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

MTCH6303

MTCH6303 Projected Capacitive Touch Controller Data Sheet

Description

Microchip's MTCH6303 is an innovative turnkey projected capacitive touch controller that provides multi-touch coordinates as well as a readymade multi-finger surface gesture suite. MTCH6303 brings modern user interface (UI) elements – such as pinch and zoom, multi-finger scrolling, and swipes – to any embedded design, with minimal host requirements.

The MTCH6303's advanced signal processing provides noise-avoidance techniques and predictive tracking for ten fingers, typically at 100 Hz each for five touches. It also combines with Microchip's MTCH652 High-voltage Line Driver to achieve a superior signal-to-noise ratio (SNR) for outstanding touch performance in noisy environments (refer to www.microchip.com/MTCH652). These capabilities are critical in demanding environments such as industrial controls, home and office automation with security control panels, thermostat, printers and lighting controls, and various consumer applications including exercise equipment and audio systems.

Features

- Multi-Touch up to Ten Touches
- Five Touches Typically at 100 Hz+ Each
- 27RX x 19TX Channels Support Approximately 8" Touch Screens (larger possible)
- Combines with MTCH652 High-Voltage Driver for Superior Signal-to-Noise Ratio (SNR)
- Integrated Single and Multi-finger Gesture Recognition Suite including Taps, Swipes, Scrolling, Pinching and Zooming
- Advanced Processing Provides Noise Avoidance Techniques
- USB and I^2C™ Communication
- Supports 3D Gestures up to 20 cm when Combined with the MGC3130 GestIC® Controller

Power Management

Example:

- 27RX 19TX Sensor
  - 27 mA full-scan rate
  - 1 mA reduced-scan rate

Applications

- Touch screen designs and touch pads that require cost effective, easy to integrate, fast time to market PCAP touch solutions
- Perfect for touch screens over displays, control panels, keypads and many other input devices
- Targeting the industrial, medical, home and office automation, and consumer markets

TABLE 1: MTCH6303 SOLUTION PART NUMBERS

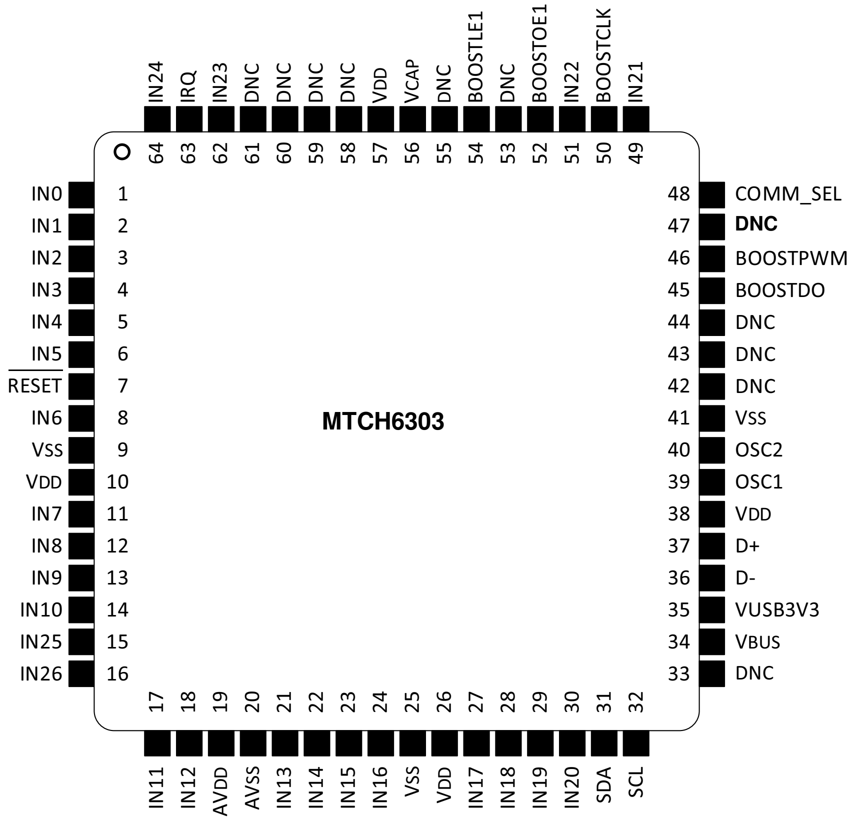
Device	Pin Count	Package Types	Touch Channels	Features
MTCH6303-I/PT	64	10 x10 mm TQFP	Up to 27 RX	Multi-touch, up to 8" sensors
MTCH6303-I/RG		9 x 9 mm QFN		
*MTCH652-I/SO	28	7.5 mm SOIC	Up to 19 TX	1.8 – 5.5V input, 6V – 18V configurable output
*MTCH652-I/SS		5.3 mm SSOP		
*MTCH652-I/MV		4 x 4 mm UQFN		

Note: *One MTCH652 high-voltage driver (boost) is required with MTCH6303.

Note: The MTCH6303 devices are pre-programmed with a Library Loader (bootloader) only. Refer to **Section 8.0, Firmware update** for more details.

PIN DIAGRAM

FIGURE 1: MTCH6303 64-PIN DIAGRAM TQFP/QFN



 Preliminary © 2015 Microchip Technology Inc.

PIN ALLOCATION TABLE

TABLE 2: MTCH6303 PINOUT DESCRIPTION

Name	Pin	Description
IN0	1	IN 0 – 5
IN1	2	
IN2	3	
IN3	4	
IN4	5	
IN5	6	
$\overline{\text{RESET}}$	7	Reset
IN6	8	IN 6
V_{SS}	9	Ground
V_{DD}	10	Power Supply Input
IN7	11	IN 7 – 10
IN8	12	
IN9	13	
IN10	14	
IN25	15	IN 25 – 26
IN26	16	
IN11	17	IN 11 – 12
IN12	18	
AV_{DD}	19	Positive supply for analog modules. This pin must be connected at all times.
AV_{SS}	20	Ground reference for analog modules
IN13	21	IN 13 – 16
IN14	22	
IN15	23	
IN16	24	
V_{SS}	25	Ground
V_{DD}	26	Power Supply Input
IN17	27	IN 17 – 20
IN18	28	
IN19	29	
IN20	30	
SDA	31	I^2C™ Data
SCL	32	I^2C Clock

TABLE 2: MTCH6303 PINOUT DESCRIPTION (CONTINUED)

Name	Pin	Description
DNC	33	Do not connect any signal to these pins.
	42	
	43	
	44	
	47	
	53	
	55	
	58	
	59	
	60	
	61	
VBUS	34	USB Bus Power Monitor
VUSB3V3	35	USB internal transceiver supply. If the USB module is not used, this pin must be connected to VDD.
D-	36	USB D-
D+	37	USB D+
VDD	38	Power Supply Input
OSC1	39	Oscillator Pin 1
OSC2	40	Oscillator Pin 2
VSS	41	Ground
BOOSTDO	45	MTCH652 DO output/DIN Input
BOOSTPWM	46	MTCH652 PWM Out/OSCIN input
COMM_SEL	48	Communication Select Pin (VDD = I^{2}C™, VSS = USB)
IN21	49	IN 21
BOOSTCLK	50	MTCH652 CLK Output
IN22	51	IN 22
BOOSTOE1	52	MTCH652 OE Output 1
BOOSTLE1	54	MTCH652 LE Output 1
VCAP	56	Capacitor for Internal Voltage Regulator
VDD	57	Power Supply Input
IN23	62	IN 23
IRQ	63	I^{2}C Interrupt
IN24	64	IN 24
MGC_TS	42	Gesture Transfer Status
MGC_SDA	43	Gesture I^{2}C Data
MGC_SCL	44	Gesture I^{2}C Clock
MGC_MCLR	61	Gesture Reset
MGC_MODE	60	Gesture Mode Control
MGC_SYNC	47	Gesture Sync

© 2015 Microchip Technology Inc.

