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

STMPE610

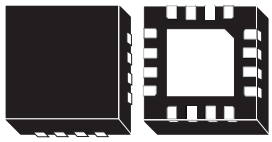
S-Touch®: advanced touchscreen controller with 6-bit port expander

Features

- 6 GPIOs
- 1.8 - 3.3 V operating voltage
- Integrated 4-wire touchscreen controller
- Interrupt output pin
- Wakeup feature on each I/O
- SPI and I^2C interface
- Up to 2 devices sharing the same bus in I^2C mode (1 address line)
- 6-input 12-bit ADC
- 128-depth buffer touchscreen controller
- Touchscreen movement detection algorithm
- 25 kV air-gap ESD protection (system level)
- 4 kV HBM ESD protection (device level)

Applications

- Portable media players
- Game consoles
- Mobile and smartphones
- GPS



QFN16
(3 x 3 mm)

Description

The STMPE610 is a GPIO (general purpose input/output) port expander able to interface a main digital ASIC via the two-line bidirectional bus (I^2C). A separate GPIO expander is often used in mobile multimedia platforms to solve the problems of the limited amount of GPIOs typically available on the digital engine.

The STMPE610 offers great flexibility, as each I/O can be configured as input, output or specific functions. The device has been designed with very low quiescent current and includes a wakeup feature for each I/O, to optimize the power consumption of the device.

A 4-wire touchscreen controller is built into the STMPE610. The touchscreen controller is enhanced with a movement tracking algorithm to avoid excessive data, 128 x 32 bit buffer and a programmable active window feature.

Table 1. Device summary

Order code	Package	Packaging
STMPE610QTR	QFN16	Tape and reel

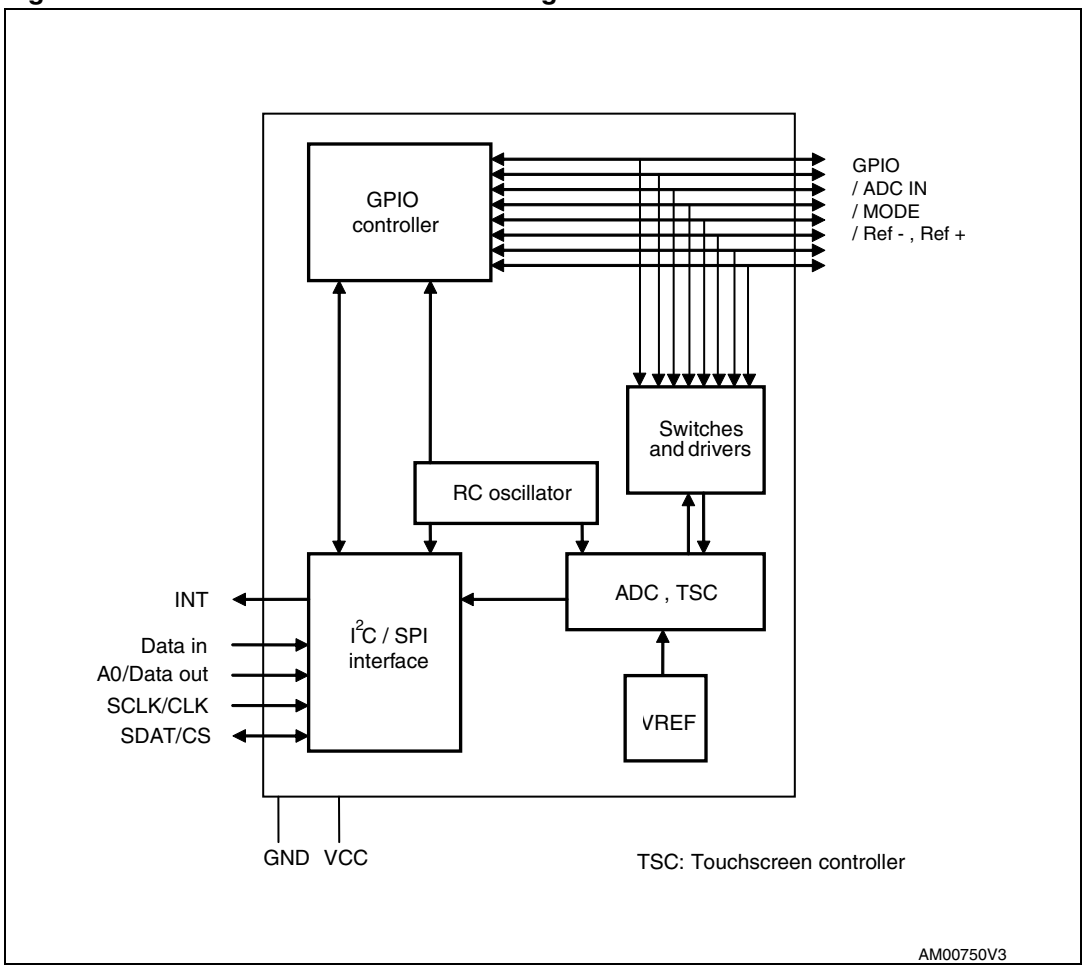
Contents

1 STMPE610 functional overview

The STMPE610 consists of the following blocks:

- I^2C and SPI interface
- Analog-to-digital converver (ADC)
- Touchscreen controller (TSC)
- Driver and switch control unit
- GPIO controller

Figure 1. STMPE610 functional block diagram



2 Pin configuration and functions

Figure 2. STMPE610 pin configuration (top through view)

