Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Atmel AT42QT1111-MU
AT42QT1111-AU

11-key QTouch® Touch Sensor IC

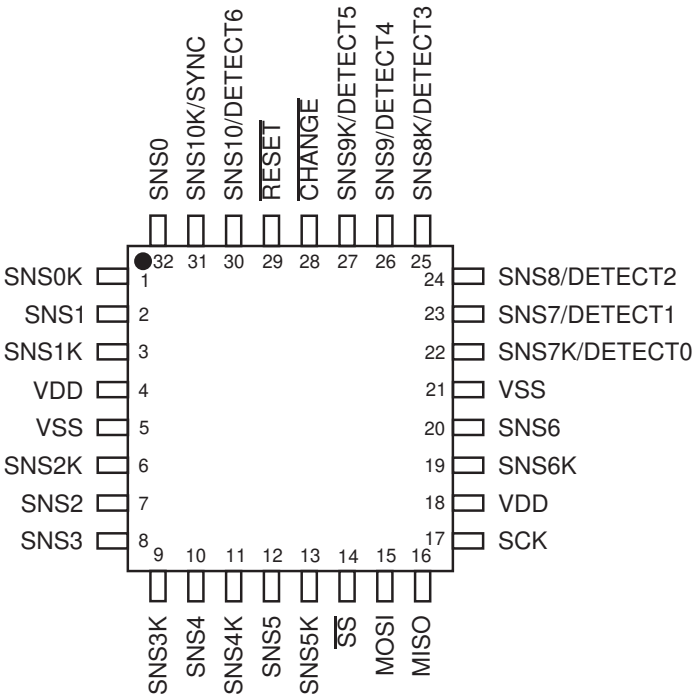
DATASHEET

Features

- Sensor Keys:
  - Up to 11 QTouch® channels
- Data Acquisition:
  - Measurement of keys triggered either by a signal applied to the SYNC pin or at regular intervals timed by the AT42QT1111 internal clock
  - Keys measured sequentially for better performance, or in parallel groups for faster operation
  - Raw data for key touches can be read as a report over the SPI interface
- Discrete Outputs:
  - Configurable “Detect” outputs indicating individual key touch (7-key mode)
- Device Setup:
  - Device configuration can be stored in EEPROM
- Technology:
  - Patented spread-spectrum charge-transfer (direct mode)
- Key Outline Sizes:
  - 6 mm × 6 mm or larger (panel thickness dependent); widely different sizes and shapes possible, including solid or ring shapes
- Key Spacings:
  - 7 mm center-to-center or more (panel thickness dependent)
- Layers Required:
  - One
- Electrode Materials:
  - Etched copper, silver, carbon, Indium Tin Oxide (ITO)
- Electrode Substrates:
  - PCB, FPCB, plastic films, glass
- Panel Materials:
  - Plastic, glass, composites, painted surfaces (low particle density metallic paints possible)
- Panel Thickness:
  - Up to 10 mm glass, 5 mm plastic (electrode size dependent)
- Key Sensitivity:
  - Individually settable via simple commands over serial interface
- Adjacent Key Suppression® (AKS®)
  - Patented AKS technology to enable accurate key detection
- Interface:
  - Full-duplex SPI slave mode (750 kHz), CHANGE pin, discrete detection outputs
- Moisture Tolerance
  - Increased moisture tolerance based on hardware design and firmware tuning
- Power:
  - 1.8 V – 5.5 V
- Package:
  - 32-pin 5 × 5 mm QFN RoHS compliant
  - 32-pin 7 × 7 mm TQFP RoHS compliant
- Signal Processing:
  - Self-calibration, auto drift compensation, noise filtering, AKS technology
- Applications:
  - Consumer and industrial applications, such as TV, media player

1. Pinout and Schematic

1.1 Pinout Configuration



1.2 Pin Descriptions

Table 1-1. Pin Listing

Pin	Name	Type	Comments	If Unused, Connect To...
1	SNS0K	I/O	Sense Pin	Leave open
2	SNS1	I/O	Sense Pin	Leave open
3	SNS1K	I/O	Sense Pin	Leave open
4	Vdd	P	Power	–
5	Vss	P	Supply Ground	–
6	SNS2K	I/O	Sense Pin	Leave open
7	SNS2	I/O	Sense Pin	Leave open
8	SNS3	I/O	Sense Pin	Leave open
9	SNS3K	I/O	Sense Pin	Leave open
10	SNS4	I/O	Sense Pin	Leave open
11	SNS4K	I/O	Sense Pin	Leave open

Table 1-1. Pin Listing

Pin	Name	Type	Comments	If Unused, Connect To...
12	SNS5	I/O	Sense Pin	Leave open
13	SNS5K	I/O	Sense Pin	Leave open
14	$\overline{SS}$	I	Enable SPI	Vss via 100 kΩ resistor to enable SPI Vdd via 100 kΩ resistor to disable SPI
15	MOSI	I	SPI Data In	Leave open
16	MISO	O	SPI Data Out	Leave open
17	SCK	I	SPI Clock	Leave open
18	Vdd	P	Power	–
19	SNS6K	I/O	Sense Pin	Leave open
20	SNS6	I/O	Sense Pin	Leave open
21	Vss	P	Supply Ground	–
22	SNS7K/DETECT0	I/O	Sense Pin/Key Status Indicator	Leave open
23	SNS7/DETECT1	I/O	Sense Pin/Key Status Indicator	Leave open
24	SNS8/DETECT2	I/O	Sense Pin / Key Status Indicator	Leave open
25	SNS8K/DETECT3	I/O	Sense Pin / Key Status Indicator	Leave open
26	SNS9/DETECT4	I/O	Sense Pin / Key Status Indicator	Leave open
27	SNS9K/DETECT5	I/O	Sense Pin / Key Status Indicator	Leave open
28	$\overline{CHANGE}$	OD	Touch Event Indicator	Leave open
29	$\overline{RESET}$	I	Reset	Vdd
30	SNS10/DETECT6	I/O	Sense Pin / Key Status Indicator	Leave open
31	SNS10K/SYNC	I/O	Sense Pin / Synchronization Input	Vdd or Vss via 100 kΩ resistor
32	SNS0	I/O	Sense Pin	Leave open