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

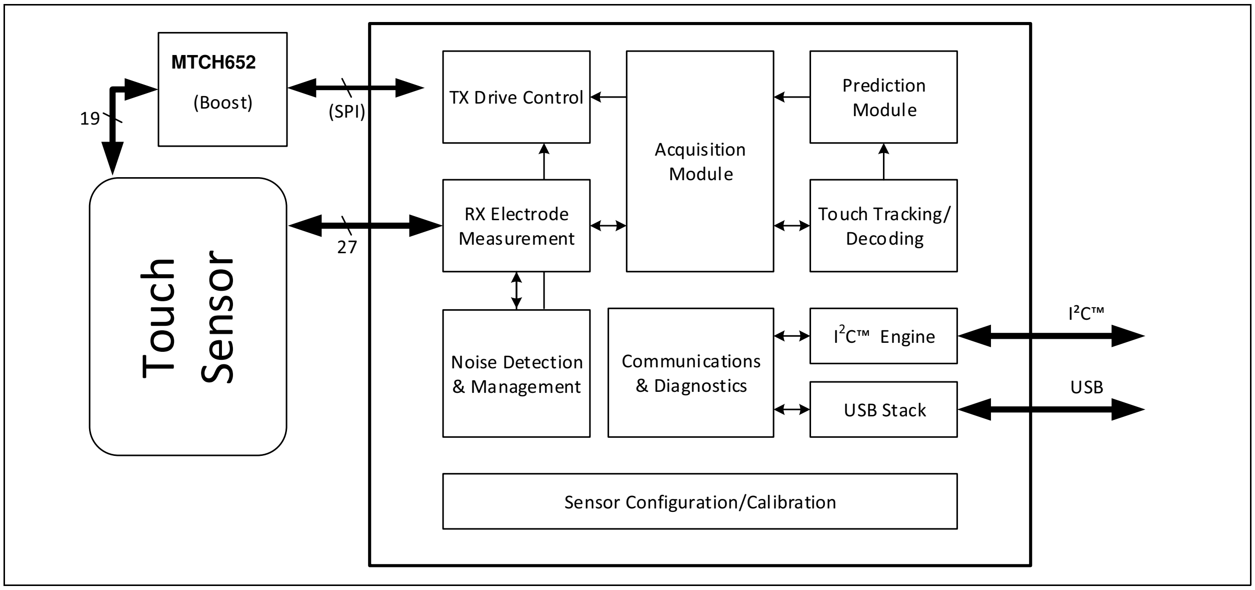
When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

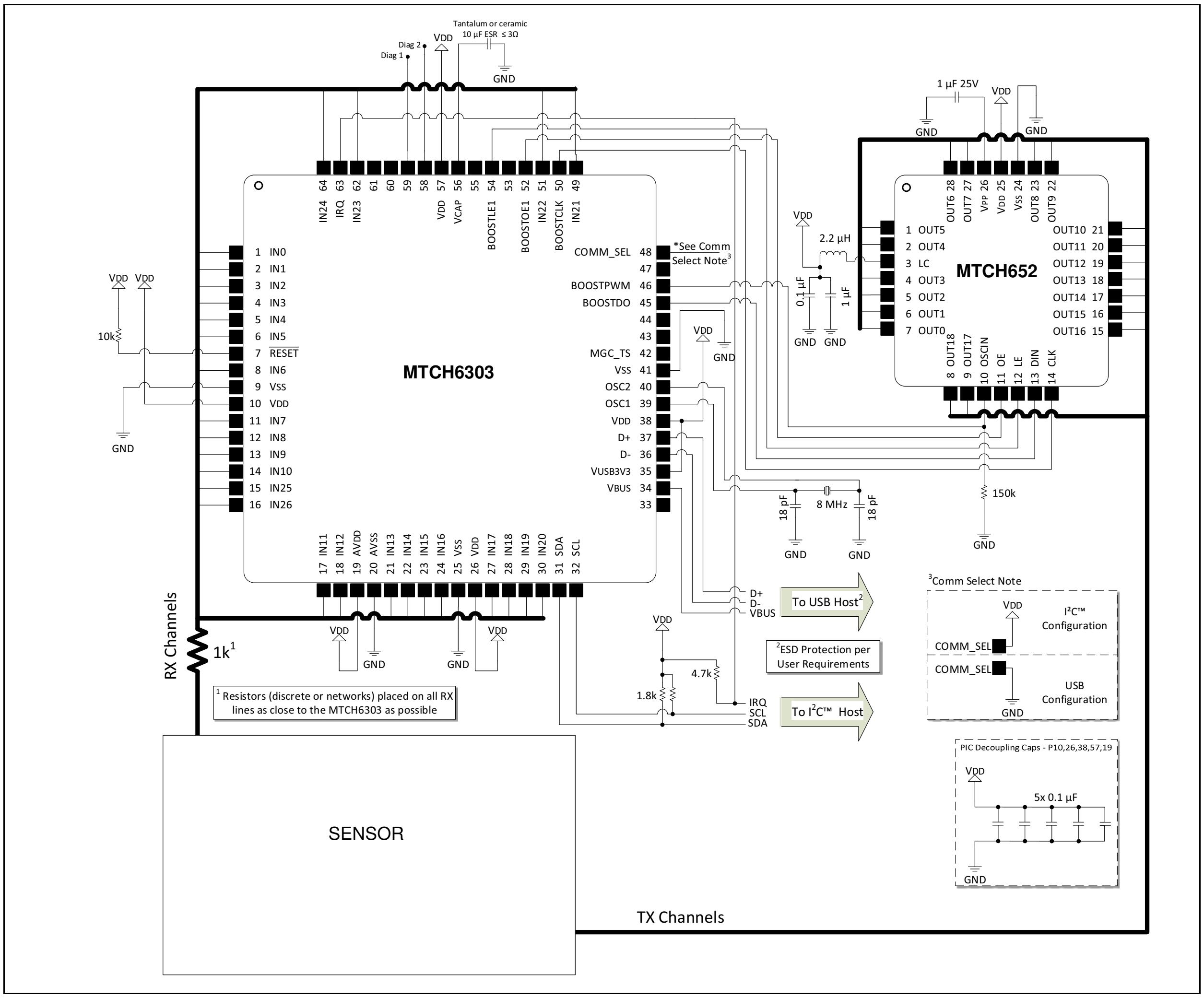
FIGURE 1-1: MTCH6303 BLOCK DIAGRAM



 Preliminary © 2015 Microchip Technology Inc.

2.0 LAYOUT

FIGURE 2-1: TYPICAL APPLICATION CIRCUIT

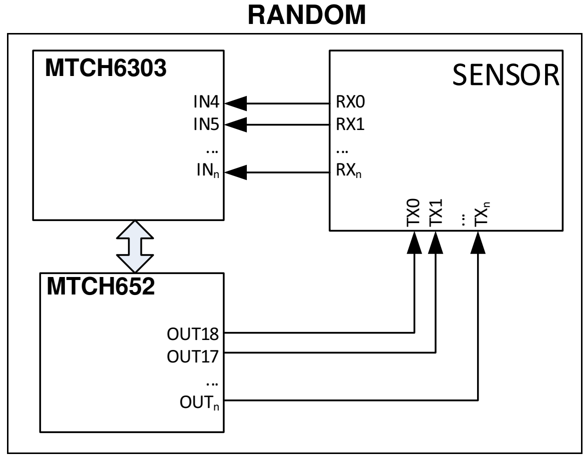


2.1 SENSOR CHANNEL NAMING CONVENTION

Throughout this document, there are references to signals such as IN, RX, OUT and TX. This is deliberately done to avoid confusion between sensor channels and physical pins on the controller. Refer to Figure 2-2 for an example of channel numbers chosen randomly.