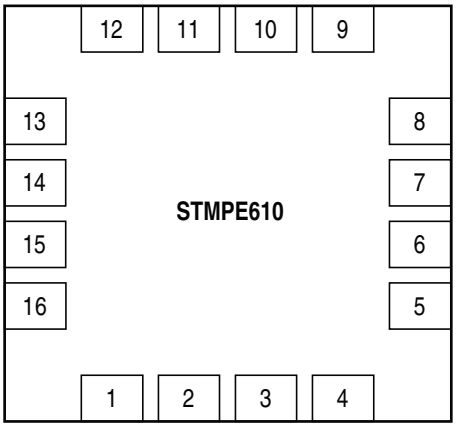


Table 2. Pin assignments

Pin	Name	Function
1	Y-	Y-/GPIO-7
2	INT	Interrupt output (V_{CC} domain, open drain)
3	A0/Data Out	I^2C address in Reset, Data out in SPI mode (V_{CC} domain)
4	SCLK	I^2C/SPI clock (V_{CC} domain)
5	SDAT	I^2C data/SPI CS (V_{CC} domain)
6	V_{CC}	1.8 –3.3 V supply voltage
7	Data in	SPI Data In (V_{CC} domain)
8	NC	–
9	Mode	MODE In RESET state, MODE selects the type of serial interface "0" - I^2C "1" - SPI
10	GND	Ground
11	IN2	IN2/GPIO-2
12	IN3	IN3/GPIO-3
13	X+	X+/GPIO-4
14	Vio	Supply for touchscreen driver and GPIO
15	Y+	Y+/GPIO-5
16	X-	X-/GPIO-6

2.1 Pin functions

The STMPE610 is designed to provide maximum features and flexibility in a very small pin-count package. Most of the pins are multi-functional. The following table shows how to select the pin's function.

Table 3. IN2, IN3 pin configuration

Pin / control register	GPIO_AF = 1	GPIO_AF = 0	
	ADC control 1 bit 1 = don't care	ADC control 1 bit 1 = 0	ADC control 1 bit 1 = 1
IN2	GPIO-2	ADC	External reference +
IN3	GPIO-3	ADC	External reference -

Table 4. X, Y pin configuration

Pin / control register	GPIO_AF = 1	GPIO_AF = 0	
	TSC control 1 bit 0 = don't care	TSC control 1 bit 0 = 0	TSC control 1 bit 0 = 1
X+	GPIO-4	ADC	TSC X+
Y+	GPIO-5	ADC	TSC Y+
X-	GPIO-6	ADC	TSC X-
Y-	GPIO-7	ADC	TSC Y-

3 I²C and SPI interface

3.1 Interface selection

The STMPE610 interfaces with the host CPU via a I²C or SPI interface. The pin IN_1 allows the selection of interface protocol at reset state.

Figure 3. STMPE610 interface

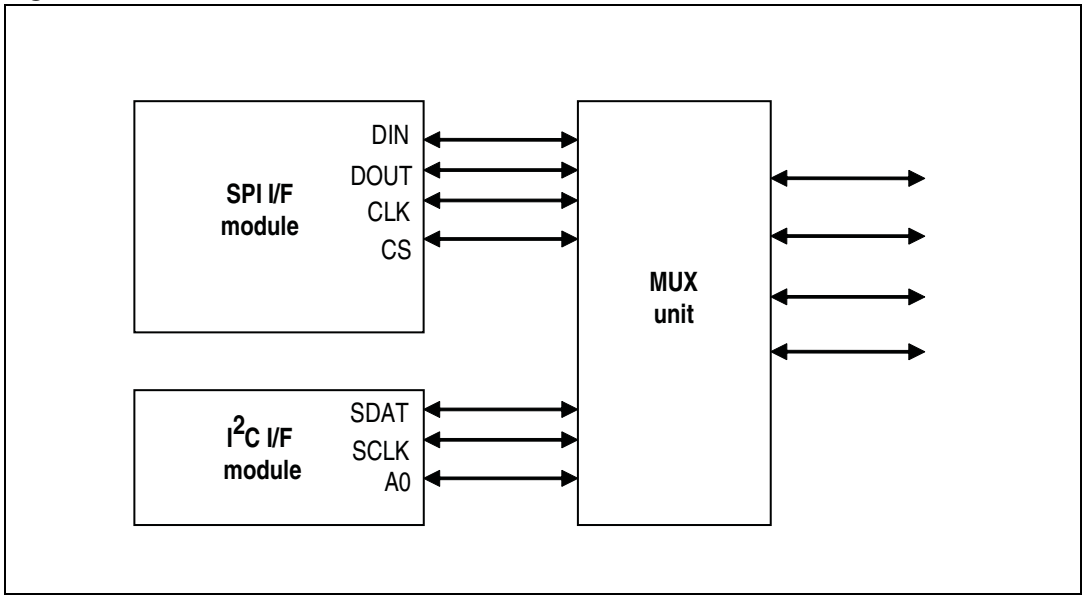


Table 5. Interface selection pins

Pin	I²C function	SPI function	Reset state
3	Address 0	Data out	CPHA for SPI
4	Clock	Clock	–
5	SDATA	CS	CPOL_N for SPI
7	–	Data in	–
9	MODE	I²C set to '0'	Set to '1' for SPI

4 I²C interface

The addressing scheme of STMPE610 is designed to allow up to 2 devices to be connected to the same I²C bus.