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

1.3 Schematics

Figure 1-1. Typical Circuit: 7 keys With Detect Outputs and No External Trigger

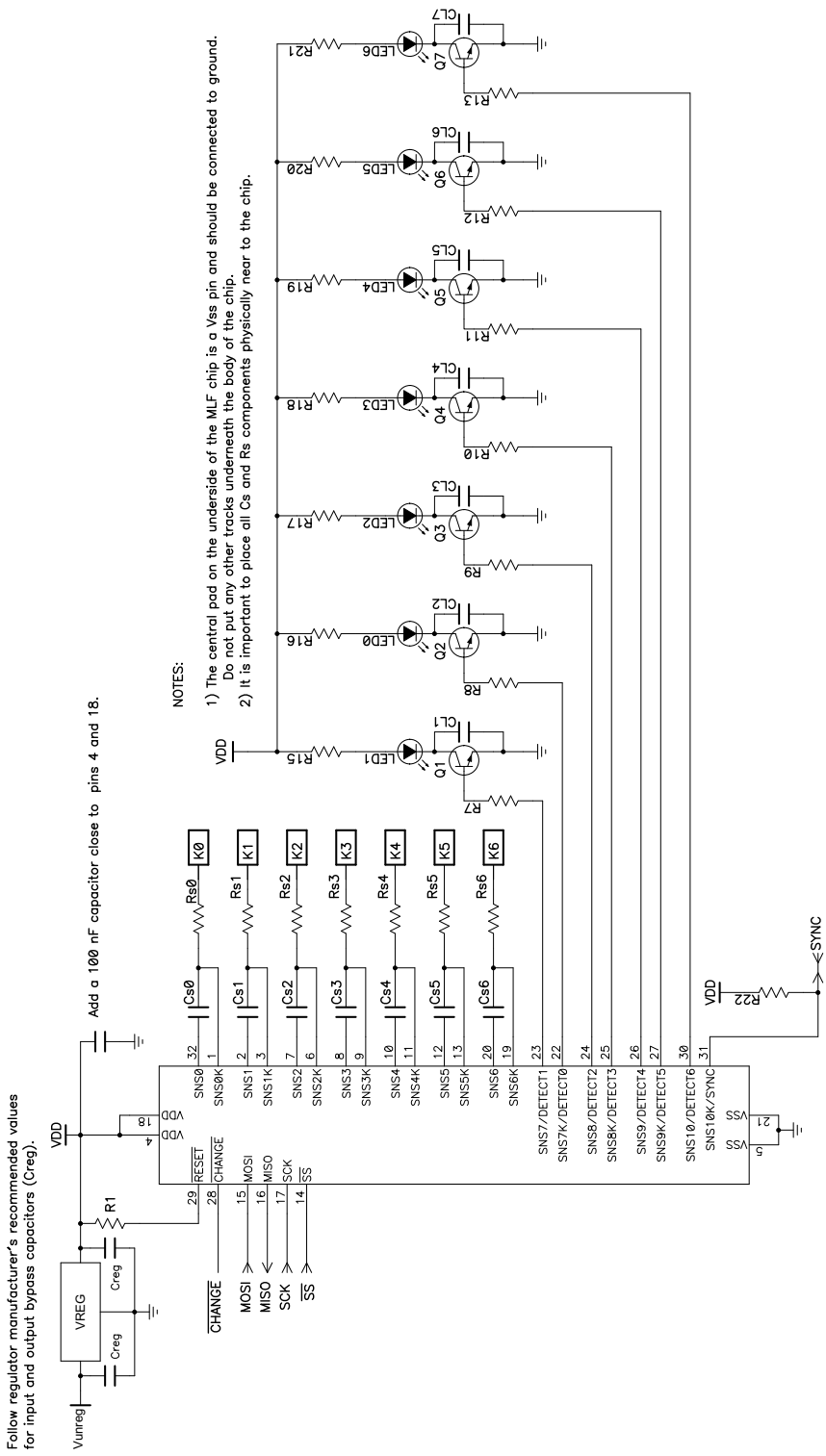


Figure 1-2. Typical Circuit: 11 Keys With No External Trigger

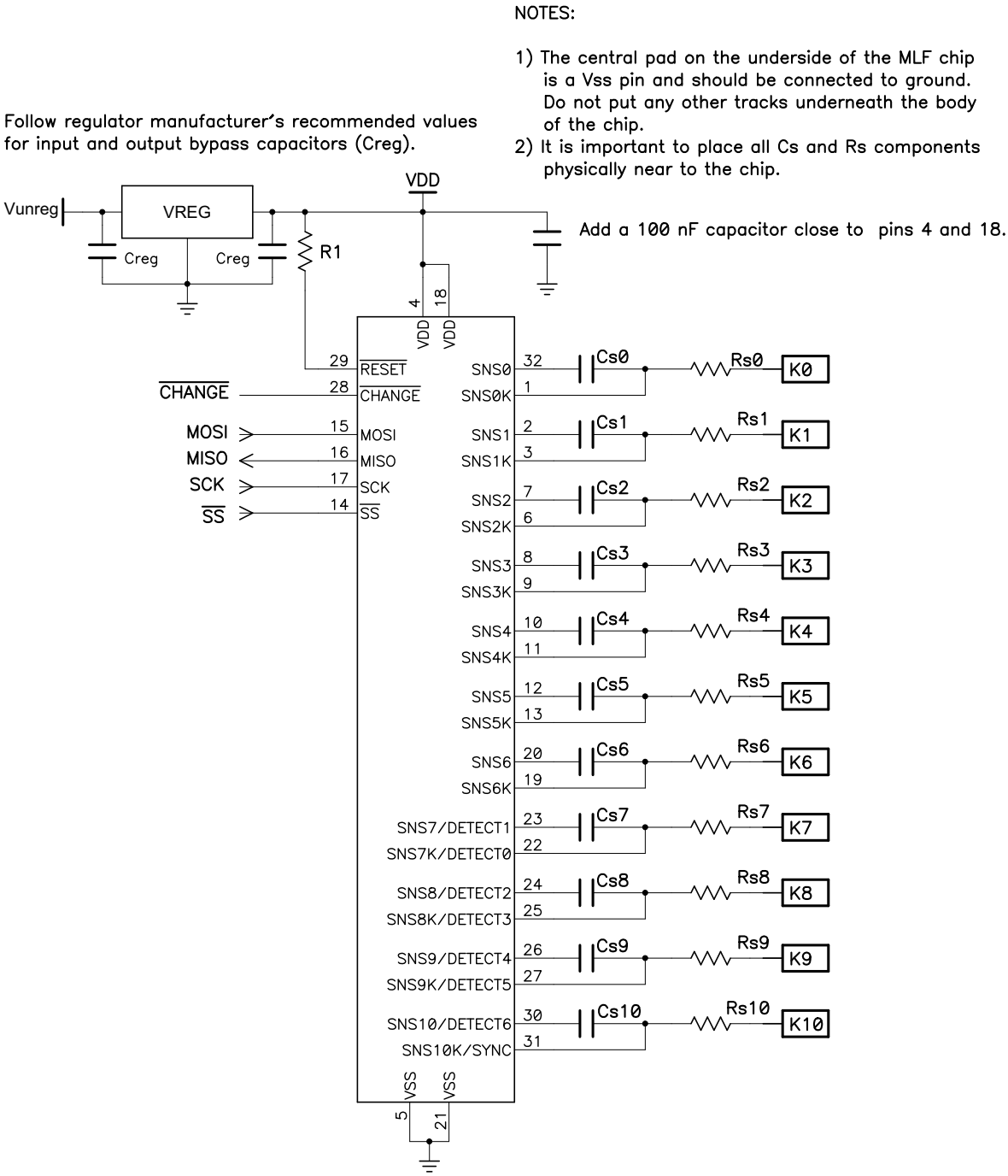
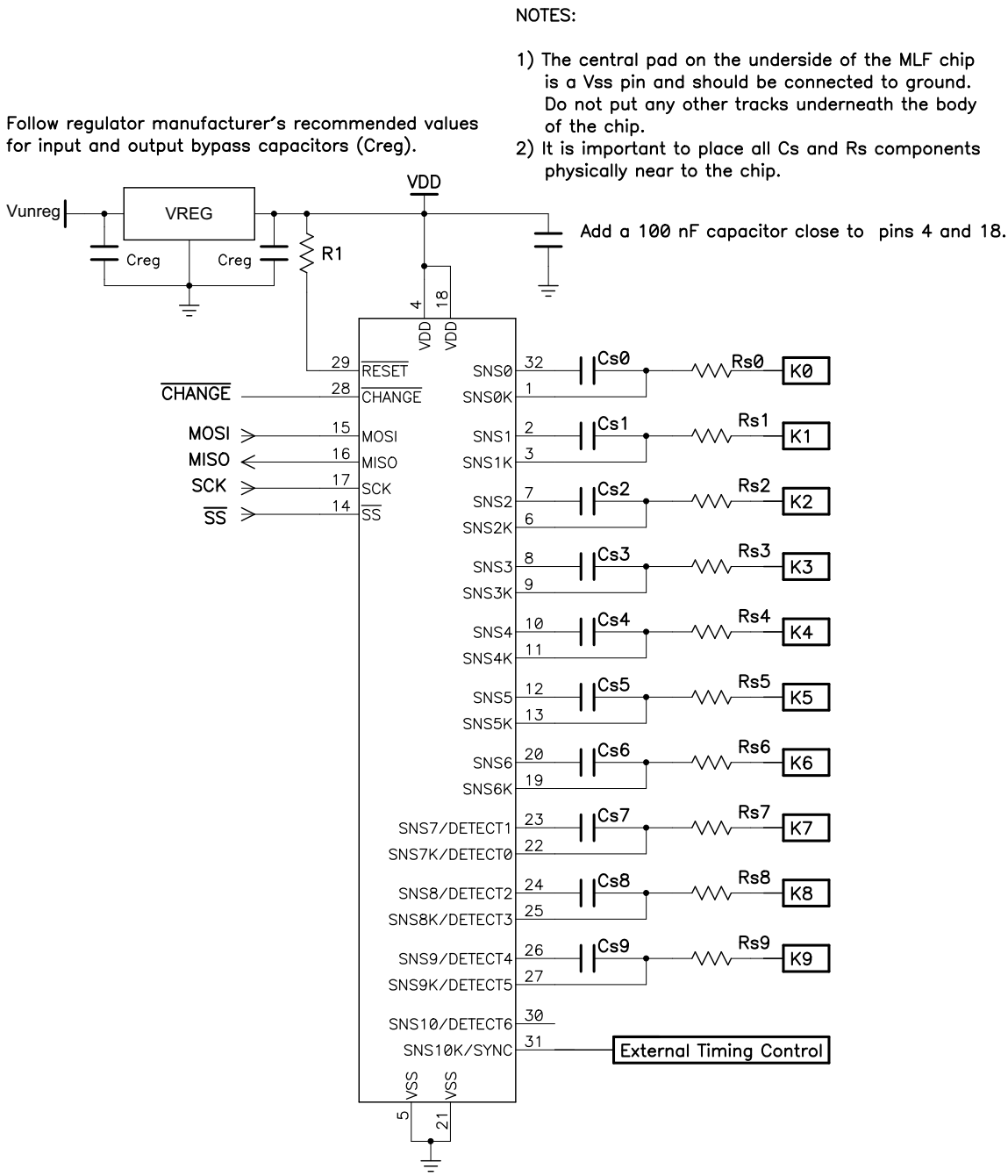