- When referring to a sensor, the channels are labeled RX0-RXn and TX0-TXn.
- When referring to the MTCH6303 controller, the INn pins connect to any RXn on the sensor.
- When referring to the MTCH652 boost converter, the OUTn pins connect to any TXn on the sensor.

FIGURE 2-2: EXAMPLE OF CHANNEL NUMBERS CHOSEN AT RANDOM



© 2015 Microchip Technology Inc.

2.2 Decoupling Capacitors

The use of decoupling capacitors on power supply pins, such as V_{DD}, V_{SS}, is required. Consider the following criteria when using decoupling capacitors.

2.2.1 VALUE AND TYPE OF CAPACITOR

A value of 0.1 µF (100 nF), 10-20V is recommended. The capacitor should be a low Equivalent Series Resistance (low ESR) capacitor and have resonance frequency in the range of 20 MHz and higher. It is further recommended that ceramic capacitors be used.

2.2.2 PLACEMENT ON THE PRINTED CIRCUIT BOARD

The decoupling capacitors should be placed as close to the pins as possible. It is recommended that the capacitors be placed on the same side of the board as the device. If space is restricted, the capacitor can be placed on another layer on the PCB; however, ensure that the trace length from the pin to the capacitor is within one-quarter of an inch (6 mm) in length.

2.2.3 HANDLING HIGH-FREQUENCY NOISE

If the board is experiencing high-frequency noise, upward of tens of MHz, add a second ceramic-type capacitor in parallel to the above described decoupling capacitor. The value of the second capacitor can be in the range of 0.01 µF to 0.001 µF. Place this second capacitor next to the primary decoupling capacitor. In high-speed circuit designs, consider implementing a decade pair of capacitances as close to the power and ground pins as possible. For example, 0.1 µF in parallel with 0.001 µF.

2.2.4 MAXIMIZING PERFORMANCE

On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins. This ensures that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB track inductance.

2.3 Bulk Capacitors

The use of a bulk capacitor is recommended to improve power supply stability. Typical values range from 4.7 µF to 47 µF. This capacitor should be located as close to the device as possible.

 © 2015 Microchip Technology Inc.

3.0 COMMUNICATION

3.1 USB/I²C™ Selection

The MTCH6303 can communicate over either USB or I²C™. The decision of which protocol is selected is made on start-up and persists until the controller is reset.

Communications are selectable between USB/I²C through the use of the COMM_SEL pin, which must be permanently tied to either VSS or VDD as follows:

TABLE 3-1: COMM_SEL SETTINGS

Setting	Communications Type
VDD	I²C™
VSS	USB

3.2 Communications Overview

Communications with the MTCH6303 fall into two main categories:

1. **Touch Data:** Data representing the current state of any contact points; this is the main function of the touch controller.
2. **Streamed Messaging:** Packet-based messaging protocol used to:

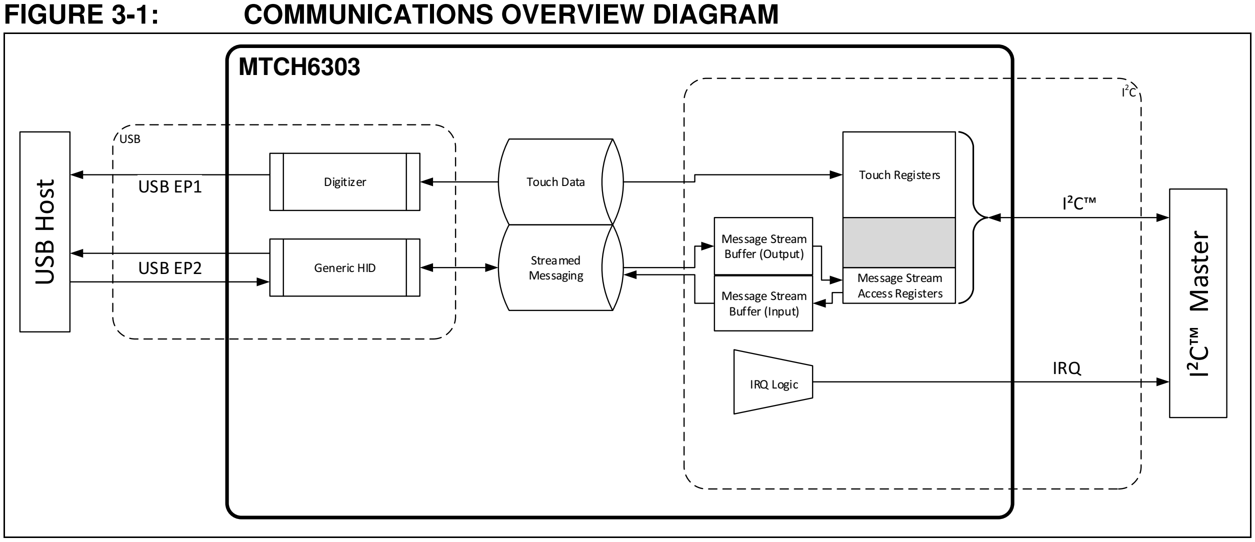
- Send controller commands
- Read/Write parameters
- Receive diagnostic reports (when enabled)
- Read 2D gesture data
- Read 3D gesture data (requires MGC3130)

Both types of data are available over either USB or I²C, as shown in the Table 3-2 below.

TABLE 3-2: COMMUNICATIONS CATEGORIES

Data Type	USB	I²C™
Touch Data	Digitizer endpoint	Register-based memory map
Streamed Messaging	Generic HID endpoint	Stream buffers accessed via I²C™ registers

FIGURE 3-1: COMMUNICATIONS OVERVIEW DIAGRAM



3.3 USB Protocol

3.3.1 HID DIGITIZER (EP 1, TOUCH DATA)

TABLE 3-3: HID DIGITIZER

Byte	7	6	5	4	3	2	1	0	
0	REPORT ID (0X01)								TOUCH 1
1	PADDING						IR	TS	
2	TOUCH ID 0								
3	X1 LSB								
4	X1 MSB								
5	Y1 LSB								
6	Y1 MSB								
7	PADDING						IR	TS	TOUCH 2
8	TOUCH ID 1								
9	X2 LSB								
10	X2 MSB								
11	Y2 LSB								
12	Y2 MSB								
..	..						..	..	TOUCHES 3-9
..	..								
..	..								
..	..								
..	..								
..	..								
47	PADDING						IR	TS	TOUCH 10
48	TOUCH ID 9								
49	X4 LSB								
50	X4 MSB								
51	Y4 LSB								
52	Y4 MSB								
53	#OF VALID TOUCHES								

Legend: IR = In Range
TS = Touch State

3.3.2 HID GENERIC (EP 2, STREAMED MESSAGES)

This generic endpoint is used to send and receive one or more messages within the payload.

FIGURE 3-2: HID GENERIC

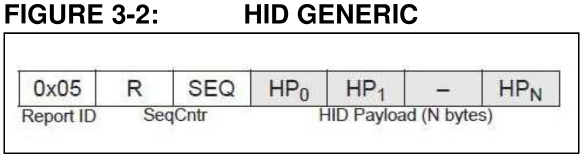


TABLE 3-4: HID GENERIC

Byte Name	Value/Description	
Report ID	0x05	0x05 (Constant)
SeqCntr [7:6]	R	[reserved]
SeqCntr [5:0]	SEQ	Sequence counter, increments on every HID packet. • Values range from 0-63 • IN and OUT packets utilize independent sequence counters

3.4 I²C™ PROTOCOL

3.4.1 OVERVIEW

The MTCH6303 uses a standard register-based read/write I²C™ protocol. This protocol is similar to many other devices such as temperature sensors and serial EEPROMs. Although data can be read at any time (polling), a configurable interrupt pin (INT) is provided for flexible integration options.