Figure 4. STMPE610 I²C interface

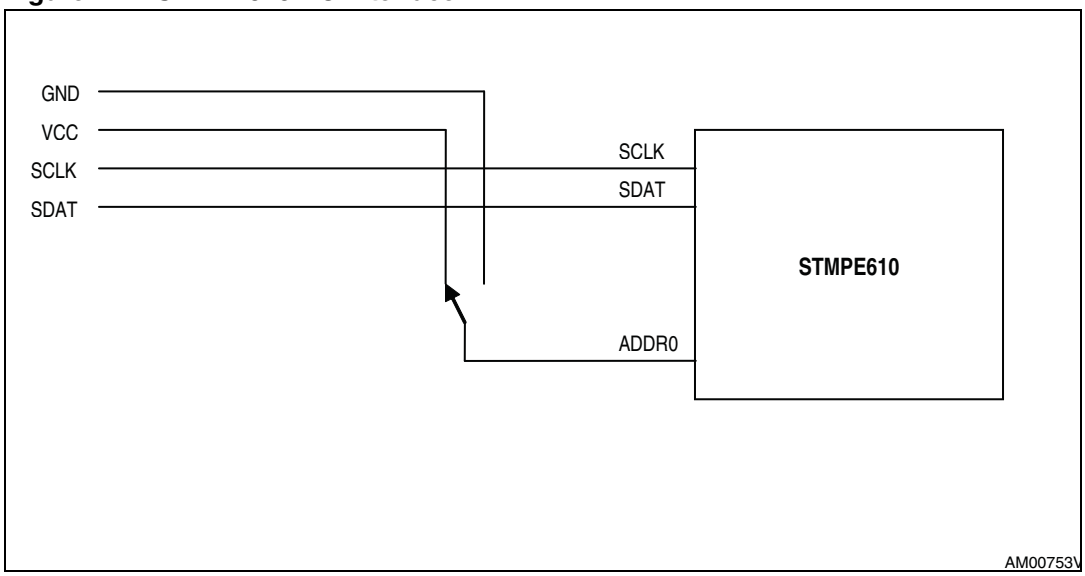


Table 6. I²C address

ADDR0	Address
0	0 x 82
1	0 x 88

For the bus master to communicate to the slave device, the bus master must initiate a Start condition and be followed by the slave device address. Accompanying the slave device adress, is a read/write bit (R/$\overline{W}$). The bit is set to 1 for read and 0 for write operation. If a match occurs on the slave device address, the corresponding device gives an acknowledge on the SDA during the 9th bit time. If there is no match, it deselects itself from the bus by not responding to the transaction.

Figure 5. I²C timing diagram

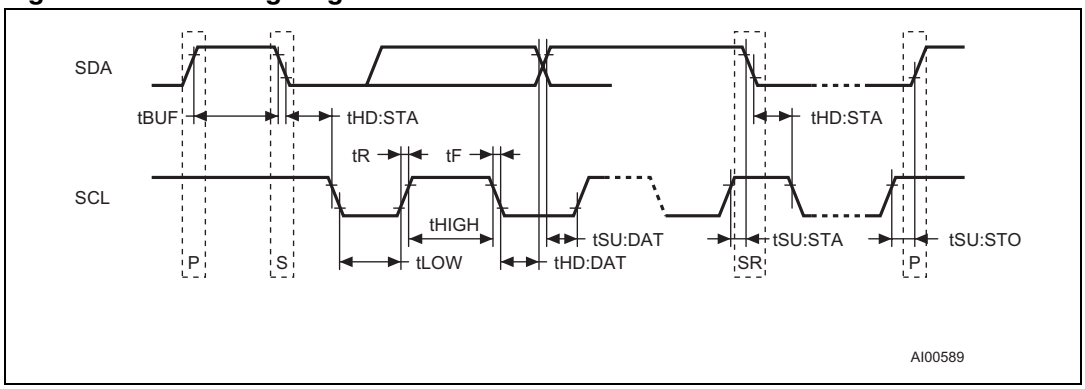


Table 7. I^2C timing

Symbol	Parameter	Min	Typ	Max	Unit
f_{SCL}	SCL clock frequency	0	–	400	kHz
t_{LOW}	Clock low period	1.3	–	–	µs
t_{HIGH}	Clock high period	600	–	–	ns
t_F	SDA and SCL fall time	–	–	300	ns
$t_{HD:STA}$	START condition hold time (after this period the first clock is generated)	600	–	–	ns
$t_{SU:STA}$	START condition setup time (only relevant for a repeated start period)	600	–	–	ns
$t_{SU:DAT}$	Data setup time	100	–	–	ns
$t_{HD:DAT}$	Data hold time	0	–	–	µs
$t_{SU:STO}$	STOP condition setup time	600	–	–	ns
t_{BUF}	Time the bus must be free before a new transmission can start	1.3	–	–	µs

4.1 I^2C features

The features that are supported by the I^2C interface are listed below:

- I^2C slave device
- Operates at 1.8 V
- Compliant to Philips I^2C specification version 2.1
- Supports standard (up to 100 Kbps) and fast (up to 400 Kbps) modes

Start condition

A Start condition is identified by a falling edge of SDATA while SCLK is stable at high state. A Start condition must precede any data/command transfer. The device continuously monitors for a Start condition and does not respond to any transaction unless one is encountered.

Stop condition

A Stop condition is identified by a rising edge of SDATA while SCLK is stable at high state. A Stop condition terminates communication between the slave device and the bus master. A read command that is followed by NoAck can be followed by a Stop condition to force the slave device into idle mode. When the slave device is in idle mode, it is ready to receive the next I^2C transaction. A Stop condition at the end of a write command stops the write operation to registers.

Acknowledge bit

The acknowledge bit is used to indicate a successful byte transfer. The bus transmitter releases the SDATA after sending eight bits of data. During the ninth bit, the receiver pulls

the SDATA low to acknowledge the receipt of the eight bits of data. The receiver may leave the SDATA in high state if it does not acknowledge the receipt of the data.

