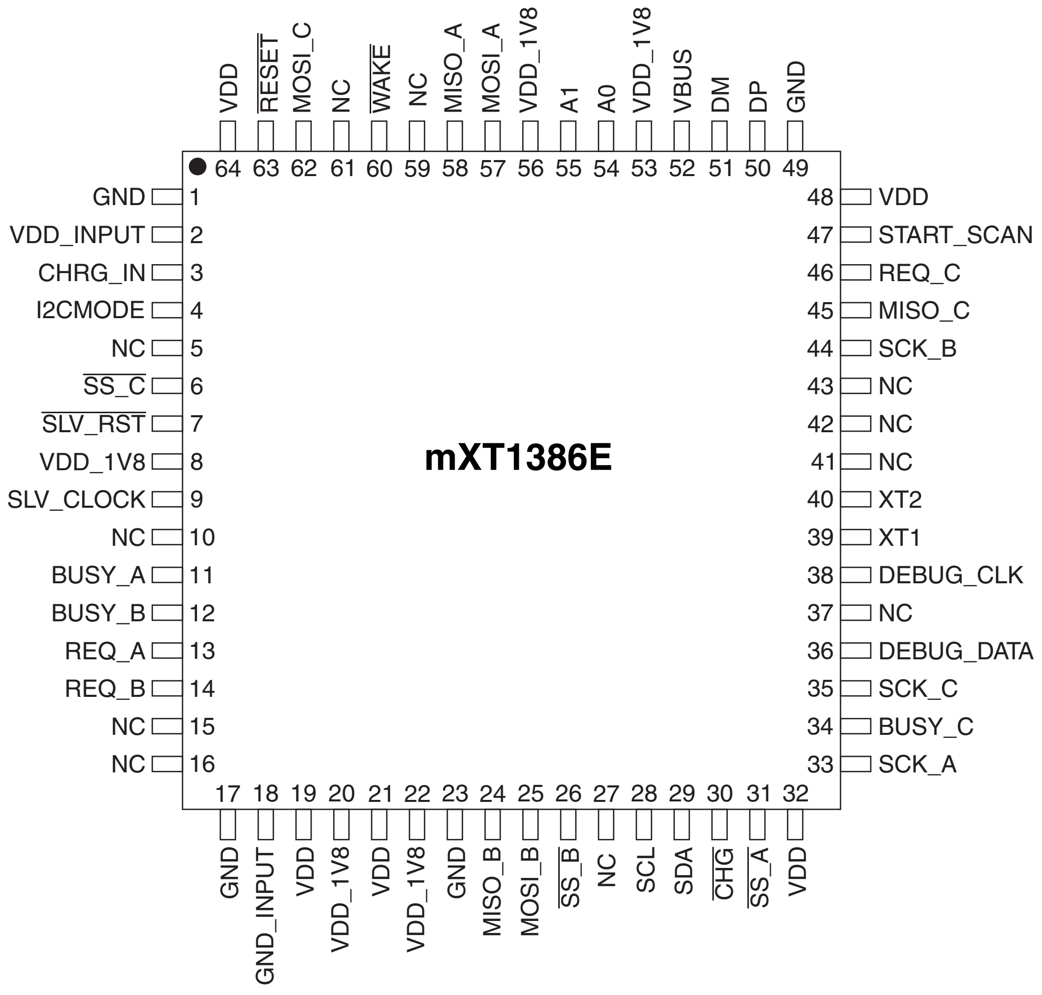
- **Configuring the chipset:**
  - Application Note: QTAN0058 – *Rejecting Unintentional Touches with the maXTouch Touchscreen Controllers*
  - Application Note: QTAN0078 – *maXTouch Stylus Tuning*

- **Miscellaneous:**
  - Application Note QTAN0050 – *Using the maXTouch Debug Port*
  - Application Note QTAN0061 – *maXTouch™ Sensitivity Effects for Mobile Devices*
  - Application Note QTAN0086 – *Touchscreen Design for Gloved Operation*
- **Touchscreen design and PCB/FPCB layout guidelines:**
  - Application Note QTAN0054 – *Getting Started with maXTouch Touchscreen Designs*
  - Application Note QTAN0048 – *maXTouch PCB/FPCB Layout Guidelines*
  - Application Note QTAN0080 – *Touchscreens Sensor Design Guide*
- **Bootloading:**
  - Application Note QTAN0051 – *Bootloading Procedure for Atmel® Touch Sensors Based on the Object Protocol*

2. Pinouts

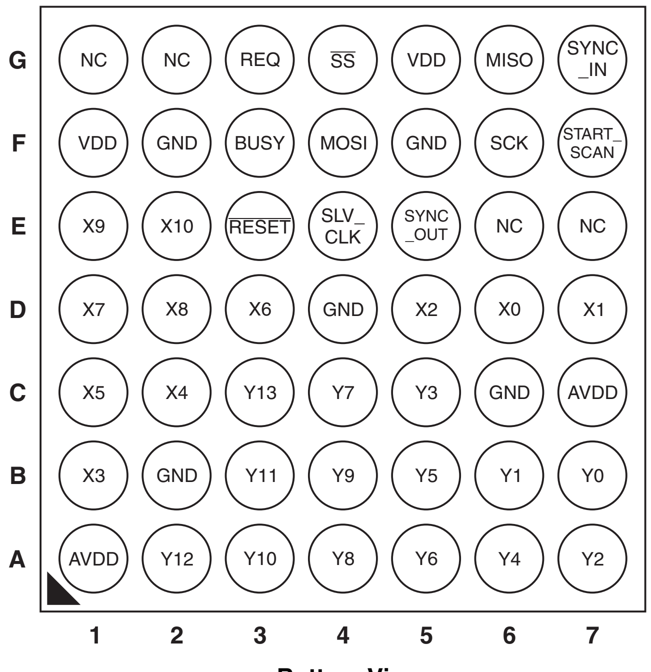
2.1 Pinout Configurations

2.1.1 Master mXT1386E – 64-pin QFN



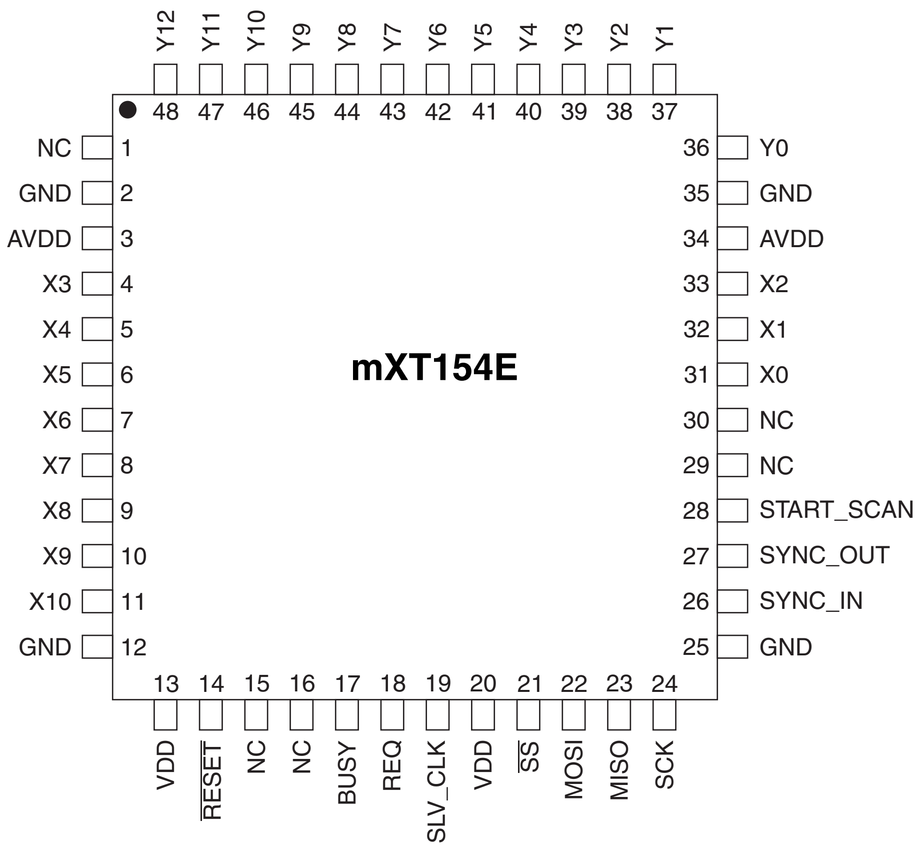
Top View

2.1.2 Slave mXT154E – 49-ball UFBGA



Bottom View

2.1.3 Slave mXT154E – 48-pin QFN



Top View

2.2 Pinout Descriptions

2.2.1 Master mXT1386E – 64-pin QFN

Table 2-1. Pin Listing

Pin	Name	Type	Comments	If Unused, Connect To...
1	GND	P	Ground	–
2	VDD_INPUT	I	Inter-chip signal; for factory use only	Vdd
3	CHRG_IN	I/O	Charger present input	Input: GND Output: leave open
4	I2CMODE (1)	I	I^{2}C-compatible protocol select – I^{2}C or HID-I^{2}C	Leave open (2)
5	NC	–	No connection	Leave open
6	$\overline{\text{SS\_C}}$	I	Inter-chip signal	–
7	$\overline{\text{SLV\_RST}}$	O	Inter-chip signal	–
8	VDD_1V8 (3)	P	Inter-chip signal	–
9	SLV_CLOCK	O	Inter-chip signal	–
10	NC	–	No connection	Leave open
11	BUSY_A	I	Inter-chip signal	–
12	BUSY_B	I	Inter-chip signal	–
13	REQ_A	O	Inter-chip signal	–
14	REQ_B	O	Inter-chip signal	–
15	NC	–	No connection	Leave open
16	NC	–	No connection	Leave open
17	GND	P	Ground	–
18	GND_INPUT	I	Inter-chip signal; for factory use only	–
19	VDD	P	Power	–
20	VDD_1V8 (3)	P	Inter-chip signal	–
21	VDD	P	Power	–
22	VDD_1V8 (3)	P	Inter-chip signal	–
23	GND	P	Ground	–
24	MISO_B	O	Inter-chip signal	–
25	MOSI_B	I	Inter-chip signal	–
26	$\overline{\text{SS\_B}}$	I	Inter-chip signal	–
27	NC	–	No connection	Leave open
28	SCL (1)	OD	Serial Interface Clock	Leave open

Table 2-1. Pin Listing (Continued)

Pin	Name	Type	Comments	If Unused, Connect To...
29	SDA (1)	OD	Serial Interface Data	Leave open
30	$\overline{\text{CHG}}$ (4)	OD	State change interrupt	Leave open
31	$\overline{\text{SS\_A}}$	I	Inter-chip signal	–
32	VDD	P	Power	–
33	SCK_A	I	Inter-chip signal	–
34	BUSY_C	I	Inter-chip signal	
35	SCK_C	I	Inter-chip signal	–
36	DEBUG_DATA	I/O	Debug port data (5)	Leave open
37	NC	–	No connection	Leave open
38	DEBUG_CLK	I/O	Debug port clock (5)	Leave open
39	XT1	I	External oscillator – 16 MHz	–
40	XT2	O	External oscillator – 16 MHz	–
41	NC	–	No connection	Leave open
42	NC	–	No connection	Leave open
43	NC	–	No connection	Leave open
44	SCK_B	I	Inter-chip signal	–
45	MISO_C	O	Inter-chip signal	–
46	REQ_C	O	Inter-chip signal	–
47	START_SCAN	O	Inter-chip signal	–
48	VDD	P	Power	–
49	GND	P	Ground	–
50	DP (1)	USB	USB device port data +	GND
51	DM (1)	USB	USB device port data -	GND
52	VBUS (1)	USB	USB VBUS monitor	GND
53	VDD_1V8 (3)	P	Inter-chip signal	–
54	A0	I	I^2C-compatible address select	Leave open
55	A1	I	I^2C-compatible address select	Leave open
56	VDD_1V8 (3)	P	Inter-chip signal	–
57	MOSI_A	I	Inter-chip signal	–
58	MISO_A	O	Inter-chip signal	–
59	NC	–	No connection	Leave open
60	$\overline{\text{WAKE}}$	I	External wake-up. Typically connected to SCL pin; see Section 6.8 on page 33 for more details	Vdd if USB used
61	NC	I/O	No connection	Leave open

Table 2-1. Pin Listing (Continued)

Pin	Name	Type	Comments	If Unused, Connect To...
62	MOSI_C	I	Inter-chip signal	–
63	$\overline{\text{RESET}}$	I	Reset low	Vdd [6]
64	VDD	P	Power	–

1. Only one interface (I^2C, USB, or HID-I^2C) can be used in any one design.
2. Leave open for standard Atmel object protocol, or connect to GND for Microsoft Windows 8 HID-I^2C protocol.
3. The mXT1386E has an internal 1.8V regulator. The host system only needs to supply the VDD rail.
4. $\overline{\text{CHG}}$ is momentarily set (approximately 100 ms) as an input after power-up or reset for diagnostic purposes.
5. See Section 5.8 on page 25 for additional information.
6. It is recommend that $\overline{\text{RESET}}$ is connected to the host system.

I	Input only	OD	Open drain output	O	Output only, push-pull
P	Ground or power	USB	USB communications		

2.2.2 Slave mXT154E – 49-ball UFBGA

Table 2-2. Pin Listing

Ball	Name	Type	Comments	If Unused, Connect To...
A1	AVDD	P	Analog power	–
A2	Y12	I	Y line connection	Leave open
A3	Y10	I	Y line connection	Leave open
A4	Y8	I	Y line connection	Leave open
A5	Y6	I	Y line connection	Leave open
A6	Y4	I	Y line connection	Leave open
A7	Y2	I	Y line connection	Leave open
B1	X3	O	X matrix drive line	Leave open
B2	GND	P	Ground	–
B3	Y11	I	Y line connection	Leave open
B4	Y9	I	Y line connection	Leave open
B5	Y5	I	Y line connection	Leave open
B6	Y1	I	Y line connection	Leave open
B7	Y0	I	Y line connection	Leave open
C1	X5	O	X matrix drive line	Leave open
C2	X4	O	X matrix drive line	Leave open
C3	Y13	I	Y line connection	Leave open
C4	Y7	I	Y line connection	Leave open
C5	Y3	I	Y line connection	Leave open
C6	GND	P	Ground	–
C7	AVDD	P	Analog power	–
D1	X7	O	X matrix drive line	Leave open

Table 2-2. Pin Listing (Continued)

Ball	Name	Type	Comments	If Unused, Connect To...
D2	X8	O	X matrix drive line	Leave open
D3	X6	O	X matrix drive line	Leave open
D4	GND	P	Ground	–
D5	X2	O	X matrix drive line	Leave open
D6	X0	O	X matrix drive line	Leave open
D7	X1	O	X matrix drive line	Leave open
E1	X9	O	X matrix drive line	Leave open
E2	X10	O	X matrix drive line	Leave open
E3	$\overline{\text{RESET}}$	I	Inter-chip signal	–
E4	SLV_CLK	I	Inter-chip signal	–
E5	SYNC_OUT	O	Inter-chip signal	–
E6	NC	–	No connection	–
E7	NC	–	No connection	Leave open
F1	VDD	P	Digital power	–
F2	GND	P	Ground	–
F3	BUSY	O	Inter-chip signal	–
F4	MOSI	O	Inter-chip signal	–
F5	GND	P	Ground	–
F6	SCK	O	Inter-chip signal	–
F7	START_SCAN	I	Inter-chip signal	–
G1	NC	–	No connection	Leave open
G2	NC	–	No connection	Leave open
G3	REQ	I	Inter-chip signal	–
G4	$\overline{\text{SS}}$	O	Inter-chip signal	–
G5	VDD	P	Digital power	–
G6	MISO	I	Inter-chip signal	–
G7	SYNC_IN	I	Inter-chip signal	–

I	Input only	O	Output only, push-pull	P	Ground or power

2.2.3 Slave mXT154E – 48-pin QFN

Table 2-3. Pin Listing

Pin	Name	Type	Comments	If Unused, Connect To...
1	Y13	I	Y line connection	Leave open
2	GND	P	Ground	–
3	AVDD	P	Analog power	–
4	X3	O	X matrix drive line	Leave open
5	X4	O	X matrix drive line	Leave open
6	X5	O	X matrix drive line	Leave open
7	X6	O	X matrix drive line	Leave open
8	X7	O	X matrix drive line	Leave open
9	X8	O	X matrix drive line	Leave open
10	X9	O	X matrix drive line	Leave open
11	X10	O	X matrix drive line	Leave open
12	GND	P	Ground	–
13	VDD	P	Digital power	–
14	$\overline{\text{RESET}}$	I	Inter-chip signal	–
15	NC	–	No connection	Leave open
16	NC	–	No connection	Leave open
17	BUSY	O	Inter-chip signal	–
18	REQ	I	Inter-chip signal	–
19	SLV_CLK	I	Inter-chip signal	–
20	VDD	P	Digital power	–
21	$\overline{\text{SS}}$	O	Inter-chip signal	–
22	MOSI	O	Inter-chip signal	–
23	MISO	I	Inter-chip signal	–
24	SCK	O	Inter-chip signal	–
25	GND	P	Ground	–
26	SYNC_IN	I	Inter-chip signal	–
27	SYNC_OUT	O	Inter-chip signal	–
28	START_SCAN	I	Inter-chip signal	–
29	NC	–	No connection	–
30	NC	–	No connection	Leave open
31	X0	O	X matrix drive line	Leave open
32	X1	O	X matrix drive line	Leave open
33	X2	O	X matrix drive line	Leave open
34	AVDD	P	Analog power	–

Table 2-3. Pin Listing (Continued)

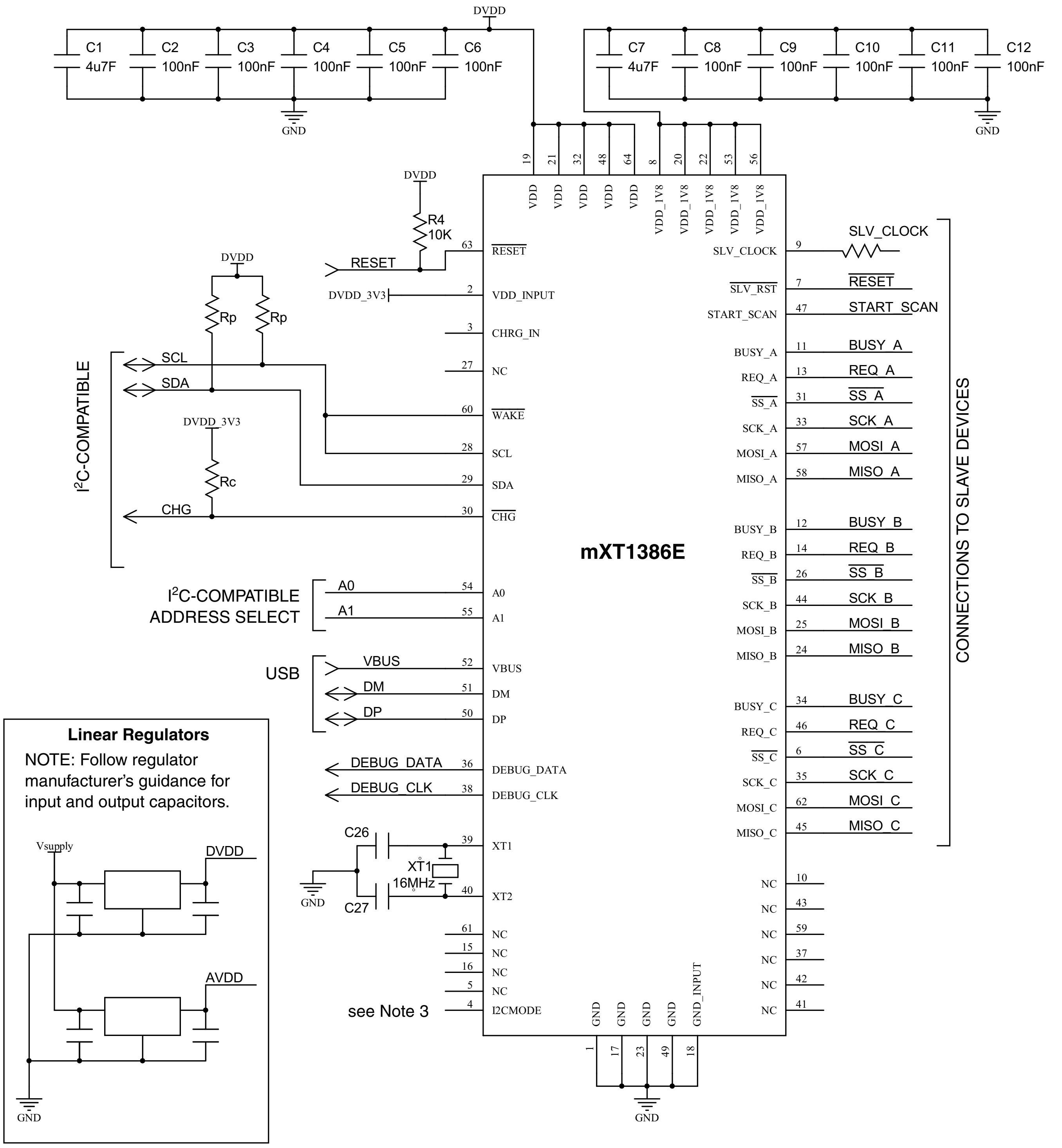
Pin	Name	Type	Comments	If Unused, Connect To...
35	GND	P	Ground	–
36	Y0	I	Y line connection	Leave open
37	Y1	I	Y line connection	Leave open
38	Y2	I	Y line connection	Leave open
39	Y3	I	Y line connection	Leave open
40	Y4	I	Y line connection	Leave open
41	Y5	I	Y line connection	Leave open
42	Y6	I	Y line connection	Leave open
43	Y7	I	Y line connection	Leave open
44	Y8	I	Y line connection	Leave open
45	Y9	I	Y line connection	Leave open
46	Y10	I	Y line connection	Leave open
47	Y11	I	Y line connection	Leave open
48	Y12	I	Y line connection	Leave open

I	Input only	O	Output only, push-pull	P	Ground or power

3. Schematics

3.1 Master Device (mXT1386E) – 64-pin QFN

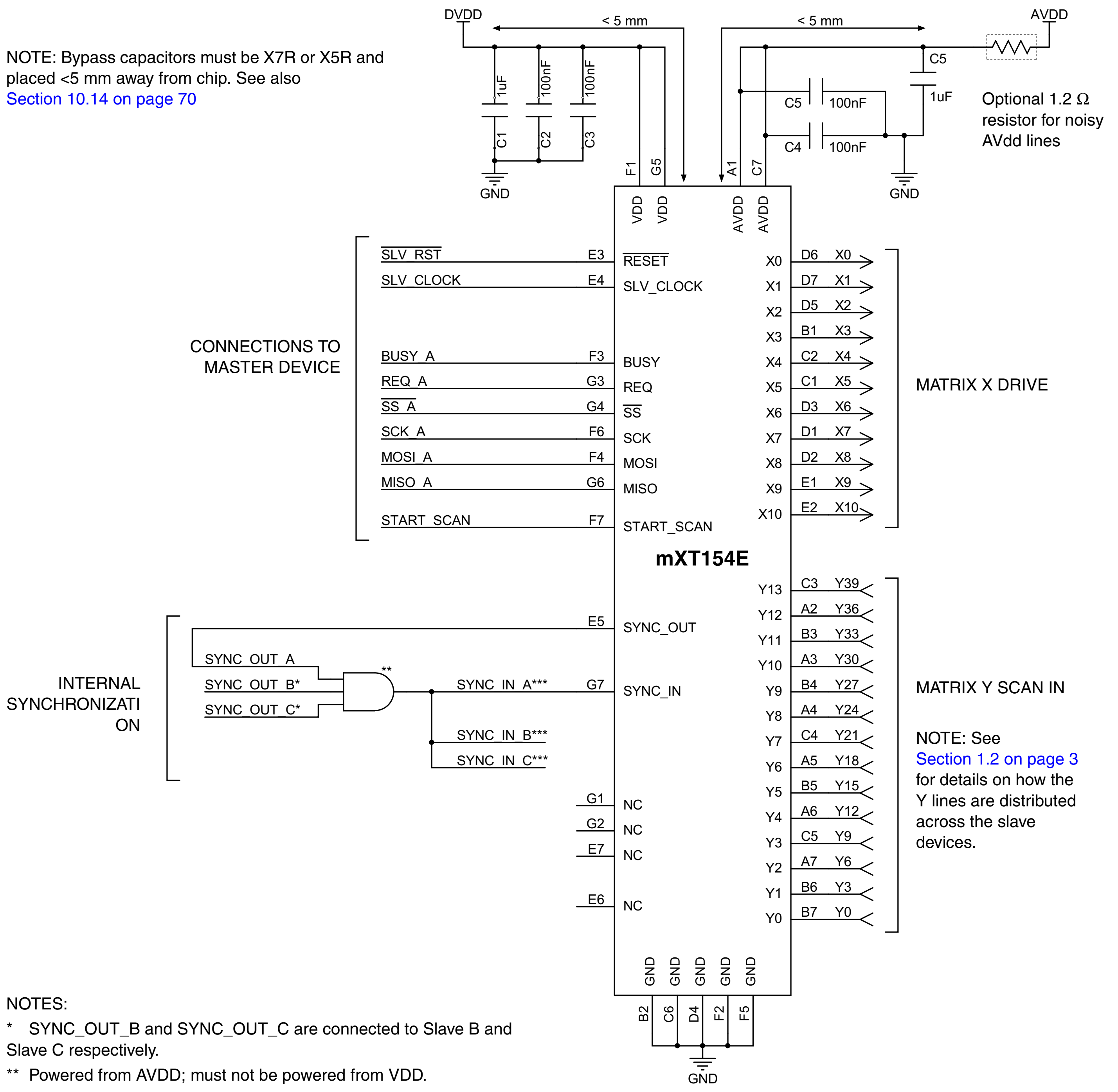
Notes:
1. Capacitors C2 – C6 and C8 – C12 must be X7R or X5R and track lengths must be <5 mm. See also Section 10.14 on page 70
2. Either I^{2}C-compatible or USB interface can be used, but only one interface should be used in any one design.
3. I2CMODE should be connected depending on protocol required: leave open for standard Atmel object protocol, or connect to GND for Microsoft Windows 8 HID-I^{2}C protocol.



3.2 Slave Devices

3.2.1 Slave Devices (3 × mXT154E) – 49-ball UFBGA

Note: Instance Slave A only is shown; Slave B and Slave C are omitted for simplicity.



