Crystal Touch I^2C Addendum

Table of Contents

Date	Page	Summary
9/16/2009	Rev: 1	Initial Release
5/11/2010	Rev: 2	Updated address

Prepared By: Tony Gray

Approved By: Marty Wakita

1. Introduction

This document describes the I^2C hardware interface for the Ocular Crystal Touch panel. This document is an addendum to the Crystal Touch User's Manual. Only the I^2C interface is described in this document. For general operation and a list of commands, refer to the Crystal Touch User's Manual.

2. Hardware Interface

2.1. I^2C Signal Descriptions

All of Ocular's Crystal Touch touch screens with I^2C interface have the following hardware pinout:

Table 1

Pin No.	Symbol	Description
1	SCL	Serial Clock
2	SDA	Serial Data
3	NC	No Connect
4	DR	Data Ready
5	NC	No Connect
6	GND	Ground
7	V_{DD}	Power Supply
8	GND	Ground
9	SNSN	Stylus input
10	GND	Ground

These signals are described below:

SCL, SDA: These pins are part of the I^2C communications interface. See section 3 I^2C Bus Interface for details.

DR: This pin indicates when the Crystal Touch controller has data to send and when the Power On Self Test (POST) has completed successfully. See the User's Manual for details.

SNSN: This pin is for a dedicated stylus input. Leave this pin unconnected (contact Ocular for details on using Crystal Touch with a stylus).

3. I^2C Bus Interface

3.1. Basic Operation

I^2C is an open-collector communication bus which allows synchronous serial communication between a Master and Slave device. Crystal Touch acts as the Slave on a two-wire I^2C bus. The two wires are clock (SCL) and data (SDA). The Master always drives the SCL line. The SDA line is bi-directional. Data is clocked on the rising edge.

All communication is initiated by the Master. The Master indicates the beginning of a packet with a START command (the Master pulls SDA low while holding SCA high). The Master indicates the end of a packet by sending a STOP command (the Master drives SDA high while holding SCA high).

For more information search the web for the Philips document “The I^2C-Bus Specification”.