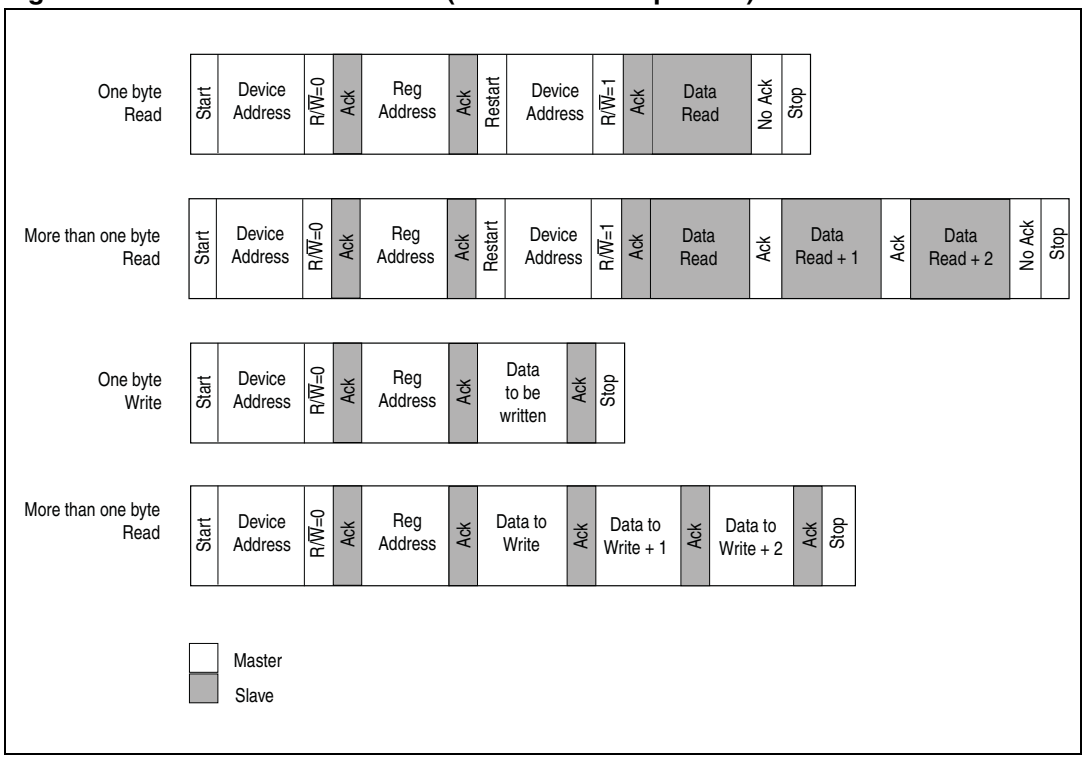
4.2 Data input

The device samples the data input on SDATA on the rising edge of the SCLK. The SDATA signal must be stable during the rising edge of SCLK and the SDATA signal must change only when SCLK is driven low.

Table 8. Operating modes

Mode	Byte	Programming sequence
Read	≥1	Start, Device address, R/$\overline{W}$ = 0, Register address to be read
		Restart, Device address, R/$\overline{W}$ = 1, Data Read, Stop
		If no Stop is issued, the Data Read can be continuously performed. If the register address falls within the range that allows an address auto-increment, then the register address auto-increments internally after every byte of data being read.
Write	≥1	Start, Device address, R/$\overline{W}$ = 0, Register address to be written, Data Write, Stop
		If no Stop is issued, the Data Write can be continuously performed. If the register address falls within the range that allows address auto-increment, then the register address auto-increments internally after every byte of data being written in. For those register addresses that fall within a non-incremental address range, the address will be kept static throughout the entire write operation. Refer to the memory map table for the address ranges that are auto and non-increment.

Figure 6. Read and write modes (random and sequential)



4.3 Read operation

A write is first performed to load the register address into the Address Counter but without sending a Stop condition. Then, the bus master sends a reStart condition and repeats the Device Address with the R/W bit set to 1. The slave device acknowledges and outputs the content of the addressed byte. If no additional data is to be read, the bus master must not acknowledge the byte and terminates the transfer with a Stop condition.

If the bus master acknowledges the data byte, then it can continue to perform the data reading. To terminate the stream of data bytes, the bus master must not acknowledge the last output byte, and be followed by a Stop condition. If the address of the register written into the Address Counter falls within the range of addresses that has the auto-increment function, the data being read will be coming from consecutive addresses, which the internal Address Counter automatically increments after each byte output. After the last memory address, the Address Counter 'rolls-over' and the device continues to output data from the memory address of 0x00. Similarly, for the register address that falls within a non-increment range of addresses, the output data byte comes from the same address (which is the address referred by the Address Counter).

Acknowledgement in read operation

For the above read command, the slave device waits, after each byte read, for an acknowledgement during the ninth bit time. If the bus master does not drive the SDA to a low state, then the slave device terminates and switches back to its idle mode, waiting for the next command.

4.4 Write operations

A write is first performed to load the register address into the Address Counter without sending a Stop condition. After the bus master receives an acknowledgement from the slave device, it may start to send a data byte to the register (referred by the Address Counter). The slave device again acknowledges and the bus master terminates the transfer with a Stop condition.

If the bus master needs to write more data, it can continue the write operation without issuing the Stop condition. Whether the Address Counter autoincrements or not after each data byte write depends on the address of the register written into the Address Counter. After the bus master writes the last data byte and the slave device acknowledges the receipt of the last data, the bus master may terminate the write operation by sending a Stop condition. When the Address Counter reaches the last memory address, it 'rolls-over' to the next data byte write.

5 SPI interface

The SPI interface in STMPE610 uses a 4-wire communication connection (DATA IN, DATA OUT, CLK, CS). In the diagram, “Data in” is referred to as MOSI (master out slave in) and “DATA out” is referred to as MISO (master in slave out).