NOTES:

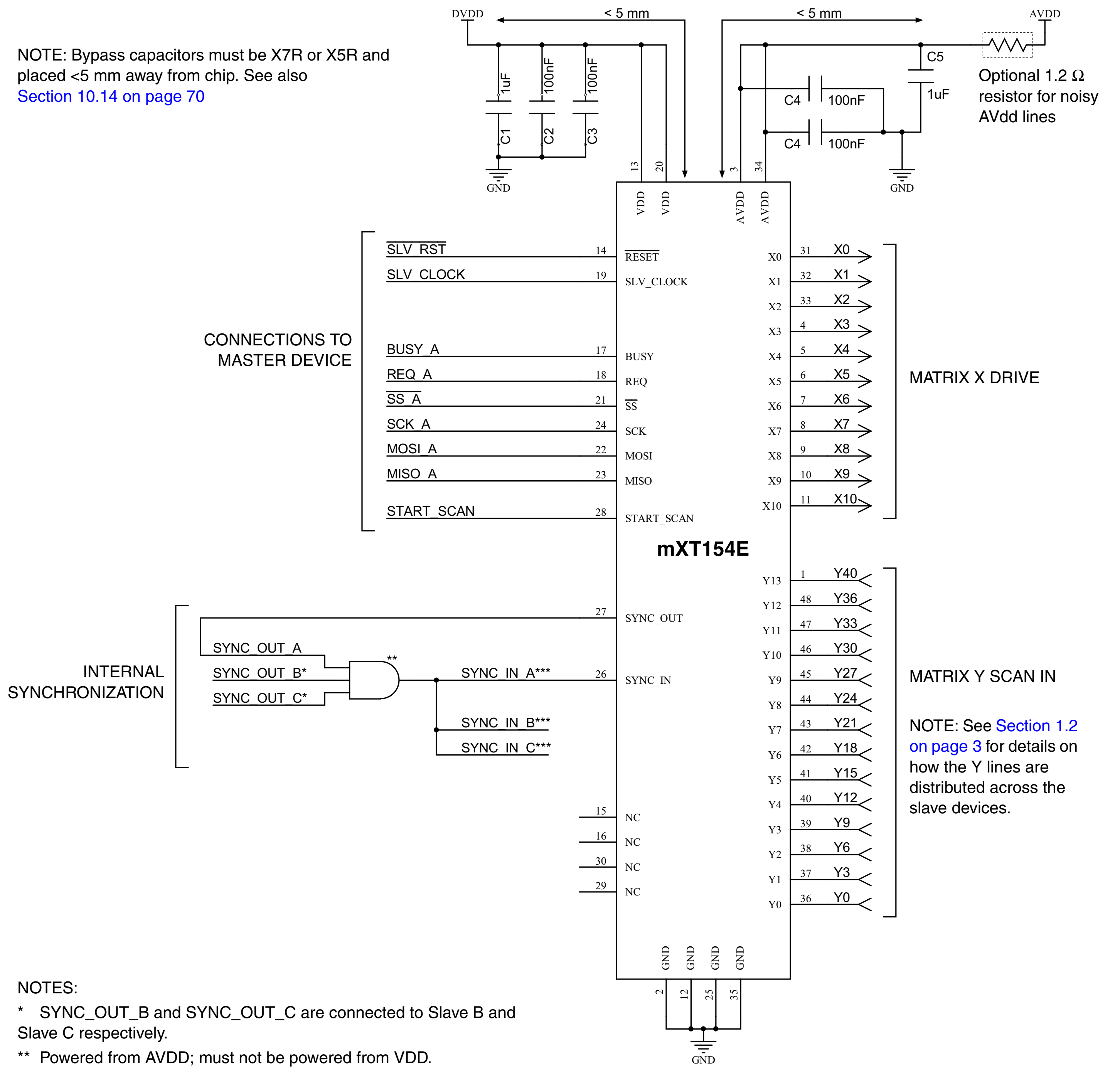
* SYNC_OUT_B and SYNC_OUT_C are connected to Slave B and Slave C respectively.

** Powered from AVDD; must not be powered from VDD.

*** SYNC_IN is also connected to Slave B and Slave C.

3.2.2 Slave Devices (3 × mXT154E) – 48-pin QFN

Note: Instance Slave A only is shown; Slave B and Slave C are omitted for simplicity.



NOTE: Bypass capacitors must be X7R or X5R and placed <5 mm away from chip. See also Section 10.14 on page 70
DVDD
< 5 mm
< 5 mm
AVDD
C1 1uF
C2 100nF
C3 100nF
C4 100nF
C4 100nF
C5 1uF
Optional 1.2 Ω resistor for noisy AVdd lines
GND
VDD
VDD
AVDD
AVDD
CONNECTIONS TO MASTER DEVICE
SLV_RST
SLV_CLOCK
BUSY_A
REQ_A
SS_A
SCK_A
MOSI_A
MISO_A
START_SCAN
RESET
SLV_CLOCK
BUSY
REQ
SS
SCK
MOSI
MISO
START_SCAN
mXT154E
MATRIX X DRIVE
X0 X1 X2 X3 X4 X5 X6 X7 X8 X9 X10
INTERNAL SYNCHRONIZATION
SYNC_OUT_A
SYNC_OUT_B*
SYNC_OUT_C*
**
SYNC_IN_A***
SYNC_IN_B***
SYNC_IN_C***
SYNC_OUT
SYNC_IN
NC
MATRIX Y SCAN IN
Y40 Y36 Y33 Y30 Y27 Y24 Y21 Y18 Y15 Y12 Y9 Y6 Y3 Y0
Y13 Y12 Y11 Y10 Y9 Y8 Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0
NOTE: See Section 1.2 on page 3 for details on how the Y lines are distributed across the slave devices.
GND

NOTES:

* SYNC_OUT_B and SYNC_OUT_C are connected to Slave B and Slave C respectively.

** Powered from AVDD; must not be powered from VDD.

*** SYNC_IN is also connected to Slave B and Slave C.

4. Touchscreen Basics

4.1 Sensor Construction

A touchscreen is usually constructed from a number of transparent electrodes. These are typically on a glass or plastic substrate. They can also be made using non-transparent electrodes, such as copper or carbon. Electrodes are normally formed by etching a material called Indium Tin Oxide (ITO). This is a brittle ceramic material, of high optical clarity and varying sheet resistance. Thicker ITO yields lower levels of resistance (perhaps tens to hundreds of Ω/square) at the expense of reduced optical clarity. Lower levels of resistance are generally more compatible with capacitive sensing. Thinner ITO leads to higher levels of resistance (perhaps hundreds to thousands of Ω/square) with some of the best optical characteristics.

Interconnecting tracks formed in ITO can cause problems. The excessive RC time constants formed between the resistance of the track and the capacitance of the electrode to ground can inhibit the capacitive sensing function. In such cases, ITO tracks should be replaced by screen printed conductive inks (non-transparent) outside the touchscreen viewing area.

A range of trade-offs also exist with regard to the number of layers used for construction. Atmel has pioneered single-layer ITO capacitive touchscreens. For many applications these offer a near optimum cost/performance balance. With a single layer screen, the electrodes are all connected using ITO out to the edges of the sensor. From there the connection is picked up with printed silver tracks. Sometimes two overprinted silver tracking layers are used to reduce the margins between the edge of the substrate and the active area of the sensor.

Two-layer designs can have a strong technical appeal where ultra-narrow edge margins are required. They are also an advantage where the capacitive sensing function needs to have a very precise cut-off as a touch is moved to just off the active sensor area. With a two-layer design the QMatrix transmitter electrodes are normally placed nearest the bottom and the receiver electrodes nearest the top. The separation between layers can range from hundreds of nanometers to hundreds of microns, with the right electrode design and considerations of the sensing environment.

4.2 Electrode Configuration

The specific electrode designs used in Atmel touchscreens are the subject of various patents and patent applications. Further information is available on request.

The chipset supports various configurations of electrodes as summarized below:

Touchscreens:	1 Touchscreen allowed
	3X × 3Y minimum (depends on screen resolution)
	33X × 42Y maximum (subject to other configurations)

4.3 Scanning Sequence

All channels are scanned in sequence by the chipset. There is full parallelism in the scanning sequence to improve overall response time. The channels are scanned by measuring capacitive changes at the intersections formed between the first X line and all the Y lines. Then the intersections between the next X line and all the Y lines are scanned, and so on, until all X and Y combinations have been measured.

The chipset can be configured in various ways. It is possible to disable some channels so that they are not scanned at all. This can be used to improve overall scanning time.

4.4 Touchscreen Sensitivity

4.4.1 Adjustment

Sensitivity of touchscreens can vary across the extents of the electrode pattern due to natural differences in the parasitics of the interconnections, control chip, and so on. An important factor in the uniformity of sensitivity is the electrode design itself. It is a natural consequence of a touchscreen pattern that the edges form a discontinuity and hence tend to have a different sensitivity. The electrodes at the far edges do not have a neighboring electrode on one side and this affects the electric field distribution in that region.

A sensitivity adjustment is available for the whole touchscreen. This adjustment is a basic algorithmic threshold that defines when a channel is considered to have enough signal change to qualify as being in detect.

4.4.2 Mechanical Stackup

The mechanical stackup refers to the arrangement of material layers that exist above and below a touchscreen. The arrangement of the touchscreen in relation to other parts of the mechanical stackup has an effect on the overall sensitivity of the screen. QMatrix technology has an excellent ability to operate in the presence of ground planes close to the sensor. QMatrix sensitivity is attributed more to the interaction of the electric fields between the transmitting (X) and receiving (Y) electrodes than to the surface area of these electrodes. For this reason, stray capacitance on the X or Y electrodes does not strongly reduce sensitivity.

Front panel dielectric material has a direct bearing on sensitivity. Plastic front panels are usually suitable up to about 1.2 mm, and glass up to about 2.5 mm (dependent upon the screen size and layout). The thicker the front panel, the lower the signal-to-noise ratio of the measured capacitive changes and hence the lower the resolution of the touchscreen. In general, glass front panels are near optimal because they conduct electric fields almost twice as easily as plastic panels.

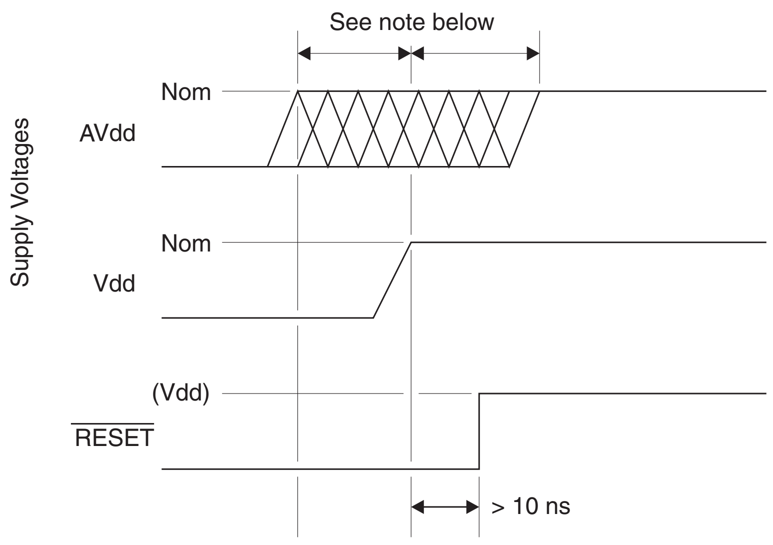
5. Detailed Operation

5.1 Power-up/Reset

The mXT1386E has an internal Power-on Reset (POR) that is executed on power up.

The device must be held in $\overline{\text{RESET}}$ (active low) while both the digital and analog power supplies (Vdd and AVdd) are powering up. If a slope or slew is applied to the digital or analog supplies, Vdd and AVdd must reach their nominal values before the $\overline{\text{RESET}}$ signal is de-asserted (that is, goes high). This is shown in Figure 5-1. See Section 10.2 on page 59 for nominal values for Vdd and AVdd.

Figure 5-1. Power Sequencing on the mXT1386E



Note: Vdd and Avdd can be powered up in either order.
There is no prerequisite for the length of time between Vdd and Avdd powering up.

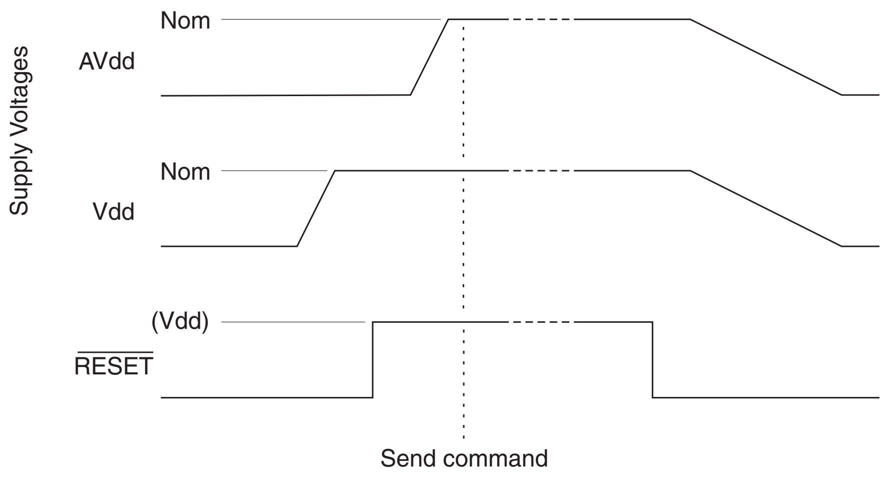
Note that there are no specific power-up, or power-down sequences required for the mXT1386E. This means that the digital or analog supplies can be applied independently and in any order during power-up.

After power-up, the mXT1386E takes ~80.8 ms before it is ready to start communications. Vdd must drop to below 1V in order to effect a proper POR. See Section 10 for further specifications.

If the $\overline{\text{RESET}}$ line is released before the AVDD supplies have reached their nominal voltage (see Figure 5-2), then some additional operations need to be carried out by the host. There are two options open to the host controller:

- Start the part in deep sleep mode and then send the command sequence to set the cycle time to wake the part and allow it to run normally. Note that in this case a calibration command is also needed.
- Send a reset command.

Figure 5-2. Power Sequencing on the mXT1386E – Late rise on AVDD



The $\overline{\text{RESET}}$ pin can be used to reset the mXT1386E whenever necessary. The $\overline{\text{RESET}}$ pin must be asserted low for at least 10 ns to cause a reset. After releasing the $\overline{\text{RESET}}$ pin the mXT1386E takes ~80.4 ms before it is ready to start communications. It is recommended to connect the $\overline{\text{RESET}}$ pin to a host controller to allow it to initiate a full hardware reset without requiring a power-down.

A software reset command can also be used to reset the chipset (refer to the Command Processor object in the *mXT1386E Protocol Guide*). A software reset takes ~231 ms. After the chipset has finished initializing it asserts the $\overline{\text{CHG}}$ line to signal to the host that a message is available. The reset flag is set in the Message Processor object to indicate to the host that it has just completed a reset cycle. This bit can be used by the host to detect any unexpected brownout events. This allows the host take any necessary corrective actions, such as reconfiguration.

A checksum check is performed on the configuration settings held in the nonvolatile memory of the master device. If the checksum does not match a stored copy of the last checksum, then this indicates that the settings have become corrupted. This is signaled to the host by setting the configuration error bit in the message data for the Command Processor object (refer to the *mXT1386E Protocol Guide* for more information).

Note that the $\overline{\text{CHG}}$ line is briefly set (approximately 100 ms) as an input after power-up or reset for diagnostic purposes. It is therefore particularly important that the line should be allowed to float high, via the $\overline{\text{CHG}}$ line pull-up resistors, during this period. It should not be driven by the host.

5.2 Calibration

Calibration is the process by which a sensor chip assesses the background capacitance on each channel. Channels are only calibrated on power-up and when:

- The channel is enabled (that is, activated).

OR

- The channel is already enabled and one of the following applies:
  - The channel is held in detect for longer than the Touch Automatic Calibration setting (refer to the *mXT1386E Protocol Guide* for more information on TCHAUTOCAL setting in the Acquisition Configuration object).
  - The signal delta on a channel is at least the touch threshold (TCHTHR) in the anti-touch direction, while no other touches are present on the channel matrix (refer to the *mXT1386E Protocol Guide* for more information on the TCHTHR field in the Multiple Touch Touchscreen object).
  - The user issues a recalibrate command.

A status message is generated on the start and completion of a calibration.

Note that the chipset performs a global calibration; that is, all the channels are calibrated together.

5.3 Operational Modes

The chipset operates in two modes: active (touch detected) and idle (no touches detected). Both modes operate as a series of burst cycles. Each cycle consists of a short burst (during which measurements are taken) followed by an inactive sleep period. The difference between these modes is the length of the cycles. Those in idle mode typically have longer sleep periods. The cycle length is configured using the IDLEACQINT and ACTVACQINT settings in the Power Configuration object. In addition, an Active to Idle timeout (ACTV2IDLETO) setting is provided.

Refer to the *mXT1386E Protocol Guide* for full information on how these modes operate, and how to use the settings provided.

5.4 Sense Lines

Each of the three mXT154E slave devices controls a maximum of 11 X lines and 14 Y lines. This makes a total of 33 X lines and 42 Y lines available for use. The X lines are distributed across the three slave devices in three sequential blocks. The Y lines, however, are distributed across the three slave devices in an interleaved manner, such that Y0 is controlled by Slave A, Y1 is controlled by Slave B, Y2 is controlled by Slave C, and so on. Each slave therefore controls the sense lines listed in Table 5-1. (See also Figure 1-1 on page 4.)

Table 5-1. Sense Lines

	Controls Sense Lines...	
Slave Device	**X**	**Y**
Slave A	X0 to X-10	Y0, Y3, Y6... Y39
Slave B	X11 to X-21	Y1, Y4, Y7... Y40
Slave C	X22 to X32	Y2, Y5 Y8... Y41

If fewer lines are required for use in the user's product, unused lines must be dropped from the slave devices in reverse sense line order, starting with the highest line.

5.5 Touchscreen Layout

5.5.1 Introduction

The physical matrix can be configured to have one or more touch objects. These are configured using the appropriate touch objects (Multiple Touch Touchscreen). It is not mandatory to have all the allowable touch objects present. The objects are disabled by default so only those that you wish to use need to be enabled. Refer to the *mXT1386E Protocol Guide* for more information on configuring the touch objects.

When designing the physical layout of the touch panel, obey the following rules:

- Each touch object should be a regular rectangular shape in terms of the lines it uses.
- The touch objects must not share the Y lines they use. The X lines can, however, be shared.
- The design of the touch objects does not physically need to be on a strict XY grid pattern.

5.6 Signal Processing

5.6.1 Detection Integrator

The chipset features a touch detection integration mechanism. This acts to confirm a detection in a robust fashion. A counter is incremented each time a touch has exceeded its threshold and has remained above the threshold for the current acquisition. When this counter reaches a preset limit the sensor is finally declared to be touched. If, on any acquisition, the signal is not seen to exceed the threshold level, the counter is cleared and the process has to start from the beginning.

The detection integrator is configured using the appropriate touch objects (Multiple Touch Touchscreen). Refer to the *mXT1386E Protocol Guide* for more information.

5.6.2 Digital Filtering and Noise Suppression

The mXT1386E supports the on-chip filtering of the acquisition data received from the sensor. Specifically, the Noise Suppression T48 object provides an algorithm to suppress the effects of noise (for example, from a noisy charger plugged into the user's product). This algorithm can automatically adjust some of the acquisition parameters on-the-fly to filter the analog-to-digital conversions (ADCs) received from the sensor. The algorithm can make use of a Grass Cutter (which rejects any samples outside a predetermined limit).

Noise suppression is triggered when a noise source is detected (typically when a charger is turned on). A hardware trigger can be implemented using the CHRG_IN pin. Alternatively, the host driver code can indicate when a noise source is present.

An alternative burst mode on the X lines, known as Dual X Drive, is provided. This improves the signal-to-noise ratio (SNR) on a closely spaced X sensor matrix (when finger touches are likely to cover more than one X line).

Refer to the *mXT1386E Protocol Guide* for more information on the Noise Suppression T48 object.

5.6.3 Stylus Support

The mXT1386E allows for the particular characteristics of stylus touches, whilst still allowing conventional finger touches to be detected. Stylus touches are configured by the Stylus T47 object. There is one instance of the Stylus T47 object for each Multiple Touch Touchscreen T9 object present on the device.

For example, stylus support ensures that the small touch area of a stylus registers as a touch, as this would otherwise by considered too small for the touchscreen. Additionally, there are controls to distinguish a stylus touch from an unwanted approaching finger (such as on the hand holding the stylus).

The touch sensitivity and threshold controls for stylus touches are configured separately from those for conventional finger touches so that both types of touches can be accommodated.

5.6.4 Grip Suppression

The mXT1386E has a grip suppression mechanism to suppress false detections when the user grips a handheld device.

Grip suppression works by specifying a boundary around a touchscreen, within which touches can be suppressed whilst still allowing touches in the center of the touchscreen. This ensures that a “rolling” hand touch (such as when a user grips a mobile device) is suppressed. A “real” (finger) touch towards the center of the screen is allowed.

Grip suppression is configured using the Grip Suppression T40 object. Refer to the *mXT1386E Protocol Guide* for more information.

5.6.5 Unintentional Touch Suppression

The Touch Suppression T42 object provides a mechanism to suppress false detections from unintentional touches from a large body area, such as from a face, ear or palm. This mechanism is enhanced by another mechanism (known as distance touch suppression) that operates in conjunction with large object suppression to suppress false touches only if they are less than a specified distance from a suppressed touch, whilst any touches greater than this distance remain unsuppressed. The Touch Suppression T42 object also provides Maximum Touch Suppression to suppress all touches if more than a specified number of touches has been detected. There is one instance of the Touch Suppression T42 object for each Multiple Touch Touchscreen T9 object present on the device. Refer to the *mXT1386E Protocol Guide* for more information.

5.6.6 Gestures

The chipset supports the on-chip processing of touches so that specific gestures can be detected. These may be a one-touch gesture (such as a tap or a drag) or they may be a two-touch gesture (such as a pinch or a rotate).

Gestures are configured using the One-touch Gesture Processor and the Two-touch Gesture Processor objects. Refer to the *mXT1386E Protocol Guide* for more information on gestures and their configuration.

5.7 Circuit Components

5.7.1 Bypass Capacitors

The mXT1386E master device requires a 4.7 µF capacitor with 100 nF ceramic X7R or X5R bypass capacitors on each of the Vdd and internal VDD_1V8 supplies.

The mXT154E slave devices require a 100 nF and a 1 µF bypass capacitor on the Vdd supply, and two 100 nF capacitors and a 1 µF capacitor on the AVdd supply. The capacitors should be ceramic X7R or X5R.

See the schematics in Section 3 on page 14 for examples of these.

The PCB traces connecting the capacitors to the pins of the mXT1386E and mXT154E devices must not exceed 5 mm in length. This limits any stray inductance that would reduce filtering effectiveness. See also Section 10.14 on page 70.

5.7.2 PCB Cleanliness

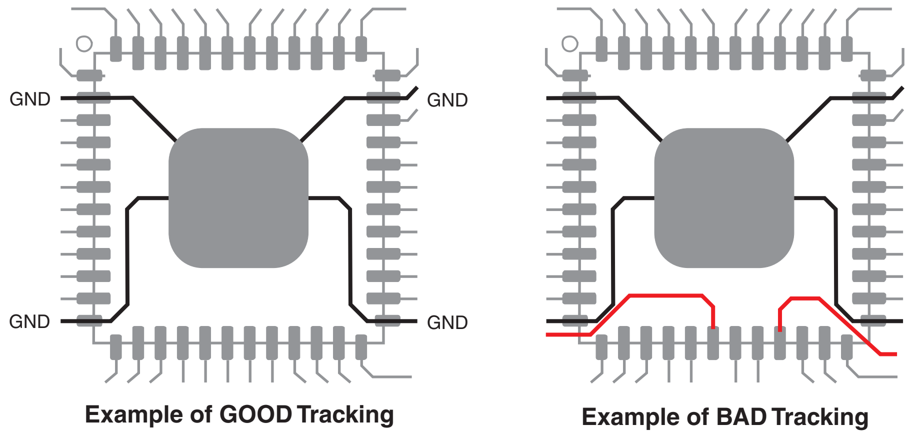
Modern no-clean-flux is generally compatible with capacitive sensing circuits.

CAUTION: If a PCB is reworked to correct soldering faults relating to any of the chipset devices, or to any associated traces or components, be sure that you fully understand the nature of the flux used during the rework process. Leakage currents from hygroscopic ionic residues can stop capacitive sensors from functioning. If you have any doubts, a thorough cleaning after rework may be the only safe option.

5.7.3 QFN Package Restrictions

The central pad on the underside of a QFN chip should be connected to ground. Do not run any tracks underneath the body of the chip, only ground. Figure 5-3 shows an example of good/bad tracking.

Figure 5-3. Examples of Good and Bad Tracking



5.7.4 Supply Quality

While the chipset has good Power Supply Rejection Ratio properties, poorly regulated and/or noisy power can significantly reduce performance. See Section 10.14 on page 70.

Always operate the chipset with a well-regulated and clean AVdd supply. It supplies the sensitive analog stages in the chipset.

There is no separate GND return pin for the analog stages. You are advised to consider return current paths from other current consumers in the system. Try to provide a separate heavy GND return trace or flood for the chipset that connects at a PSU star-point or connector pin. This helps to avoid inductive transient voltages coupling into the capacitive measurements made by the chip.

It is still recommended, however, that a low noise supply is used to prevent cross-talk into the analog sections.

5.7.5 Supply Sequencing

Vdd and AVdd can be powered independently of each other without damage to the chipset. Vdd and AVdd should be supplied with the same voltage unless specified by Atmel.

Make sure that any lines connected to the chipset are below or equal to Vdd during power-up. For example, if $\overline{\text{RESET}}$ is supplied from a different power domain to the mXT1386E master device Vdd pin, make sure that it is held low when Vdd is off. If this is not done, the $\overline{\text{RESET}}$ signal could parasitically couple power via the mXT1386E $\overline{\text{RESET}}$ pin into the Vdd supply.

5.7.6 Oscillator

The chipset requires an 16 MHz crystal oscillator connected to the master device. A crystal oscillator with a minimum accuracy of 100 ppm must be used.

5.7.7 Inter-slave Synchronization

Synchronization between the three slave devices is achieved using the SYNC_IN and SYNC_OUT pins (see Figure 1-1). These should be connected to a common AND gate. This means that when all the SYNC_OUT lines have been asserted, the SYNC_IN line is triggered simultaneously on each of the three slaves. An example of this is shown in the schematic in Section 3 on page 14.

Note that the logic levels for the AND gate are GND/AVDD (not VDD).

5.8 Debugging

The chipset provides a mechanism for obtaining raw data for development and testing purposes by reading data from the Diagnostic Debug object. Refer to the *mXT1386E Protocol Guide* for more information on this object.

A second mechanism is provided that allows the host to read the real-time raw data using the low-level debug port. This can be accessed via the SPI interface or the USB interface. Note that if the USB interface is used for normal communications, the debug data is output on the USB interface. Refer to QTAN0050, *Using the maXTouch Debug Port*, for more information on the debug port.

There is also a Self Test object that runs self-test routines in the mXT1386E to find hardware faults on the sense lines and the electrodes. Refer to the *mXT1386E Protocol Guide* for more information.

5.9 Communications

Communication with the host is achieved using either the I^{2}C-compatible interface (see Section 6 on page 27), the HID-I^{2}C-compatible interface (see Section 8 on page 47), or the USB interface (see Section 7 on page 35). Any interface can be used, depending on the needs of the user's project, but only one interface should be used in any one design. The selection of the I^{2}C or the HID-I^{2}C interface is determined by the I2CMODE pin.

Note that you only need to connect those pins that are actually required for use with the chosen communications interface. See Section 2.2 on page 8 for details on what should be done with the unconnected pins. This ensures optimal power consumption and correct functioning.

5.10 Configuring the Chipset

The chipset has an object-based protocol that organizes the features of the chipset into objects that can be controlled individually. This is configured using the Object Protocol common to many of the Atmel touch sensor devices. For more information on the Object Protocol and its implementation on the chipset, refer to the *mXT1386E Protocol Guide*.

6. I^{2}C-compatible Communications

6.1 Communications Protocol

The chipset can use an I^{2}C-compatible interface for communication. See Appendix E on page 86 for details of the I^{2}C-compatible protocol.

The I^{2}C-compatible interface is used in conjunction with the $\overline{\text{CHG}}$ line. The $\overline{\text{CHG}}$ line going active signifies that a new data packet is available. This provides an interrupt-style interface and allows the chipset to present data packets when internal changes have occurred.

6.2 I^{2}C-compatible Addresses

The chipset supports four I^{2}C-compatible device addresses. These are selected at start-up using the A0 and A1 pins on the mXT1386E master device (see Table 6-1). The address pins should be connected to GND to signal a logic “0”, and either left open or connected to VDD_3v3 to signal a logic 1 [1].

Table 6-1. I^{2}C-compatible Device Addresses

A1	A0	Address
0	0	0x4C
0	1	0x4D
1	0	0x5A
1	1	0x5B

The addresses are shifted left to form the SLA+W or SLA+R address when transmitted over the I^{2}C-compatible interface (see Table 6-2).