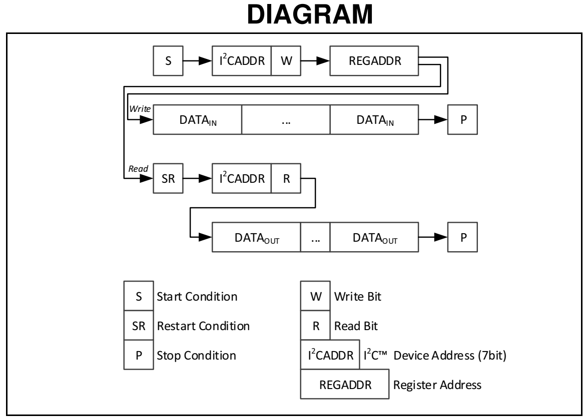
3.4.2 READING/WRITING REGISTERS

To access memory (both to read or write), the I²C transaction must start by addressing the chip with the WRITE bit set, then writing out a single byte of data representing the memory address to be operated on. After that, the host can choose to do either of the following:

1. To write memory, continue writing “n” data bytes.
2. To read memory, restart the I²C transaction (via either a Stop and Start or Restart), then address the chip with the READ bit set. Continue to read “n” data bytes.

During either of these transactions, multiple bytes may be read or written due to the device’s address auto-increment feature.

FIGURE 3-3: I²C™ TRANSACTION DIAGRAM



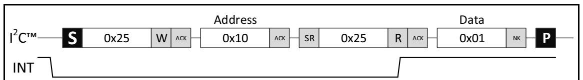
 Preliminary © 2015 Microchip Technology Inc.

3.4.3 DEVICE ADDRESSING

The device's 7-bit base address is 0x25. Each transmission must be prefixed with this address, as well as a bit signifying whether the transmission is a MASTER WRITE (0) or MASTER READ (1). After appending this read/write bit to the base address, this first byte becomes either 0x4A (write) or 0x4B (read).

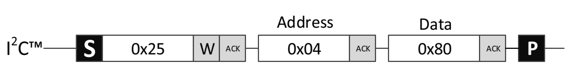
Note:	If this address conflicts with another in the system, it may be possible to customize the device. Contact Microchip support for more information.

FIGURE 3-4: EXAMPLE I²C™ READ TRANSACTION



Note: Reading one byte from address 0x10, byte value is 0x01.

FIGURE 3-5: EXAMPLE I²C™ WRITE TRANSACTION



Note: Writing one byte to address 0x04, value 0x80.

TABLE 3-5: I²C™ MEMORY MAP

ADDR	NAME	7	6	5	4	3	2	1	0	Description
		TOUCH								
0x00	TOUCHSTATUS	R	MGC	GST	STR	NUMTOUCHES				MGC = GestIC® data, GST = Gestures Ready, STR = Stream Ready
0x01	TOUCH 0							IR	TS	IR = In Range, TS = Touch State
0x02		TOUCH ID 1								ID = touch ID, 0-16
0x03		X1 LSB								
0x04		X1 MSB								
0x05		Y1 LSB								
0x06		Y1 MSB								
0x07	TOUCH 1							IR	TS	
0x08		TOUCH ID 1								
0x09		X1 LSB								
0x0A		X1 MSB								
0x0B		Y1 LSB								
0x0C		Y1 MSB								
0x0D	(TOUCH 2)	...								(format follows from above)
0x13	(TOUCH 3)	...								
0x19	(TOUCH 4)	...								
0x1F	(TOUCH 5)	...								
0x25	(TOUCH 6)	...								
0x2B	(TOUCH 7)	...								
0x31	(TOUCH 8)	...								
0x37	(TOUCH 9)	...								
0x42 — 0x7F		*[RESERVED]*								
		STREAM BUFFER								
0xF0 — 0xFA		*[RESERVED]*								
0xFB	RX Bytes Ready	RXRDY								Space available (bytes) for writing into RX buffer
0xFC	RX Buffer	RXBUFF								Pointer to RX Buffer
0xFD	TX Bytes Left	TXRDY								Bytes ready to be read from TX buffer
0xFE	TX Buffer	TXBUFF								Pointer to TX Buffer

© 2015 Microchip Technology Inc.

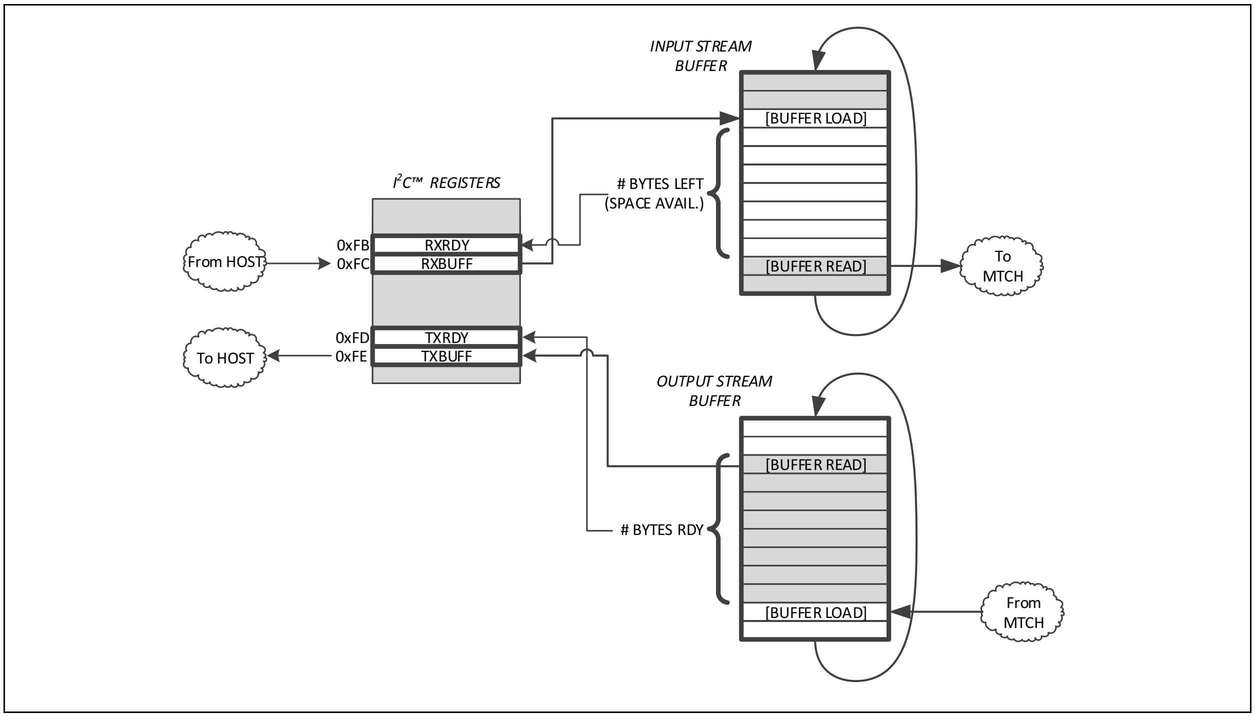
3.4.4 TOUCH REGISTERS

Touch data can be read out of the touch registers at any time, and is ensured to represent the latest state of the sensor. Use of the IRQ pin can improve efficiency by letting the host controller only read data when necessary. (See **Section 6.0, Communication Examples** for more details.)