5.1 SPI protocol definition

The SPI (serial peripheral interface) follows a byte sized transfer protocol. All transfers begin with an assertion of CS_n signal (falling edge). The protocol for reading and writing is different and the selection between a read and a write cycle is dependent on the first captured bit on the slave device. A '1' denotes a read operation and a '0' denotes a write operation. The SPI protocol defined in this section is shown in Figure 3.

The following are the main features supported by this SPI implementation.

- Support of 1 MHz maximum clock frequency.
- Support for autoincrement of address for both read and write.
- Full duplex support for read operation.
- Daisy chain configuration support for write operation.
- Robust implementation that can filter glitches of up to 50 ns on the CS_n and SCL pins.
- Support for all 4 modes of SPI as defined by the CPHA, CPOL bits on SPICON.

5.1.1 Register read

The following steps need to be followed for register read through SPI.

1. Assert CS_n by driving a '0' on this pin.
2. Drive a '1' on the first SCL launch clock on MOSI to select a read operation.
3. The next 7 bits on MOSI denote the 7-bit register address (MSB first).
4. The next address byte can now be transmitted on the MOSI. If the autoincrement bit is set, the following address transmitted on the MOSI is ignored. Internally, the address is incremented. If the autoincrement bit is not set, then the following byte denotes the address of the register to be read next.
5. Read data is transmitted by the slave device on the MISO (MSB first), starting from the launch clock following the last address bit on the MOSI.
6. Full duplex read operation is achieved by transmitting the next address on MOSI while the data from the previous address is available on MISO.
7. To end the read operation, a dummy address of all 0's is sent on MOSI.

5.1.2 Register write

The following steps need to be followed for register write through SPI.

1. Assert CS_n by driving a '0' on this pin.
2. Drive a '0' on the first SCL launch clock on MOSI to select a write operation.
3. The next 7 bits on MOSI denote the 7-bit register address (MSB first).
4. The next byte on the MOSI denotes data to be written.
5. The following transmissions on MOSI are considered byte-sized data. The register address to which the following data is written depends on whether the autoincrement bit in the SPICON register is set. If this bit has been set previously, the register address is incremented for data writes.

5.1.3 Termination of data transfer

A transfer can be terminated before the last launch edge by deasserting the CS_n signal. If the last launch clock is detected, it is assumed that the data transfer is successful.

5.2 SPI timing modes

The SPI timing modes are defined by CPHA and CPOL,CPHA and CPOL are read from the "SDAT" and "A0" pins during power-up reset. The following four modes are defined according to this setting.

Table 9. SPI timing modes

CPOL_N (SDAT pin)	CPOL	CPHA (ADDR pin)	Mode
1	0	0	0
1	0	1	1
0	1	0	2
0	1	1	3

The clocking diagrams of these modes are shown in ON reset. The device always operates in mode 0. Once the bits are set in the SPICON register, the mode change takes effect on the next transaction defined by the CS_n pin being deasserted and asserted.

5.2.1 SPI timing definition

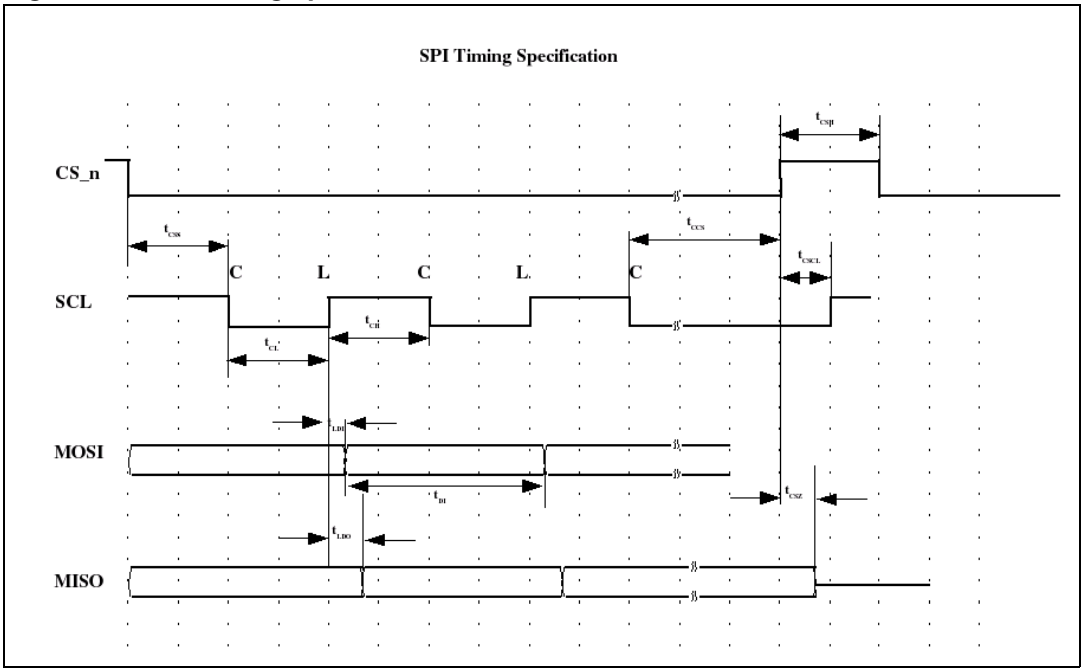
Table 10. SPI timing specification

Symbol	Description	Timing			Unit
		Min	Typ	Max	
t_{CSS}	CS_n falling to first capture clock	1	–	–	µs
t_{CL}	Clock low period	500	–	–	ns
t_{CH}	Clock high period	500	–	–	ns

Table 10. SPI timing specification (continued)

Symbol	Description	Timing			Unit
		Min	Typ	Max	
t_{LDI}	Launch clock to MOSI data valid	–	–	20	ns
t_{LDO}	Launch clock to MISO data valid	–	–	330	µs
t_{DI}	Data on MOSI valid	1	–	–	µs
t_{CCS}	Last clock edge to CS_n high	1	–	–	µs
t_{CSH}	CS_n high period	2	–	–	µs
t_{CSCL}	CS_n high to first clock edge	300	–	–	ns
t_{CSZ}	CS_n high to tri-state on MISO	1	–	–	µs

Figure 7. SPI timing specification



6 STMPE610 registers

This section lists and describes the registers of the STMPE610 device, starting with a register map and then provides detailed descriptions of register types.