Table 6-2. Format of SLA+W and SLA+R

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Address (see Table 6-1)							Read/write

6.3 Writing To the Chipset

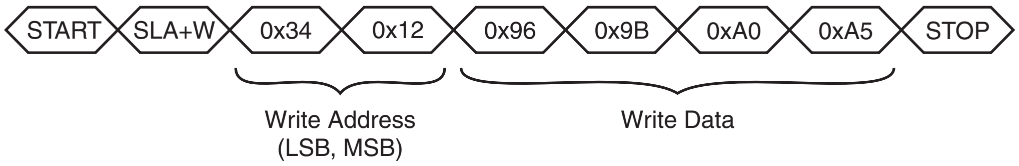
A WRITE cycle to the chipset consists of a START condition followed by the I^{2}C-compatible address of the device (SLA+W). The next two bytes are the address of the location into which the writing starts. The first byte is the Least Significant Byte (LSByte) of the address, and the second byte is the Most Significant Byte (MSByte). This address is then stored as the address pointer.

Subsequent bytes in a multibyte transfer form the actual data. These are written to the location of the address pointer, location of the address pointer +1, location of the address pointer + 2, and so on. The address pointer returns to its starting value when the WRITE cycle STOP condition is detected.

1. No external pull-down resistors are required on the A0 and A1 pins.

Figure 6-1 shows an example of writing four bytes of data to contiguous addresses starting at 0x1234.

Figure 6-1. Example of a Four-byte Write Starting at Address 0x1234

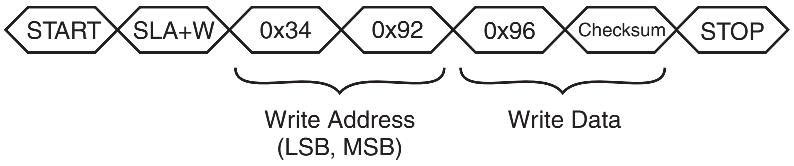


6.4 I²C-compatible Writes in Checksum Mode

In I²C-compatible checksum mode an 8-bit CRC is added to all I²C-compatible writes. The CRC is sent at the end of the data write as the last byte before the STOP condition. All the bytes sent are included in the CRC, including the two address bytes. Any command or data sent to the chipset is processed even if the CRC fails.

To indicate that a checksum is to be sent in the write, the most significant bit of the MSByte of the address is set to 1. For example, the I²C-compatible command shown in Figure 6-2 writes a value of 150 (0x96) to address 0x1234 with a checksum. The address is changed to 0x9234 to indicate checksum mode.

Figure 6-2. Example of a Write To Address 0x1234 With a Checksum



6.5 Reading From the Chipset

Two I²C-compatible bus activities must take place to read from the chipset. The first activity is an I²C-compatible write to set the address pointer (LSByte then MSByte). The second activity is the actual I²C-compatible read to receive the data. The address pointer returns to its starting value on detection of the NACK condition immediately before the STOP condition.

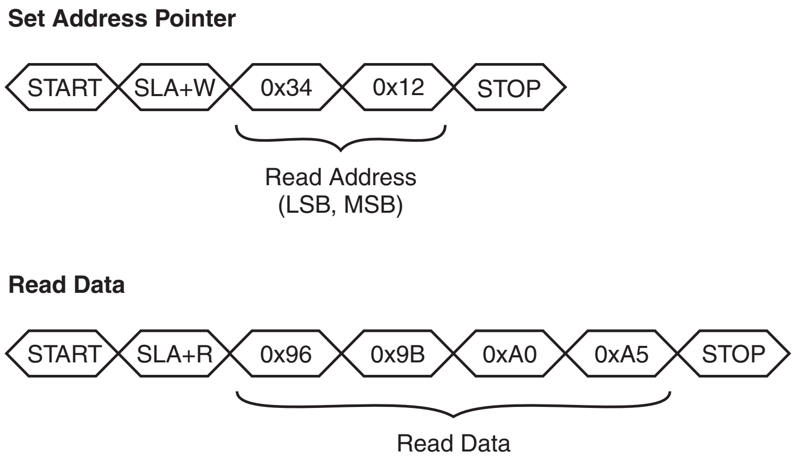
It is not necessary to set the address pointer before every read. The address pointer is updated automatically after every read operation. The address pointer will be correct if the reads occur in order. In particular, when reading multiple messages from the Message Processor object, the address pointer is automatically reset to allow continuous reads (see Section 6.6).

The WRITE and READ cycles consist of a START condition followed by the I²C-compatible address of the device (SLA+W or SLA+R respectively).

Figure 6-3 shows the I²C-compatible commands to read four bytes starting at address 0x1234.

Note: Although some chips may tolerate an illegal ACK before a STOP condition, the mXT1386E will not tolerate this. The correct I²C-specified sequence to terminate a read transfer is a NACK followed by a STOP condition.

Figure 6-3. Example of a Four-byte Read Starting at Address 0x1234



6.6 Reading Status Messages with DMA

The device facilitates the easy reading of multiple messages using a single continuous read operation. This allows the host hardware to use a direct memory access (DMA) controller for the fast reading of messages, as follows:

1. The host uses a write operation to set the address pointer to the start of the Message Count object, if necessary [1]. If a checksum is required on each message, the most significant bit of the MSByte of the read address must be set to 1.
2. The host starts the read operation of the message by sending a START condition.
3. The host reads the Message Count object (one byte) to retrieve a count of the pending messages (refer to the *mXT1386E Protocol Guide* for details).
4. The host calculates the number of bytes to read by multiplying the message count by the size of the Message Processor object. [2]

   Note that the size of the Message Processor object as recorded in the Object Table includes a checksum byte. If a checksum has not been requested, one byte should be deducted from the size of the object. That is: number of bytes = count × (size – 1).
5. The host reads the calculated number of message bytes. It is important that the host does *not* send a STOP condition during the message reads, as this will terminate the continuous read operation and reset the address pointer. No START and STOP conditions must be sent between the messages.
6. The host sends a STOP condition at the end of the read operation after the last message has been read. The NACK condition immediately before the STOP condition resets the address pointer to the start of Message Count object.

Figure 6-4 shows an example of using a continuous read operation to read three messages from the device without a checksum. Figure 6-5 on page 31 shows the same example with a checksum.

1. The STOP condition at the end of the read resets the address pointer to its initial location, so it may already be pointing at the Message Count object following a previous message read.
2. The host should have already read the size of the Message Processor object in its initialization code.

Figure 6-4. Continuous Message Read Example – No Checksum

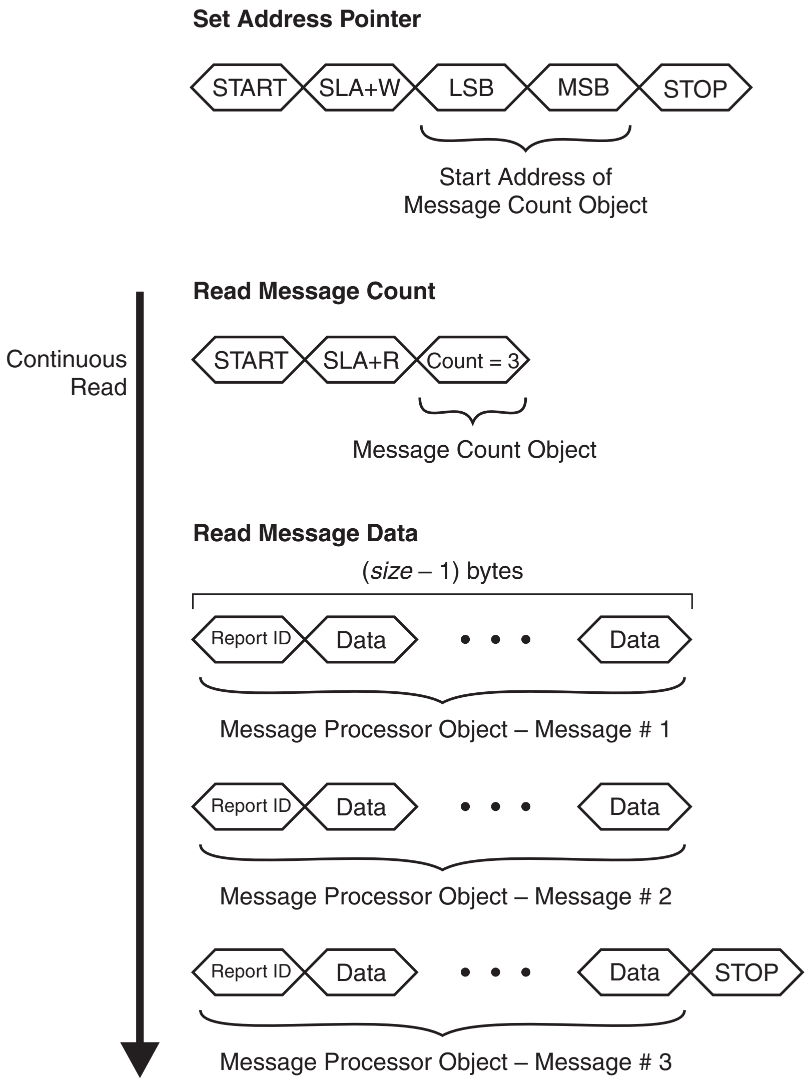
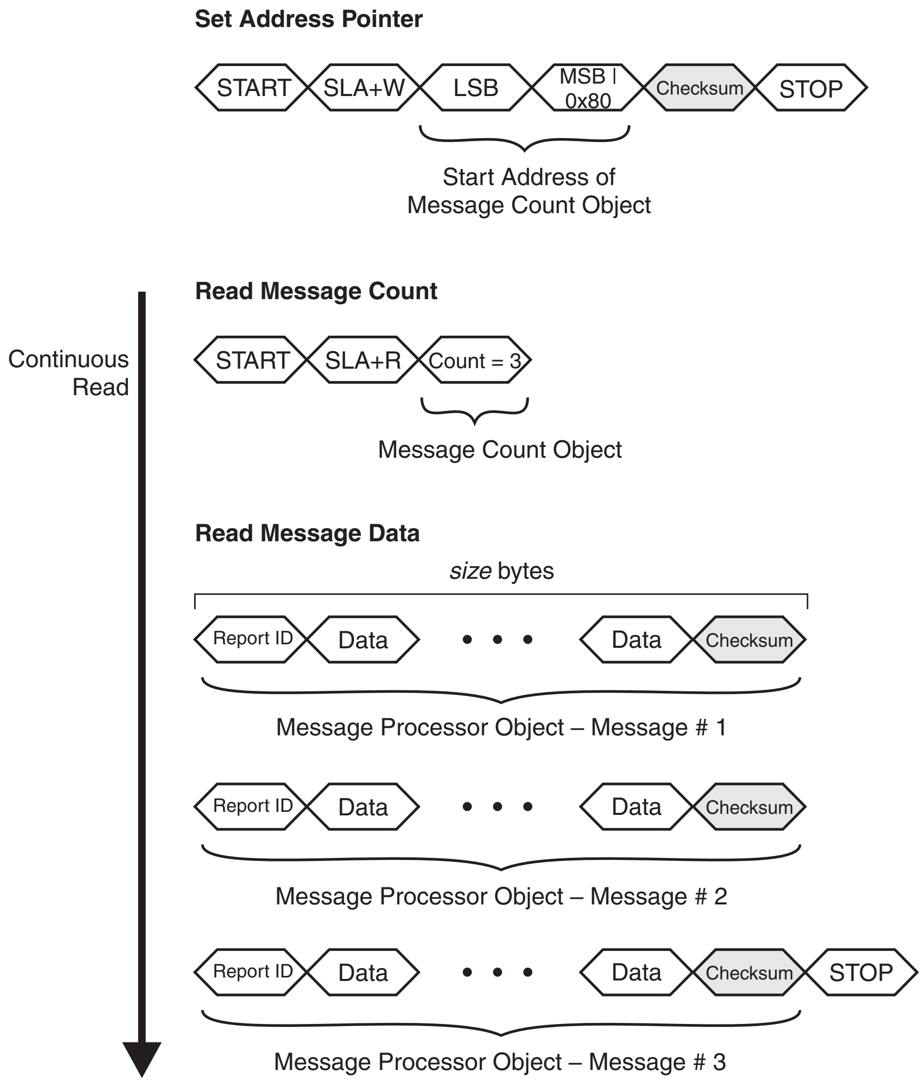


Figure 6-5. Continuous Message Read Example – I^{2}C-compatible Checksum Mode



There are no checksums added on any other I^{2}C-compatible reads. An 8-bit CRC can be added, however, to all I^{2}C-compatible writes, as described in Section 6.4 on page 28.

An alternative method of reading messages using the $\overline{\text{CHG}}$ line is given in Section 6.7.

6.7 $\overline{\text{CHG}}$ Line

The $\overline{\text{CHG}}$ line is an active-low, open-drain output that is used to alert the host that a new message is available in the Message Processor T5 object. This provides the host with an interrupt-style interface with the potential for fast response times. It reduces the need for wasteful I^{2}C-compatible communications.

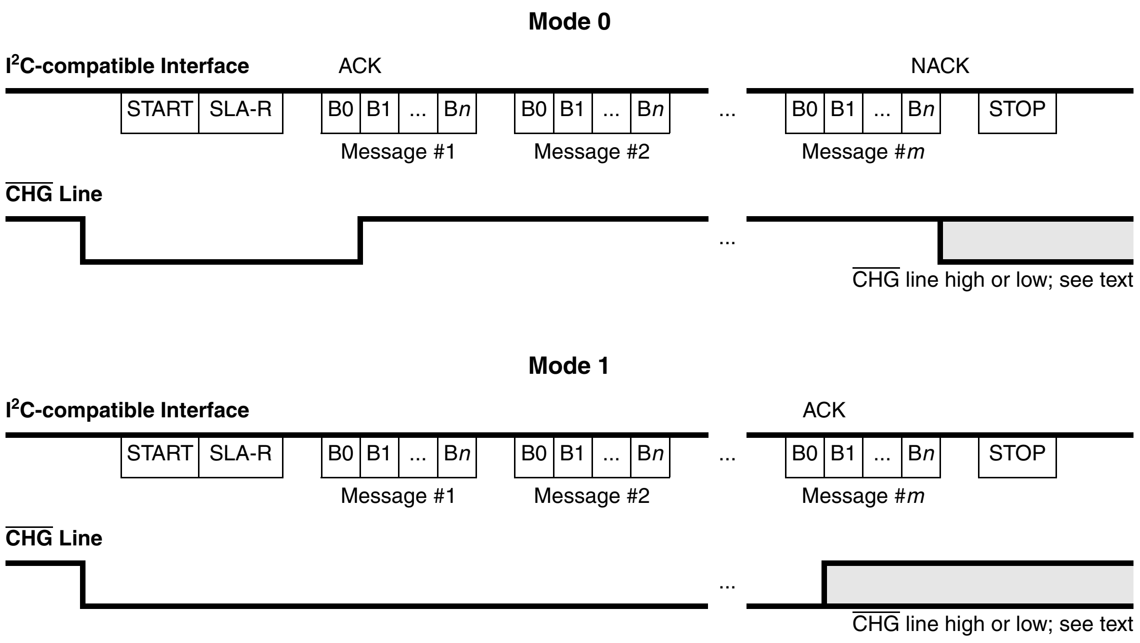
The $\overline{\text{CHG}}$ line remains low as long as there are messages to be read. The host should be configured so that the $\overline{\text{CHG}}$ line is connected to an interrupt line that is level-triggered. The host should not use an edge-triggered interrupt as this means adding extra software precautions.

The $\overline{\text{CHG}}$ line should be allowed to float during normal usage. This is particularly important after power-up or reset (see Section 5.1 on page 19).

A pull-up resistor is required, typically 10 kΩ to Vdd.

The $\overline{\text{CHG}}$ line operates in two modes, as defined by the Communications Configuration T18 object (refer to the *mXT1386E Protocol Guide*).

Figure 6-6. $\overline{\text{CHG}}$ Line Modes for I^{2}C-compatible Transfers



In Mode 0:

1. The $\overline{\text{CHG}}$ line goes low to indicate that a message is present.
2. The $\overline{\text{CHG}}$ line goes high when the first byte of the first message (that is, its report ID) has been sent and acknowledged (ACK sent) and the next byte has been prepared in the buffer.
3. The STOP condition at the end of an I^{2}C-compatible transfer causes the $\overline{\text{CHG}}$ line to stay high if there are no more messages. Otherwise the $\overline{\text{CHG}}$ line goes low to indicate a further message.

Mode 0 allows the host to continually read messages. Message reading ends when a report ID of 255 (“invalid message”) is received. Alternatively the host ends the transfer by sending a NACK after receiving the last byte of a message, followed by a STOP condition. If and when there is another message present, the $\overline{\text{CHG}}$ line goes low, as in step 1. In this mode the state of the $\overline{\text{CHG}}$ line does not need to be checked during the I^{2}C-compatible read.

In Mode 1:

1. The $\overline{\text{CHG}}$ line goes low to indicate that a message is present.
2. The $\overline{\text{CHG}}$ line remains low while there are further messages to be sent after the current message.
3. The $\overline{\text{CHG}}$ line goes high again only once the first byte of the last message (that is, its report ID) has been sent and acknowledged (ACK sent) and the next byte has been prepared in the output buffer.

Mode 1 allows the host to continually read the messages until the $\overline{\text{CHG}}$ line goes high, and the state of the $\overline{\text{CHG}}$ line determines whether or not the host should continue receiving messages from the chipset.

Note: The state of the $\overline{\text{CHG}}$ line should be checked only between messages and not between the bytes of a message. The precise point at which the $\overline{\text{CHG}}$ line changes state cannot be predicted and so the state of the $\overline{\text{CHG}}$ line cannot be guaranteed between bytes.

6.8 $\overline{\text{WAKE}}$ Line

The $\overline{\text{WAKE}}$ line is an active-low input that is used to wake the mXT1386E up from deep sleep mode before communicating with it via the I^2C-compatible interface. It can be used to minimize current consumption when the mXT1386E is in deep sleep mode. Refer to the *mXT1386E Protocol Guide* for information on deep sleep mode.

Note that the $\overline{\text{WAKE}}$ line is not used when the mXT1386E is not in deep sleep mode.

This pin must be connected in one of the following ways:

- It can be connected to the I^2C-compatible SCL pin.
- It can be connected to a GPIO pin on the host.
- It can also be left permanently low (connected to GND), but at the expense of increased power consumption in deep sleep mode.

The mXT1386E is ready to accept I^2C-compatible communications 25 ms after the $\overline{\text{WAKE}}$ line is asserted. This means that if the $\overline{\text{WAKE}}$ line is connected to a GPIO line, the line must be asserted 25 ms before the host attempts to communicate with the mXT1386E.

If the $\overline{\text{WAKE}}$ line is connected to the SCL pin, the mXT1386E will send a NACK on the first attempt to address it; the host must then retry 25 ms later.

The mXT1386E remains ready to accept I^2C-compatible communications for 2 seconds after the $\overline{\text{WAKE}}$ line is asserted, after which time the chip will timeout and return to deep sleep mode. This timeout period is reset every time there is an I^2C-compatible communication with the mXT1386E, or if the $\overline{\text{WAKE}}$ line is held asserted.

Note that when the mXT1386E is sent into deep sleep mode, it goes to sleep immediately. In this case the two-second timeout does not apply until the $\overline{\text{WAKE}}$ pin is asserted.

Note: In USB mode, (that is, when the I^2C-compatible interface is not being used), the $\overline{\text{WAKE}}$ pin should be connected to Vdd.

6.9 SDA, SCL

The I^2C-compatible bus transmits data and clock with SDA and SCL, respectively. These are open-drain. The I^2C-compatible master and slave devices can only drive these lines low or leave them open. The termination resistors (Rp) pull the line up to Vdd if no I^2C-compatible device is pulling it down.

The termination resistors commonly range from 1 kΩ to 10 kΩ. They should be chosen so that the rise times on SDA and SCL meet the I^2C-compatible specifications (see Section 10.9 on page 67).

6.10 Clock Stretching

The chipset supports clock stretching in accordance with the I^2C specification. It may also instigate a clock stretch if a communications event happens during a period when the chipset is busy internally.

The chipset has an internal bus monitor that can reset the internal I^2C-compatible hardware if SDA or SCL is stuck low. This means that if a prolonged clock stretch is seen by the chipset, then any ongoing transfers with the chipset may be corrupted. The bus monitor is enabled or disabled using the Communications Configuration object. Refer to the *mXT1386E Protocol Guide* for more information.

7. USB Communications

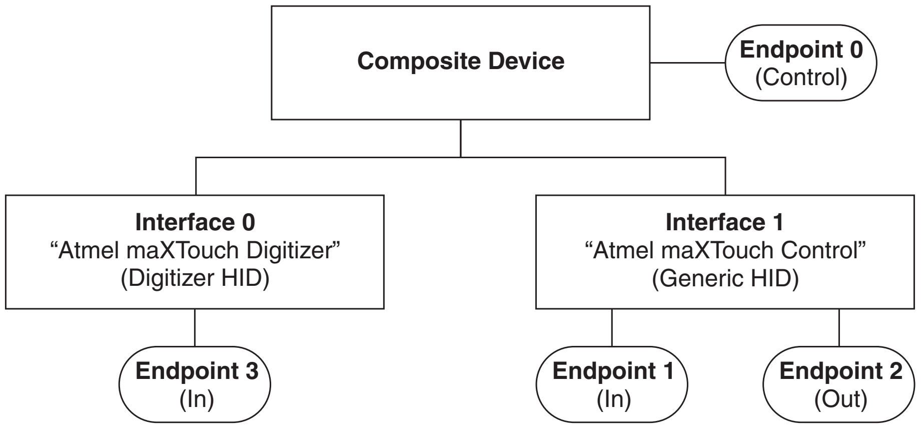
7.1 Communications Protocol

The chipset is a composite USB device with two Human Interface Device (HID) interfaces:

- **Interface 0** – This interface provides a Digitizer HID that supplies touch information to the Host for passing on to a PC operating system. This interface is supported by Microsoft® Windows® 7 without the need for additional software. The HID identifier string is "Atmel maXTouch Digitizer".
- **Interface 1** – This interface provides a Generic HID that allows the host to communicate with the chipset using the Object Protocol. The HID identifier string is "Atmel maXTouch Control".

The topography of the USB device is shown in Figure 7-1.

Figure 7-1. USB Topography


Composite Device
Endpoint 0 (Control)
Interface 0 "Atmel maXTouch Digitizer" (Digitizer HID)
Interface 1 "Atmel maXTouch Control" (Generic HID)
Endpoint 3 (In)
Endpoint 1 (In)
Endpoint 2 (Out)

Communication takes place using Full-speed USB at 12 Mbps.

For more information on the USB HID specifications visit www.usb.org.

7.2 Endpoint Addresses

The endpoint addresses are listed in Table 7-1.

Table 7-1. Endpoint Addresses

Endpoint	Direction	Address
Endpoint 0	Bidirectional (control)	–
Endpoint 1	In	0x81
Endpoint 2	Out	0x02
Endpoint 3	In	0x83

7.3 Composite Device

The composite device is a USB 2.0-compliant USB composite device running at full speed (12 Mbps). It has the following specification:

Vendor ID:	0x03EB (Atmel)
Product ID:	0x212A (mXT1386E)

Version:	16-bit Version & Build Identifier in the form 0xVVBB, where:
	VV = Version Major (Upper 4 bits) / Minor (Lower 4 bits)
	BB = Build number

The composite device has one bidirectional endpoint: the Control Endpoint (Endpoint 0). It is used by the USB Host to interrogate the USB device for details on its configurations, interfaces and report structures. It is also used to apply general device settings relating to USB Implementation.

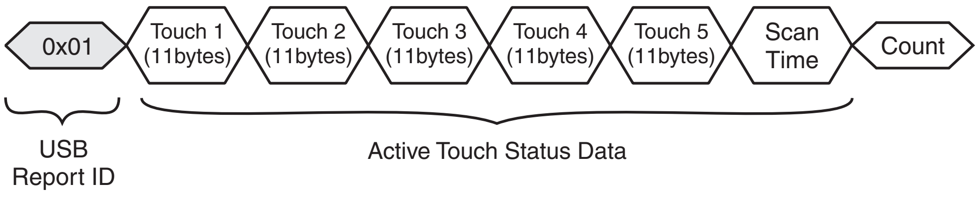
7.4 Interface 0 (Digitizer HID)

Interface 0 is a Digitizer-class HID, compliant with HID specification 1.11 with amendments. (1)

This interface consists of a single interrupt-In endpoint (Endpoint 3).

The format of an input report is shown in Figure 7-2. Each input report start with a USB Report ID (2) (value 0x01). This is followed by 5 sets of data (11 bytes each) that describe the status of up to 5 active touches. The input report is terminated by a single byte that contains the number of active touches.

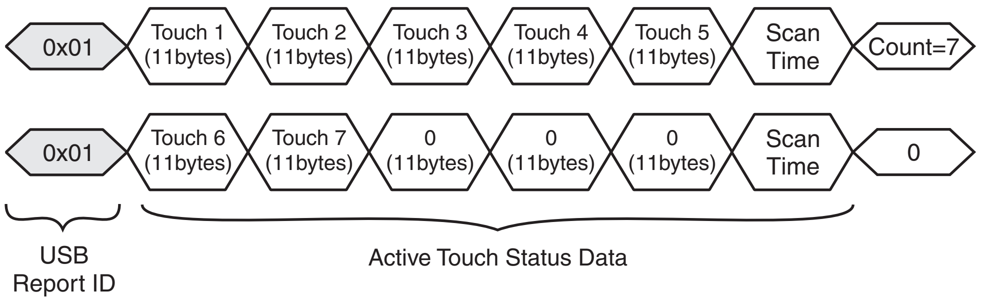
Figure 7-2. Input Report Packet



Any unused touch data bytes are set to zero (for example, the data for one active touch would be followed by 44 zeroed bytes). If there are more than five active touches to be reported, further input reports are sent with the remaining touch data. In this case, the count (for all touches) is sent in the count byte of the first report and the count byte in the following reports is zero. An example of the input report packets for 7 active touches is shown in Figure 7-3 on page 37.

1. This is an implementation of Microsoft's USB HID specification for Multitouch digitizers.
2. The term USB Report ID should not be confused with the term Report Id as used in the Object Protocol; the two are entirely different concepts.

Figure 7-3. Example Input Report Packets for 7 Active Touches



The input report format depends on the geometry calculation field (TCHGEOMEN) of the Digitizer HID Configuration T43 object. Table 7-2 and Table 7-3 gives the detailed format of an input report packet.

Table 7-2. Input Report Format when TCHGEOMEN is Enabled

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	USB Report ID							
1	Touch ID (first touch)					Reserved		Status
2	X Position LSByte (first touch)							
3	0	0	0	0	X Position MSBits (first touch)			
4	X Position LSByte (first touch)							
5	0	0	0	0	X Position MSBits (first touch)			
6	Y Position LSByte (first touch)							
7	0	0	0	0	Y Position MSBits (first touch)			
8	Y Position LSByte (first touch)							
9	0	0	0	0	Y Position MSBits (first touch)			
10	Touch width							
11	Touch height							
12 – 22	Touch data for second touch – same format as bytes 1 – 11							
23 – 33	Touch data for third touch – same format as bytes 1 – 11							
34 – 44	Touch data for fourth touch – same format as bytes 1 – 11							
45 – 55	Touch data for fifth touch – same format as bytes 1 – 11							
56 – 57	Scan time							
58	Contact count							

Table 7-3. Input Report Format when TCHGEOMEN is Disabled

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	USB Report ID							
1	Touch ID (first touch)					Reserved		Status
2	X Position LSByte (first touch)							
3	0	0	0	0	X Position MSBits (first touch)			
4 – 5	Reserved							

Table 7-3. Input Report Format when TCHGEOMEN is Disabled

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6	Y Position LSByte (first touch)							
7	0	0	0	0	Y Position MSBits (first touch)			
8 – 9	Reserved							
10 –11	Reserved							
12 – 22	Touch data for second touch – same format as bytes 1 – 11							
23 – 33	Touch data for third touch – same format as bytes 1 – 11							
34 – 44	Touch data for fourth touch – same format as bytes 1 – 11							
45 – 55	Touch data for fifth touch – same format as bytes 1 – 11							
56 – 57	Scan time							
58	Contact count							

In Table 7-3:

- **Byte 1:**

  Status: 1 = In detect, 0 = Not in detect.