Figure 1-3. Typical Circuit: 10 Keys With External Trigger (SYNC Mode)



For component values in Figure 1-1, Figure 1-2 and Figure 1-3, check the following sections:

- Section 3.1 on page 8: Cs capacitors (Cs0 – Cs10)
- Section 3.2 on page 8: Sample resistors (Rs0 – Rs10)
- Section 3.5 on page 8: Voltage levels
- Section 3.3 on page 8: LED traces

2. Overview of the AT42QT1111

2.1 Introduction

The AT42QT1111 (QT1111) is a digital burst mode charge-transfer (QT™) capacitive sensor driver designed for any touch-key applications.

The keys can be constructed in different shapes and sizes. Refer to the *Touch Sensors Design Guide* and Application Note QTAN0002, *Secrets of a Successful QTouch Design*, for more information on construction and design methods (both downloadable from the Atmel website).

The device includes all signal processing functions necessary to provide stable sensing under a wide variety of changing conditions, and the outputs are fully debounced. Only a few external parts are required for operation.

The QT1111 modulates its bursts in a spread-spectrum fashion in order to suppress heavily the effects of external noise, and to suppress RF emissions.

2.2 Configurations

The QT1111 is designed as a versatile device, capable of various configurations. There are two basic configurations for the QT1111:

- 11-key QTouch. The device can sense up to 11 keys.
- 7-key QTouch with individual outputs for each key. The device can sense up to 7 keys and drive the matching Detect outputs to a user-configurable PWM.

Both configurations allow for a choice of acquisition modes, thus providing a variety of possibilities that will satisfy most applications (see the following sections for more information).

Additionally, the SYNC line can be used as an external trigger input. Note that in 11-key mode the SYNC line replaces one key, thus allowing only 10 keys.

See Section 4.7 on page 17 for more information.

2.3 Guard Channel

The device has a guard channel option (available in all key modes), which allows one key to be configured as a guard channel to help prevent false detection. See Section 4.9 on page 19 for more information.

2.4 Self-test Functions

The QT1111 has two types of self-test functions:

- Internal Hardware tests – check for hardware failures in the device internal memory.
- Functional checks – confirm that the device is operating within expected parameters.

See Section 4.10 on page 19 for more information.

2.5 Moisture Tolerance

The presence of water (condensation, sweat, spilt water, and so on) on a sensor can alter the signal values measured and thereby affect the performance of any capacitive device. The moisture tolerance of QTouch devices can be improved by designing the hardware and fine-tuning the firmware following the recommendations in the application note Atmel *AVR3002: Moisture Tolerant QTouch Design* (www.atmel.com/Images/doc42017.pdf).

3. Wiring and Parts

3.1 Cs Sample Capacitors

Cs0 – Cs10 are the charge sensing sample capacitors. Normally they are identical in nominal value. The optimal Cs values depend on the thickness of the panel and its dielectric constant. Thicker panels require larger values of Cs. Values can be in the range 2.2 nF (for faster operation) to 33 nF (for best sensitivity); typical values are 4.7 nF to 10 nF.

The value of Cs should be chosen so that a light touch on a key produces a reduction of ~20 to 30 in the key signal value (see Section 6.8 on page 25). The chosen Cs value should never be so large that the key signals exceed ~1000, as reported by the chip in the debug data.

The Cs capacitors must be X7R or PPS film type, for stability. For consistent sensitivity, they should have a 10 percent tolerance. Twenty percent tolerance may cause small differences in sensitivity from key to key and unit to unit. If a key is not used, the Cs capacitor may be omitted.

3.2 Rs Resistors

The series resistors Rs0 – Rs10 are inline with the electrode connections and should be used to limit electrostatic discharge (ESD) currents and to suppress radio frequency (RF) interference. Values should be approximately 2 kΩ to 20 kΩ each; a typical value is 4.7 kΩ.

Although these resistors may be omitted, the device may become susceptible to external noise or radio frequency interference (RFI). For details of how to select these resistors see the Application Note QTAN0002, *Secrets of a Successful QTouch Design*, downloadable from the Touch Technology area of the Atmel website, www.atmel.com.

3.3 LED Traces and Other Switching Signals

Digital switching signals near the sense lines can induce transients into the acquired signals, deteriorating the SNR performance of the device. Such signals should be routed away from the sensing traces and electrodes, or the design should be such that these lines are not switched during the course of signal acquisition (bursts).

LED terminals which are multiplexed or switched into a floating state, and which are within, or physically very near, a key (even if on another nearby PCB) should be bypassed to either Vss or Vdd with at least a 1 nF capacitor. This is to suppress capacitive coupling effects which can induce false signal shifts. The bypass capacitor does not need to be next to the LED, in fact it can be quite distant. The bypass capacitor is noncritical and can be of any type.

LED terminals which are constantly connected to Vss or Vdd do not need further bypassing.

3.4 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 the QT1111, 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.

3.5 Power Supply

See Section 8.2 on page 37 for the power supply range. If the power supply fluctuates slowly with temperature, the device tracks and compensates 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 usual power supply considerations with QT parts apply to the device. The power should be clean and come from a separate regulator if possible. However, this device is designed to minimize the effects of unstable power, and, except in extreme conditions, should not require a separate Low Dropout (LDO) regulator.

See underneath Figure 1.3 on page 4 for suggested regulator manufacturers.

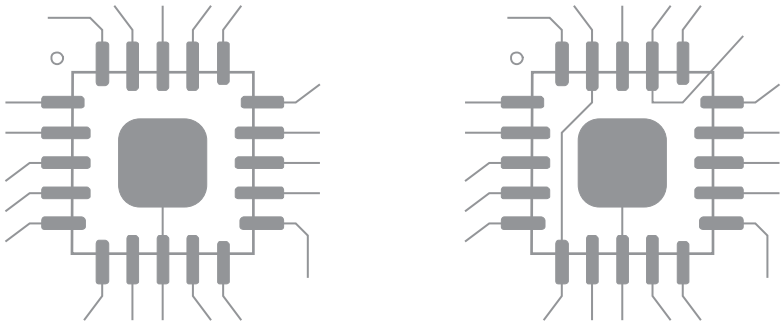
	**Caution:** A regulator IC shared with other logic can result in erratic operation and is **not** advised. A single ceramic 0.1 µF bypass capacitor, with short traces, should be placed very close to the power pins of the IC. Failure to do so can result in device oscillation, high current consumption, or erratic operation.

It is assumed that a larger bypass capacitor (like1 µF) is somewhere else in the power circuit; for example, near the regulator.

3.6 QFN Package Restrictions

The central pad on the underside of the QFN chip should be connected to ground. Do not run any tracks underneath the body of the chip, only ground. Figure 3-1 shows examples of good and bad tracking.

Figure 3-1. Examples of Good and Bad Tracking



Example of GOOD tracking **Example of BAD tracking**

4. Detailed Operations

4.1 Communications

4.1.1 Introduction

All communication with the device is carried out over the Serial Peripheral Interface (SPI). This is a synchronous serial data link that operates in full-duplex mode. The host communicates with the QT controller over the SPI using a master-slave relationship, with the QT1111 acting in slave mode.

4.1.2 SPI Operation

The SPI uses four logic signals:

- Serial Clock (SCK) – output from the host.
- Master Output, Slave Input (MOSI) – output from the host, input to the QT controller. Used by the host to send data to the QT controller.
- Master Input, Slave Output (MISO) – input to the host, output from the QT controller. Used by the QT device to send data to the host.
- Slave Select ($\overline{SS}$) – active low output from the host.