Table 11. Register summary map table

Address	Register name	Bit	Type	Reset value	Function
0x00	CHIP_ID	16	R	0x0811	Device identification
0x02	ID_VER	8	R	0x03	Revision number 0x01 for engineering sample 0x03 for final silicon
0x03	SYS_CTRL1	8	R/W	0x00	Reset control
0x04	SYS_CTRL2	8	R/W	0x0F	Clock control
0x08	SPI_CFG	8	R/W	0x01	SPI interface configuration
0x09	INT_CTRL	8	R/W	0x00	Interrupt control register
0x0A	INT_EN	8	R/W	0x00	Interrupt enable register
0x0B	INT_STA	8	R	0x10	interrupt status register
0x0C	GPIO_EN	8	R/W	0x00	GPIO interrupt enable register
0x0D	GPIO_INT_STA	8	R	0x00	GPIO interrupt status register
0x0E	ADC_INT_EN	8	R/W	0x00	ADC interrupt enable register
0x0F	ADC_INT_STA	8	R	0x00	ADC interrupt status register
0x10	GPIO_SET_PIN	8	R/W	0x00	GPIO set pin register
0x11	GPIO_CLR_PIN	8	R/W	0x00	GPIO clear pin register
0x12	GPIO_MP_STA	8	R/W	0x00	GPIO monitor pin state register
0x13	GPIO_DIR	8	R/W	0x00	GPIO direction register
0x14	GPIO_ED	8	R/W	0x00	GPIO edge detect register
0x15	GPIO_RE	8	R/W	0x00	GPIO rising edge register
0x16	GPIO_FE	8	R/W	0x00	GPIO falling edge register
0x17	GPIO_AF	8	R/W	0x00	Alternate function register
0x20	ADC_CTRL1	8	R/W	0x9C	ADC control
0x21	ADC_CTRL2	8	R/W	0x01	ADC control
0x22	ADC_CAPT	8	R/W	0xFF	To initiate ADC data acquisition
0x30	ADC_DATA_CH0	16	R	0x0000	ADC channel 0
0x32	ADC_DATA_CH1	16	R	0x0000	ADC channel 1

Table 11. Register summary map table (continued)

Address	Register name	Bit	Type	Reset value	Function
0x38	ADC_DATA_CH4	16	R	0x0000	ADC channel 4
0x3A	ADC_DATA_CH5	16	R	0x0000	ADC channel 5
0x3C	ADC_DATA_CH6	16	R	0x0000	ADC channel 6
0x3E	ADC_DATA_CH7	16	R	0x0000	ADC channel 7
0x40	TSC_CTRL	8	R/W	0x90	4-wire touchscreen controller setup
0x41	TSC_CFG	8	R/W	0x00	Touchscreen controller configuration
0x42	WDW_TR_X	16	R/W	0x0FFF	Window setup for top right X
0x44	WDW_TR_Y	16	R/W	0x0FFF	Window setup for top right Y
0x46	WDW_BL_X	16	R/W	0x0000	Window setup for bottom left X
0x48	WDW_BL_Y	16	R/W	0x0000	Window setup for bottom left Y
0x4A	FIFO_TH	8	R/W	0x00	FIFO level to generate interrupt
0x4B	FIFO_STA	8	R/W	0x20	Current status of FIFO
0x4C	FIFO_SIZE	8	R	0x00	Current filled level of FIFO
0x4D	TSC_DATA_X	16	R	0x0000	Data port for touchscreen controller data access
0x4F	TSC_DATA_Y	16	R	0x0000	Data port for touchscreen controller data access
0x51	TSC_DATA_Z	8	R	0x0000	Data port for touchscreen controller data access
0x52	TSC_DATA_XYZ	32	R	0x00000000	Data port for touchscreen controller data access
0x56	TSC_FRACT_XYZ	8	RW	0x00	Select the range and accuracy of the pressure measurement
0x57	TSC_DATA	8	R	0x00	Data port for touchscreen controller data access
0x58	TSC_I_DRIVE	8	R/W	0x00	Touchscreen controller drive I
0x59	TSC_SHIELD	8	R/W	0x00	Touchscreen controller shield

7 System and identification registers

Table 12. System and identification registers map

Address	Register name	Bit	Type	Reset	Function
0x00	CHIP_ID	16	R	0x0811	Device identification
0x02	ID_VER	8	R	0x03	Revision number 0x01 for engineering sample 0x03 for final silicon
0x03	SYS_CTRL1	8	R/W	0x00	Reset control
0x04	SYS_CTRL2	8	R/W	0x0F	Clock control
0x08	SPI_CFG	8	R/W	0x01	SPI interface configuration

CHIP_ID — Device identification

Address: 0x00

Type: R

Reset: 0x0811

Description: 16-bit device identification

ID_VER — Revision number

Address: 0x02

Type: R

Reset: 0x03

Description: 16-bit revision number

SYS_CTRL1 — Reset control

7	6	5	4	3	2	1	0
RESERVED						SOFT_RESET	HIBERNATE

Address: 0x03

Type: R/W

Reset: 0x00

Description: The reset control register enables to reset the device

[7:2] RESERVED

[1] SOFT_RESET: Reset the STMPE610 using the serial communication interface

[0] HIBERNATE: Force the device into hibernation mode.
Forcing the device into hibernation mode by writing ‘1’ to this bit would disable the hot-key feature. If the hot-key feature is required, use the default auto-hibernation mode.