  Touch ID: Identifies the touch for which this is a status report (starting from 1).

- **Bytes 2 to 9:**

  X and Y positions: These are scaled to 12-bit resolution. This means that the upper four bits of the MSByte will always be zero.
  Bytes 4, 5, 8, 9, 10 and 11 are Reserved when TCHGEOMEN field is set to 0.

- **Byte 10:**

  Touch Width: Reports the width of the detected touch.

- **Byte 11:**

  Touch Height: Reports the height of the detected touch.

- **Byte 56 to 57:**

  Scan Time: Timestamp associated with the current report frame with a 10 kHz resolution.

  Note that the scan time for each report packet of a single frame is same.

- **Byte 58:**

  Contact Count: Non-zero value in the first report packet of a frame indicating the total number of report packets in the frame. Zeros in the subsequent report packets within the frame.

There are two update conditions:

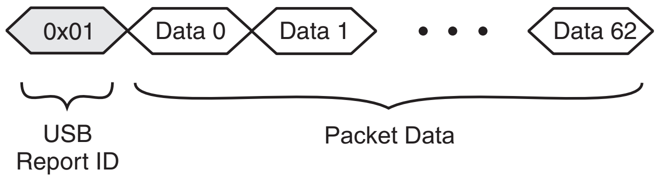
- **Change:** A change in status of any contact (touch) triggers a touch update message to be sent to the host.
- **Idle:** The idle delay of the Digitizer Interface may be controlled via the Control Endpoint as per the HID 1.11 specification (Set Idle command). By default this is set to a delay of 2 (8 ms).

7.5 Interface 1 (Generic HID)

Interface 1 is a Generic Human Interface Device, compliant with HID specification 1.11 with amendments. [1]

It consists of two endpoints: an interrupt-In endpoint (Endpoint 1) and an interrupt-out endpoint (Endpoint 2). The data packet in each case contains a 1-byte USB Report ID followed by 63 bytes of data, totalling 64 bytes (see Figure 7-4).

Figure 7-4. Data Packet for Interface 1



Commands are sent by the application software over the Interrupt-out endpoint, Endpoint 2. The command is sent as the first data byte of the packet data (data byte 0), followed by conditions and/or data.

The supported commands are as follows:

- Read/write Memory Map
- Send Auto-return messages
- Start debug monitoring
- End debug monitoring

Responses from the device are sent via the interrupt-In endpoint, Endpoint 1.

7.5.1 Read/Write Memory Map

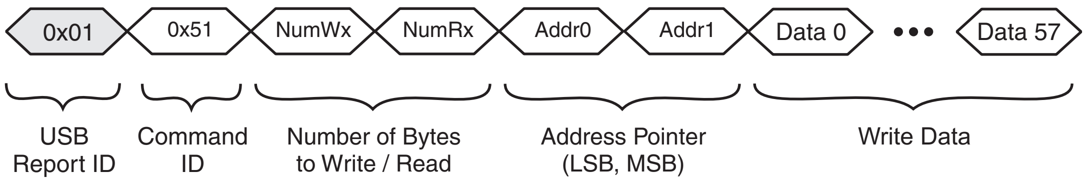
7.5.1.1 Introduction

This command is used to carry out a write/read operation on the memory map of the chipset.

The USB Report ID is 0x01.

The command packet has the generic format given in Figure 7-5. The following sections give examples on using the command to write to the memory map and to read from the memory map.

Figure 7-5. Generic Command Packet Format



In Figure 7-5:

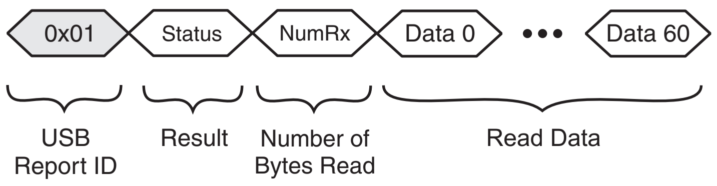
- **NumWx** is the number of data bytes to write to the memory map (may be zero). If the address pointer is being sent, this must include the size of the address pointer.

1. This is an implementation of the Microsoft USB HID specification for Multitouch digitizers.

- **NumRx** is the number of data bytes to read from the memory map (may be zero).
- **Addr 0** and **Addr 1** form the address pointer to the memory map (where necessary; may be zero if not needed).
- **Data 0** to **Data 57** are the bytes of data to be written (in the case of a write). Note that data locations beyond the number specified by NumWx will be ignored.

The response packet has the generic format given in Figure 7-6.

Figure 7-6. Response Packet Format



In Figure 7-6:

- **Status** indicates the result of the command:

  0x00 = read and write completed; read data returned

  0x04 = write completed; no read data requested
- **NumRx** is the number of bytes following that have been read from the memory map (in the case of a read). This will be the same value as NumRx in the command packet.
- **Data 0** to **Data 60** are the data bytes read from the memory map.

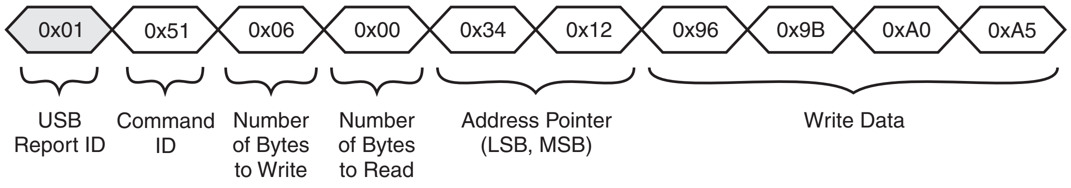
7.5.1.2 *Writing To the Chipset*

A write operation cycle to the chipset consists of sending a packet that contains six header bytes. These specify the USB report ID, the Command ID, the number of bytes to read, the number of bytes to write, and the 16-bit address pointer.

Subsequent bytes in a multibyte transfer form the actual data. These are written to the location of the address pointer, location of the address pointer +1, location of the address pointer + 2, and so on.

Figure 7-7 shows an example command packet to write four bytes of data to contiguous addresses starting at 0x1234.

Figure 7-7. Example of a Four-byte Write Starting at Address 0x1234

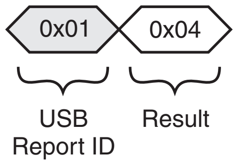


In Figure 7-7:

- The number of bytes to read is set to zero as this is a write-only operation.
- The number of bytes to write is six: that is, four data bytes plus the two address pointer bytes.

Figure 7-8 shows the response to this command. Note that the result status returned is 0x04 (that is, the write operation was completed but no read data was requested).

Figure 7-8. Response to Example Four-byte Write

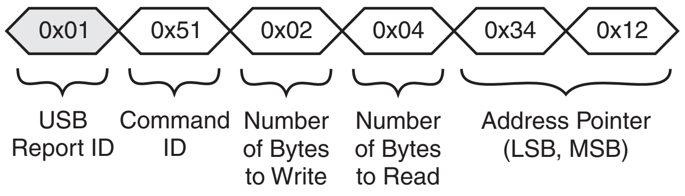


7.5.1.3 *Reading From the Chipset*

A read operation consists of sending a packet that contains the six header bytes only and no write data.

Figure 7-9 on page 41 shows an example command packet to read four bytes starting at address 0x1234. Note that the address pointer is included in the number of bytes to write, so the number of bytes to write is set to 2 as there are no other data bytes to be written.

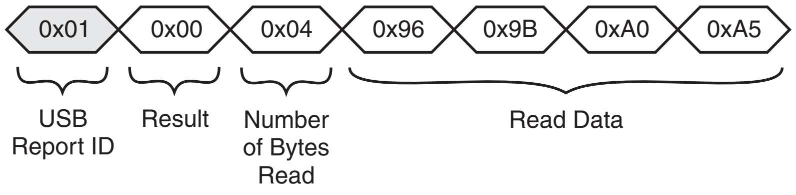
Figure 7-9. Example of a Four-byte Read Starting at Address 0x1234



It is not necessary to set the address pointer before every read. The address pointer is updated automatically after every read operation, so the address pointer will be correct if the reads occur in order.

Figure 7-10 shows the response to this command. The result status returned is 0x00 (that is the write operation was completed and the data was returned). The number of bytes returned will be the same as the number requested (4 in this case).

Figure 7-10. Response to Example Four-byte Read

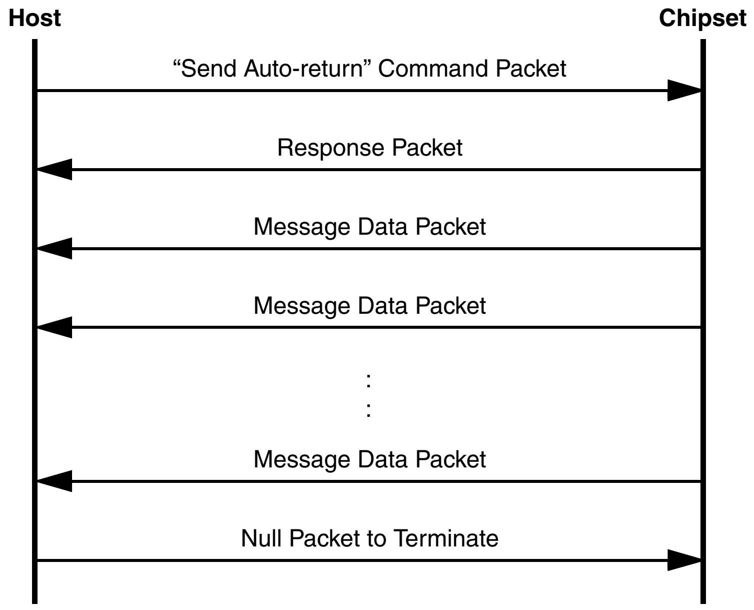


7.5.2 Send Auto-return Messages

7.5.2.1 Introduction

With this command the chipset can be configured to return new messages from the Message Processor object autonomously. The packet sequence to do this is shown in Figure 7-11.

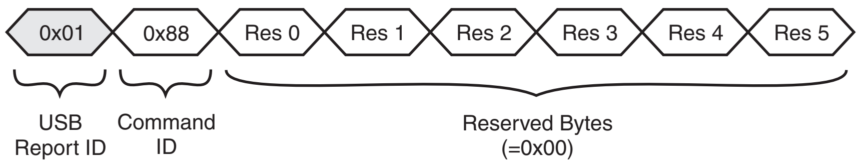
Figure 7-11. Packet Sequence for "Send Auto-return" Command.



The USB Report ID is 0x01.

The command packet has the format given in Figure 7-12.

Figure 7-12. Command Packet Format

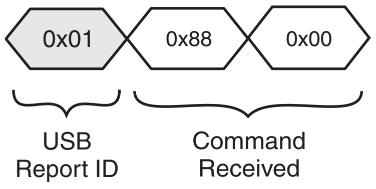


In Figure 7-12:

- **Res 0** to **Res 5** are reserved bytes with a value of 0x00.

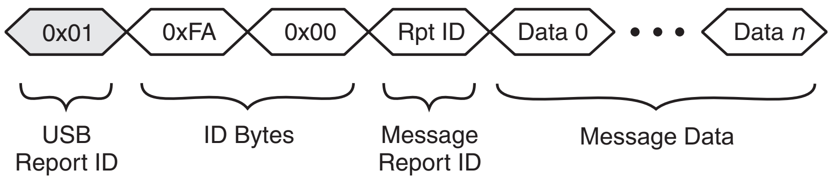
The response packet has the format given in Figure 7-13. Note that with this command, the command packet does not include an address pointer as the chipset already knows the address of the Message Processor object.

Figure 7-13. Response Packet Format



Once the chipset has responded to the command, it starts sending message data. Each time a message is generated in the Message Processor object, the chipset automatically sends a message packet to the host with the data. The message packets have the format given in Figure 7-14.

Figure 7-14. Message Packet Format



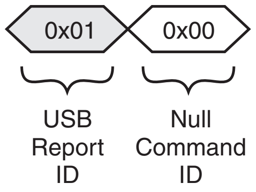
In Figure 7-14:

- **ID Bytes** identify the packet as an auto-return message packet.
- **Rpt ID** is the Report ID returned by the Message Processor object. [1]
- **Message Data** bytes are the bytes of data returned by the Message Processor. The size of the data depends on the source object for which this is the message data. Refer to the *mXT1386E Protocol Guide* for more information.

To stop the sending of the messages, the host can send a null command packet. This consists of two bytes: a report ID of 0x01 and a command byte of 0x00 (see Figure 7-15).

1. This is the Report ID used in the Object Protocol and should not be confused with the USB Report ID. Refer to the *mXT1386E Protocol Guide* for more information on the use of Report IDs in the Object Protocol.

Figure 7-15. Null Command Packet Format

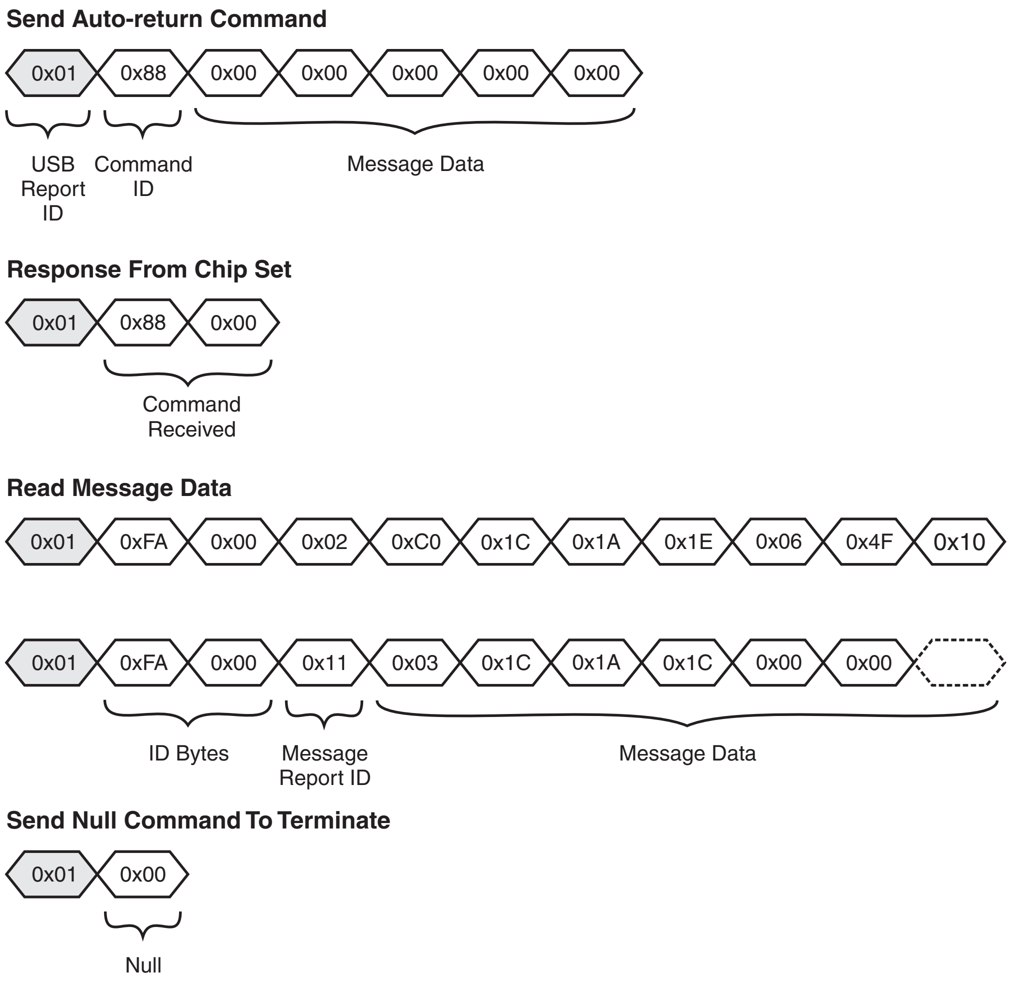


Note that the "Start Debug Monitoring" command may also terminate any currently enabled auto-return mode (see Section 7.5.3).

7.5.2.2 Reading Status Messages

Figure 7-9 shows an example sequence of packets to receive messages from the Message Processor object using the "Send Auto-return" command.

Figure 7-16. Example Auto-return Command Packet



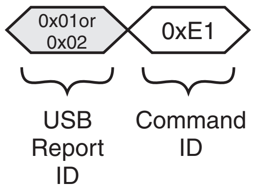
7.5.3 Start Debug Monitoring

This command instructs the device to return debug-monitoring data packets using the debug port, if this feature has been enabled in the Command Processor object.

The USB Report ID can be either 0x01 or 0x02. This allows the source of the request to be identified. The main difference is that a USB Report ID of 0x01 will terminate any currently enabled auto-return mode (see Section 7.5.2 on page 42).

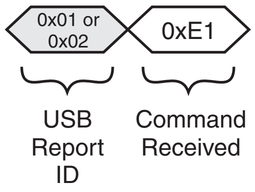
The command packet has the format given in Figure 7-17.

Figure 7-17. Command Packet Format



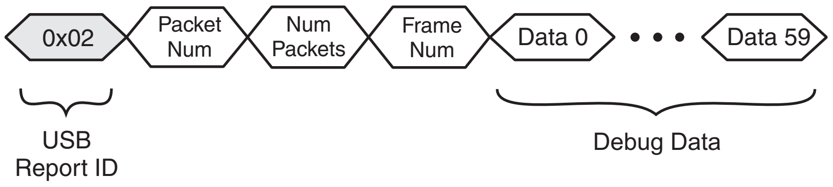
The response packet has the format given in Figure 7-18. Note that the USB Report ID will be the same as that used in the command packet.

Figure 7-18. Response Packet Format



The debug data packet has the format given in Figure 7-19.

Figure 7-19. Debug Data Packet Format



In Figure 7-19:

- **PacketNum** is the number of this USB packet in the debug data frame (full set of debug data). Refer to QTAN0050, *Using the maXTouch Debug Port*, for more information on the format of the debug data.
- **NumPackets** is the total number of USB packets that make up a debug data frame.
- **FrameNum** is the ID number of this frame.
- **Data 0** to **Data 59** are 60 bytes of debug data.

7.5.4 Stop Debug monitoring

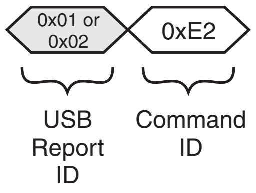
This command instructs the device to cease returning debug-monitoring data packets.

The command packet has the following format:

The USB Report ID is either 0x01 or 0x02.

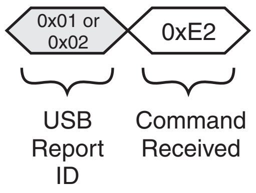
The command packet has the format given in Figure 7-20.

Figure 7-20. Command Packet Format



The response packet has the format given in Figure 7-21.

Figure 7-21. Response Packet Format



7.6 USB Suspend Mode

When the mXT1386E is used in USB configuration, the USB *System Suspend* event can be used to minimize current consumption. Note that it is possible to put the mXT1386E into deep sleep mode without also sending a *System Suspend* event on the USB bus, but the current consumption is not as low. The USB controller must send a USB *System Wake Up* event on the bus to bring the mXT1386E out of suspend mode.

The mXT1386E can also be configured to respond to USB *Remote Wakeup* requests. In this case, if the operating system enables remote wakeup and the mXT1386E is suspended, the chipset will continue to scan at a preset sensor refresh rate. Use of the remote wake up feature and the sensor refresh rate are configured using the Digitizer HID Configuration T43 object (refer to the *mXT1386E Protocol Guide* for more information).

8. HID-I^{2}C-compatible Communications

8.1 Communications Protocol

The chipset is an HID-I^{2}C device presenting two top-level collections (TLCs):

Interface 0 (Digitizer HID-I^{2}C) – supplies touch information to the host. This interface is supported by Microsoft Windows 8 without the need for additional software.

Note that touch messages are reported using the HID report ID of 0x01.

Interface 1 (Generic HID-I^{2}C) – This interface provides a generic HID-I^{2}C interface that allows the host to communicate with the chipset using the object protocol.

This interface can be used to configure the device and to receive object protocol messages using the Microsoft Windows Inbox driver and native Microsoft Windows APIs. This interface uses the HID report ID of 0x06, this value is subject to change and to guarantee correct operation the host software should use the value reported in the report descriptor.

Other features are identical to standard I^{2}C communication described in Section 6.1 on page 27.

8.2 I^{2}C-compatible Addresses

See Section 6.2 on page 27.

8.3 Device

The device is compliant with HID-I^{2}C specification V0.91. It has the following specification:

Vendor ID:	0x03EB (Atmel)
Product ID:	0x212A

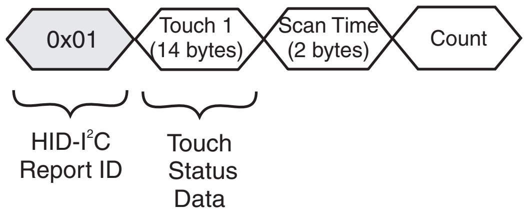
Version:	16-bit Version & Build Identifier in the form 0xVVBB, where:
	VV = Version Major (Upper 4 bits) / Minor (Lower 4 bits)
	BB = Build number

HID descriptor address:	0x0000.

8.4 Interface 0 (Digitizer HID-I^{2}C)

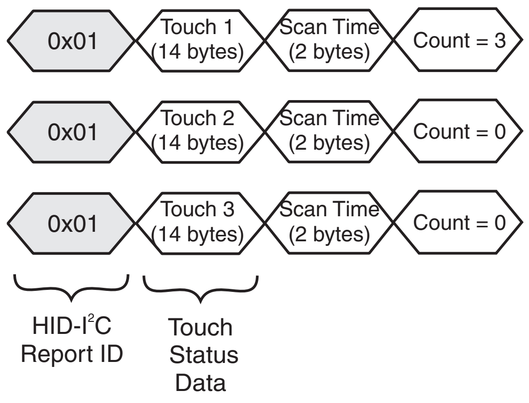
The format of an input report is shown in Figure 8-1. Each input report starts with a report ID and each input report message report contains data of one touch.

Figure 8-1. Input Report Packet



An example of the input report packets for 3 active touches is shown in Figure 8-2.

Figure 8-2. Example Input Report Packets for 3 Active Touches



Each input report consists of a HID-I^{2}C report ID followed by 17 bytes of that describe the status of one active touch. The input report format depends on the geometry calculation field (TCHGEOMEN) of the Digitizer HID Configuration T43 object. Table 8-1 and Table 8-2 explains the detailed format of an input report packet..

Table 8-1. Input Report Format when TCHGEOMEN = 1

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	HID-I2C Report ID							
1	Reserved							Status
2	Touch ID							
3 – 4	X Position							
5 – 6	X' Position							
7 – 8	Y Position							
5 – 6	X' Position							
7 – 8	Y Position							
9 – 10	Y' Position							
11	Touch Width							
12	Reserved							
13	Touch Height							
14	Reserved							
15 – 16	Scan Time HRTS							
17	Contact Count							
4 – 5	Reserved							

Table 8-2. Input Report Format when TCHGEOMEN = 0

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	HID-I2C Report ID							
1	Reserved							Status
2	Touch ID							
3 – 4	X Position							
5 – 6	Reserved							
7 – 8	Y Position							
5 – 6	Reserved							
7 – 8	Y Position							
9 – 10	Reserved							
11	Reserved							
12	Reserved							
13	Reserved							
14	Reserved							
15 – 16	Scan Time							
17	Contact Count							
4 – 5	Reserved							

- **Byte 2:**

  Touch ID: Identifies the touch for which this is a status report (starting from 0).
- **Bytes 3 to 10:**

  X and Y positions: These are scaled to 12-bit resolution. This means that the upper four bits of the MSByte will always be zero.

  Bytes 5, 6, 9, 10 are Reserved when TCHGEOMEN is set to 0.
- **Byte 11:**

  Touch Width: Reports the width of the detected touch when TCHGEOMEN is set to 1.
- **Byte 13:**

  Touch Height: Reports the height of the detected touch when TCHGEOMEN is set to 1.
- **Byte 15 to 16:**

  Scan Time: Timestamp associated with the current report packet with a 10 kHz resolution.
- **Byte 17:**

  Contact Count: Number of active touches.

8.5 Interface 1 (Generic HID-I^{2}C)

Interface 1 is a generic human interface device. It supports an input report for receiving data from the device and an output report for sending data to the device.

This interface is used for configuring the device using Atmel tools, or the customer can create their own tools or software that access the interface using the Microsoft Windows Inbox driver.

Commands are sent by the host using the output reports. Responses from the device are sent using input reports.

Supported commands are:

- Read/Write Memory Map
- Send Auto-return Messages.

If this interface generates an input report it will be loaded into the input buffer, the $\overline{\text{CHG}}$ line will be asserted and the Microsoft Windows Inbox driver will automatically receive it.

8.5.1 Read/Write Memory Map

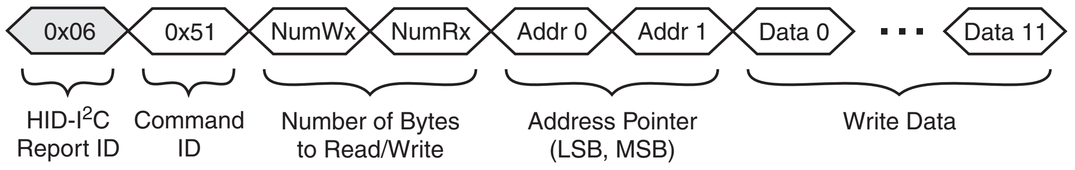
8.5.1.1 Introduction

This command is used to carry out a write/read operation using the object protocol on the memory map of the chipset.

The HID-I^{2}C Report ID is 0x06.

The command packet has the generic format given in Figure 8-3. The following sections give examples on using the command to write to the memory map and to read from the memory map.

Figure 8-3. Generic Command Packet Format

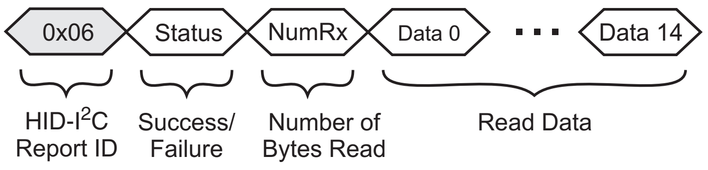


In Figure 8-3:

- **NumWx** is the number of data bytes to write to the memory map (may be zero). If the address pointer is being sent, this must include the size of the address pointer. The maximum value is 14.
- **NumRx** is the number of data bytes to read from the memory map (may be zero). The maximum value is 15.
- **Addr 0** and **Addr 1** form the address pointer to the memory map (where necessary; may be zero if not needed).
- **Data 0** to **Data 11** are the bytes of data to be written (in the case of a write). Note that data locations beyond the number specified by NumWx will be ignored.

The response packet has the generic format given in Figure 8-4.

Figure 8-4. Response Packet Format



In Figure 8-4:

- **Status** indicates the result of the command:

  0x00 = read and write completed; read data returned

  0x04 = write completed; no read data requested
- **NumRx** is the number of bytes following that have been read from the memory map (in the case of a read). This will be the same value as NumRx in the command packet.
- **Data 0** to **Data 14** are the data bytes read from the memory map.

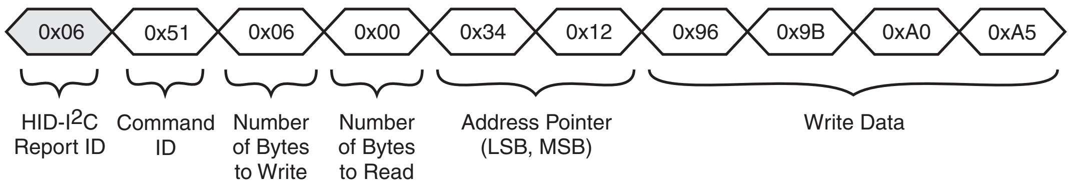
8.5.1.2 *Writing To the Chipset*

A write operation cycle to the chipset consists of sending a packet that contains six header bytes. These specify the HID-I^{2}C report ID, the Command ID, the number of bytes to read, the number of bytes to write, and the 16-bit address pointer.