At each byte, the master pulls $\overline{SS}$ low and generates 8 clock pulses on SCK. With these 8 clock pulses, a byte of data is transmitted from the master to the slave over MOSI, most significant bit (MSB) first.

Simultaneously a byte of data is transmitted from the slave to the master over MISO, also most significant bit first.

The slave reads the status of MOSI at the leading edge of each clock pulse, and the master reads the slave data from MISO at the trailing edge.

The QT1111 requires that the clock idles “high”, meaning that the data on MOSI and MISO pins are set at the falling edges and sampled at the rising edges.

The QT1111 SPI interface can operate at any SCK frequency up to 750 kHz.

In multibyte communications, the master must pause for a minimum delay of 300 µs between the completion of one byte exchange and the beginning of the next.

Note that the number of bytes to be transmitted depends on the initial command sent by the host. This sets the mode on the QT1111 so that the QT1111 knows how to respond to, or how to interpret, the following bytes. If there is a delay of >100 ms between bytes while the QT1111 is waiting for data, or waiting to send data, then the incomplete transmission is discarded and the device resets its SPI state machine. It will then interpret the next byte it receives as a fresh command.

When the QT1111 SPI interface is receiving a new command, it returns the *Idle* status code (`0x55`) on MISO during the first byte exchange to indicate to the master that it is in the correct state for receiving instructions.

4.1.3 CRC Bytes

If enabled, a CRC checking procedure is implemented on all communications between the SPI master and the QT1111. In this case, each command or report request sent by the master must have a byte appended containing the CRC checksum of the data sent. The QT1111 will not respond to commands until the CRC byte has been received and verified.

Sample C code showing the algorithm for calculating the CRC of the data can be found in Appendix A..

When the QT1111 is expecting a CRC byte, it returns (on MISO) the calculated CRC byte which it expects to receive. This is sent simultaneously with the QT1111 receiving the CRC byte from the master (that is, during the same byte exchange). This allows both devices to confirm that the data was sent correctly.

All data returned by the QT1111 is also be followed by a CRC byte, allowing the master to confirm the integrity of the data transmission.

4.1.4 SPI Commands

There are three types of communication between the SPI master and the QT1111:

- Control commands (see Section 5. on page 21)
  - To send control instructions to the QT1111
- Report requests (see Section 6. on page 23)
  - To reading status information from the QT1111
- Setup commands (see Section 7. on page 27)
  - To set configuration options (“Set” instructions)
  - To read configuration options (“Get” instructions)

Additionally the NULL command (`0x00`) is transmitted by the host device as it is receiving data from the QT1111.

4.1.4.1 Control Commands

A control command is an instruction sent to the QT1111 that controls operations of the device, and for which no response is required. Examples of control commands are: *Reset*, *Calibrate*, *Send Setups*.

With the exception of *Send Setups*, control commands normally require a single byte exchange, unless CRC checking is enabled, in which case a second byte must be transmitted by the host with the calculated CRC of the command byte.

Figure 4-1. Sleep Command – CRC Disabled

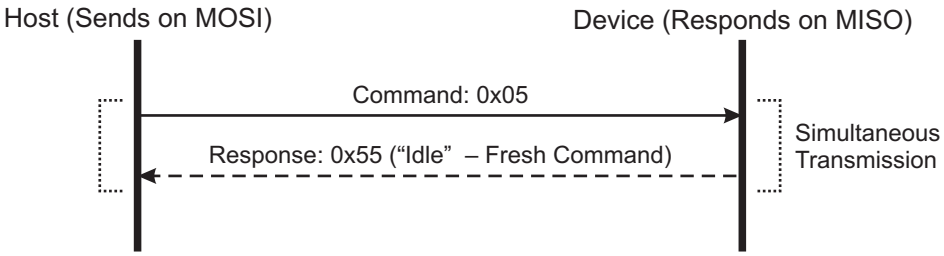
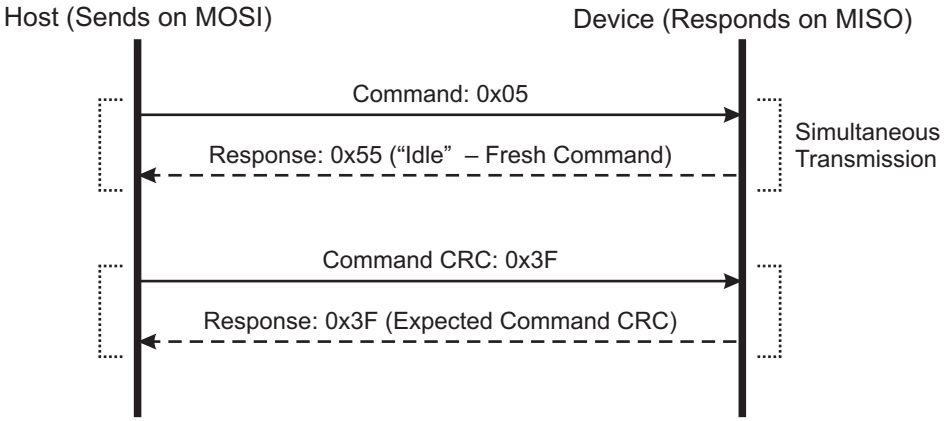


Figure 4-2. Sleep Command – CRC Enabled



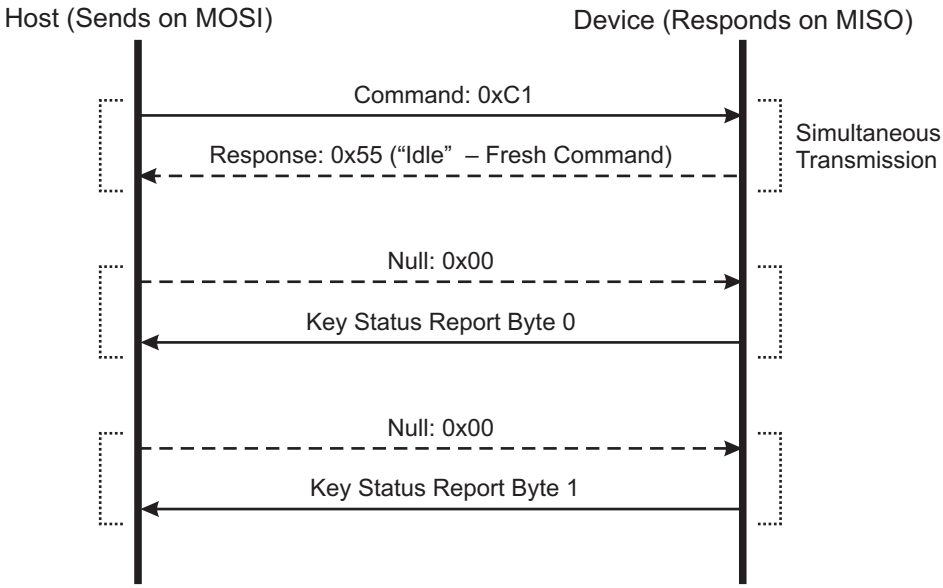
When the *Send Setups* command is received, the QT1111 stops measurement of QTouch sensors and waits for 42 bytes of data to be sent. Only when all 42 bytes have been received (and the CRC byte, if CRC is enabled), the QT1111 applies all the settings to RAM and resumes measurement. In this case, if CRC is enabled, the CRC byte is calculated for all the data sent by the host, including the command byte `0x01`.

Control Commands are specified in detail in Section 5. on page 21.

4.1.5 Report Requests

Report Requests are sent by the Host to instruct the QT1111 to return status information. The host sends the appropriate *Report Request* command, then transmits Null bytes on MOSI while the QT1111 returns the report data on MISO.