SYS_CTRL2 Clock control

7	6	5	4	3	2	1	0
-	-	-	-	RESERVED	GPIO_OFF	TSC_OFF	ADC_OFF

Address: 0x04

Type: R/W

Reset: 0x0F

Description: This register enables to switch off the clock supply

[7:3] RESERVED

[2] GPIO_OFF: Switch off the clock supply to the GPIO
1: Switches off the clock supply to the GPIO

[1] TSC_OFF: Switch off the clock supplyto the touchscreen controller
1: Switches off the clock supply to the touchscreen controller

[0] ADC_OFF: Switch off the clock supply to the ADC
1: Switches off the clock supply to the ADC

SPI_CFG SPI interface configuration

7	6	5	4	3	2	1	0
RESERVED					AUTO_INCR	SPI_CLK_MOD1	SPI_CLK_MOD0

Address: 0x08

Type: R/W

Reset: 0x01

Description: SPI interface configuration register

[7:3] RESERVED

[2] AUTO_INCR:
This bit defines whether the SPI transaction follows an addressing scheme that internally autoincrements or not

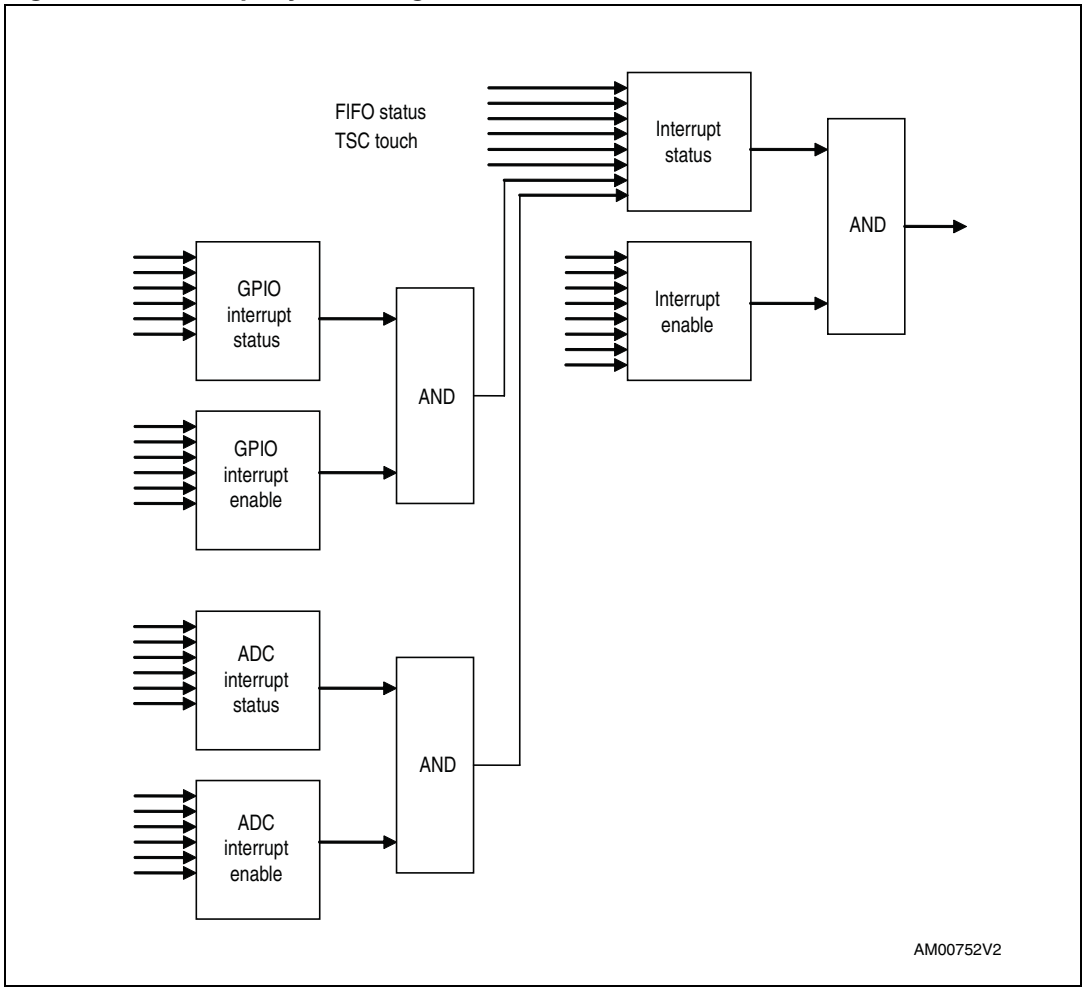
[1] SPI_CLK_MOD1:
This bit reflects the value of the SCAD/A0 pin during power-up reset

[0] SPI_CLK_MOD0:
This bit reflects the value of the SCAD/A0 pin during power-up reset

8 Interrupt system

The STMPE610 uses a 2-tier interrupt structure. The ADC interrupts and GPIO interrupts are ganged as a single bit in the “interrupt status register”. The interrupts from the touchscreen controller can be seen directly in the interrupt status register.

Figure 8. Interrupt system diagram



INT_CTRL Interrupt control register

7	6	5	4	3	2	1	0
RESERVED					INT_POLARITY	INT_TYPE	GLOBAL_INT

Address: 0x09

Type: R/W

Reset: 0x00

Description: The interrupt control register is used to enable the interruption from a system-related interrupt source to the host.