Subsequent bytes in a multibyte transfer form the actual data. These are written to the location of the address pointer, location of the address pointer +1, location of the address pointer + 2, and so on.

Figure 8-5 shows an example command packet to write four bytes of data to contiguous addresses starting at 0x1234.

Figure 8-5. Example of a Four-byte Write Starting at Address 0x1234

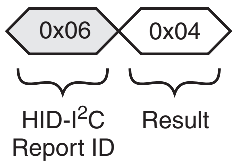


In Figure 8-5:

- The number of bytes to read is set to zero as this is a write-only operation.
- The number of bytes to write is six: that is, four data bytes plus the two address pointer bytes.

Figure 8-6 shows the response to this command. Note that the result status returned is 0x04 (that is, the write operation was completed but no read data was requested).

Figure 8-6. Response to Example Four-byte Write

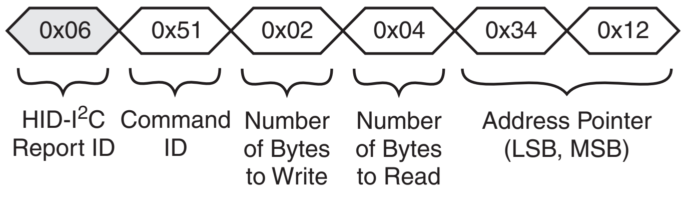


8.5.1.3 *Reading From the Chipset*

A read operation consists of sending a packet that contains the six header bytes only and no write data.

Figure 8-7 shows an example command packet to read four bytes starting at address 0x1234. Note that the address pointer is included in the number of bytes to write, so the number of bytes to write is set to 2 as there are no other data bytes to be written.

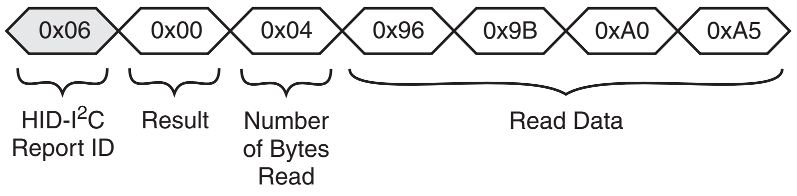
Figure 8-7. Example of a Four-byte Read Starting at Address 0x1234



It is not necessary to set the address pointer before every read. The address pointer is updated automatically after every read operation, so the address pointer will be correct if the reads occur in order.

Figure 8-8 shows the response to this command. The result status returned is 0x00 (that is the write operation was completed and the data was returned). The number of bytes returned will be the same as the number requested (4 in this case).

Figure 8-8. Response to Example Four-byte Read

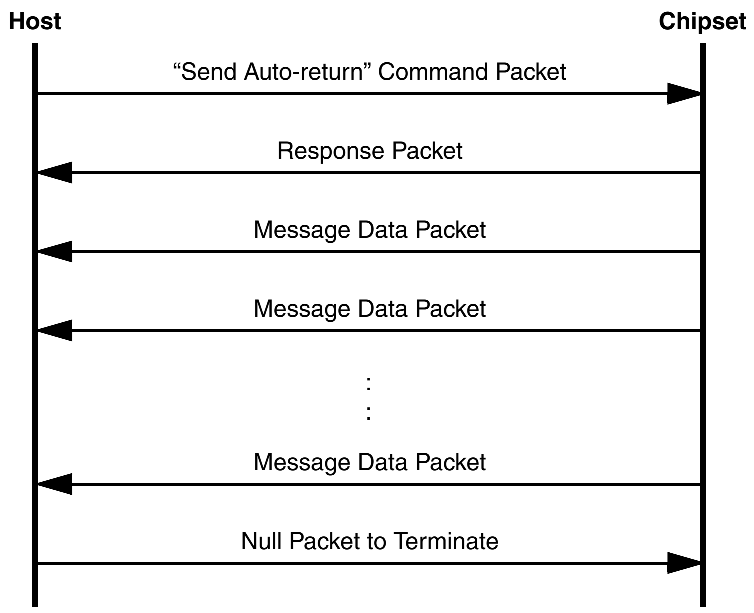


8.5.2 Send Auto-return Messages

8.5.2.1 Introduction

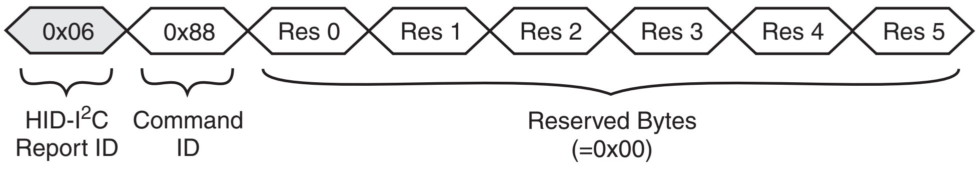
With this command the chipset can be configured to return new messages from the Message Processor T5 object autonomously. The packet sequence to do this is shown in Figure 8-9.

Figure 8-9. Packet Sequence for "Send Auto-return" Command.



The command packet has the format given in Figure 8-10.

Figure 8-10. Command Packet Format

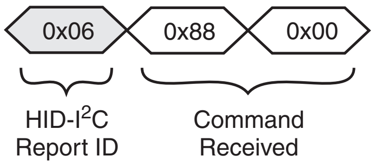


In Figure 8-10:

- **Res 0** to **Res 5** are reserved bytes with a value of 0x00.

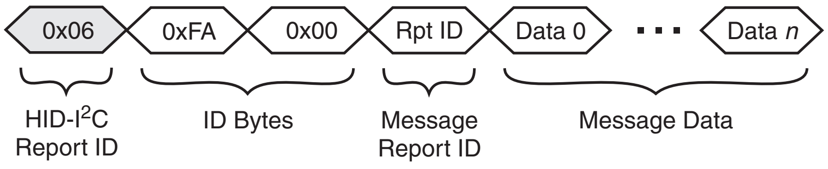
The response packet has the format given in Figure 8-11. Note that with this command, the command packet does not include an address pointer as the chipset already knows the address of the Message Processor object.

Figure 8-11. Response Packet Format



Once the chipset has responded to the command, it starts sending message data. Each time a message is generated in the Message Processor object, the chipset automatically sends a message packet to the host with the data. The message packets have the format given in Figure 8-12.

Figure 8-12. Message Packet Format



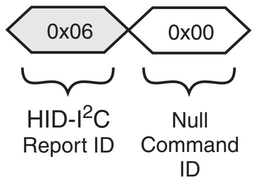
In Figure 8-12:

- **ID Bytes** identify the packet as an auto-return message packet.
- **Rpt ID** is the Report ID returned by the Message Processor object. [(1)]
- **Message Data** bytes are the bytes of data returned by the Message Processor. The size of the data depends on the source object for which this is the message data. Refer to the *mXT1386E Protocol Guide* for more information.

To stop the sending of the messages, the host can send a null command packet. This consists of two bytes: a report ID of 0x01 and a command byte of 0x06 (see Figure 8-13).

1. This is the Report ID used in the Object Protocol and should not be confused with the USB Report ID. Refer to the *mXT1386E Protocol Guide* for more information on the use of Report IDs in the Object Protocol.

Figure 8-13. Null Command Packet Format

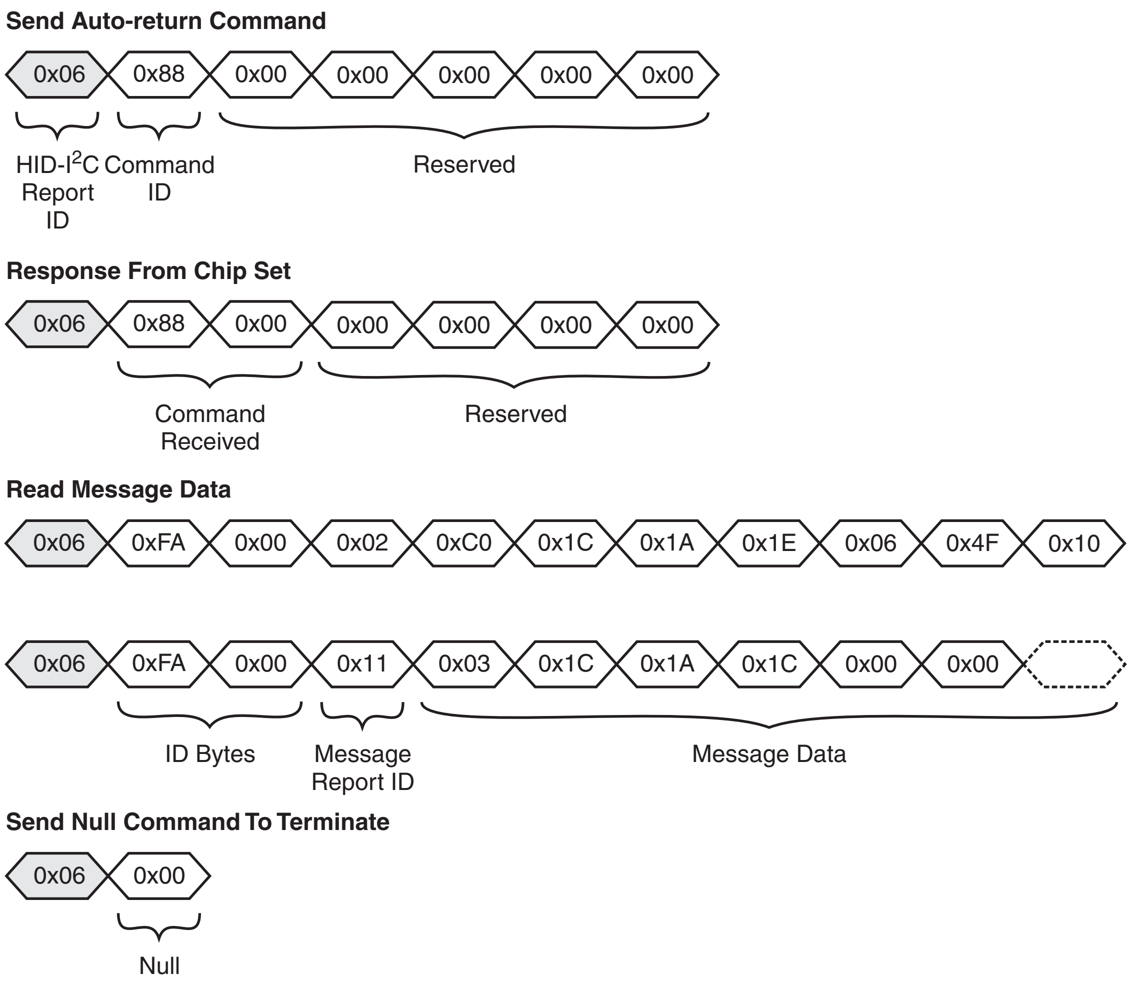


Note that any read or write will also terminate any currently enabled auto-return mode (see Section 7.5.3).

8.5.2.2 *Reading Status Messages*

Figure 8-14 shows an example sequence of packets to receive messages from the Message Processor object using the “Send Auto-return” command.

Figure 8-14. Example Auto-return Command Packet


Send Auto-return Command
0x06 0x88 0x00 0x00 0x00 0x00 0x00
HID-I2C Report ID
Command ID
Reserved
Response From Chip Set
0x06 0x88 0x00 0x00 0x00 0x00 0x00
Command Received
Reserved
Read Message Data
0x06 0xFA 0x00 0x02 0xC0 0x1C 0x1A 0x1E 0x06 0x4F 0x10
0x06 0xFA 0x00 0x11 0x03 0x1C 0x1A 0x1C 0x00 0x00
ID Bytes
Message Report ID
Message Data
Send Null Command To Terminate
0x06 0x00
Null

8.6 $\overline{\text{CHG}}$ Line

The $\overline{\text{CHG}}$ line is an active-low, open-drain output that is used to alert the host that a new message is available in the Input Buffer. This provides the host with an interrupt-style interface with the potential for fast response times as it reduces the need for wasteful I^2C-compatible communications.

Further information on the $\overline{\text{CHG}}$ line is given in Section 6.7 on page 31.

8.7 $\overline{\text{WAKE}}$ Line

Identical to standard I^2C operation. See Section 6.8 on page 33.

8.8 SDA, SCL

Identical to standard I^2C operation. See Section 6.8 on page 33.

8.9 Clock Stretching

Identical to standard I^2C operation. See Section 6.10 on page 34.

8.10 Microsoft Windows 8 Compliance

The mXT1386E has algorithms within the Digitizer HID Configuration T43 and Multiple Touch Touchscreen T9 specifically to ensure Microsoft Windows 8 compliance.

The mXT1386E also supports Microsoft Touch Hardware Quality Assurance (THQA) in the Serial Data Command T68 object. Refer to the Microsoft whitepaper on "*How to Design and Test Multitouch Hardware Solutions for Windows 8*".

These, and other device features, may need specific tuning.

9. Getting Started With the mXT1386E

9.1 Establishing Contact

9.1.1 Communication with the Host

The host can use either the I^{2}C-compatible bus (see Section 6.1 on page 27) or the USB interface (see Section 7.1 on page 35), or the HID-I^{2}C bus (see Section 8.1 on page 47) to communicate with the chipset.

9.1.2 I^{2}C-compatible Interface

On power-up, the $\overline{\text{CHG}}$ line goes low to indicate that there is new data to be read from the Message Processor object. If the $\overline{\text{CHG}}$ line does not go low, there is a problem with the chipset.

The host should attempt to read any available messages to establish that the chipset is present and running following power-up or a reset. Examples of messages include reset or calibration messages. The host should also check that there is no configuration error reported.

9.1.3 USB Interface

The host can establish contact with the chipset as specified in the USB 2.0 specification and the USB HID specification (both available from www.usb.org).

9.1.4 HID-I^{2}C Interface

On power up the $\overline{\text{CHG}}$ line will assert to indicate that there is a DIR Reset message in the input buffer, the Microsoft Windows system will enumerate the device, and the Atmel tools or customer software are free to access the configuration interface using the Microsoft Windows Inbox driver – although this is not necessary, Microsoft Windows will take care of all initialization steps needed.

9.2 Using the Object Protocol

The chipset has an object-based protocol that is used to communicate with the chipset. Typical communication includes configuring the chipset, sending commands to the chipset, and receiving messages from the chipset. Refer to the *mXT1386E Protocol Guide* for more information.

The host must perform the following initialization so that it can communicate with the chipset:

1. Read the start positions of all the objects in the chipset from the Object Table and build up a list of these addresses.
2. Use the Object Table to calculate the report IDs so that messages from the chipset can be correctly interpreted.

9.3 Writing to the Chipset

There are three mechanisms for writing to the chipset:

- Using an I^{2}C-compatible write operation (see Section 7.5.1.2 on page 41).
- Using the USB Generic HID *Read/Write Memory Map* command (see Section 7.5.1 on page 39).
- Using the Generic HID read/write command (see Section 8.5.1 on page 50).

To communicate with the chipset, you write to the appropriate object:

- To send a command to the chipset, you write the appropriate command to the Command Processor T6 object (refer to the *mXT1386E Protocol Guide*).
- To configure the chipset, you write to an object. For example, to configure the chipset power consumption you write to the global Power Configuration T7 object, and to set up a touchscreen you write to a Multiple Touch Touchscreen T9 object. Some objects are optional and need to be enabled before use. Refer to the *mXT1386E Protocol Guide* for more information on the objects.

9.4 Reading from the Chipset

Status information is stored in the Message Processor T5 object. This object can be read to receive any status information from the chipset. The I^2C-compatible interface, the HID-I^2C interface, and the USB interface all provide an interrupt-style interface for reading messages in the Message Processor T5 object.

When using the I^2C-compatible interface, the $\overline{\text{CHG}}$ line is asserted whenever a new message is available in the Message Processor T5 object (see Section 6.7 on page 31). See Section 6.5 on page 28 for information on the format of the I^2C-compatible read operation.

When using the USB interface, or the HID-I^2C interface, the configuration interfaces provide the auto-return mechanism that sends messages automatically. (See Section 7.5.2 on page 42 and Section 8.5.2 on page 52).

Note that in both cases the host should always wait to be notified of messages. The host should not poll the chipset for messages.

The USB Digitizer HID and the HID-I^2C interface provides an alternative interrupt-style mechanism for reading a subset of the touch data. See Section 7.4 on page 36 for more information. When using Microsoft Windows this is used automatically.

9.5 Configuring the Chipset

The objects are designed such that a default value of zero in their fields is a “safe” value that typically disables functionality. The objects must be configured before use and the settings written to the nonvolatile memory using the Command Processor T6 object. Refer to the *mXT1386E Protocol Guide* for more information.

Perform the following actions for each object:

1. Enable the object, if the object requires it.
2. Configure the fields in the object, as required.
3. Enable reporting, if the object supports messages, to receive messages from the object.

Refer to the *mXT1386E Protocol Guide* for more information on configuring the objects.

The following objects are read-only and require no configuration:

- Debug Objects
  - Diagnostic Debug T37
- General objects:
  - Message Processor T5
- Support objects:

- – User Data T38
  - – Message Count T44

The following objects must be configured before use:

- General objects
  - – Power Configuration T7
  - – Acquisition Configuration T8

The following objects should be checked and configured as necessary:

- General objects:
  - – Command Processor T6
- Support objects:
  - – Communications Configuration T18
  - – CTE Configuration T46

The following objects should also be enabled and configured, as required:

- Touch objects:
  - – Multiple Touch Touchscreen T9
- Signal processing objects:
  - – One-touch Gesture Processor T24
  - – Two-touch Gesture Processor T27
  - – Grip Suppression T40
  - – Touch Suppression T42
  - – Stylus T47
  - – Shieldless T56
  - – maXCharger T62
  - – Lens Bending T65
  - – Custom Gesture Processor T67
- Support objects:
  - – Digitizer HID Configuration T43
  - – Self Test T25
  - – Golden References T66
  - – Serial Data Command T68

10. Specifications

10.1 Absolute Maximum Specifications

Vdd	3.6 V
AVdd	3.6 V
DP, DM and VBUS pins	5.5 V
Max continuous pin current, any control or drive pin	20 mA
Voltage forced onto any pin	–0.5 V to (Vdd or AVdd) + 0.5 V
Configuration parameters maximum writes	10,000

⚠ **CAUTION:** Stresses beyond those listed under *Absolute Maximum Specifications* may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum specification conditions for extended periods may affect device reliability.

10.2 Recommended Operating Conditions

Operating temp	–20°C to +85°C
Storage temp	–65°C to +150°C
Vdd	3.3 V ±5%
AVdd	3.3 V ±5%
Vdd vs AVdd power sequencing	No sequencing required
Supply ripple + noise	See Section 10.14 on page 70
Cx transverse load capacitance per channel	0.63 pF to 5 pF

10.3 DC Specifications

10.3.1 Digital Power (DVdd_3V3)

Parameter	Description	Min	Typ	Max	Units	Notes
Vdd	Operating limits	3.14	3.3	3.47	V	Common to master and slaves

10.3.2 Analog Power (AVdd_3V3_A/B/C)

Parameter	Description	Min	Typ	Max	Units	Notes
AVdd	Operating limits	3.14	3.3	3.47	V	See Section 5.7.4 on page 25
Slew rate	Minimum slew rate	1			V/100 µs	

Note: AVdd must be stable and have a nominal tolerance in the host system of ±5% or better.

10.3.3 Input/Output

Ta (ambient temperature) = recommended range, unless otherwise noted

Parameter	Description	Min	Typ	Max	Units	Notes
Vil	Low input logic level	–0.3		+0.8	V	Vdd = 1.8 V to 3.3 V
Vih	High input logic level	2		3.6	V	Vdd = 1.8 V to 3.3 V
Vol	Low output voltage			0.4	V	Vdd = 1.8 V to 3.3 V
Voh	High output voltage	Vdd – 0.4			V	Vdd = 1.8 V to 3.3 V
Iil	Input leakage current			1	μA	

10.4 ESD Information

1.5 kΩ/100 pF/3 pulses		
Parameter	**Value**	**Notes**
Electrostatic Discharge Human Body Model ESD HBM	±2000V	Reference standard: MIL– STD883 Method 3015.7

10.5 Supply Current

Parameters used: XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 μs), IDLESYNCSPERX/ACTSYNCSPERX = 16, Shieldless T56 = On, Golden References T66 = On, INCBIAS =1, Lens Bending T65 = Off

10.5.1 Digital Supply – I^{2}C-compatible Interface

Vdd = 3.3V,Ta = recommended range, unless otherwise noted

Parameter	Description	Min	Typ	Max	Units	Notes
Idd	Active average supply current		26.3		mA	100 Hz, 1 touch
	Idle average supply current		4.8		mA	16 Hz, no touches
	Sleep average supply current		0.013		mA	Deep sleep

10.5.2 Digital Supply – USB Bus

Vdd = 3.3V,Ta = recommended range, unless otherwise noted

Parameter	Description	Min	Typ	Max	Units	Notes
Idd	Active average supply current		30.1		mA	100 Hz, 1 touch
	Idle average supply current		8.6		mA	16 Hz, no touches
	Sleep average supply current		4.8		mA	Deep sleep

10.5.3 Digital Supply – HID-I^{2}CInterface

Vdd = 3.3V,Ta = recommended range, unless otherwise noted

Parameter	Description	Min	Typ	Max	Units	Notes
Idd	Active average supply current		27.8		mA	100 Hz, 1 touch
	Idle average supply current		4.8		mA	16 Hz, no touches
	Sleep average supply current		1.3		mA	HID-I^{2}C sleep

10.5.4 Analog Supply – I^{2}C-compatible Interface

AVdd = 3.3V,Ta = recommended range, unless otherwise noted

Parameter	Description	Min	Typ	Max	Units	Notes
AIdd	Active average supply current		17.3		mA	100 Hz, 1 touch
	Idle average supply current		2.7		mA	16 Hz, no touches
	Sleep average supply current		0.00003		mA	Deep sleep

10.5.5 Analog Supply – USB Bus

AVdd = 3.3V,Ta = recommended range, unless otherwise noted

Parameter	Description	Min	Typ	Max	Units	Notes
AIdd	Active average supply current		17.3		mA	100 Hz, 1 touch
	Idle average supply current		2.7		mA	16 Hz, no touches
	Sleep average supply current		0.00002		mA	Deep sleep

10.5.6 Analog Supply – HID-I^{2}C Interface

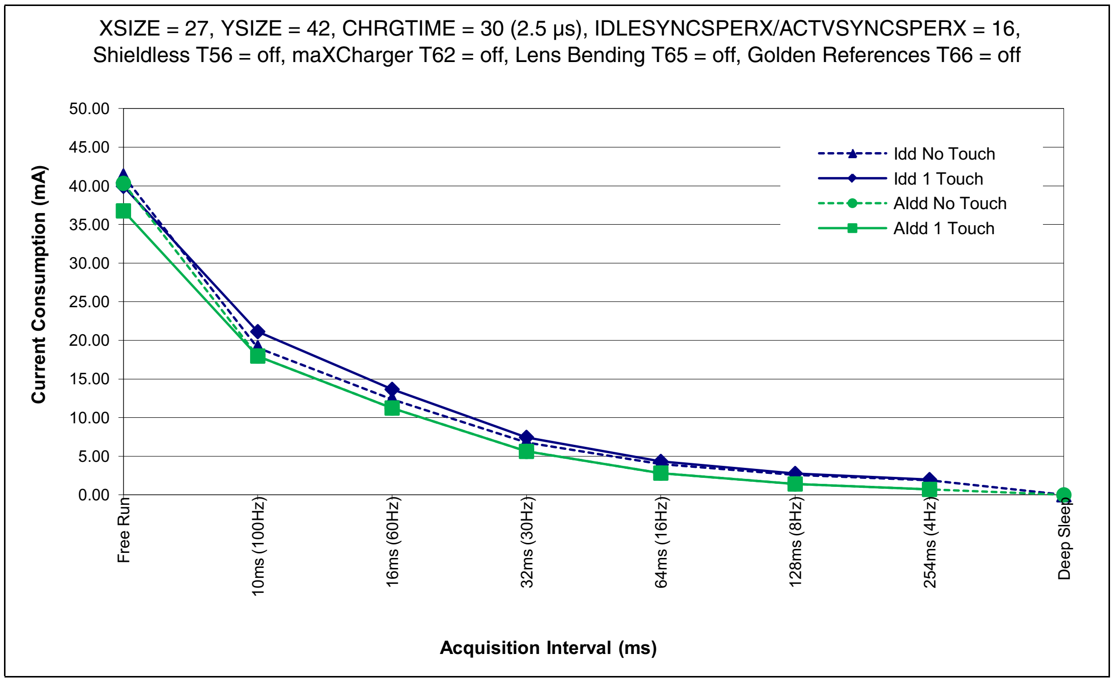
AVdd = 3.3V,Ta = recommended range, unless otherwise noted

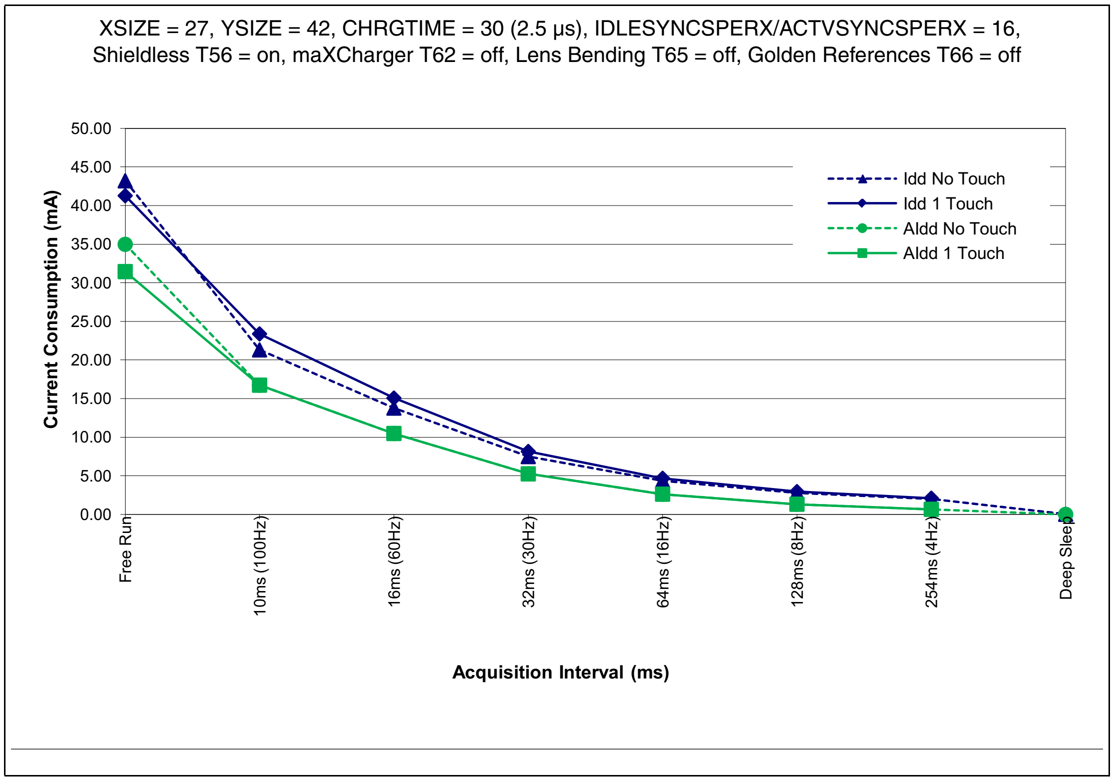
Parameter	Description	Min	Typ	Max	Units	Notes
AIdd	Active average supply current		17.3		mA	100 Hz, 1 touch
	Idle average supply current		2.7		mA	16 Hz, no touches
	Sleep average supply current		0.00003		mA	HID-I^{2}C sleep