Figure 4-3. All Keys Report – CRC Disabled



For example, Figure 4-3 on page 12 shows the exchange that takes place to read the 2-byte *All Keys* report. In this exchange, the host sends:

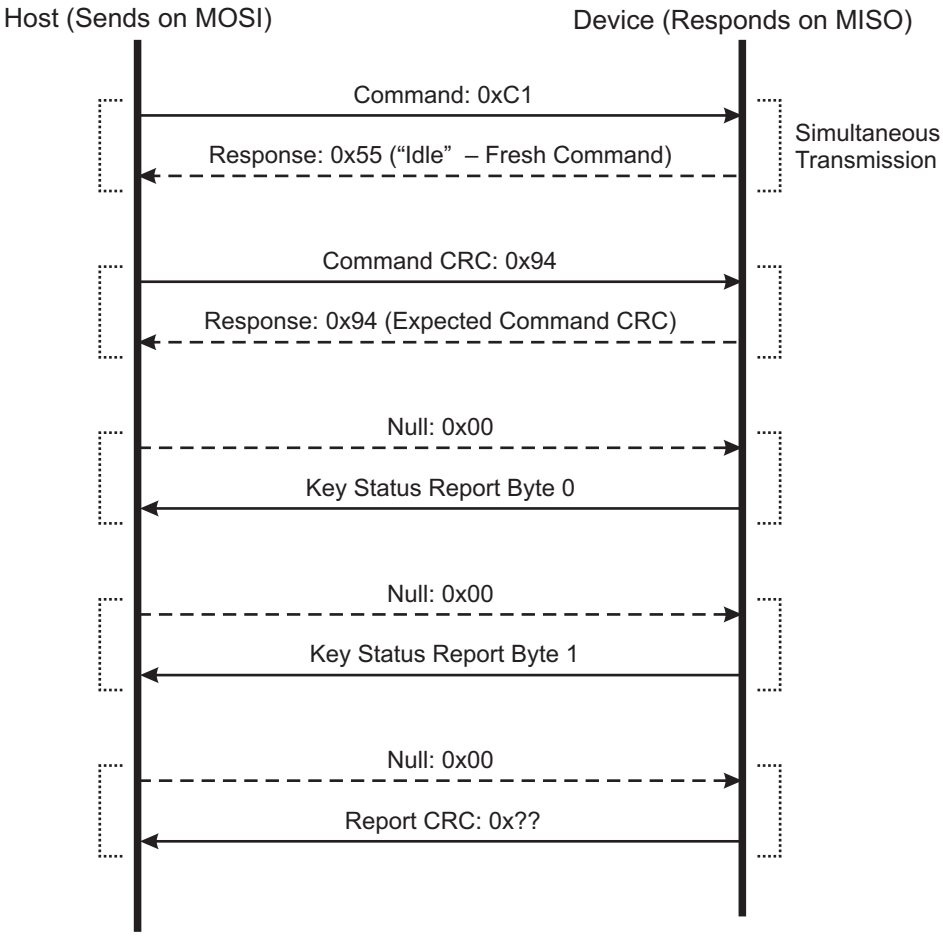
`0xC1` — `0x00` — `0x00`

and the QT1111 returns (simultaneously):

`0x55` — Report Byte 0 — Report Byte 1

If CRC is enabled, this exchange is extended to 5 bytes, as shown in Figure 4-4.

Figure 4-4. All Keys Report – CRC Enabled



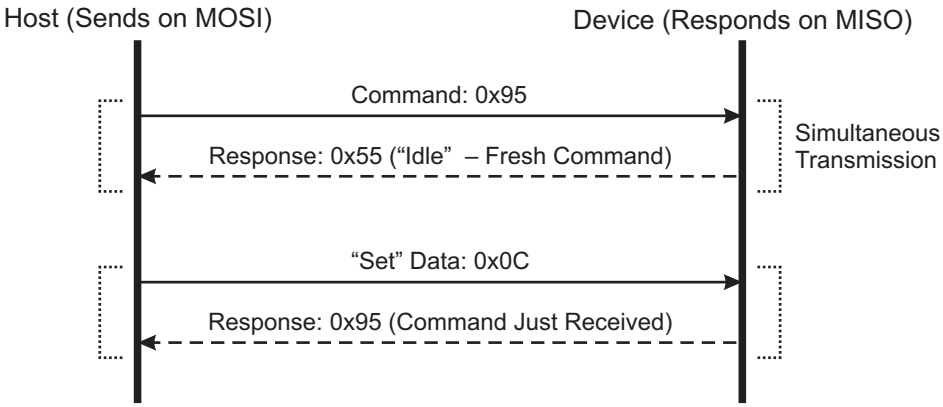
4.1.5.1 Set Instructions

Set Instructions are 2-byte transmissions by the host that are used to send settings to individual locations in the device memory map.

At the first byte, the QT1111 returns `0x55` (*Idle*) to confirm that it will interpret the byte as a new command. At the second byte, the QT1111 returns the *Set* command it has just received.

For example, to set the *Positive Recalibration Delay* to 1920 ms, address 5 in the memory map is set to 12 (`0x0C`). This is done with the *Set* command for address 5 (command code `0x95`), as shown in Figure 4-5 on page 13.

Figure 4-5. Positive Recalibration Delay Set Instruction – CRC Disabled

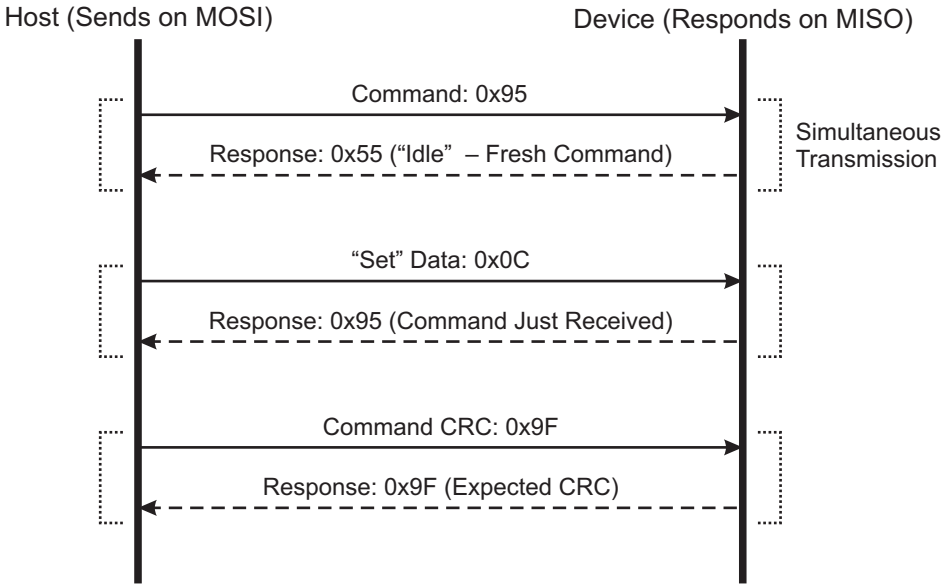


With CRC Enabled, a CRC byte is also required (Figure 4-6). This is calculated for the two transmitted bytes (that is, the *Set* command and the data byte).

For example, for the sequence shown in Figure 4-5 (`0x95` – `0x0C`), the CRC Byte is `0x9F`. As is the case with the other command types, when the QT1111 is expecting a CRC byte from the host, it calculates that byte in advance and returns the expected value to the host in the same transmission as the host sends the CRC byte.

The sent data is not applied to the memory location until the CRC byte has been received and verified.

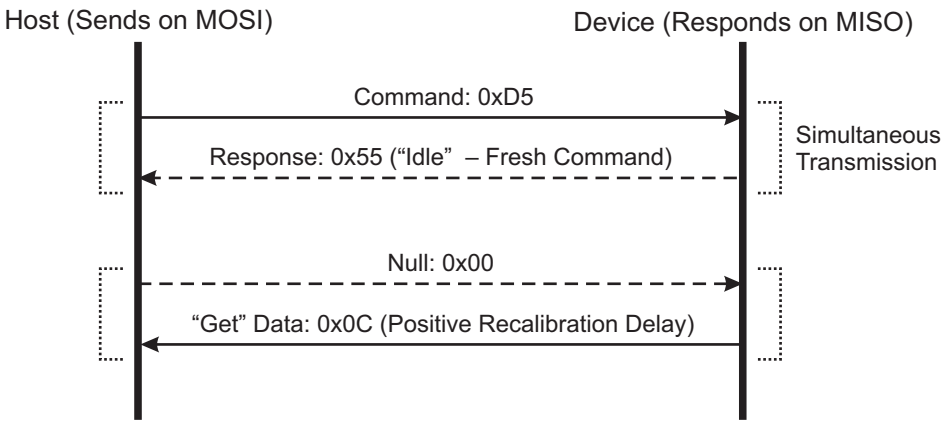
Figure 4-6. Positive Recalibration Delay Set Instruction – CRC Enabled



4.1.5.2 Get Instructions

Get instructions are instructions that read the data from a location in the QT1111 memory map.