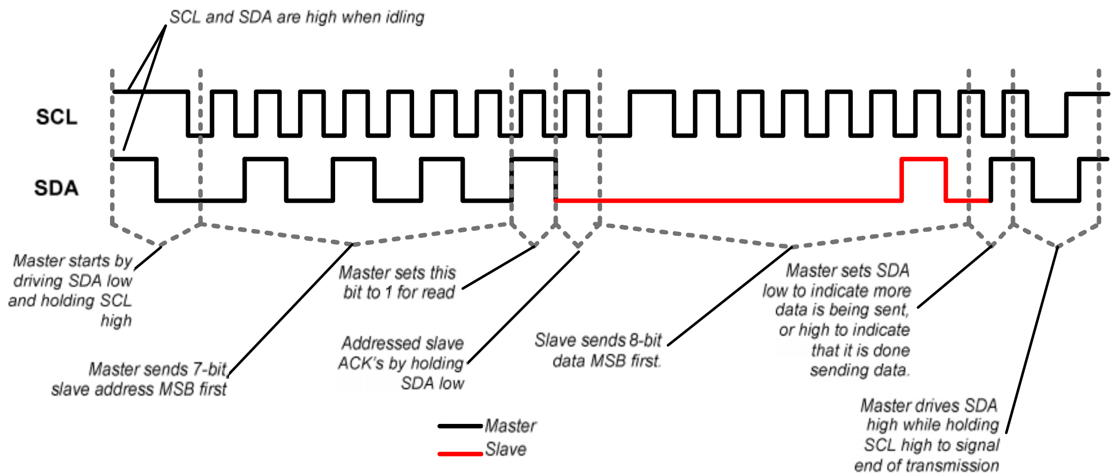

Figure 1: I^2C Read Cycle

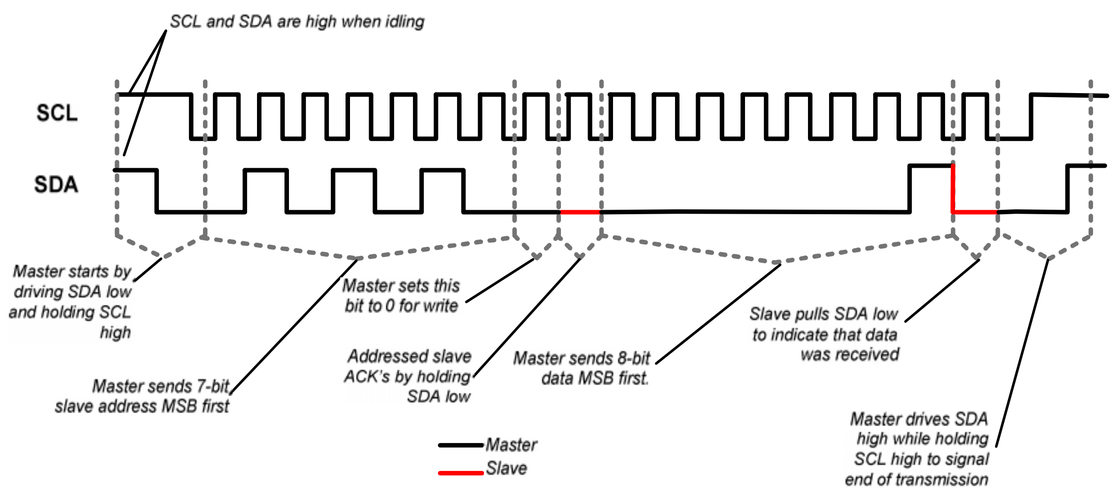

Figure 2: I^2C Write Cycle

3.2. I^2C Timing Requirements

Timing for the I^2C data lines is shown below (see Figure 3).

Table 2

Item	Symbol	Min.	Max.	Unit	Note
Output fall time for V_{IHmin} to V_{ILmax} with a bus capacitance from 10pF to 400pF	t_f	$20+0.1C_b$	250	ns	1
Pulse width of spikes which must be suppressed by the input filter	t_{SP}	0	50	ns	
Input current at each I/O pin with an input voltage between 0.1 V_{DD} and $0.9V_{DDmax}$	I_i	-10	10	µA	2
Capacitance for each I/O pin	C_i	—	10	pF	
SCL clock frequency	F_{SCL}	0	400	kHz	
Hold time (repeated) START condition. After this period, the first clock pulse is generated	$T_{HD;STA}$	0.6	—	µs	
Low period of the SCL clock	t_{LOW}	1.3	—	µs	
High period of the SCL clock	t_{HIGH}	0.6	—	µs	
Setup time for a repeated START condition	$t_{SU;STA}$	0.6	—	µs	
Data hold time	$T_{HD;DAT}$	0	0.9	µs	3
Data setup time	$t_{SU;DAT}$	100	—	ns	4
Rise time of both SCA and SDA	t_r	$20+0.1C_b$	300	ns	1
Fall time of both SDA and SCL	t_f	$20+0.1C_b$	300	ns	1
Setup time for STOP condition	$t_{SU;DAT}$	0.6	—	µs	
Bus free time between a STOP and START condition	t_{BUF}	1.3	—	µs	
Capacitive load for each bus line	C_b	—	400	pF	
Noise margin at the LOW level for each connected device (including hysteresis)	V_{nL}	$0.1V_{DD}$	—	V	
Noise margin fat the HIGH level for each connected device (including hysteresis)	V_{nH}	$0.2V_{DD}$	—	V	

Note 1: C_b is the capacitance of one bus line in pF.

Note 2: I/O pins of fast mode devices must not obstruct the SDA and SCL lines if V_{DD} is switched off.

Note 3: The maximum $t_{HD;DAT}$ has only to be met if the device does not stretch the Low period (t_{LOW}) of the SCL signal.

Note 4: A fast mode device can be used on the same bus as standard mode devices, but the requirement of $t_{SU;DAT} \geq 250$ns must be met. This will automatically be the case if the device does not stretch the Low period of the SCL signal. If such a device does stretch the Low period of the SCL signal, it must output the next data bit to the SDA line $t_{rmax} = t_{SU;DAT}$ = 1000+250 = 1250ns before the SCL line is released.

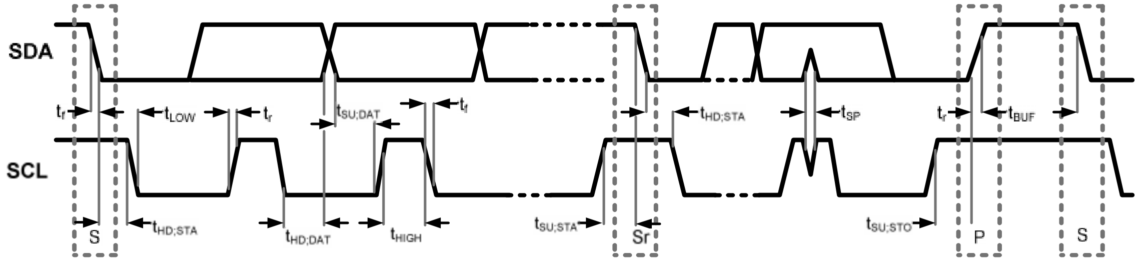

Figure 3: I^2C Timing

3.3. I^2C Read/Write vs. Register Read/Write

When using I^2C to communicate to a Crystal Touch touch screen, there are two pairs of read/write commands used. The first is the I^2C read/write command which is indicated by the lowest bit of the first byte sent to the touch panel. The second is the Register Access Protocol (RAP) read/write command which is indicated by the lowest bit of the second byte sent to the touch panel. To avoid confusion, these are referred to as the I^2C read/write or the RAP read/write for the rest of this document.