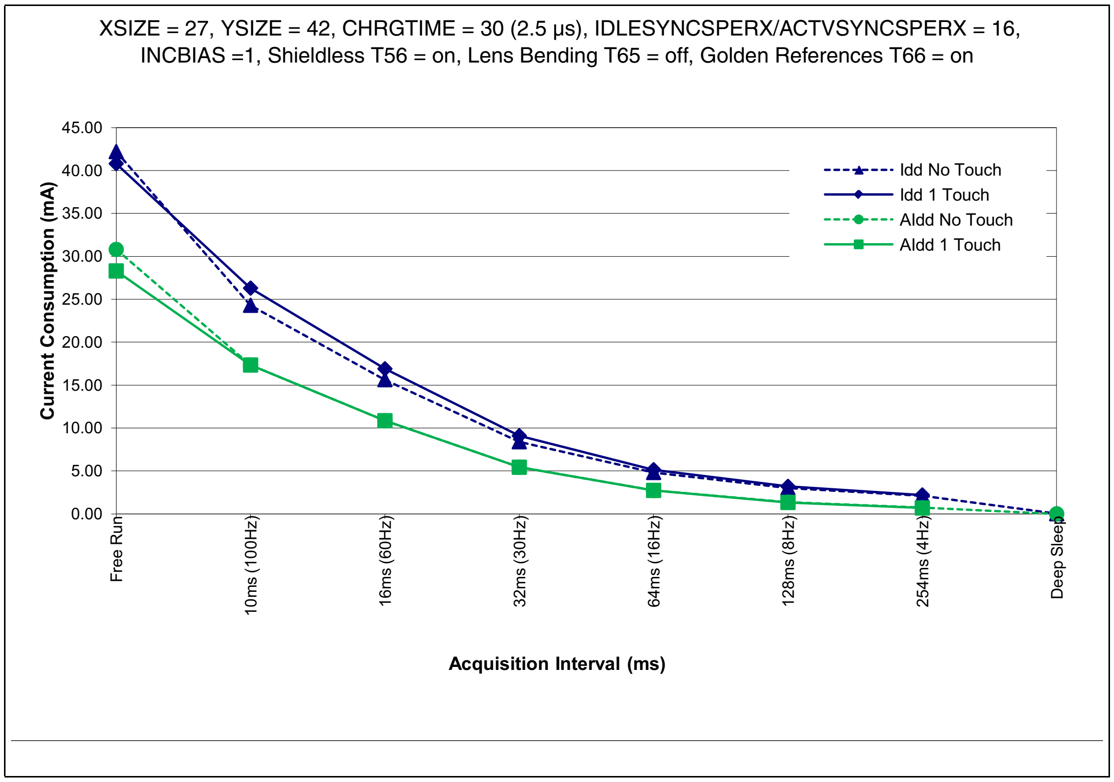
10.6 Power Consumption

Vdd = 3.3 V and AVdd = 3.3 V

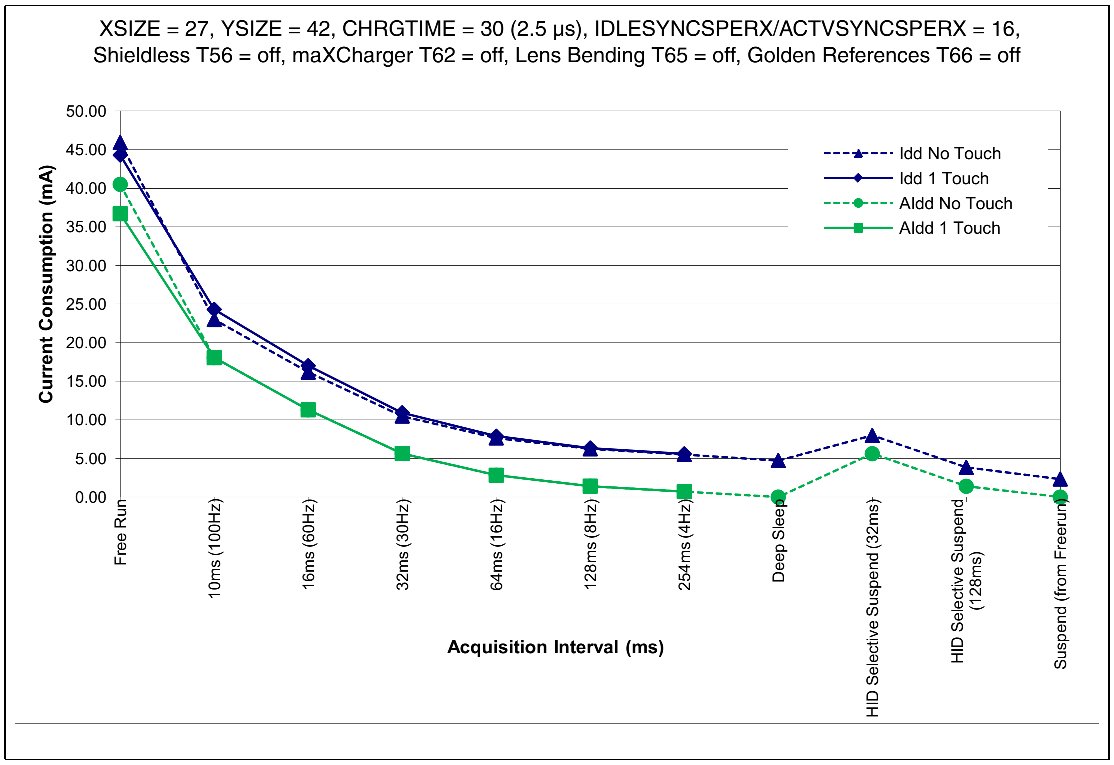
10.6.1 I^2C-compatible Interface



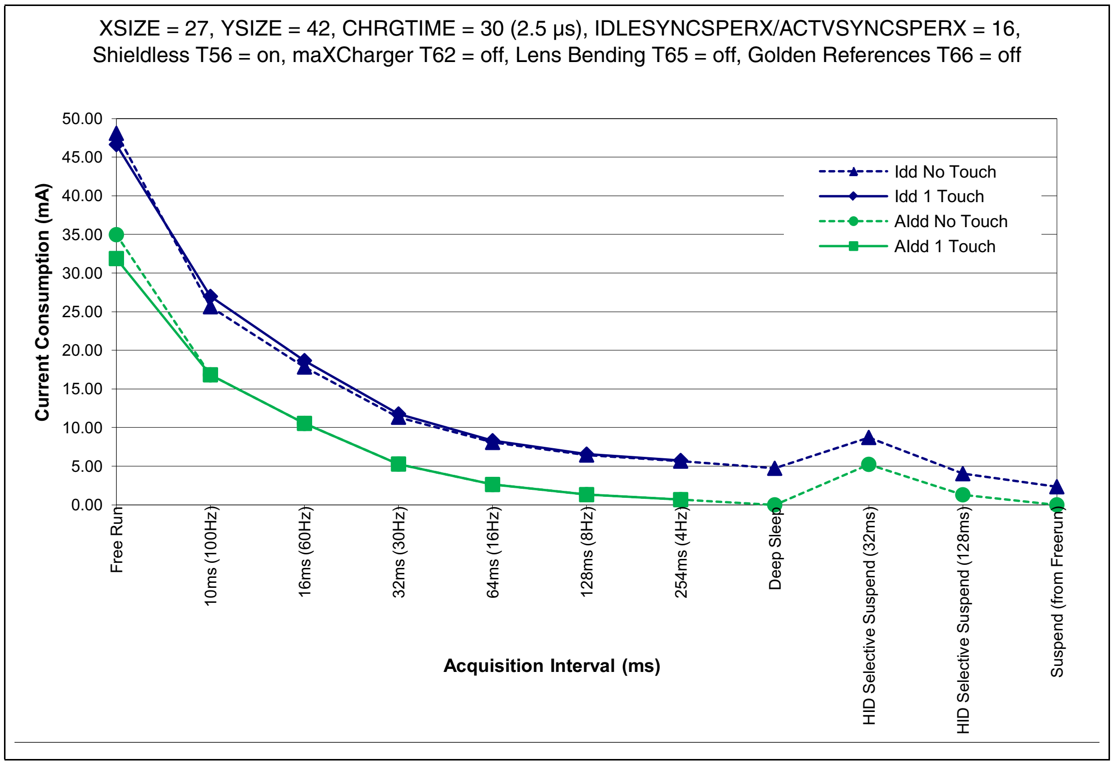


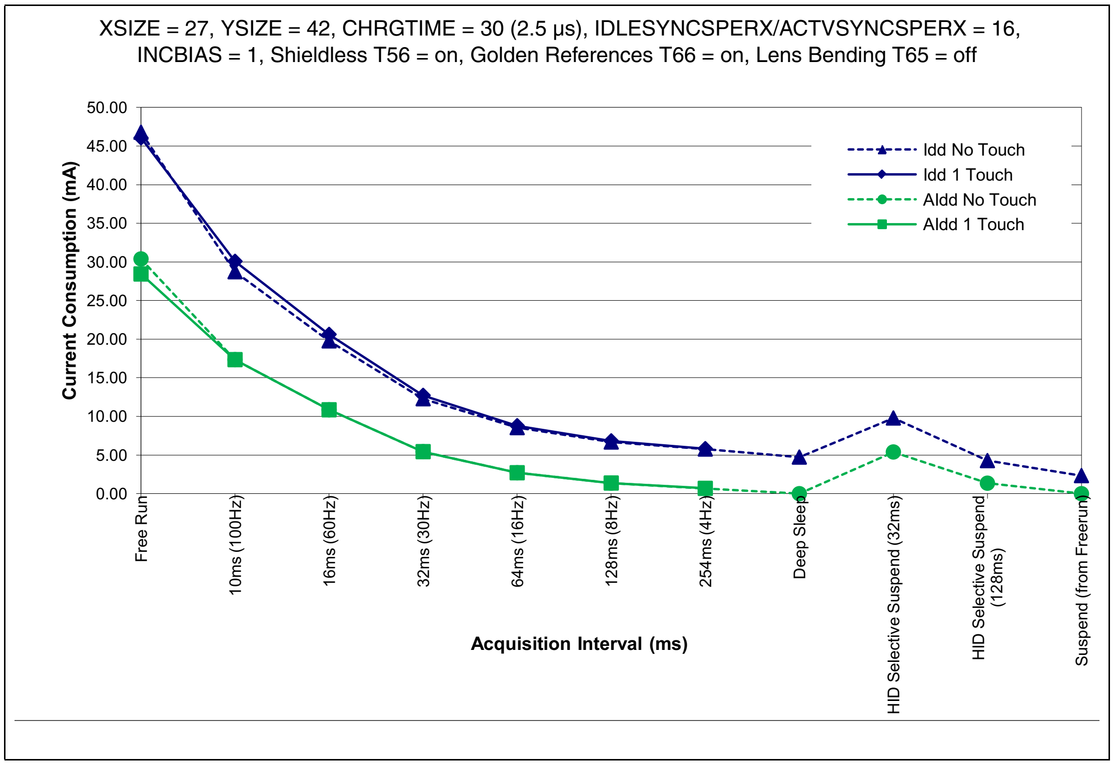


10.6.2 USB Interface

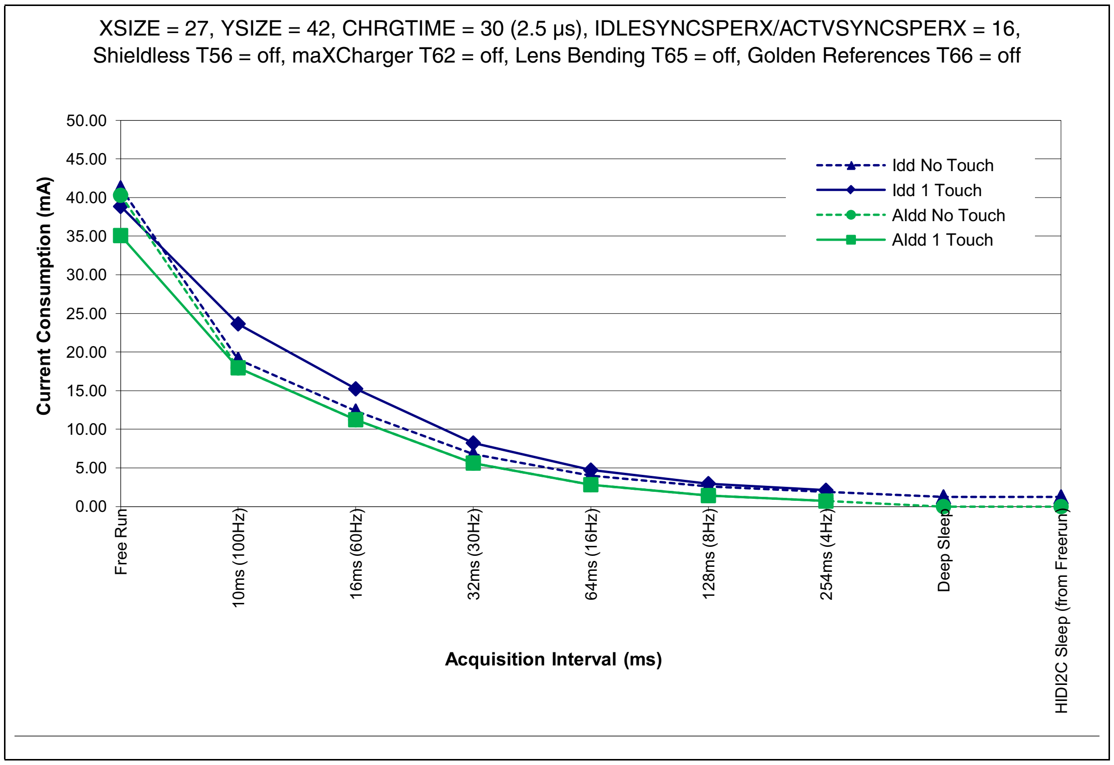


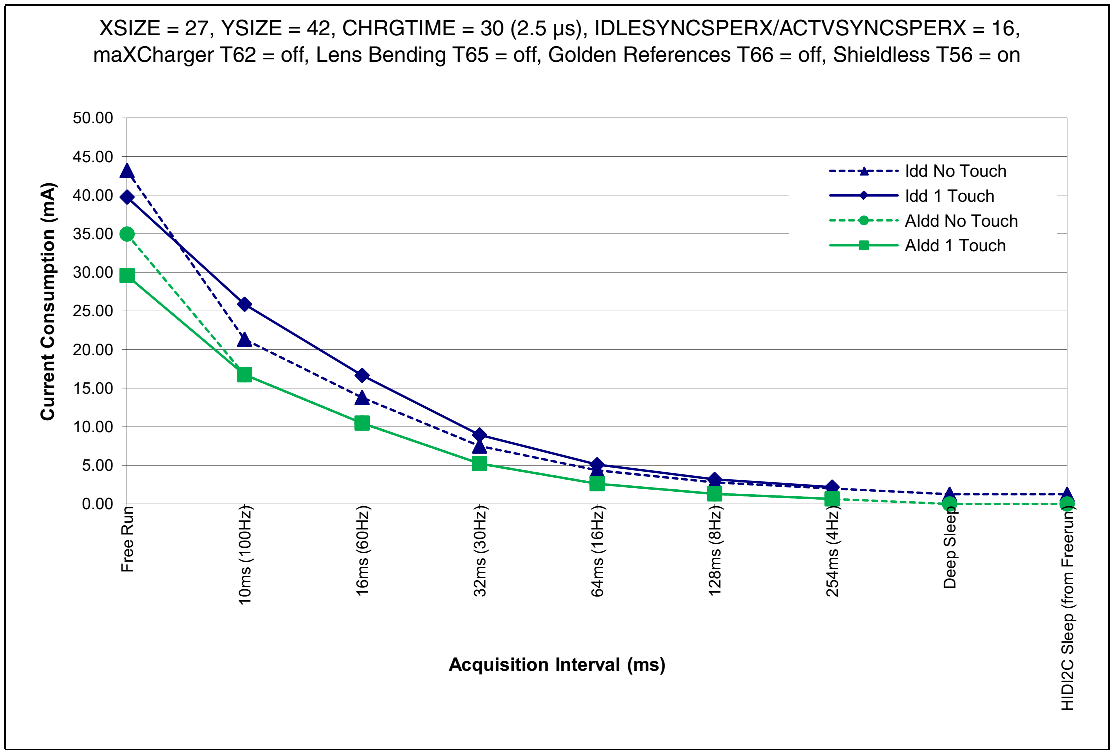
HID

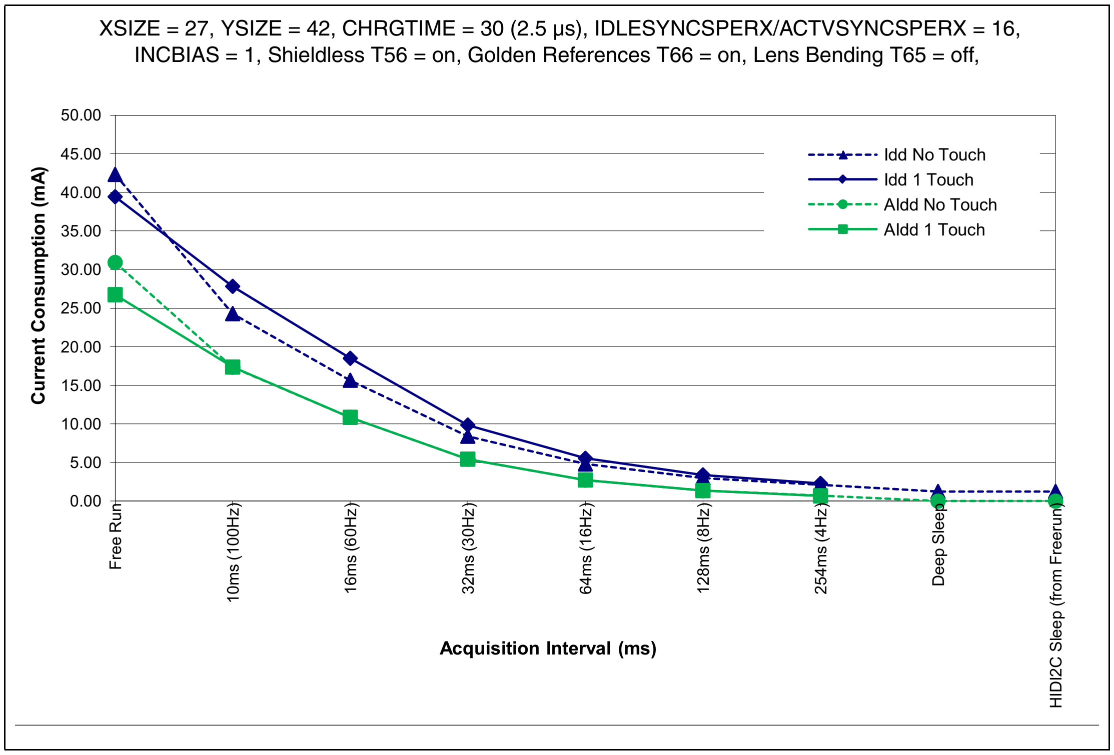

XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 µs), IDLESYNCSPERX/ACTVSYNCSPERX = 16,
Shieldless T56 = on, maXCharger T62 = off, Lens Bending T65 = off, Golden References T66 = off
Current Consumption (mA)
50.00
45.00
40.00
35.00
30.00
25.00
20.00
15.00
10.00
5.00
0.00
Idd No Touch
Idd 1 Touch
AIdd No Touch
AIdd 1 Touch
Free Run
10ms (100Hz)
16ms (60Hz)
32ms (30Hz)
64ms (16Hz)
128ms (8Hz)
254ms (4Hz)
Deep Sleep
HID Selective Suspend (32ms)
HID Selective Suspend (128ms)
Suspend (from Freerun)
Acquisition Interval (ms)


XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 µs), IDLESYNCSPERX/ACTVSYNCSPERX = 16,
INCBIAS = 1, Shieldless T56 = on, Golden References T66 = on, Lens Bending T65 = off
Current Consumption (mA)
50.00
45.00
40.00
35.00
30.00
25.00
20.00
15.00
10.00
5.00
0.00
Idd No Touch
Idd 1 Touch
AIdd No Touch
AIdd 1 Touch
Free Run
10ms (100Hz)
16ms (60Hz)
32ms (30Hz)
64ms (16Hz)
128ms (8Hz)
254ms (4Hz)
Deep Sleep
HID Selective Suspend (32ms)
HID Selective Suspend (128ms)
Suspend (from Freerun)
Acquisition Interval (ms)

10.6.3 HID-I^{2}C


XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 µs), IDLESYNCSPERX/ACTVSYNCSPERX = 16,
Shieldless T56 = off, maXCharger T62 = off, Lens Bending T65 = off, Golden References T66 = off
Current Consumption (mA)
50.00
45.00
40.00
35.00
30.00
25.00
20.00
15.00
10.00
5.00
0.00
Free Run
10ms (100Hz)
16ms (60Hz)
32ms (30Hz)
64ms (16Hz)
128ms (8Hz)
254ms (4Hz)
Deep Sleep
HIDI2C Sleep (from Freerun)
Acquisition Interval (ms)
Idd No Touch
Idd 1 Touch
AIdd No Touch
AIdd 1 Touch


XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 µs), IDLESYNCSPERX/ACTVSYNCSPERX = 16,
maXCharger T62 = off, Lens Bending T65 = off, Golden References T66 = off, Shieldless T56 = on
Current Consumption (mA)
50.00
45.00
40.00
35.00
30.00
25.00
20.00
15.00
10.00
5.00
0.00
Free Run
10ms (100Hz)
16ms (60Hz)
32ms (30Hz)
64ms (16Hz)
128ms (8Hz)
254ms (4Hz)
Deep Sleep
HIDI2C Sleep (from Freerun)
Acquisition Interval (ms)
Idd No Touch
Idd 1 Touch
AIdd No Touch
AIdd 1 Touch



10.7 Timing Specifications

Values measured at room temperature, Golden References T66 = on

Parameter	Description	Min	Typ	Max	Units	Notes
Tlatency	83 Hz	7.7		20.8	ms	IDLEACQINT/ACTVACQINT = 12
	100 Hz	7.7		18.9	ms	IDLEACQINT/ACTVACQINT = 10
	167 Hz	7.7		15.0	ms	IDLEACQINT/ACTVACQINT = 6

10.8 Reset Timings

Golden References T66 = on

10.8.1 I2C Interface

Parameter	Min	Typ	Max	Units	Notes
Power on to $\overline{\text{CHG}}$ line low		80.8		ms	
Hardware reset to $\overline{\text{CHG}}$ line low		80.8		ms	
Software reset to $\overline{\text{CHG}}$ line low		231		ms	

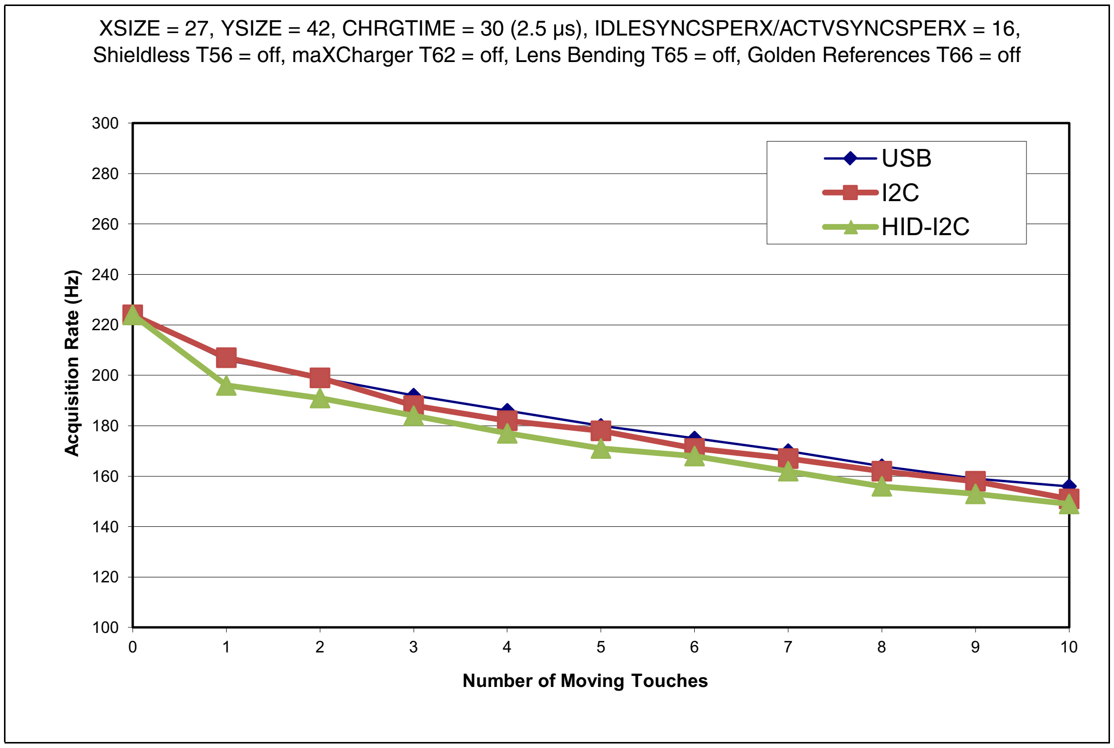
10.8.2 USB Interface

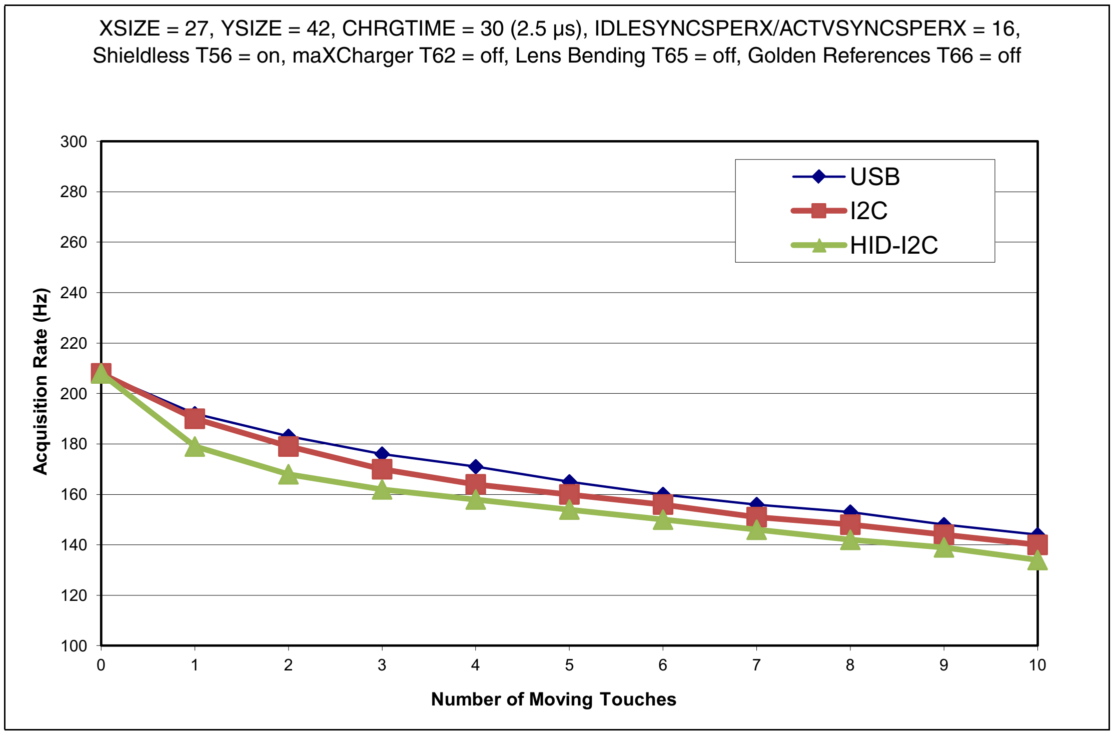
Parameter	Min	Typ	Max	Units	Notes
Power on to $\overline{\text{CHG}}$ line low		81.2		ms	
Hardware reset to $\overline{\text{CHG}}$ line low		80.8		ms	
Software reset to $\overline{\text{CHG}}$ line low		226		ms	