Figure 4-7. Positive Recalibration Delay Get Instruction – CRC Disabled

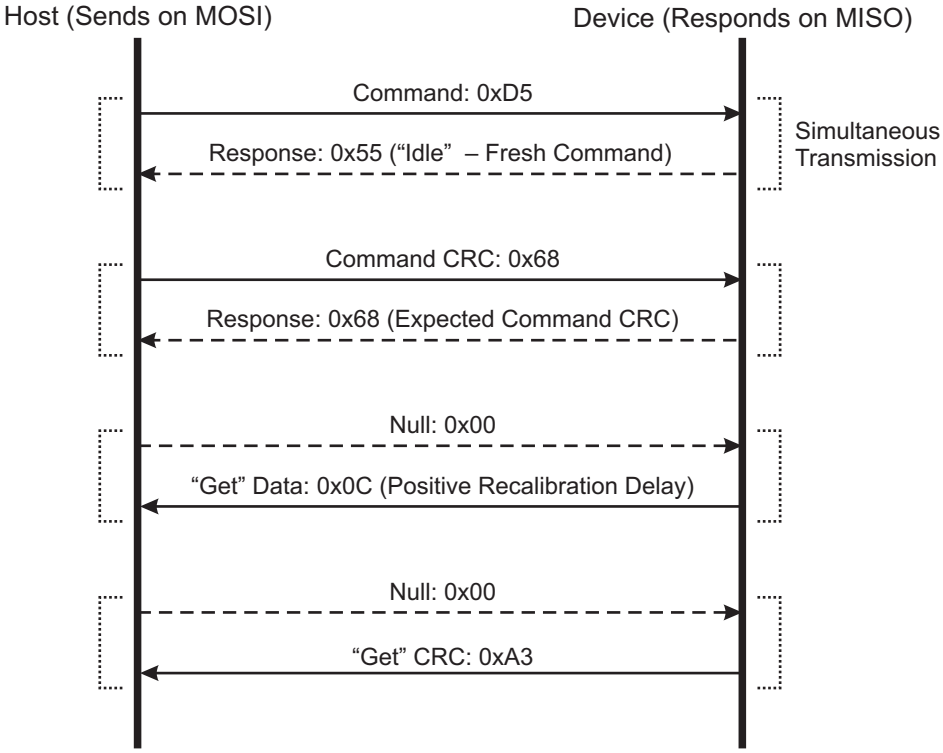


The host sends the appropriate *Get* command, followed by a *Null* byte. The QT1111 returns the contents of the addressed memory location.

Figure 4-7 on page 14 shows the exchange for a report on the positive recalibration delay (assuming that the data byte is `0x0C`).

With CRC Enabled, this exchange takes 4 bytes, with a command CRC transmitted by the host and a report CRC returned by the QT1111 (see Figure 4-8).

Figure 4-8. Positive Recalibration Delay Get Instruction – CRC Enabled



4.1.6 Quick SPI Mode

4.1.6.1 Introduction

In Quick SPI Mode, the QT1111 sends a 7-byte key report at each exchange. No host commands are required over SPI in this mode; the host clocks the data bytes out in sequence. Quick SPI mode is enabled by setting the *SPI_EN* bit in the Comms Options setup byte (see Section 7.5 on page 29).

4.1.6.2 Quick SPI Report

The 7 report bytes are in the format given in Table 4-1.

Table 4-1. Device Status Report Format

Byte	Description	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Counter	Counter – increments from 0 to 255							
1	Detect status, channels 0 – 3	Channel 3		Channel 2		Channel 1		Channel 0	
2	Detect status, channels 4 – 7	Channel 7		Channel 6		Channel 5		Channel 4	
3	Detect status, channels 8 – 10	Reserved		Channel 10		Channel 9		Channel 8	
4	Error status, channels 0 – 3	Channel 3		Channel 2		Channel 1		Channel 0	
5	Error status, channels 4 – 7	Channel 7		Channel 6		Channel 5		Channel 4	
6	Error status, channels 8 – 10	Reserved		Channel 10		Channel 9		Channel 8	

where:

- Byte 0 is a counter that increments from 0 to 254 on successive exchanges to confirm that firmware is operating correctly.
- Bytes 1 – 3 indicate the detect status of channels 0 – 3, 4 – 7 and 8 – 10 respectively (two bits per channel), as follows:
  - 00 = Channel not in detect
  - 01 = Channel in detect
  - 10 = Not Allowed
  - 11 = Invalid Signal (Channel disabled)
- Bytes 4 – 6 indicate the error status of channels 0 – 3, 4 – 7 and 8 – 10 respectively (two bits per channel), as follows:
  - 00 = No error
  - 01 = Not allowed
  - 10 = Error on channel
  - 11 = Invalid signal (channel disabled)

Successive byte exchanges in Quick SPI mode cycle through the 7 bytes of status information. If synchronization is lost, the host must either re-synchronize by identifying the incrementing counter byte (byte 0) or pausing communications for at least 100 ms so the QT1111 will reset its SPI state.

4.1.6.3 Commands in Quick SPI Mode

Only two host commands are recognized under Quick SPI mode. These are shown in Table 4-2.

Table 4-2. Host Commands in Quick SPI Mode

Command	Code	Purpose
Store to EEPROM	0x0A	Allows for “Quick SPI mode” to be stored as the default start-up mode
Enable Full SPI	0x36	Enables full SPI mode

CRC checking is not implemented in Quick SPI mode for host commands or return data.

4.1.6.4 Quick SPI Mode timing

In Quick SPI mode, the minimum time between byte exchanges is reduced to 100 µS.

If a pause in communications of 100 ms is detected during reading of the 7-byte report, the QT1111 resets the exchange, and on the next byte read it returns byte 0 of the report.

4.2 Reset

The QT1111 can be reset using one of two methods:

- **Hardware reset:** An external reset logic line can be used if desired, fed into the $\overline{\text{RESET}}$ pin. However, under most conditions it is acceptable to tie $\overline{\text{RESET}}$ to Vdd.
- **Software reset:** A software reset can be forced using the “Reset” control command.

For both methods, the device will follow the same initialization sequence. If there any saved settings in the EEPROM, these are loaded into RAM. Otherwise the default settings are applied.

Note: The SPI interface becomes active after the QT1111 has completed its startup sequence, taking approximately 320 ms after power on/reset.

4.3 Sleep Mode

The QT1111 can be put into a very low power sleep mode (typically < 2 µA). During sleep mode, no keys are measured and the DETECT outputs are all put into high impedance mode to minimize current consumption. The device remains in sleep mode until a falling edge is detected on either the $\overline{SS}$ pin or the $\overline{CHANGE}$ pin. When the QT1111 wakes from sleep mode, it continues to operate as it was before it was put into sleep mode. The QT1111 requires approximately 100 µs to wake from sleep mode and will not respond correctly to SPI communications until the wake-up procedure is complete. The low level on the $\overline{SS}$ or $\overline{CHANGE}$ pin that is used to wake the device must be maintained for 100 µs to ensure correct operation.

Note: If the device is set to sleep mode for an extended period, the host should initiate a recalibration immediately after waking the QT1111.

4.4 Calibration

The device can be forced to recalibrate the sensor keys at any time. This can be useful where, for example, a portable device is plugged into mains power, or during product development when settings are being tuned.

The QT1111 can also be configured to automatically recalibrate if it remains in detection for too long. This avoids keys becoming “stuck” after a prolonged period of uninterrupted detection. See Section 7.17 on page 36 for details.

4.5 $\overline{CHANGE}$ Pin

The $\overline{CHANGE}$ pin can be configured using the Comms Options setup byte (see Section 7.5 on page 29) to act in one of two modes:

- Data mode
  - The $\overline{CHANGE}$ pin is asserted (pulled low) when the detection status of a key changes from that last sent to the host; that is when a key-touch or key-release event occurs.
  - The $\overline{CHANGE}$ pin is pulled low when a key status changes and is only released when the “Send All keys” report is requested (`0xC1`), or the key status information bytes are read in Quick SPI mode (see Section 7.5 on page 29).
- Touch mode
  - The $\overline{CHANGE}$ pin is pulled low when one or more keys are in detect. The $\overline{CHANGE}$ pin remains low as long as there is a key in detect, regardless of communications.
  - The $\overline{CHANGE}$ pin is released when there are no keys in detect. No host communications are required to release the $\overline{CHANGE}$ pin.

4.6 Stand-alone Mode

The QT1111 can operate in a stand-alone mode without the use of the SPI interface. The settings are loaded from EEPROM and the device operates in 7-key mode using the Detect outputs.