3.4.5 MESSAGE STREAM ACCESS

For sending and receiving stream messages (described further on in this document), register-based access to the message stream is provided as shown in Figure 3-6.

FIGURE 3-6: MESSAGE STREAM ACCESS



3.4.5.1 Reading Stream Messages Over I^2C

The host discovers that data is ready to be read from the stream by reading a non-zero value from the TXRDY register. This register should be queried after one of the following events:

- IRQ activity
- STR bit of TOUCHSTATUS register is set
- Polled at a random interval (of the host's choosing)

To read the data, an I^2C register read should be started at the address of TXBUFF. The host can choose to read any amount of bytes (up to the value in TXRDY).

3.4.5.2 Writing Stream Messages Over I^2C

The host can write messages directly into the address of RXBUFF. Before writing, the host should check the amount of space available for writing by reading the RXRDY register.

3.4.5.3 Interrupt Pin

To alert the host that new data is ready, an interrupt pin (IRQ) is provided. The IRQ is an 'open-drain' output that is pulled to GND when asserted, and high-impedance (tri-state) when not asserted. A suitable pull-up resistor should be used on this output.

The IRQ can be configured using the parameters in Table 3-6 below (refer to **Section 5.0, Parameters** for accessing).

TABLE 3-6: IRQ CONFIGURATION PARAMETERS

Parameter	Default	Description
irqMode	1	Overall IRQ mode 0 = IRQ deactivated 1 = IRQ level maintained until data read 2 = IRQ pulsed for [irqPulseWidth] msec
irqPolarity	0	IRQ Polarity control 0 = Active-Low, 1 = Active-High
irqPulseWidth	5	Value (msec) to pulse IRQ when irqMode is set to ‘2’
irqTrigger	2	Event control for IRQ activity 0 = Off 1 = Every touch decoding frame 2 = Any touch is present 3 = Only when touch is changed

 © 2015 Microchip Technology Inc.

4.0 MESSAGE PROTOCOL

4.1 Overview

The MTCH6303 messaging protocol is used to send and receive streamed messages. Full or partial (fragment) messages may be exchanged with this protocol.

Messages are transmitted in an overall 'block' size of 64 and must be split up accordingly. Refer to **Section 6.0, Communication Examples** for depictions of messages being fragmented.

FIGURE 4-1: MESSAGE PROTOCOL

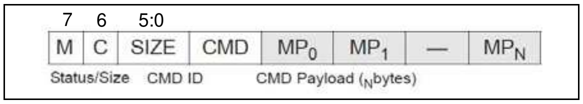


TABLE 4-1: MTCH6303 MESSAGE FORMAT

Name	Description
Status/ Size	**B5-0 SZ** Size of message fragment. If 63 (0x3f), the fragment is incomplete and uses up ALL of the parent transport layer packet
	B6 C 1 = Continued (from last fragment) 0 = Not continued (start of message)
	B7 M 1 = More messages to follow in this block 0 = Last message
CMD ID	Command ID, **only sent on first fragment of message.** For fragments after, this is a normal payload byte.
CMD Payload	Data bytes of message fragment.

4.2 Message Definitions

Messages starting with REP are reports sent from the MTCH6303 to the host. Messages starting with CMD are commands sent from the host to the MTCH6303. Messages that require further clarification are expanded upon in the following section.

© 2015 Microchip Technology Inc.

TABLE 4-2: MESSAGE DEFINITIONS

ID	Name	Payload size	Payload Description (assume uint8 unless noted)	Gated by NVDM[1]	Description
0x04	REP_Echo	<varies>	[data]...[datan]	[NO GATE]	It will echo the exact payload of a received ‘echo’ command
0x17	REP_FlashContents	<varies>	[data]...[datan]	[NO GATE]	Flash contents readback (invoked by CMD_ReadFlash)
0x60	REP_AdcDbg	132	[rx] [tx] [freq] [RSVD] [*uint16* D0] [*uint16* D1]...[*uint16* Dn]	NVDM_ADC	Raw sample output from ADC
0x90	REP_Trace	2	[location][event]	NVDM_DIAG	—
0xA0	REP_Swipe	2	[flags][fingers]	NVDM_GESTURE	Swipe gesture
0xA1	REP_Scroll	8	[fingers][diamHI][*uint16* diameter][*uint16* centerx][*uint16* centery]	NVDM_GESTURE	Scroll gesture
0xA2	REP_Tap	2	[flags][fingers]	NVDM_GESTURE	Tap gesture
0xB0	REP_Noise	<varies>	[subID][data]...[datan]	NVDM_NOISE	Noise messages (see below)
0xC3	REP_MutNormSection	2+2*nodes	[rx][tx][*uint16* node0][*uint16* node1]...[*uint16* noden]	NVDM_MUTCACHE	Sends out a dynamic amount of nodes (from 1 to full RX electrode)
0xCF	REP_ParameterRead	2+len	[*uint16* address][data] (up to ‘len’ bytes)	[NO GATE]	Parameter read response
0xF0	REP_Ack	1	[command ID]	[NO GATE]	Acknowledgment of receipt of command
0xF2	REP_TouchFiltered	5*i	[STATE/ID][uint16 X][uint16 Y]	NVDM_FINGERPOS	Filtered (but not scaled) touch coordinates
0xF3	REP_TouchPredict	9	[ID][*uint16* X0][*uint16* Y0][*uint16* Xpred][*uint16* Ypred]	NVDM_RAWPOS	Prediction value for a touch
0xF4	REP_TouchRaw	5*i	[STATE/ID][*uint16* X][*uint16* Y]	NVDM_RAWPOS	Raw touch report (pre-filter)
0xF5	REP_TouchPos16	5*i	[PEN/ID][*uint16* X][*uint16* Y]	NVDM_FINGERPOS	Final scaled touch report – first byte has touch status as bit 7
0xFA	REP_SelfRaw	2*numRXch	[uint16 self0][uint16 self1]...[uint16 selfn]	NVDM_SELFRAW	Self measurements (raw)
0xFD	REP_SelfNorm	2*numRXch	[*uint16* self0][*uint16* self1]...[*uint16* selfn]	NVDM_SELFNORM	Self measurements (normalized)
0xFE	REP_ForwardGestIC	<varies>	[data]...[datan]	NVDM_GESTIC	Packet from GestIC® (direct)
0xFF	REP_FwVersion	<varies>	[fwVersionInfo]	[NO GATE]	Large array of bytes denoting all firmware information
0x04	CMD_Echo	<varies>	[data]...[datan]	n/a	Firmware will echo back any payload sent
0x17	CMD_ReadFlash	6	[uin32 address][uint16 size]	n/a	Allows host to read Flash contents of device (fw dump)
0x55	CMD_EnterBootLoader	0	(none)	n/a	Commands firmware to enter the bootloader – ACK will be sent before jumping
0xE0	CMD_SetParameter	10	[*uint16* address][*uint8*[4] data][*uint8*[4] mask]	n/a	Writes a parameter
0xE1	CMD_GetParameter	2	[*uint16* address]	n/a	Reads a parameter
0xFB	CMD_ForceBaseline	0	(none)	n/a	Forces a baseline
0xFC	CMD_ResetGestIC	0	(none)	n/a	Resets GestIC immediately
0xFD	CMD_GestIC	<varies>	(gestic command)	n/a	Sends packet directly on to GestIC
0xFF	CMD_QueryVersion	0	(none)	n/a	Requests all firmware version information – bytes 124:127 represent Rev[2].Minor.Major

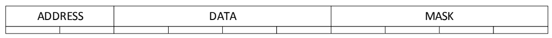
Note: Refer to parameter documentation for explanation of NVDM bitfields.