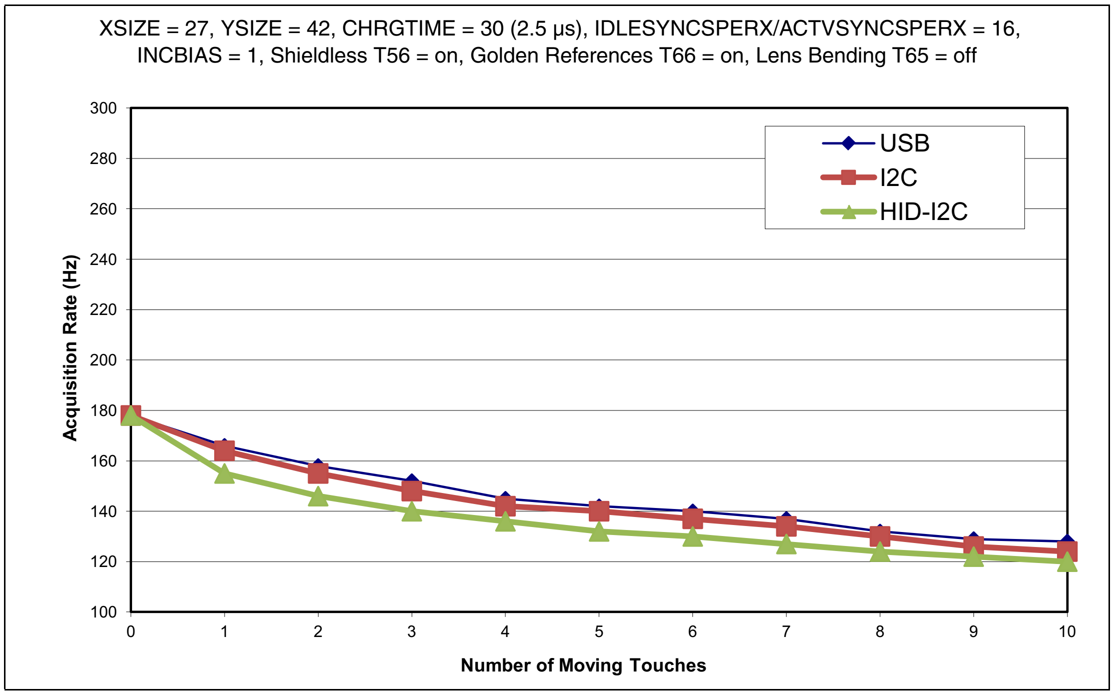
10.8.3 HID-I2C Interface

Parameter	Min	Typ	Max	Units	Notes
Power on to first X burst		85.2		ms	The maXTouch communication port is ready few ms prior to the first X burst
Hardware reset to first X burst		84.8		ms	
Software reset to first X burst		231		ms	

10.9 Speed




XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 µs), IDLESYNCSPERX/ACTVSYNCSPERX = 16,
Shieldless T56 = on, maXCharger T62 = off, Lens Bending T65 = off, Golden References T66 = off
Acquisition Rate (Hz)
Number of Moving Touches
USB
I2C
HID-I2C
300
280
260
240
220
200
180
160
140
120
100
0
1
2
3
4
5
6
7
8
9
10


XSIZE = 27, YSIZE = 42, CHRGTIME = 30 (2.5 µs), IDLESYNCSPERX/ACTVSYNCSPERX = 16,
INCBIAS = 1, Shieldless T56 = on, Golden References T66 = on, Lens Bending T65 = off
Acquisition Rate (Hz)
Number of Moving Touches
USB
I2C
HID-I2C
300
280
260
240
220
200
180
160
140
120
100
0
1
2
3
4
5
6
7
8
9
10

10.10 I^{2}C-compatible Bus Specifications

Parameter	Operation
Addresses	0x4C, 0x4D, 0x5A or 0x5B
Maximum bus speed (SCL)	100 kHz (standard mode) 400 kHz (F/S mode)
Hold time START condition	<600 ns (400 kHz) >4000 ns (100 kHz)
Setup time for STOP condition	<600 ns (400 kHz) >4000 ns (100 kHz)
SDA/SCL rise time	<300 ns (400 kHz) <1000 ns (100 kHz)
I^{2}C specification	Version 2.1

10.11 USB Bus Specification

Parameter	Operation
Endpoint Addresses	0x81 (Endpoint 1) 0x02 (Endpoint 2) 0x83 (Endpoint 3)
Maximum bus speed	12 Mbps
Vendor ID	0x03EB (Atmel)
Product ID	0x212A (mXT1386E)
USB specification	USB 2.0 HID specification 1.11 with amendments for multitouch digitizers

10.12 HID-I^{2}C Bus Specification

Parameter	Operation
Maximum bus speed	400 kHz
Vendor ID	0x03EB (Atmel)
Product ID	0x212A (mXT1386E)
HID descriptor address	0x0000
HID-I^{2}C specification	Version 0.91

10.13 Touch Accuracy and Repeatability

Touchscreen pitch= 4.7 mm, front panel = 1 mm, touch size = 8 mm

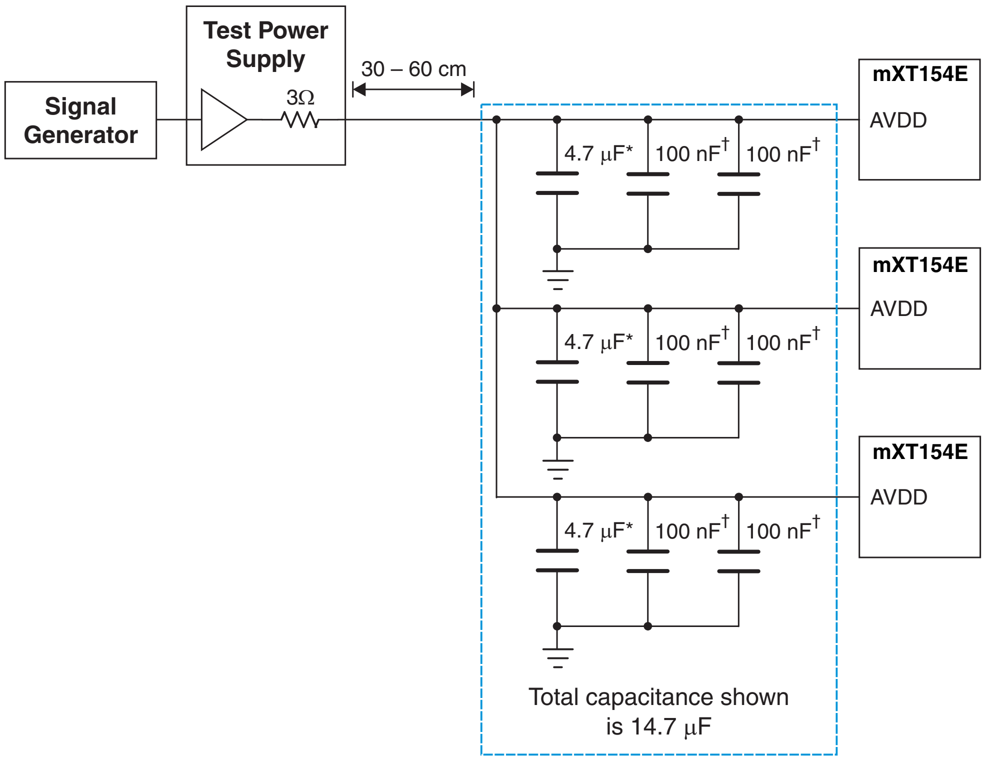
Parameter	Min	Typ	Max	Units	Notes
Linearity		±0.5		mm	
Accuracy		±1		mm	
Accuracy edge		±2		mm	
Repeatability		±0.25		mm	X axis with 12-bit resolution

10.14 Power Supply Ripple and Noise

Parameter	Min	Typ	Max	Units	Notes
Vdd			±50	mV	Across frequency range 1 Hz to 10 MHz
AVdd (Noise Suppression T48 disabled)			±25	mV	Across frequency range 1 Hz to 10 MHz
AVdd (Noise Suppression T48 enabled)			±40	mV	Across frequency range 1 Hz to 10 MHz

The test circuit used is shown in Figure 10-1.

Figure 10-1. Circuit Used for Power Supply Ripple Characterization Charts



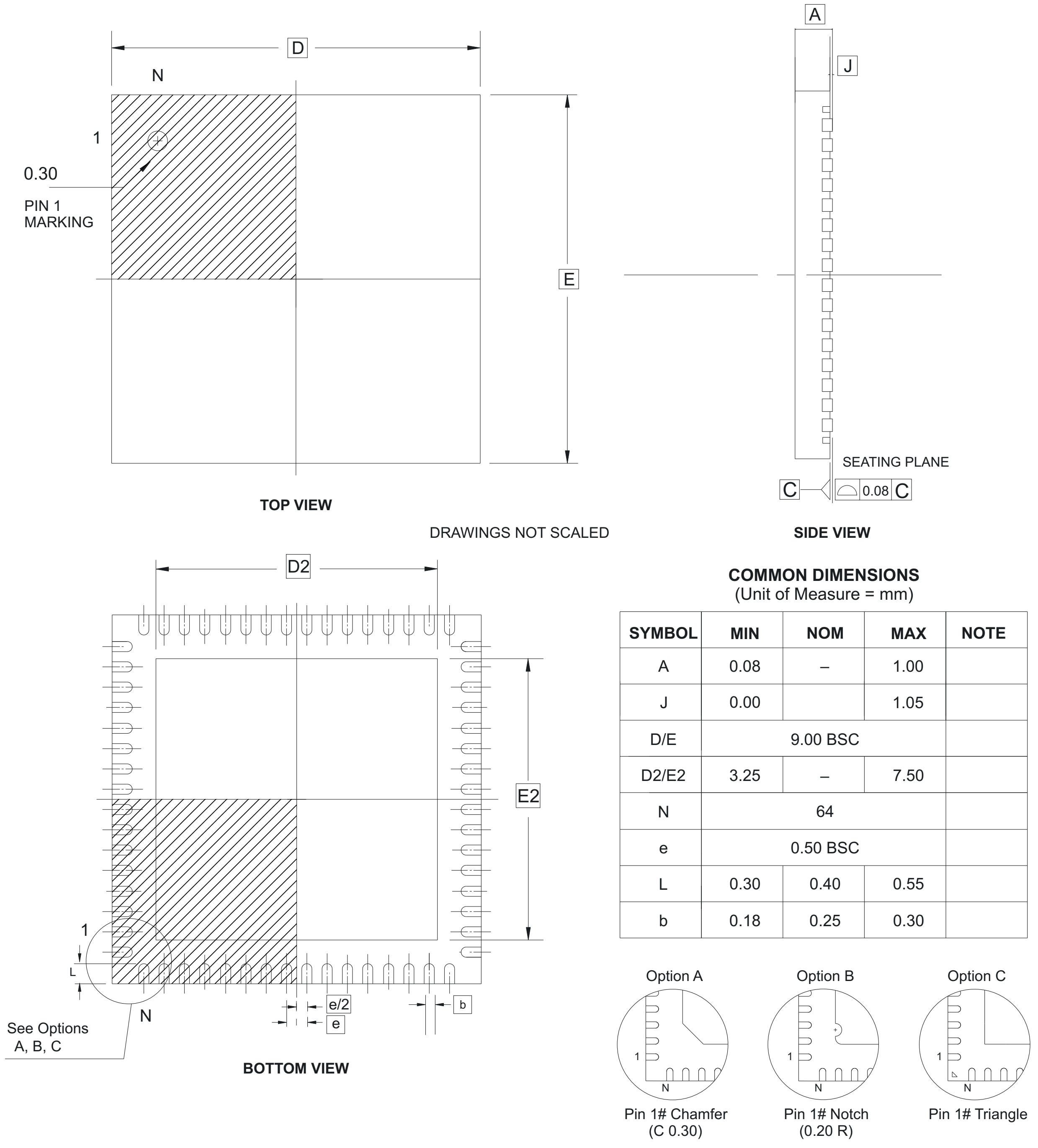
NOTES: *Bypass capacitors are <5 mm away from the chip.
† Bypass capacitors are <2 mm away from the chip.

10.15 Soldering Profile

Profile Feature	Green Package
Average Ramp-up Rate (217°C to Peak)	3°C/s max
Preheat Temperature 175°C ±25°C	150 – 200°C
Time Maintained Above 217°C	60 s – 150 s
Time within 5°C of Actual Peak Temperature	30 s
Peak Temperature Range	260°C
Ramp down Rate	6°C/s max
Time 25°C to Peak Temperature	8 minutes max

10.16 Mechanical Dimensions

10.16.1 mXT1386E-Z2U – 64-pin QFN



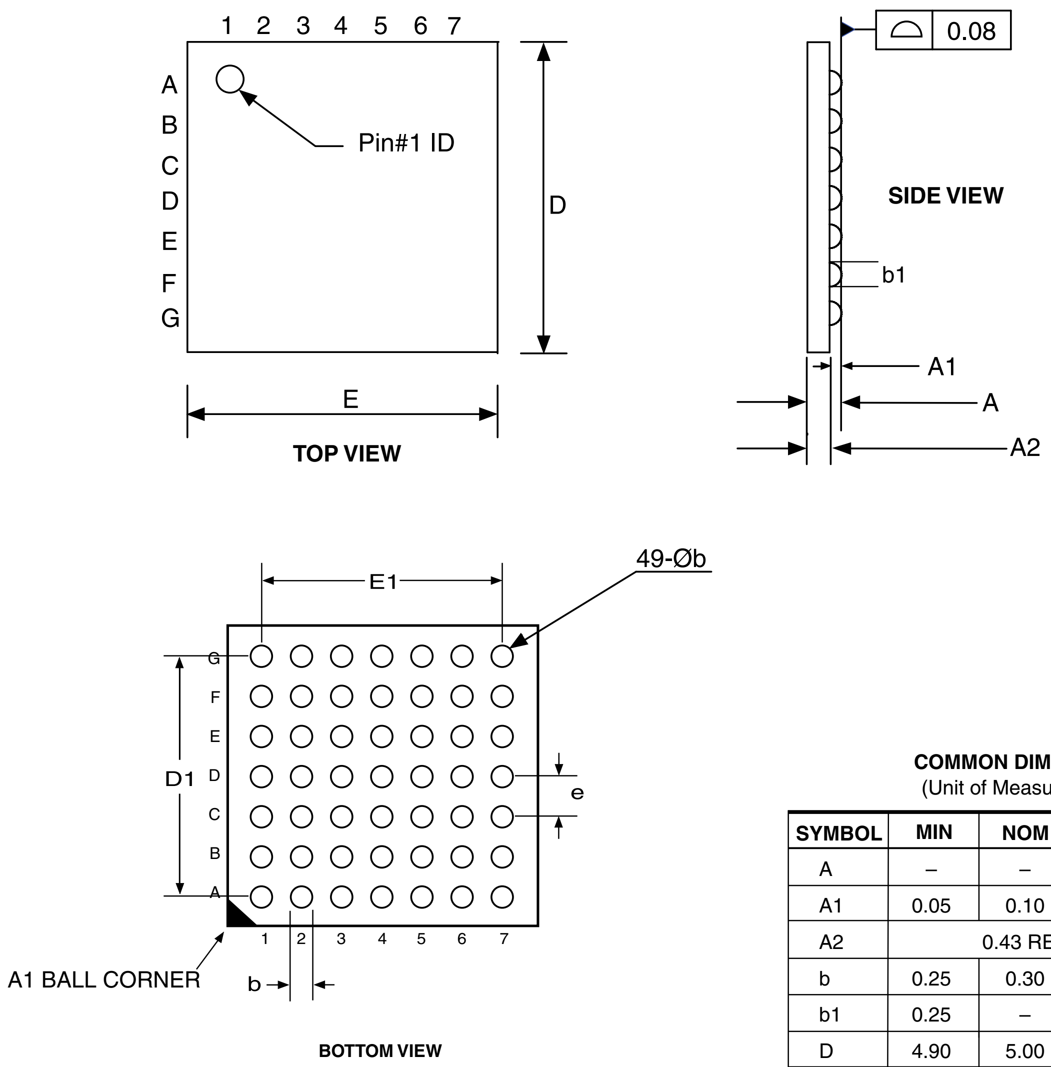
COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.08	–	1.00	
J	0.00		1.05	
D/E		9.00 BSC		
D2/E2	3.25	–	7.50	
N		64		
e		0.50 BSC		
L	0.30	0.40	0.55	
b	0.18	0.25	0.30	

6/24/10

Package Drawing Contact: touch@atmel.com	TITLE **64Z2,** 64 Leads - body 9.0 x 9.0 mm - pitch 0.5 mm Quad Flat No Lead Package (QFN)	DRAWING NO. 64Z2	REV. A

10.16.2 ATMXT154E-CCU – 49-ball UFBGA



COMMON DIMENSIONS
(Unit of Measure = mm)

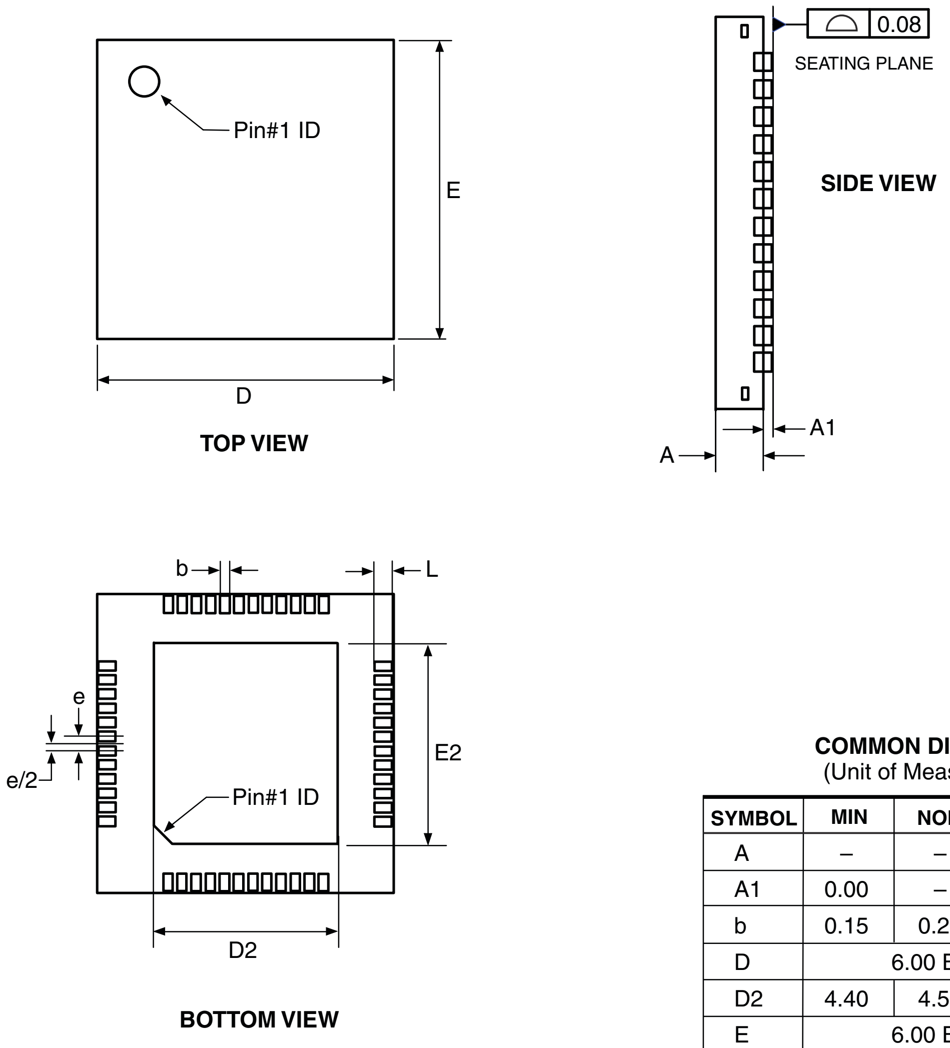
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	0.60	
A1	0.05	0.10	0.15	
A2		0.43 REF		
b	0.25	0.30	0.35	1
b1	0.25	–	–	2
D	4.90	5.00	5.10	
D1		3.90 BSC		
E	4.90	5.00	5.10	
E1		3.90 BSC		
e		0.65 BSC		

Note 1: Dimension "b" is measured at the maximum ball dia. in a plane parallel to the seating plane.
Note 2: Dimension "b1" is the solderable surface defined by the opening of the solder resist layer.

08/03/09

	TITLE	GPC	DRAWING NO.	REV.
Package Drawing Contact: packagedrawings@atmel.com	**49CU1,** 49-ball (7 x 7 Array), 5 x 5 x 0.6 mm body, lead pitch 0.65 mm, Ultra Thin, Fine-Pitch bump Ball Grid Array Package (UFBGA)	CDG	49CU1	A

10.16.3 ATMXT154E-MAH/MAHI – 48-pin QFN



COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	0.60	
A1	0.00	–	0.05	
b	0.15	0.20	0.25	2
D		6.00 BSC		
D2	4.40	4.50	4.60	
E		6.00 BSC		
E2	4.40	4.50	4.60	
e	–	0.40	–	
L	0.35	0.40	0.45	

Note1: Refer to JEDEC Drawing MO-248, variation UHHE-1 (saw singulation).

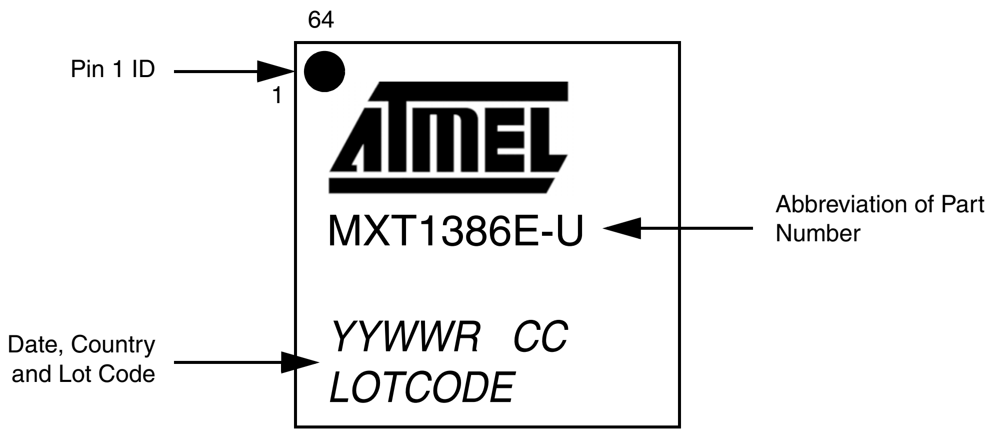
Note2: Dimension "b" refers to metalized terminal and is measured between 0.15 and 0.30mm from the terminal tip. If the terminal has the optional radius on the other end of the terminal, the dimension should not be measured in that radius area.

30/06/09

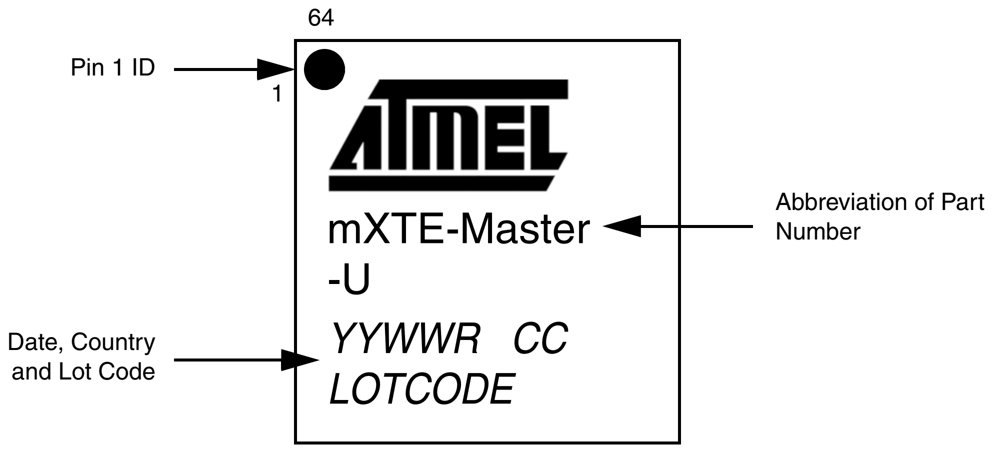
	TITLE	GPC	DRAWING NO.	REV.
Package Drawing Contact: packagedrawings@atmel.com	**48MA1,** 48 pad, 6 x 6 x 0.6mm body, 0.40mm pitch,4.5 x 4.5mm exposed pad, Saw singulated Thermally enhanced plastic Ultra thin quad flat no lead package (UQFN).	ZAL	48MA1	A

10.17 Part Markings

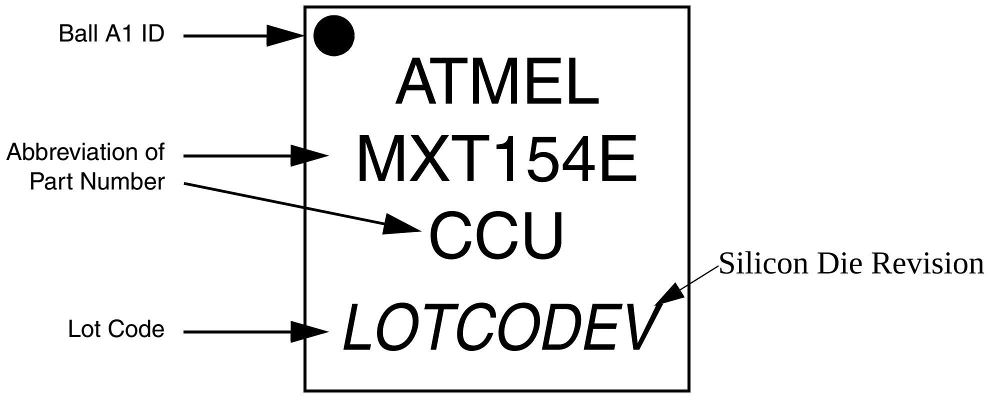
10.17.1 mXT1386E-Z2U/Z2UI – 64-pin QFN



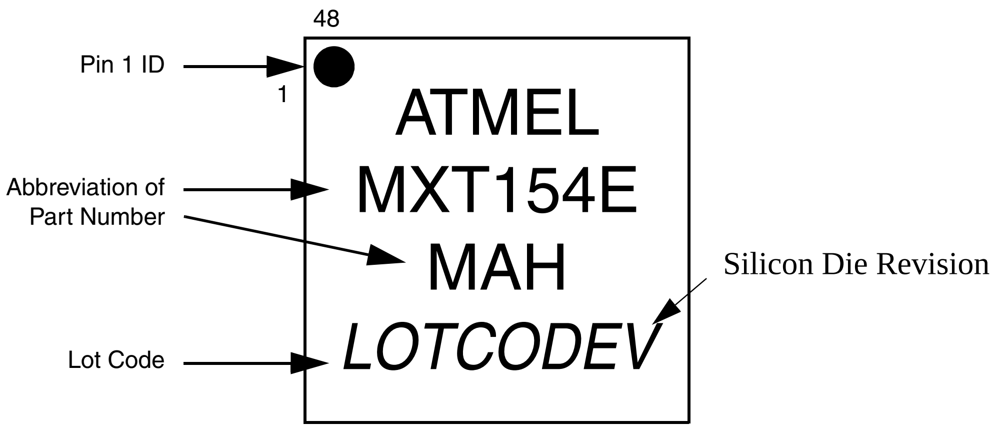
or



10.17.2 ATMXT154E-CCU/CCUI – 49-ball UFBGA



10.17.3 ATMXT154E-MAH/MAHI – 48-pin QFN



10.18 Part Numbers

Orderable Part Number	QS Number	Description
ATMXT1386E-Z2UI (supplied in trays)	QS643	64-pin 9 x 9 mm QFN RoHS compliant
ATMXT1386E-Z2UIR (supplied in tape and reels)	QS643	64-pin 9 x 9 mm QFN RoHS compliant
ATMXT154E-CCUI (supplied in trays)	QS593	49-ball 5 x 5 mm UFBGA RoHS compliant
ATMXT154E-CCUIR (supplied in tape and reels)	QS593	49-ball 5 x 5 mm UFBGA RoHS compliant
ATMXT154E-MAHI (supplied in trays)	QS593	48-pin 6 x 6 mm QFN RoHS compliant
ATMXT154E-MAHIR (supplied in tape and reels)	QS593	48-pin 6 x 6 mm QFN RoHS compliant

10.19 Moisture Sensitivity Level (MSL)

MSL Rating	Peak Body Temperature	Specifications
MSL3	260°C	IPC/JEDEC J-STD-020

Appendix A. PCB Design Considerations

A.1 Introduction

The following sections give the design considerations that should be adhered to when designing a PCB layout for use with the mXT1386E. Of these, power supply and ground tracking considerations are the most critical.