4.7 Key Modes

4.7.1 11-key Mode

In 11-key mode, the device can sense up to 11 keys. Alternatively, one key can be replaced by the SYNC line as an external trigger input (see Section 4.8.2 on page 18).

11-key mode is configured by setting the *MODE* bit in the Device Mode setup byte (see Section 7.4 on page 28).

Key acquisition can be triggered in one of two ways: using the internal clock to trigger acquisition either at a fixed repetition period or in a continuous “free run” mode (see Section 4.8.1), or using the SYNC pin to provide an external trigger (see Section 4.8.2 on page 18),

4.7.2 7-key Mode

In 7-key mode, the detect outputs DETECT0 to DETECT6 become active on pins 22 – 27 and 30. These outputs provide configurable PWM signals that indicate when each of the keys is touched.

7-key mode is configured by clearing the *MODE* bit in the Device Mode setup byte (see Section 7.4 on page 28).

Each DETECT output can be individually configured to output a PWM signal while the matching key is in detect or out of detect. This signal can be one of nine levels, ranging from low (PWM = 0%) to high (PWM = 100%). This allows for the use of an indicating LED. This is achieved by enabling the appropriate bit in the Key to LED setup byte (see Section 7.14 on page 34), and setting the desired outputs levels or PWMs in setup addresses 9 to 15 (see Section 7.12 on page 32).

4.8 Trigger Modes

4.8.1 Timed Trigger

In 11-key mode, The QT1111 can be configured to use the internal clock as a timed trigger. In this case, the QT1111 is configured with a cycle period, such that each acquisition cycle starts a specified length of time after the start of the previous cycle. If the cycle period is set to 0, each acquisition cycle starts as soon as the previous one has finished, resulting in the acquisition cycles running back-to-back in a “free run” mode.

The use of a timed trigger, and the cycle period to be used, is set in the Device Mode setup byte (see Section 7.4 on page 28).

4.8.2 Synchronized Trigger

Alternatively, the QT1111 can operate in “synchronized” mode. In this mode, SNS10K is used as a SYNC pin to trigger key acquisition, rather than using the device internal clock. In this case the maximum number of keys is reduced to 10.

The SYNC pin can use one of two methods to trigger key measurements, selectable via bit 4 of the Device Mode setup byte (see Section 7.4 on page 28): Low Level and Rising Edge.

With the Low Level method the QT1111 operates in “free run” mode for as long as the SYNC pin is read as a logical 0. When the SYNC pin goes high, the current measurement cycle will be finished and no more key measurements will be taken until the SYNC pin goes low again. The low level trigger should be a minimum of 1 ms so that there is sufficient time for the device to detect the low level.

With the Rising Edge method all enabled keys are measured once when a rising edge is detected on the SYNC pin. This allows key measurements to be synchronized to an external event or condition.

Note: In SYNC mode a single acquisition burst is carried out for self-testing at start-up. No further bursts occur unless triggered via the SYNC pin.

For example, the SYNC pin can be used by the host to synchronize several devices to each other. This would ensure that only one of the devices outputs pulses at any given time and signals from one QT1111 do not interfere with the measurements from another.

Another use for synchronizing to the rising edge is to steady the signals when the device is running off a mains transformer with insufficient mains frequency filtering that is causing a 50 Hz or 60 Hz ripple on Vdd. If the mains voltage is scaled down with a simple voltage divider and connected to the SYNC pin, then the key measurement can be triggered by the rising edge detected at a positive going zero-crossing. Note that in this case, each key signal will be taken at the same point in the cycle, so Vdd will be the same at each measurement for a given key and the signals will be steadier.

4.9 Guard Channel Option

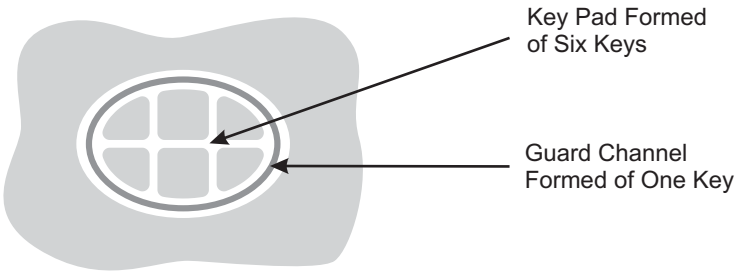
The device has a guard channel option (available in all key modes), which allows one key to be configured as a guard channel to help prevent false detection (see Figure 4-9 on page 19). Guard channel keys should be more sensitive than the other keys (physically bigger or larger Cs), subject to burst length limitations (see Section 4.11.2 on page 20).

With guard channel enabled, the designated key is connected to a sensor pad which detects the presence of touch and overrides any output from the other keys using the chip AKS feature. The guard channel option is enabled by the Guard Key setup byte (see Section 7.5 on page 29).

With the guard channel not enabled, all the keys work normally.

Note: If a key is already “in detect” when the guard channel becomes active, that key will remain in detect and the guard key will not activate until the active key goes out of detect.

Figure 4-9. Guard Channel Example



4.10 Self-test Functions

4.10.1 Internal Hardware Tests

Internal hardware tests check for hardware failure in the device internal memory areas and data paths. Any failure detected in the function or contents of application ROM, RAM or registers causes the device to reset itself.

The application code is scanned with a CRC check routine to confirm that the application data is all correct.

The RAM and registers are checked periodically (every 10 seconds) for dynamic and static failures.

4.10.2 Functional Checks

Functional checks confirm that the device is operating within expected parameters; any failure detected in these tests is notified to the system host. The device will continue to operate in the event that such functional failures are detected.

The functional tests are:

- Check that the channel-measurement signals are within the defined range.
- Confirm that data stored in the EEPROM is valid.

These tests are carried out as the particular functions are used. For example, the EEPROM is checked when the device attempts to load data from EEPROM, and the channel signals are checked when a measurement is carried out.

Note: If a particular channel is unused, the threshold of that channel should be set to 0 to prevent the incorrect reporting of the unused channel as being in an error state.

4.11 Signal Processing

4.11.1 Detection Integrator

The device features a detection integration mechanism, which acts to confirm a detection in a robust fashion. A per-key counter is incremented each time the key has exceeded its threshold. When this counter reaches a preset limit the key is finally declared to be touched. For example, if the DI limit is set to 10, then a key signal must fall by more than the key threshold, and remain below that level for 10 acquisitions, before the key is declared to be touched.

Similarly, the DI is applied to a key that is going out of detect: it must take 10 acquisitions where the signal has not exceeded its detect threshold before it is declared to leave touch.

4.11.2 Burst Length Limitations

The maximum burst length is 2048 pulses. The recommended design is to use a capacitor that gives a signal of <1000 pulses.

The number of pulses in the burst can be obtained by reading the key signal (that is, the number of pulses to complete measurement of the key signal) over the SPI interface (see Section 6.8 on page 25). Alternatively, a scope can be used to measure the entire burst, and then the burst length divided by the time for a single pulse.

Note that the keys are independent of each other. It is therefore possible, for example, to have a signal of 100 on one key and a signal of 1000 on another.

4.11.3 Adjacent Key Suppression Technology

The device includes the Atmel patented Adjacent Key Suppression (AKS) technology to allow the use of tightly spaced keys on a keypad with no loss of selectability by the user.

AKS is enabled or disabled for each key individually; only one key out of those enabled for AKS may be reported as touched at any one time. The first key touched dominates and stays in detect until it is released, even if another stronger key is reported. Once it is released, the next strongest key is reported. If two keys are simultaneously detected, the strongest key is reported, allowing a user to slide a finger across multiple keys with only the dominant key reporting touch.

Each key can be enabled for AKS processing via the AKS mask (see Section 7.11 on page 32). Keys outside the group of enabled keys may be in detect simultaneously.

5. Control Commands

5.1 Introduction

The QT1111 control commands are those commands that affect the device operation.

The control commands are listed in Table 5-1 and are described individually in the following sections.

Table 5-1. Control Commands

Command	Code	Note
Send Setups	0x01	Configures the device to receive setup data
Calibrate All	0x03	Calibrates all keys
Reset	0x04	Resets the device
Sleep	0x05	Sleep (dead) mode
Store to EEPROM	0x0A	Stores RAM setups to EEPROM
Restore from EEPROM	0x0B	Copies EEPROM setups to RAM (automatically done at startup)
Erase EEPROM	0x0C	Erases EEPROM setups
Recover EEPROM	0x0D	Restores last EEPROM settings (after erase)
Calibrate Key *k*	0x1*k*	Calibrates one key (key *k*)

Note: Commands are implemented immediately upon reception, so a suitable delay is required for the operation to be completed before communications can be re-established.