Preliminary

© 2015 Microchip Technology Inc.

4.2.1 SET PARAMETER COMMAND

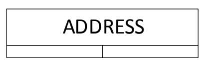
FIGURE 4-2: SET PARAMETER COMMAND



Address:	16bit address location of parameter to set
Data:	32bit (4 bytes) data value to write. For data sizes smaller than 4 bytes, pad with 0x00
Mask:	32 bit (4 bytes) mask value to mask off bits that should not be set. (usually set to 0xFFFFFFFF)

4.2.2 GET PARAMETER COMMAND

FIGURE 4-3: GET PARAMETER COMMAND



Address: 16bit address location of parameter to retrieve

5.0 PARAMETERS

5.1 Operation

Default parameters are loaded on start-up, as shown in the parameter table section. These values can be modified during runtime, but will not be restored on Reset. To permanently modify parameters, the MTCH6303 Utility should be used to export and Flash a new configuration. Refer to the MTCH6303 Utility documentation for more information.

5.2 Parameter Table

Many parameters are tuned by the MTCH6303 Utility itself, so descriptions are not provided. Table 5-1 is provided for reference only.

TABLE 5-1: PARAMETER TABLE

Module	Name	Address	Format	Default	Description
pub	mgc3130	0x0102	uint8_t	0	1 = MTC3130 is present
pub	numberOfRXChannels	0x0100	uint8_t	27	Number of RX channels currently in use
pub	numberOfTXChannels	0x0101	uint8_t	19	Number of TX channels currently in use
pub	diagMask	0x0080	uint16_t	[see NVDM]	[see NVDM]
pub	activeModules	0x0081	uint16_t	[see NVAM]	[see NVAM]
pub	streamingMode	0x0082	uint8_t	0	see Operating Modes
pub	swipeDistance	0x0501	uint16_t	4*256	See Gesture definition
pub	swipeTimeout	0x0500	uint32_t	msec2ticks(1500)(1)	See Gesture definition
pub	swipeBorder	n/a (struct)		n/a	See Gesture definition
pub	swipeBorder.left	0x0502	uint16_t	3*256	See Gesture definition
pub	swipeBorder.right	0x0503	uint16_t	24*256	See Gesture definition
pub	swipeBorder.top	0x0504	uint16_t	3*256	See Gesture definition
pub	swipeBorder.bottom	0x0505	uint16_t	16*256	See Gesture definition
pub	swipeExtBorder	n/a (struct)		n/a	See Gesture definition
pub	swipeExtBorder.left	0x0506	uint16_t	2*256	See Gesture definition
pub	swipeExtBorder.right	0x0507	uint16_t	25*256	See Gesture definition
pub	swipeExtBorder.top	0x0508	uint16_t	2*256	See Gesture definition
pub	swipeExtBorder.bottom	0x0509	uint16_t	17*256	See Gesture definition
pub	tapBorder	n/a (struct)		n/a	See Gesture definition
pub	tapBorder.left	0x0540	uint16_t	1*256	See Gesture definition
pub	tapBorder.right	0x0541	uint16_t	26*256	See Gesture definition
pub	tapBorder.top	0x0542	uint16_t	1*256	See Gesture definition
pub	tapBorder.bottom	0x0543	uint16_t	18*256	See Gesture definition
pub	tapTimeout	0x0544	uint32_t	mSec2Ticks(200)(1)	See Gesture definition
pub	dblTapTimeout	0x0545	uint32_t	mSec2Ticks(500) (1)	See Gesture definition
pub	commSelectMode	0x0584	uint8_t	0	0 = use COMMSEL pin, 1 = force I^2C™, 2 = force USB
pub	irqPolarity	0x0581	uint8_t	0	0 = Active-Low, 1 = Active-High
pub	irqPulseWidth	0x0582	uint8_t	5	Value in msec to pulse (when mode 2)
pub	irqTrigger	0x0583	uint8_t	2	0 = Off, 1 = Set on frame, 2 = Set on touch, 3 = Set on touch changed
pub	irqMode	0x0580	uint8_t	1	0 = Off, 1 = Level-trigger, 2 = Pulse-trigger
pub	idleTime2D	0x0103	uint16_t	100	Scan period while 2D is idle (in msec)
map	txSelfTape	0x02c0	uint16_t [66]	[see below]	
map	rxPinMap	0x0200	uint8_t[27]	[see below]	
map	rxPrechargePinMap	0x0240	uint8_t[27]	[see below]	
map	txPinMap	0x0280	uint8_t[36]	[see below]	
acq	baseUpdateTime	0x0802	uint32_t	mSec2Ticks(10000) (1)	Calibration update rate
acq	selfScanPhase	0x0812	uint16_t[4]	{52,45,40,40}	Self measurement period
acq	selfScanISRPhase	0x0816	uint16_t[4]	{59,49,46,45}	Self measurement phase
acq	mutScanPeriode	0x0803	uint16_t[4]	{122,105,104,100}	Mutual measurement period

 © 2015 Microchip Technology Inc.

TABLE 5-1: PARAMETER TABLE (CONTINUED)

Module	Name	Address	Format	Default	Description
acq	mutScanPhase	0x0807	uint16_t[4]	{68,60,59,55}	Mutual measurement phase
acq	mutFreqHopping	0x080B	uint8_t	0	Frequency hopping control (0 = enabled, 1-4 = lock to F0-F3)
acq	mutFreqHoppingLevel	0x080C	int8_t[4]	{0,0,0,0}	Linear gain to apply to results from each frequency
acq	diagRxChannel	0x0800	uint8_t	0xff	
acq	diagTxChannel	0x0801	uint8_t	0xff	
acq	syncRxChannel	0x081A	uint8_t	0xff	
acq	syncTxChannel	0x081B	uint8_t	0xff	
acq	fullScanRxStart	0x081C	uint8_t	0	
acq	fullScanRxStop	0x081D	uint8_t	27	
acq	fullScanTxStart	0x081E	uint8_t	0	
acq	fullScanTxStop	0x081F	uint8_t	19	
dec	penDownTimer	0x0403	uint16_t	781	
dec	penUpTimer	0x0404	uint16_t	781	
dec	selfTouchThres	0x0400	uint8_t	60	
dec	mutTouchThres	0x0401	uint8_t	60	
dec	minCuspDelta	0x040b	uint8_t	25	
dec	weightThreshold	0x0402	uint8_t	20	
dec	minTouchDistance	0x040c	uint8_t	5*8	
dec	fatThreshold	0x040d	uint8_t	95	
dec	nbSampleSelf	0x0407	uint8_t	64	
dec	touchActiveHysteresis2D	0x0409	uint16_t	1000	
dec	touchActiveHysteresis2D3D	0x0401	uint16_t	50	
rep	flipState	0x0041	uint8_t	0b010	
rep	rxScale	n/a (struct)		n/a	
rep	rxScale.shift	0x0042	uint8_t	7	
rep	rxScale.divide	0x0043	uint8_t	27	
rep	rxScale.offset	0x0044	uint16_t	0	
rep	txScale	n/a (struct)		n/a	
rep	txScale.shift	0x0045	uint8_t	7	
rep	txScale.divide	0x0046	uint8_t	19	
rep	txScale.offset	0x0047	uint16_t	0	
mtc	mtch65x_active_config	none	uint32_t	0x27	
mtc	mtch65x_periode_fast_rise	0x0900	uint16_t	10	
mtc	mtch65x_periode_fast_rise_oc	0x0901	uint16_t	7	
mtc	mtch65x_fast_rise_delay	0x0902	uint16_t	300	
mtc	mtch65x_periode_self_measurement	0x090D	uint16_t[4]	{20,20,20,20}	
mtc	mtch65x_periode_self_measurement_oc	0x0911	uint16_t[4]	{10,10,10,10}	
mtc	mtch65x_periode_mutu_measurement	0x0905	uint16_t[4]	{66,60,59,58}	
mtc	mtch65x_periode_mutu_measurement_oc	0x0909	uint16_t[4]	{16,15,14,14}	

Note 1: mSec2Ticks(ms) = (((ms) * 625 + 2) / 4)

© 2015 Microchip Technology Inc.