3.4. Slave Address

Crystal Touch has a predefined I^2C Slave address of 0x2A. This is shifted into the upper seven bits of the first transmitted byte. The lowest bit is used to indicate an I^2C read (1) or a write (0). Therefore, the first transmitted byte is either (0x2A << 1) | 0x01 = 0x55 for a read, or (0x2A << 1) = 0x54 for a write.

3.5. Register Access Protocol

Crystal Touch uses a communication method known as Register Access Protocol (RAP). RAP only has two commands: READ and WRITE. The registers have five bit addresses ranging from 0x00 to 0x1F. Commands are formatted as follows.

Table 3

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Read (1) Write (0)	Register Address Bit 4	Register Address Bit 3	Register Address Bit 2	Register Address Bit 1	Register Address Bit 0

3.6. Register Addresses

Only nine of the thirty two registers are used. The rest are reserved and should not be read or written.

Table 4

Register Address	Description
0x02	Status
0x03	Power and Reset
0x04	Control
0x07	Calibrate
0x0A	Z Idle
0x14	X Position Low Byte
0x15	Y Position Low Byte
0x16	X and Y Position High Nibbles
0x17	Z Level

Refer to the Crystal Touch User's Manual for more information on the available bits in each register.

3.7. Reading From a Register

Reading a register requires two I^2C packets. The Master sends a START followed by the seven bit Slave address and an I^2C write command (0x54). Then it sends the register address being accessed and a RAP read command. The Master then terminates the I^2C packet by sending a STOP.

The second packet begins with a START followed by the Slave address and an I^2C read command. For the second byte the touch screen clocks the eight bits of data to the Master. The Master then terminates the I^2C packet by sending a STOP.

Example: Read the contents of the Status register (0x02):

As shown in section 3.5 Register Access Protocol, *a RAP Read command is formatted as 101xxxxx. When OR'ed with the register address 0x02, the value is 0xA2:*

0xA0 | 0x02 = 0xA2

Start the packet. Send 0x54 to indicate a Slave read followed by 0xA2. Stop the packet. Start another packet. Send 0x55 to indicate Slave write. The touch screen then transmits the 8 bits of data. Stop the packet.

Table 5

Byte	Data	Source
	START	Master
1	0x54	Master
2	0xA2	Master
	STOP	Master
	START	Master
3	0x55	Master
4	Contents of register 0x02	Slave
	STOP	Master

3.8. Writing To a Register

Writing a register requires a single I^2C packet. The Master sends a START followed by the seven bit Slave address and an I^2C write command (0x54). Then it sends the register address being accessed and a RAP write command. Finally, the Master sends the byte being written. The Master then terminates the I^2C packet by sending a STOP.

Example: Write the value 0x03 to the Control register (0x04):

As shown in section 3.5 Register Access Protocol*, a RAP Write command is formatted as 100xxxxx. When OR'ed with the register address 0x04, the value is 0x84:*

0x80 | 0x04 = 0x84

Start the packet. Send 0x54 to indicate a Slave write. Send 0x84 to write to register 0x04. Then send the 8 bits of data. Stop the packet.

Table 6

Byte	Data	Source
	START	Master
1	0x54	Master
2	0x84	Master
3	0x03	Master
	STOP	Master

NOTE: Many registers contain reserved bits that are not used for this application. Changing any reserved bits could result in unexpected behavior. When writing to a register, always use a read-modify-write algorithm that preserves the values of all reserved bits.

4. Sleep Mode Errata

Older versions of the Cirque touch panel controller have an intermittent failure when coming out of sleep mode while using the I^2C communications interface. This failure causes the controller to sometimes lock up when switching from Sleep mode to Active mode. When this occurs, a power cycle is required to reset the controller chip.

When looking at the last two digits of the marking on the Cirque controller chip, any chips that end with B2 have this problem. Any chips marked with B4 or later do not.

This problem only affects sleep mode. If sleep mode is never invoked, this failure will never occur.

This failure does not occur under any circumstances when using the SPI interface.