By observing the following design rules, and with careful preparation for the PCB layout exercise, designers will achieve a better and correctly-functioning product.

A.2 Printed Circuit Board

Atmel recommends the use of a four layer printed circuit board for mXT1386E applications. This, together with careful layout, will ensure that the board meets relevant EMC requirements for both noise radiation and susceptibility, as laid down by the various national and international standards agencies.

A.3 Supply Rails and Ground Tracking

Power supply and clock distribution are the most critical parts of any board layout. Because of this, it is advisable that these be completed before any other tracking is undertaken. After these, supply decoupling, and analog and high speed digital signals should be addressed. Track widths for all signals, especially power rails should be kept as wide as possible in order to reduce inductance.

The Power and Ground planes themselves can form a useful capacitor. Flood filling for either or both of these supply rails, therefore, should be used where possible. It is important to ensure that there are no floating copper areas remaining on the board: all such areas should be connected to the 0V plane. The flood filling should be done on the outside layers of the board.

In applications where the USB bus supplies power to the board, care should be taken to ensure that suitable capacitive decoupling is provided close to the USB connector. The tracking to the on-board regulators should also be kept as short as possible.

It should also be remembered that the screen of the USB cable is not intended to be connected to the ground or 0V supply of a remote device. It should either be left open circuit (being connected only at the host computer end) or decoupled with a suitable high voltage capacitor (typically 4.7 nF, 250V) and a parallel resistor (typically 1 MΩ). Note that these components may not be required when the USB cabling is internal and permanently wired, and is routed away from the noisier parts of the system.

A.4 Power Supply Decoupling

As a rule, a suitable decoupling capacitor should be placed on each and every supply pin on all digital devices. It is important that these capacitors are placed as close to the chip supply pins as possible (less than 5 mm away). The ground connection of these capacitors should be tracked to 0V by the shortest, heaviest traces possible.

Capacitors with a Type II dielectric, such as X5R or X7R and with a value of at least 100 nF, should be used for this purpose.

In addition, at least one ‘bulk’ tantalum decoupling capacitor, with a minimum value of 4.7 µF should be placed on each power rail, close to where the supply enters the board.

Surface mounting capacitors are preferred to wire leaded types due to their lower ESR and ESL. It is often possible to fit these decoupling capacitors underneath and on the opposite side of the PCB to the digital ICs. This will provide the shortest tracking, and most effective decoupling possible.

Refer to the application note *Selecting Decoupling Capacitors for Atmel PLDs* (doc0484.pdf; available on the Atmel website) for further general information on decoupling capacitors.

A.5 Suggested Voltage Regulator Manufacturers

The AVdd supply stability is critical for the mXT1386E because this supply interacts directly with the analog front end. Atmel therefore recommends that the supply for the analog section of the board be supplied by a regulator that is separate from the logic supply regulator. This reduces the amount of noise injected into the sensitive, low signal level parts of the design.

A single low value series resistor (around 1Ω) is required from the regulator output to the analog supply input on the mXT1386E device. This, together with the regulator output capacitor, and the capacitors at the DC input to the device, forms a simple filter on the supply rail.

A low noise device should be chosen for the regulator. If possible this should have provision for adding a capacitor across the internal reference for further noise reduction. Reference should be made to the manufacturer's datasheet.

The voltage regulators listed in Table 10-1 have been tested and found to work well with the mXT1386E. They have compatible footprints and pin-out specifications, and are available in the SOT-23 package.

Table 10-1. Recommended Voltage Regulators

Manufacturer	Part Number
Linear Technology	LT1761
National Semiconductor	LP2981
Micrel	MIC5255
Torex	XC6204

Note some manufacturers claim that minimal or no capacitance is required for correct regulator operation. However, in all cases, a minimum of a 1.0 µF ceramic, low ESR capacitor at the input and output of these devices should be used. The manufacturers' datasheets should always be referred to when selecting capacitors for these devices and the typical recommended values, types and dielectrics adhered to.

A.6 Crystal Oscillator

If a crystal oscillator is used, its placement is critical to the performance of the design. The connecting leads between the mXT1386E and the crystal should be as short as possible. These tracks, together with the crystal itself, should be placed above a suitable ground plane. It is also important that no other signal tracks are placed close to, or under, these tracks. The crystal input pins are at a relatively high impedance and cross-talk from other signals will seriously affect oscillator stability and accuracy. The crystal case should also be connected to ground if possible.

If an oscillator module is used, care still needs to be taken when tracking to the mXT1386E. The clock signal should be kept as short as possible, with a solid ground return underneath the clock output.

A.7 Analog I/O

In general, tracking for the analog I/O signals from the mXT1386E device should be kept as short as possible. These normally go to a connector which interfaces directly to the touchscreen.

Ensure that adequate ground-planes are used. An analog ground plane should be used in addition to a digital one. Care should be taken to ensure that both ground planes are kept separate and are connected together only at the point of entry for the power to the PCB. This is usually at the input connector.

A.8 Component Placement

It is important to orient all devices so that the tracking for important signals (such as power and clocks) are kept as short as possible. This simple point is often overlooked when initially planning a PCB layout and can save hours of work at a later stage.

A.9 Digital Signals

In general, when tracking digital signals, it is advisable to avoid sharp directional changes, sensitive signal tracks (such as analog I/O) and any clock or crystal tracking.

A good ground return path for all signals should be provided, where possible, to ensure that there are no discontinuities in the ground return path.

A.10 EMC and Other Observations

The following recommendations are not mandatory, but may help in situations where particularly difficult EMC or other problems are present:

- A small common mode choke is recommended on the differential USB data pair. This should be placed directly at the USB connector, between the connector and the relevant mXT1386E pins. Tracking lengths for the USB data pair should be kept as short as possible.
- Try to keep as many signals as possible on the inside layers of the board. If suitable ground flood fills are used on the top and bottom layers, these will provide a good level of screening for noisy signals, both into and out of the PCB.
- Ensure that the on-board regulators have sufficient tracking around and underneath the devices to act as a heatsink. This heatsink will normally be connected to the 0V or ground supply pin. Increasing the width of the copper tracking to any of the device pins will aid in removing heat. There should be no solder mask over the copper track underneath the body of the regulators.
- Ensure that the decoupling capacitors, especially tantalum, or high capacity ceramic types, have the requisite low ESR, ESL and good stability/temperature properties. Refer to the regulator manufacturer’s datasheet for more information.

Appendix B. Reference Configuration

The values listed below are used in the reference unit to validate the interfaces and derive the characterization data provided in Section 10.

The fields that are not listed have their values set to 0 (which is replaced by a default value). See *mXT1386E Protocol Guide* for information about the individual objects and their fields.

The values for the user's application will depend on the circumstances of that particular project and will vary from those listed here. Further tuning will be required to achieve an optimal performance..

Field	Value
Power Configuration T7 – GEN_POWERCONFIG_T7 (Instance 0)	
IDLEACQINT	255
ACTVACQINT	255
Acquisition Configuration T8 – GEN_ACQUISITIONCONFIG_T8 (Instance 0)	
CHRGTIME	30
TCHDRIFT	20
DRIFTST	20
ATCHCALSTHR	1
Multiple Touch Touchscreen T9 – TOUCH_MULTITOUCHSCREEN_T9 (Instance 0)	
CTRL	131
XSIZE	27
YSIZE	42
BLEN	32
TCHTHR	75
NUMTOUCH	16
Digitizer HID Configuration T43 – SPT_DIGITIZER_T43 (Instance 0)	
CTRL	4
XLENGTH	125
YLENGTH	222
RWKRATE	32
CTE Configuration T46 – SPT_CTECONFIG_T46 (Instance 0)	
CTRL	64
IDLESYNCSPERX	8
ACTVSYNCSPERX	8
Shieldless T56 – PROCI_SHIELDLESS_T56 (Instance 0)	
CTRL	2
OPTINT	1
INTTIME	60
INTDELAY[0]	25

Field	Value
INTDELAY[1]	25
INTDELAY[2]	25
INTDELAY[3]	25
INTDELAY[4]	25
INTDELAY[5]	25
INTDELAY[6]	25
INTDELAY[7]	25
INTDELAY[8]	25
INTDELAY[9]	25
INTDELAY[10]	25
INTDELAY[11]	25
INTDELAY[12]	25
INTDELAY[13]	25
INTDELAY[14]	25
INTDELAY[15]	29
INTDELAY[16]	29
INTDELAY[17]	29
INTDELAY[18]	29
INTDELAY[19]	29
INTDELAY[20]	29
INTDELAY[21]	29
INTDELAY[22]	29
INTDELAY[23]	29
INTDELAY[24]	29
INTDELAY[25]	29
INTDELAY[26]	25
NCNCL	1

Appendix C. Glossary of Terms

Channel

One of the capacitive measurement points at which the sensor controller can detect capacitive change.

Jitter

The peak-to-peak variance in the reported location for an axis when a fixed touch is applied. Typically jitter is random in nature and has a Gaussian [1] distribution, therefore measurement of peak-to-peak jitter must be conducted over some period of time, typically a few seconds. Jitter is typically measured as a percentage of the axis in question.

For example a 100 x 100 mm touchscreen that shows ±0.5% jitter in X and ±1% jitter in Y would show a peak deviation from the average reported coordinate of ±0.5 mm in X and ±1 mm in Y. Note that by defining the jitter relative to the average reported coordinate, the effects of linearity are ignored.

Linearity

The measurement of the peak-to-peak deviation of the reported touch coordinate in one axis relative to the absolute position of touch on that axis. This is often referred to as the nonlinearity. Non-linearities in either X or Y axes manifest themselves as regions where the perceived touch motion along that axis (alone) is not reflected correctly in the reported coordinate giving the sense of moving too fast or too slow. Linearity is measured as a percentage of the axis in question.

For each axis, a plot of the true coordinate versus the reported coordinate should be a perfect straight line at 45°. A non-linearity makes this plot deviate from this ideal line. It is possible to correct modest non-linearities using on-chip linearization tables, but this correction trades linearity for resolution in regions where stronger corrections are needed (because there is a stretching or compressing effect to correct the nonlinearity, so altering the resolution in these regions). Linearity is typically measured using data that has been sufficiently filtered to remove the effects of jitter. For example, a 100 mm slider with a nonlinearity of ±1% reports a position that is, at most, 1 mm away in either direction from the true position.

Resolution

The measure of the smallest movement on a slider or touchscreen in an axis that causes a change in the reported coordinate for that axis. Resolution is normally expressed in bits and tends to refer to resolution across the whole axis in question. For example, a resolution of 10 bits can resolve a movement of 0.0977 mm on a slider 100 mm long. Jitter in the reported position degrades usable resolution.

Touchscreen

A two-dimensional arrangement of electrodes whose capacitance changes when touched, allowing the location of touch to be computed in both X and Y axes. The output from the XY computation is a pair of numbers, typically 12-bits each, ranging from 0 to 4095, representing the extents of the touchscreen active region.

1. Sometimes called Bell-shaped or Normal distribution.

Resolution

The measure of the smallest movement on a slider or touchscreen in an axis that causes a change in the reported coordinate for that axis. Resolution is normally expressed in bits and tends to refer to resolution across the whole axis in question. For example, a resolution of 10 bits can resolve a movement of 0.0977 mm on a slider 100 mm long. Jitter in the reported position degrades usable resolution.

Touchscreen

A two-dimensional arrangement of electrodes whose capacitance changes when touched, allowing the location of touch to be computed in both X and Y axes. The output from the XY computation is a pair of numbers, typically 12-bits each, ranging from 0 to 4095, representing the extents of the touchscreen active region.

Two-touch Gesture

A touch gesture that consists of two simultaneous touches. The change in position of the two touches in relation to each other characterizes a specific gesture. For example, a pinch gesture is characterized by two long-duration touches that have a decreasing distance between them (that is, they are moving closer together).

Appendix D. QMatrix Primer

D.1 Acquisition Technique

QMatrix capacitive acquisition uses a series of pulses to deposit charge into a sampling capacitor, Cs. The pulses are driven on X lines from the controller. The rising edge of the pulse causes current to flow in the mutual capacitance, Cx, formed between the X line and a neighboring receiver electrode or Y line. While one X line is being pulsed, all others are grounded. This leads to excellent isolation of the particular mutual capacitances being measured [(1)], a feature that makes for good inherent touchscreen performance.

After a fixed number of pulses (known as the burst length) the sampling capacitor's voltage is measured to determine how much charge has accumulated. This charge is directly proportional to Cx and therefore changes if Cx [(2)] changes. The transmit-receive charge transfer process between the X lines and Y lines causes an electric field to form that loops from X to Y. The field itself emanates from X and terminates on Y. If the X and Y electrodes are fixed directly [(3)] to a dielectric material like plastic or glass, then this field tends to channel through the dielectric with very little leakage of the field out into free-space (that is, above the panel). Some proportion of the field does escape the surface of the dielectric, however, and so can be influenced during a touch.

When a finger is placed in close proximity (a few millimeters) or directly onto the dielectric's surface, some of this stray field and some of the field that would otherwise have propagated via the dielectric and terminated onto the Y electrode, is diverted into the finger and is conducted back to the controller chip via the human body rather than via the Y line.

This means that less charge is accumulated in Cs, and hence the terminal voltage present on Cs, after all the charge transfer pulses are complete, becomes less. In this way, the controller can measure changes in Cx during touch. This means that the measured capacitance Cx goes down during touch, because the coupled field is partly diverted by the touching object.

The spatial separation between the X and Y electrodes is significant to make the electric field to propagate well in relation to the thickness of the dielectric panel.

D.2 Moisture Resistance

A useful side effect of the QMatrix acquisition method is that placing a floating conductive element between the X and Y lines tends to increase the field coupling and so increases the capacitance Cx. This is the opposite change direction to normal touch, and so can be quite easily ignored or compensated for by the controller. An example of such floating conductive elements is the water droplets caused by condensation.

As a result, QMatrix-based touchscreens tend not to go into false detect when they are covered in small non-coalesced water droplets. Once the droplets start to merge, however, they can become large enough to bridge the field across to nearby ground return paths (for example, other X lines not currently driven, or ground paths in mechanical chassis components). When this happens, the screen's behavior can become erratic.

1. A common problem with other types of capacitive acquisition technique when used for touchscreens, is that this isolation is not so pronounced. This means that when touching one region of the screen, the capacitive signals also tend to change slightly in nearby channels too, causing small but often significant errors in the reported touch position.
2. To a first approximation.
3. Air gaps in front of QMatrix sensors massively reduce this field propagation and kill sensitivity. Normal optically clear adhesives work well to attach QMatrix touchscreens to their dielectric front panel.

There are some measures used in these controllers to help with this situation, but in general there comes a point where the screen is so contaminated by moisture that false detections become inevitable. It should also be noted that uniform condensation soon becomes non-uniform once a finger has spread it around. Finger grease renders the water highly conductive, making the situation worse overall.

In general, QMatrix has industry-leading moisture tolerance but there comes a point when even the best capacitive touchscreen suffers due to moisture on the dielectric surface.

D.3 Interference Sources

D.3.1 Power Supply

The chipset can tolerate short-term power supply fluctuations. If the power supply fluctuates slowly with temperature, the chipset tracks and compensate for these changes automatically with only minor changes in sensitivity. If the supply voltage drifts or shifts quickly, the drift compensation mechanism is not able to keep up, causing sensitivity anomalies or false detections.

The chipset itself uses the AVdd power supply as an analog reference, so the power should be very clean and come from a separate regulator. A standard inexpensive Low Dropout (LDO) type regulator should be used that is not also used to power other loads, such as LEDs, relays, or other high current devices. Load shifts on the output of the LDO can cause AVdd to fluctuate enough to cause false detection or sensitivity shifts. The digital Vdd supply is far more tolerant to noise.

CAUTION: A regulator IC shared with other logic can result in erratic operation and is not advised.

Noise on AVdd can appear directly in the measurement results. Vdd should be checked to ensure that it stays within specification in terms of noise, across a whole range of product operating conditions.

Ceramic bypass capacitors on AVdd and Vdd, placed very close (<5 mm) to the chip are recommended. A bulk capacitor of at least 1 µF and a higher frequency capacitor of around 10 nF to 100 nF in parallel are recommended; both must be X7R or X5R dielectric capacitors.

D.3.2 Other Noise Sources

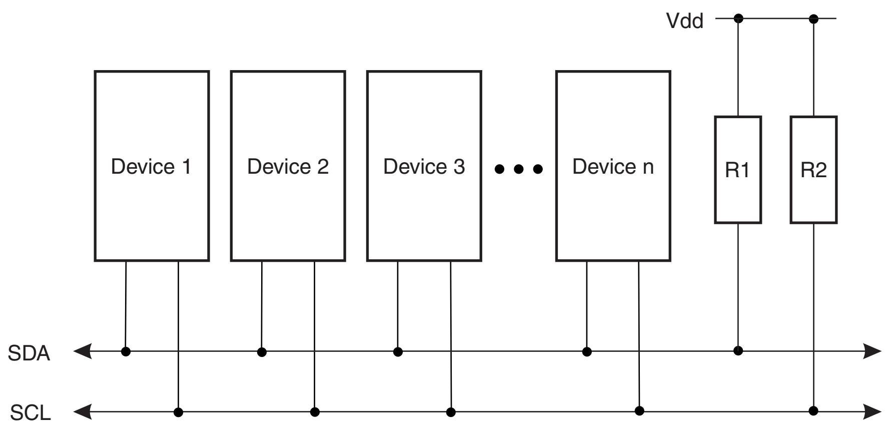
Refer to QTAN0079, *Buttons, Sliders and Wheels Sensor Design Guide*, for information (downloadable from the Touch Technology area of the Atmel website).

Appendix E. I²C Basics (I²C-compatible Operation)

10.20 Interface Bus

The device communicates with the host over an I²C bus. The following sections give an overview of the bus; more detailed information is available from www.i2C-bus.org. Devices are connected to the I²C bus as shown in Figure E-1. Both bus lines are connected to Vdd via pull-up resistors. The bus drivers of all I²C devices must be open-drain type. This implements a wired AND function that allows any and all devices to drive the bus, one at a time. A low level on the bus is generated when a device outputs a zero.

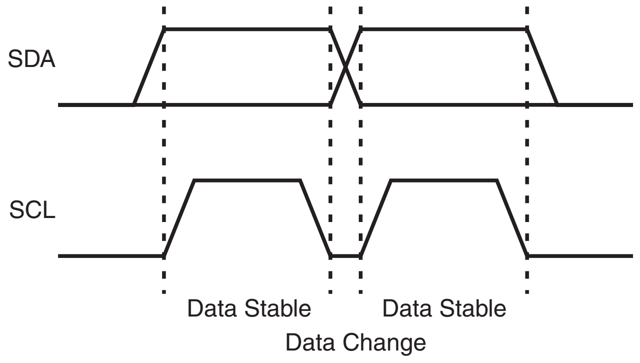
Figure E-1. I²C Interface Bus



E.1 Transferring Data Bits

Each data bit transferred on the bus is accompanied by a pulse on the clock line. The level of the data line must be stable when the clock line is high; the only exception to this rule is for generating START and STOP conditions.

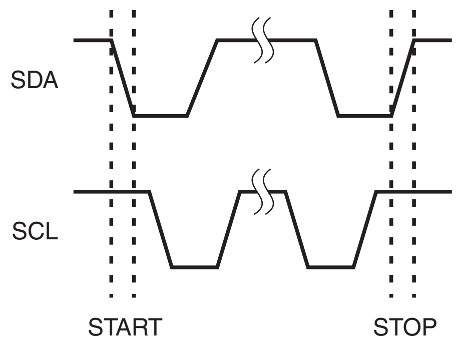
Figure E-2. Data Transfer



E.2 START and STOP Conditions

The host initiates and terminates a data transmission. The transmission is initiated when the host issues a START condition on the bus, and is terminated when the host issues a STOP condition. Between the START and STOP conditions, the bus is considered busy. As shown in Figure E-3, START and STOP conditions are signaled by changing the level of the SDA line when the SCL line is high.

Figure E-3. START and STOP Conditions

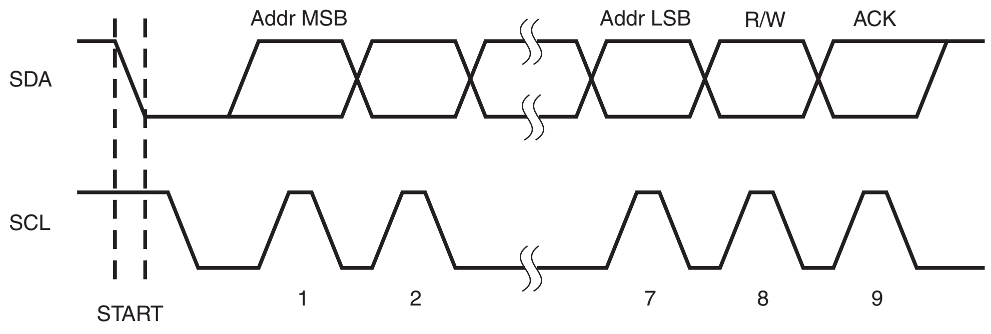


E.3 Address Byte Format

All address bytes are 9 bits long, consisting of 7 address bits, one READ/WRITE control bit and an acknowledge bit. If the READ/WRITE bit is set, a read operation is performed, otherwise a write operation is performed. When the device recognizes that it is being addressed, it will acknowledge by pulling SDA low in the ninth SCL (ACK) cycle. An address byte consisting of a slave address and a READ or a WRITE bit is called SLA+R or SLA+W, respectively.

The most significant bit of the address byte is transmitted first. The address sent by the host must be consistent with that selected with the option jumpers.

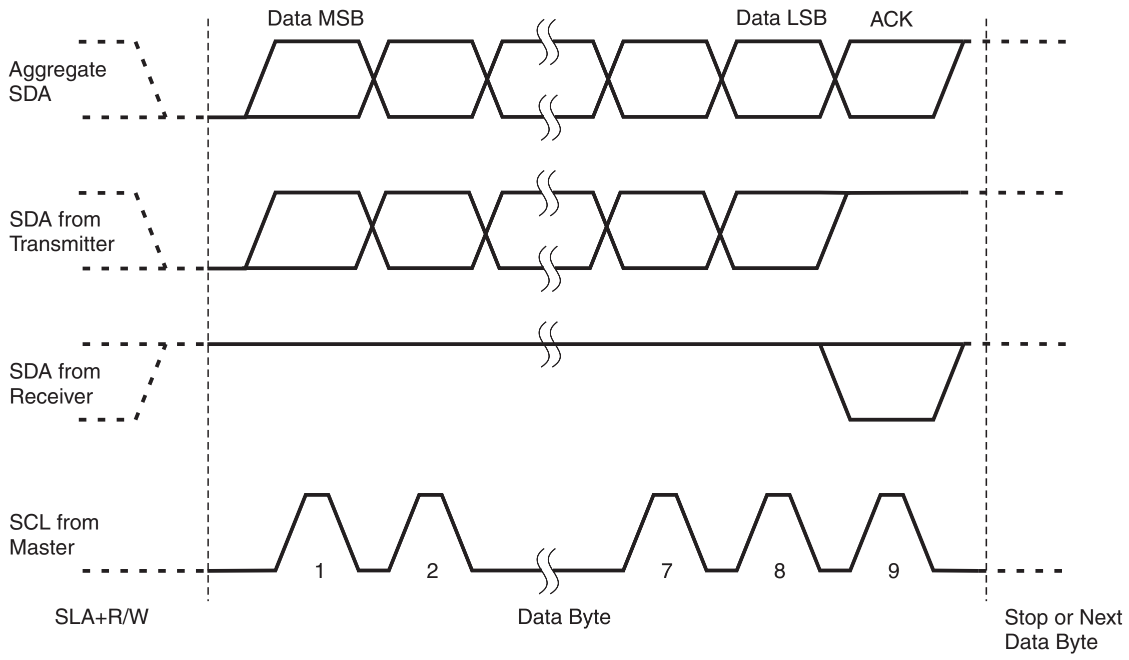
Figure E-4. Address Byte Format



E.4 Data Byte Format

All data bytes are 9 bits long, consisting of 8 data bits and an acknowledge bit. During a data transfer, the host generates the clock and the START and STOP conditions, while the receiver is responsible for acknowledging the reception. An acknowledge (ACK) is signaled by the receiver pulling the SDA line low during the ninth SCL cycle. If the receiver leaves the SDA line high, a NACK is signaled.

Figure E-5. Data Byte Format



E.5 Combining Address and Data Bytes into a Transmission

A transmission consists of a START condition, an SLA+R/W, one or more data bytes and a STOP condition. The wired "ANDing" of the SCL line is used to implement handshaking between the host and the device. The device extends the SCL low period by pulling the SCL line low whenever it needs extra time for processing between the data transmissions.

Note: Each write or read cycle must end with a stop condition. The device may not respond correctly if a cycle is terminated by a new start condition.

Figure E-6 shows a typical data transmission. Note that several data bytes can be transmitted between the SLA+R/W and the STOP.

Figure E-6. Byte Transmission

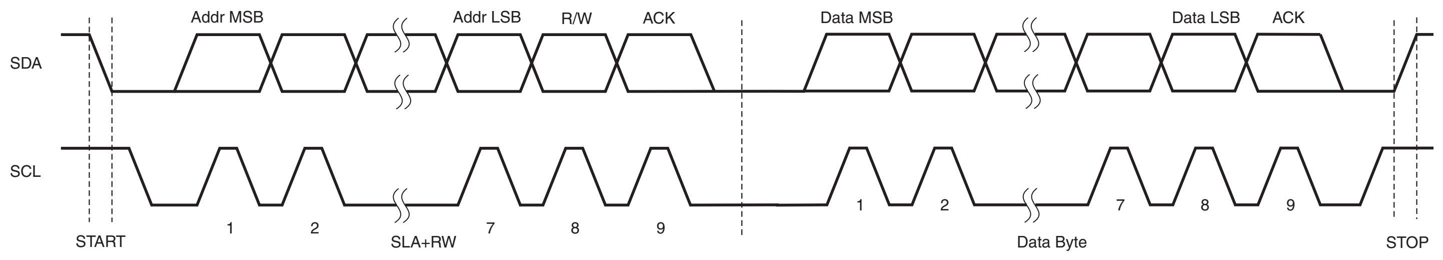


Table of Contents

Revision History

Revision Number	History
Revision AX – August 2012	Initial release for firmware revision 2.9
Revision BX – September 2012	Minor updates to firmware revision 2.9 • Added power consumption figures • Updated reset timings • Updated latency figures • Updated speed charts

Headquarters

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
USA
Tel: (+1) (408) 441-0311
Fax: (+1) (408) 436-4314

International

Atmel Asia
Unit 01-05 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong
Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Touch Technology Division
1560 Parkway
Solent Business Park
Whiteley
Fareham
Hampshire
PO15 7AG
UNITED KINGDOM
Tel: (+44) (0) 844 894 1920
Fax: (+44) (0) 1489 557 066

Atmel Munich GmbH
Business Campus
Parkring 4
D- 85748 Garching b.
MUNICH
Tel: (+49) 89-31970-111
Fax: (+49) 89-3194621

Atmel Japan
16F Shin-Osaki Kangyo Building
1-6-4 Osaki
Shinagawa-ku, Tokyo 141-0032
JAPAN
Tel: (+81) (3) 6417-0300
Fax: (+81) (3) 6417-0370

Product Contact

Web Site
www.atmel.com

Technical Support
touch@atmel.com

Sales Contact
www.atmel.com/contacts