EXAMPLE 5-1: COMPLICATED INITIALIZATIONS

```
rxPinMap = {(15), (14), (13), (12), (11), (10), (9), (8), (7), (6), (0), (1), (2), (3), (4), (5),
(19), (18), (17), (16), (27), (23), (22), (21), (20), (26), (24)}
rxPrechargePinMap = {(24), (24), (24), (24), (24), (24), (24), (24), (24), (24), (24), (24), (24),
(15), (15), (15), (15), (15), (15), (15), (15), (15), (15), (15), (15), (15), (15)}
txPinMap = {(0+ 17), (0+ 18), (0+ 0), (0+ 1), (0+ 2), (0+ 3), (0+ 4), (0+ 5), (0+ 6), (0+ 7), (0+
8), (0+ 9), (0+ 10), (0+ 11), (0+ 12), (0+ 13), (0+ 14), (0+ 15), (0+ 16)}
txSelfTape = {0x0000,
0x0F,0x0010,0x0010,0x7110,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0x0010,0x0310,0x8110,0x00,0x00,0x00
,0x00,0x00,0x00,0x0F,0x0010,0x1C10,0x0110,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0x0010,0xE010,0x011
0,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0x0F10,0x0010,0x0110,0x00,0x00,0x00,0x00,0x00,0x00,0x0C,0x0
000,0x0000,0,0,0,0,0,0,0,0,0,0,0,0}
```

5.3 Special Parameters

5.3.1 ACTIVE MODULES REGISTER (NVAM)

REGISTER 5-1: ACTIVE MODULES REGISTER (NVAM)

U-x	U-x	U-x	U-x	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	—	DECODE	DIGITIZER	AUTOBASE	BESTFREQ
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-x	U-x	R/W-0	U-x	R/W-1
AW_EVENT	SW_EVENT	FL_EVENT	—	—	FULLSCAN	—	GESTURE
bit 7							bit 0

Legend:		
R = Readable bit	x = Bit is unknown	-n = Value after initialization (default)
W = Writable bit	U = Unimplemented bit	q = Conditional
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 15-12	**Unused**	
bit 11	**DECODE:**	Turns touch decoding logic on or off
bit 10	**DIGITIZER:**	Turns digitizer/I^{2}C™ register output on or off
bit 9	**AUTOBASE:**	Turns on or off automatic baseline functionality
bit 8	**BESTFREQ:**	Turns on or off bestfrequency selection algorithms
bit 7	**AW_EVENT:**	Events related to GestIC airwheel
bit 6	**SW_EVENT:**	Events related to GestIC swipes
bit 5	**FL_EVENT:**	Events related to GestIC flicks
bit 4-3	**Unused**	
bit 2	**FULLSCAN:**	Turns on full mutual scanning
bit 1	**Unused**	
bit 0	**GESTURE:**	Turns on 2d gesture recognition

 © 2015 Microchip Technology Inc.

5.3.2 DIAGNOSTIC MODULES REGISTER (NVDM)

REGISTER 5-2: ACTIVE DIAGNOSTICS MODULES REGISTER (NVDM)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
GESTIC	DIAG	CUSTOM	GESTURE	FINGERPOS	RAWPOS	NOISE	TRACE
bit 15							bit 8

U-x	U-x	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	ADC_COR	ADC	MUTRAW	SELFRAW	MUTCACHE	SELFNORM
bit 7							bit 0

Legend:		
R = Readable bit	x = Bit is unknown	-n = Value after initialization (default)
W = Writable bit	U = Unimplemented bit	q = Conditional
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 15	**GESTIC:**	Forward GestIC® packets to host, also packets from host to GestIC
bit 14	**DIAG:**	Diagnostic Messages
bit 13	**CUSTOM:**	Custom Messages
bit 12	**GESTURE:**	Gesture Messages
bit 11	**FINGERPOS:**	Filtered Touch Data
bit 10	**RAWPOS:**	Unfiltered Touch Data
bit 9	**NOISE:**	Noise Messages
bit 8	**TRACE:**	Trace Messages
bit 7-6	**Unused**	
bit 5	**ADC_COR:**	Use ADC Offsets
bit 4	**ADC:**	ADC Messages
bit 3	**MUTRAW:**	Mutual Raw Data
bit 2	**SELFRAW:**	Self Raw Data
bit 1	**MUTCACHE:**	Mutual Normalized Data
bit 0	**SELFNORM:**	Self Normalized Data

6.0 COMMUNICATION EXAMPLES

6.1 Reading Touch Data

The following examples show a frame of data communicating three Touch ID contact points:

TABLE 6-1: READING TOUCH DATA

Touch ID	ID5
5	Contact at (2345,4657)
8	Contact at (9823,0023)
13	Touch Removed (last contact 7264,1893)

TABLE 6-2: READING TOUCH DATA HID REPORT

0	0x01 REPID	0x03 STATUS0	0x05 ID0	0x29 XLSB0	0x09 XMSB0	0x31 YLSB0	0x12 YLSB0	0x03 STATUS1	7
8	0x08 ID1	0x5F XLSB1	0x26 XMSB1	0x17 YLSB1	0x00 YMSB1	0x02 STATUS2	0x0D ID2	0x60 XLSB2	15
16	0x1C XMSB2	0x65 YLSB2	0x07 YMSB2	0x00 STATUS3	— ID3	— XLSB3	— XMSB3	— YLSB3	23
24	— YMSB3	— STATUS4	— ID4	— XLSB4	— XMSB4	— YLSB4	— YMSB4	— STATUS5	31
32	— ID5	— XLSB5	— XMSB5	— YLSB5	— YMSB5	— STATUS6	— ID6	— XLSB6	39
40	— XMSB6	— YLSB6	— YMSB6	— STATUS7	— ID7	— XLSB7	— XMSB7	— YLSB7	47
48	— YMSB7	— STATUS8	— ID8	— XLSB8	— XMSB8	— YLSB8	— YMSB8	— STATUS9	55
56	— ID9	— XLSB9	— XMSB9	— YLSB9	— YMSB9	0x03 #VALID	— —	— —	

6.1.1 READING TOUCH DATA (USB)

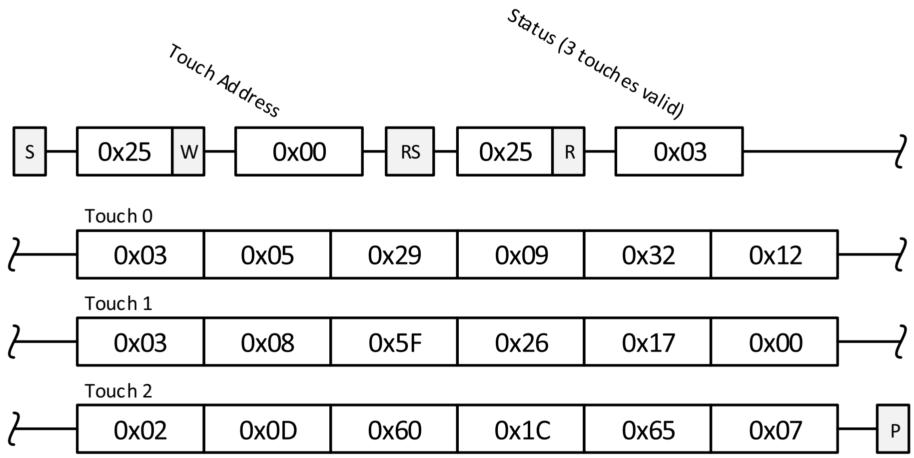
Touch data is populated in the HID report (refer to **Section 3.3.2, HID Generic (EP 2, Streamed Messages)**).