[7:3] RESERVED

[2] INT_POLARITY: This bit sets the INT pin polarity
1: Active high/rising edge
0: Active low/falling edge

[1] INT_TYPE: This bit sets the type of interrupt signal required by the host
1: Edge interrupt
0: Level interrupt

[0] GLOBAL_INT: This is master enable for the interrupt system
1: Global interrupt
0: Stops all interrupts

INT_EN Interrupt enable register

7	6	5	4	3	2	1	0
GPIO	ADC	RESERVED	FIFO_EMPTY	FIFO_FULL	FIFO_0FLOW	FIFO_TH	TOUCH_DET

Address: 0x0A

Type: R/W

Reset: 0x00

Description: The interrupt enable register is used to enable the interruption from a system related interrupt source to the host.

[7] GPIO: Any enabled GPIO interrupts

[6] ADC: Any enabled ADC interrupts

[5] RESERVED

[4] FIFO_EMPTY: FIFO is empty

[3] FIFO_FULL: FIFO is full

[2] FIFO_OFLOW: FIFO is overflowed

[1] FIFO_TH: FIFO is equal or above threshold value

[0] TOUCH_DET: Touch is detected

INT_STA Interrupt status register

7	6	5	4	3	2	1	0
GPIO	ADC	RESERVED	FIFO_EMPTY	FIFO_FULL	FIFO_OFLOW	FIFO_TH	TOUCH_DET

Address: 0x0B

Type: R

Reset: 0x10

Description: The interrupt status register monitors the status of the interruption from a particular interrupt source to the host. Regardless of whether the INT_EN bits are enabled, the INT_STA bits are still updated. Writing '1' to this register clears the corresponding bits. Writing '0' has no effect.

[7] GPIO: Any enabled GPIO interrupts

[6] ADC: Any enabled ADC interrupts

[5] RESERVED

[4] FIFO_EMPTY: FIFO is empty

[3] FIFO_FULL: FIFO is full

[2] FIFO_OFLOW: FIFO is overflowed

[1] FIFO_TH: FIFO is equal or above threshold value.
This bit is set when FIFO level equals to threshold value. It will only be asserted again if FIFO level drops to < threshold value, and increased back to threshold value.

[0] TOUCH_DET: Touch is detected

GPIO_INT_EN GPIO interrupt enable register

7	6	5	4	3	2	1	0
IEG[x]							

Address: 0x0C

Type: R/W

Reset: 0x10

Description: The interrupt status register monitors the status of the interruption from a particular interrupt source to the host. Regardless of whether the IER bits are enabled, the ISR bits are still updated. Writing '1' to this register clears the corresponding bits. Writing '0' has no effect.

[7:0] IEG[x]: Interrupt enable GPIO mask (where x = 7 to 0)
1: Writing ‘1’ to the IE[x] bit enables the interruption to the host

GPIO_INT_STA GPIO interrupt status register

7	6	5	4	3	2	1	0
ISG[x]							

Address: 0x0D

Type: R/W

Reset: 0x00

Description: The GPIO interrupt status register monitors the status of the interruption from a particular GPIO pin interrupt source to the host. Regardless of whether or not the GPIO_STA bits are enabled, the GPIO_STA bits are still updated. The ISG[7:0] bits are the interrupt status bits corresponding to the GPIO[7:0] pins. Writing '1' to this register clears the corresponding bits. Writing '0' has no effect.

[7:0] ISG[x]: GPIO interrupt status (where x = 7 to 0)
Read:
Interrupt status of the GPIO[x]. Reading the register will clear any bits that have been set to '1'
Write:
Writing to this register has no effect

9 Analog-to-digital converter

An 8-input,12-bit analog-to-digital converter (ADC) is integrated in the STMPE610. The ADC can be used as a generic analog-to-digital converter, or as a touchscreen controller capable of controlling a 4-wire resistive touchscreen.

Table 13. ADC controller register summary table

Address	Register name	Size	Description
0x20	ADC_CTRL1	8	ADC control
0x21	ADC_CTRL2	8	ADC control
0x22	ADCCapture	8	To initiate ADC data acquisition
0x30	ADC_DATA_CH0	8	ADC channel 0 (IN3/GPIO-3)
0x32	ADC_DATA_CH1	8	ADC channel 1 (IN2/GPIO-2)
0x38	ADC_DATA_CH4	8	ADC channel 4 (TSC)
0x3A	ADC_DATA_CH5	8	ADC channel 5 (TSC)
0x3C	ADC_DATA_CH6	8	ADC channel 6 (TSC)
0x3E	ADC_DATA_CH7	8	ADC channel 7 (TSC)

ADC_CTRL1 ADC control 1

7	6	5	4	3	2	1	0
RESERVED	SAMPLE_TIME2	SAMPLE_TIME1	SAMPLE_TIME0	MOD_12B	RESERVED	REF_SEL	RESERVED

Address: 0x20

Type: R/W

Reset: 0x9C

Description: ADC control register

[7] RESERVED

[6:4] SAMPLE_TIMEn: ADC conversion time in number of clock
000: 36
001: 44
010: 56
011: 64
100: 80
101: 96
110: 124
111: Not valid

[3] MOD_12B: Selects 10 or 12-bit ADC operation
1: 12 bit ADC
0: 10 bit ADC

[2] RESERVED

[1] REF_SEL: Selects between internal or external reference for the ADC
1: External reference
0: Internal reference

[0] RESERVED

ADC_CTRL2 ADC control 2

7	6	5	4	3	2	1	0
RESERVED						ADC_FREQ_1	ADC_FREQ_0

Address: 0x21

Type: R/W

Reset: 0x01

Description: ADC control.

[7] RESERVED

[6] RESERVED

[5] RESERVED

[4] RESERVED

[3] RESERVED

[2] RESERVED