5.2 Send Setups (0x01)

This command initiates the upload of the full settings table to the QT1111 (see Section 7. on page 27).

When this command is received, the QT1111 stops key measurement and waits until 42 bytes of setup data have been received. Key acquisition will restart after all the setup data has been received.

If enabled, a CRC check byte is transmitted (both ways) after the 42 bytes to confirm that they have been received correctly.

If CRC checking is not enabled, it is recommended that the host request a dump of setup data from the QT1111, and confirms that the data correctly matches the data sent.

The host must wait for at least 300 µs for the operation to be completed before communications can be re-established.

5.3 Calibrate All (0x03)

This command initiates the recalibration of all sensor keys.

The host must wait for at least 300 µs for the operation to be completed before communications can be re-established.

5.4 Reset (0x04)

The Reset command forces the QT1111 to reset. If the setups data is present in the EEPROM, the setups are loaded into the device. Otherwise default settings are applied.

The host must wait for at least 320 ms for the operation to be completed before communications can be re-established.

5.5 Sleep (0x05)

The Sleep command puts the device into sleep mode (see Section 4.3 on page 17).

The host must wait for at least 300 µs after a low signal is applied to the $\overline{SS}$ or $\overline{CHANGE}$ pin to wake the device before communications can be re-established.

5.6 Store to EEPROM (0x0A)

Stores the current RAM contents to the QT1111 internal EEPROM. When the device is reset, it will automatically reload these settings.

The host must wait for at least 200 ms for the operation to be completed before communications can be re-established.

5.7 Restore from EEPROM (0x0B)

Settings stored in EEPROM are automatically loaded into RAM when the device is reset. If desired, these settings can be re-loaded into RAM using the *Restore from EEPROM* command.

The host must wait for at least 150 ms for the operation to be completed before communications can be re-established.

5.8 Erase EEPROM (0x0C)

This command erases the settings stored in EEPROM and then resets the QT1111. This causes the QT1111 to revert to its default settings.

The host must wait for at least 50 ms for the operation to be completed before communications can be re-established.

Note that under the default settings, the CRC is disabled. By erasing the EEPROM, therefore, the default settings are restored and the QT1111 is put into non-CRC mode regardless of the previous setting.

5.9 Recover EEPROM (0x0D)

This command “undeletes” the setup data that was previously stored in the device EEPROM and has been erased using the “Erase EEPROM” command.

Note: If valid settings have not previously been stored in the device EEPROM, the QT1111 continues to operate under the default settings.

The host must wait for at least 50 ms for the operation to be completed before communications can be re-established.

5.10 Calibrate Key (0x1k)

This command recalibrates the key specified by *k*. For example, to calibrate key 4, the host sends `0x14`; to calibrate key 10, the host sends `0x1A`.

The host must wait for at least 300 µs for the operation to be completed before communications can be re-established.

6. Report Requests

6.1 Introduction

The host can request reports from the QT1111, as summarized in Table 6-1.

Table 6-1. Report Requests

Command	Code	Note	Data Returned
Send First Key	0xC0	Returns the first detected key	1 byte
Send All keys	0xC1	Returns all keys	2-byte bitfield
Device Status	0xC2	Returns the device status	1-byte bitfield
EEPROM CRC	0xC3	Returns the EEPROM CRC	1 byte
RAM CRC	0xC4	Returns the RAM CRC	1 byte
Error Keys	0xC5	Returns the error keys	2-byte bitfield
Signal for Key “k”'	0x2*k*	Returns the signal for key “k”	2-byte number
Reference for Key “k”	0x4*k*	Returns the reference for key “k”	2-byte number
Status for Key “k”	0x8*k*	Returns error conditions/touch indication	1 byte
Detect Output States	0xC6	Returns the detect output states	1 byte
Last Command	0xC7	Returns the last command sent to QT1111	1 byte
Setups	0xC8	Returns the setup data	42 bytes
Device ID	0xC9	Returns the device ID	1 byte
Firmware Version	0xCA	Returns the firmware version	1 byte

Note that SPI communications are full-duplex, so the host must transmit on the MOSI pin to keep the communications active, while reading data from the QT1111 on the MOSI pin. Failure to do this within 100 ms will cause the device to assume that the exchange has been abandoned and reset the SPI interface. The host should therefore send one or two “NULL” bytes, as appropriate, on the MOSI line as it receives the 1- or 2-byte report data from the device.

6.2 First Key (0xC0)

This command returns 1-byte report in the format shown in Table 6-2.

Table 6-2. Send First Key Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	DETECT	NUMKEY	ERROR		KEY_NUM			

DETECT: 0 = no key in detect; 1 = there is a key in detect.

NUMKEY: indicates the number of keys in detect:

0 = only one key is in detect (specified by “KEY_NUM”)

1 = more than one key in detect.

ERROR: 0 = there are no keys in an error state; 1 = at least one key is in error state.

KEY_NUM: the key number (0 to 10) of the key in detect (if there is only one), or the number of the first key to go into detection when there are more than one.

6.3 All Keys (0xC1)

Returns a 2-byte bit-field report indicating the detection status of all 11 keys.

Table 6-3. Send All Keys Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0						KEY_10	KEY_9	KEY_8
Byte 1	KEY_7	KEY_6	KEY_5	KEY_4	KEY_3	KEY_2	KEY_1	KEY_0

KEY_*n*: 0 = key *n* out of detect, 1 = key *n* in detect (where *n* is 0 – 10).

6.4 Device Status (0xC2)

This command returns a 1-byte bit-field report indicating the overall status of the QT1111.

Table 6-4. Device Status Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	1	DETECT	CYCLE	ERROR	CHANGE	EEPROM	RESET	GUARD

Bits 7 is always 1; the other bits are as follows:

DETECT: 0 = no key in detect, 1 = at least 1 key in detect.

CYCLE: 0 = cycle time is good, 1 = cycle time over-run. A cycle time over-run occurs when it takes longer to measure and process all the keys than the assigned cycle time.

ERROR: 0 = no key in error state, 1 = at least 1 key in error.

CHANGE: 0 = $\overline{\text{CHANGE}}$ pin is asserted, 1 = $\overline{\text{CHANGE}}$ pin is floating.

EEPROM: 0 = EEPROM is good, 1 = EEPROM has an error. If there are no settings stored in EEPROM, the EEPROM error bit is set and a zero EEPROM CRC is returned.

RESET: set to 1 after power-on or reset, cleared when “Device Status” is read.

GUARD: 0 = guard channel is not in detect, 1 = guard channel is active or in detect. This bit will be zero if the guard channel is not enabled.

6.5 EEPROM CRC (0xC3)

This command returns a 1-byte CRC checksum for the setup data in EEPROM.

6.6 RAM CRC (0xC4)

This command returns a 1-byte CRC checksum for the setup data in RAM.

6.7 Error Keys (0xC5)

This command returns a 2-byte bit-field report indicating the error status of all 11 keys. Note that disabled keys do not report errors.

Table 6-5. Send All Keys Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0						KEY_10	KEY_9	KEY_8
Byte 1	KEY_7	KEY_6	KEY_5	KEY_4	KEY_3	KEY_2	KEY_1	KEY_0

KEY_*n*: 0 = key *n* status good, 1 = key *n* in error (where *n* is 0–10).

6.8 Signal for Key *k* (0x2*k*)

This command returns a 2-byte report containing the most recent measured signal for key *k*. The signal is returned as a 16-bit number, MSB first.

Table 6-6. Signal for Key *k* Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	Signal MSB							
Byte 1	Signal LSB							

6.9 Reference for Key *k* (0x4*k*)

This command returns a 2-byte report containing the reference signal for key *k*. The reference is returned as a 16-bit number, MSB first.

Table 6-7. Reference for Key *k* Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	Reference MSB							
Byte 1	Reference LSB							

6.10 Status for Key *k* (0x8*k*)

This command returns a 1-byte report containing the status for key *k*.

Table 6-8. Status for Key *k* Report Format

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	DETECT	LBL	MBL			AKS_EN	CAL	KEY_EN

DETECT: 0ut of detect, 1 = in detect.

LBL: 0 = lower burst limit is good, 1 = lower burst limit has error.

MBL: 0 = maximum burst limit is good, 1 = maximum burst limit has error. The maximum burst limit is fixed at 2048 pulses.