Preliminary
© 2015 Microchip Technology Inc.

6.1.2 READING TOUCH DATA (I^2C)

Reading touch data over I^2C must be performed in one single transaction to ensure the data is all from the same frame.

FIGURE 6-1: READING TOUCH DATA (I^2C™)



Note: The host could continue to read all 10 touches, but there is no need since the first byte indicates only three touches are valid.

6.2 Message Send/Receive

In these examples, a message setting the current number of RX channels is sent, and the response received is shown. (including acknowledgment).

6.2.1 MESSAGE TO SEND

Message ID

0xE0 (CMD_SetParameter)

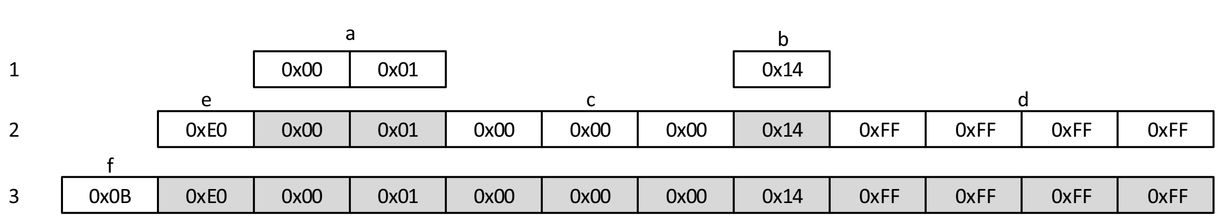
Payload (message specific)

Address: 0x0100

Data: 0x14

First, the message must be created according to the message format in Figure 6-2.

FIGURE 6-2: MESSAGE TO SEND



6.2.1.1 Steps

1. Parameter address (a) and value to write (b)
2. Message ID is added (e).

Fill bytes are added to value to make it 32 bits (c).

Data mask is added (d) – note that since the parameter is only one byte, only the last byte of the mask actually affects the behavior.

3. Status byte is added:
   - size is 11 (0x0B)
   - “more messages” is set to 0
   - “is continued” ID set to 0 (this is the start of message)

6.2.2 EXPECTED RESPONSE

Every message sent to the controller also contains an acknowledgment message back (ACK), which follows this format:

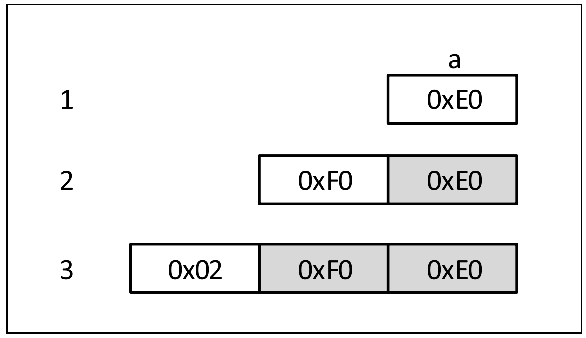
Message ID

0xF0 (REP_Ack)

Payload

0xE0 (command received was CMD_SetParameter)

FIGURE 6-3: EXPECTED RESPONSE

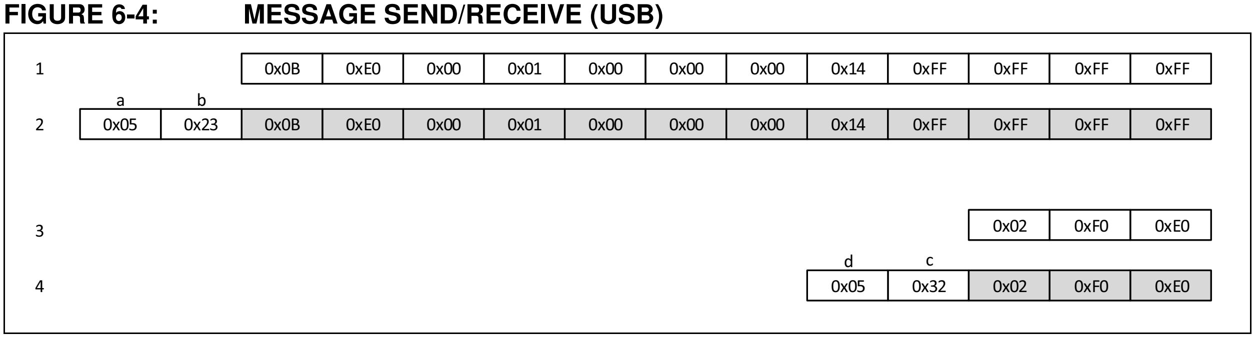


6.2.2.1 Steps

1. Expected payload for an ACK message is an echo of the command being ACK’d – in this case, 0xE0
2. Message ID is added
3. Status byte is added:
   - Size = 2
   - More messages = 0
   - Continued = 0

6.2.3 MESSAGE SEND/RECEIVE (USB)

FIGURE 6-4: MESSAGE SEND/RECEIVE (USB)



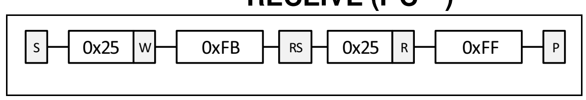
6.2.3.1 Steps

1. Message to send (from previous section)
2. Adding sequence ID (b), which was chosen at random for this example. Adding reportID (always 0x05)
3. Response expected (from previous section)
4. Adding sequence ID (c), which was chosen at random for this example. Adding reportID (always 0x05).

6.2.4 MESSAGE SEND/RECEIVE (I^2C)

First, the host must query the RXRDY buffer to ensure there is enough space to write the command. In this case, the controller is reporting that 255 bytes are available for writing:

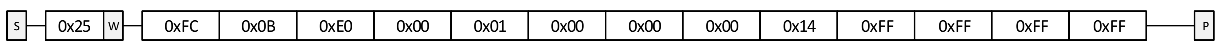
FIGURE 6-5: MESSAGE SEND/RECEIVE (I^2C™)



 © 2015 Microchip Technology Inc.

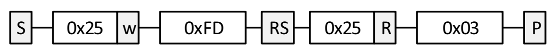
Next, the host writes the command into the controller's RXBUFF register (Figure 6-6).

FIGURE 6-6: HOST WRITE TO RXBUFF REGISTER



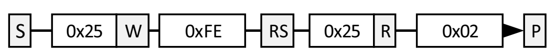
The host may now query the TXRDY buffer to see if the response is ready, either after a set amount of time or by observing IRQ (Figure 6-7).

FIGURE 6-7: HOST READ FROM TXRDY REGISTER



Since there are three bytes ready to be read, the host should now read those three bytes out of the TXBUFF register (Figure 6-8).

FIGURE 6-8: HOST READ FROM TXBUFF REGISTER



Note: The two previous steps could have been completed in a single read by a host capable of performing decisions during a read (based on the value of TXRDY).

Reading address 0xFD auto-increments the address pointer to 0xFE, the stream buffer. Further bytes read will all be from within the stream buffer, maintaining the 0xFE address. The first byte read, 0x03, would indicate that three more bytes are within the stream buffer and may be read immediately.

© 2015 Microchip Technology Inc.