GT911 Programming Guide

(Applicable to firmware of version 1040 or later)

====== Disclaimer ======

The information concerning the device and the like in this publication is intended for you only and is subject to change without prior notice. It is your responsibility to ensure its application complies with technical specifications. Shenzhen Huiding Technology Co., Ltd. (hereafter referred to as “GOODIX”) makes no representation or guarantee for this information, either expressed or implied, written or verbal, statutory or otherwise including but not limited to representation or guarantee for its application, quality, performance, merchantability or fitness for a particular purpose. GOODIX shall assume no responsibility for this information and relevant consequences arising out of the use of such information. Without written consent of GOODIX, it is prohibited to use GOODIX products as critical components in any life support system. This document conveys no licenses, implicitly or otherwise, to any intellectual property rights belonging to GOODIX.

Contents

1. Description on Interface

GT911 interfaces with the host via 6 pins: VDD, GND, SCL, SDA, INT and RESET.

The INT pin of the host can be rising/falling-edge Triggered. In addition, when INT is in input state, the host should set INT to be floating, with no internal pull-up or pull-down; the host controls the RESET pin of the GT911 by outputting high or low level. To ensure reliable reset, it is recommended that RESET pin outputs low for longer than 100μs.

GT911 communicates with the host via standard I^2C interface, with a maximum transmission rate of 400K bps. When the host communicates at rates exceeding 200K bps, it is required to pay special attention to the resistance of the external pull-up resistor of I^2C interface to ensure the edges of SCL and SDA are steep enough. GT911 invariably serves as slave device in communication and its I^2C device address consists of 7 device address bits and 1 Read/Write control bit. The high 7 bits are device address while bit 0 is Read/Write control bit. GT911 supports two slave device addresses which are shown below:

7 bit address	8 bit write address	8 bit read address
0x5D	0xBA	0xBB
0x14	0x28	0x29

Upon each power-on or reset, it is required to select I^2C address using INT pin. For specific configuration method, please refer to section 4.1 and section 4.2.

2. Communication Timings

2.1 Timing for Write Operation

S	Address_W	A C K	Register_H	A C K	Register_L	A C K	Data_1	A C K	••••••	Data_n	A C K	E

S: Start condition.

Address_W: slave device address with Write control bit.

ACK: Acknowledgement signal.

Register_H, Register_L: 16-bit register address from which Write Operation starts

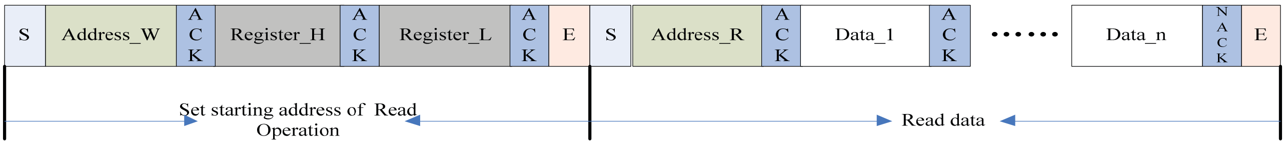
Data_1 to Data_n: Data bytes 1-n.

E: Stop condition.

After setting the starting register address for Write operation, it is allowed to write one or more bytes at a time. GT911 will automatically increase the address of register and store the data bytes in sequence.

2.2 Timing for Read Operation

First, set register address from which Read Operation starts based on the aforesaid Wirte Operation timing sequence. Then, resend start condition to perform Read addressing and read register data.



Address_R: Slave device address with Read control bit.

NACK: Host issues NACK after reading the last byte.

After setting read operation register addresses, the host can read one or more than one byte at a time. GT911 will automatically increase register address and send subsequent data in sequence.

The stop condition (the first E signal as shown in the above diagram) after setting read operation register address is optional. However, the start condition to restart I^{2}C communication has to be resent.

3. Register Map

3.1 Real-time command (Write only)

Addr	Name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x8040	Command	0: Read coordinates status; 1: Read diff data or raw data; 2: Read diff data or raw data; 3: Reference capacitance update (Internal test); 4: Reference capacitance calibration (Internal test); 5: Screen off; 6: Enter Charge mode; 7: Exit Charge mode 8 : Gesture mode. 0x20: Enter HotKnot Slave Approach mode 0x21：Enter HotKnot Master Approach mode 0x22：Enter Receive mode 0x28：Exit Slave Approach mode 0x29：Exit Master Approach mode 0x2A：Exit Receive mode 0xAA: ESD protection mechanism enabled; driver writes 0xAA to 0x8040 and reads and checks the value of 0x8040 regularly; other values are invalid.							
0x8041	ESD_Check	ESD protection mechanism enabled; reset to 0 upon initialization; after that, driver writes 0xAA to 0x8040 and reads and checks the value of 0x8040 regularly.							
0x8046	Command_Check	For commands greater than 0x07, it is required to write the command to 0x8046 before writing to 0x8040, to improve anti-ESD capability.							

3.2 Configuration information (R/W)

Register	Config Data	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x8047	Config_ Version	The version number of configuration documents (configuration parameters will be updated only when the version number of the new release is later than that of the previous one, or equal to that of the previous one but there are changes in contents; documents are numbered sequentially from 'A' to 'Z'; Send 0x00 and the version number is reset to 'A')							
0x8048	X Output Max (Low Byte)	Resolution of X axis							
0x8049	X Output Max (High Byte)								
0x804A	Y Output Max (Low Byte)	Resolution of Y axis							
0x804B	Y Output Max (High Byte)								
0x804C	Touch Number	Reserved				Touch points supported: 1 to 5			
0x804D	Module_ Switch1	Driver_ Resersal (Y2Y)	Sensor_ Resersal （X2X）	Stretch_rank		X2Y (X,Y axis switch-over)	Sito (Software noise reduction)	INT triggering mechanism 00: rising edge 01: falling edge 02: Low level 03: High level	
0x804E	Module_ switch2	Reserved		FirstFilter_Dis	Reserved		Approch_ En	HotKnot _En	Touch_ Key
0x804F	Shake_Count	De-jitter frequency when touch is being released				De-jitter frequency when touch is pressing down			
0x8050	Filter	First_Filter		Normal_Filter (Filter threshold for original coordinates, coefficient is 4)					
0x8051	Large_Touch	Number of large-area touch points							
0x8052	Noise_ Reduction	Reserved				Noise reduction value (0-15 valid, coefficient is 1)			
0x8053	Screen_ Touch_Level	Threshold for touch to be detected							
0x8054	Screen_ Leave_Level	Threshold for touch to be released							
0x8055	Low_Power_ Control	Reserved				Interval to enter lower power consumption mode (0s to 15s)			
0x8056	Refresh_Rate	Pulse width setting for gesture wakeup				Coordinates report rate (period: 5+N ms)			
0x8057	x_threshold	X coordinate output threshold: 0-255 (Based on the last reported coordinates; If configured to 0, GT911 will keep outputting coordinates continuously)							
0x8058	y_threshold	Y coordinate output threshold: 0-255 (Based on the last reported coordinates. If configured to 0, GT911 will keep outputting coordinates continuously)							
0x8059	X_Speed_Limit	Reserved							
0x805A	Y_Speed_Limit								

0x805B	Space	Space of border top (coefficient: 32)		Space of border bottom (coefficient: 32)	
0x805C		Space of border left (coefficient: 32)		Space of border right (coefficient: 32)	
0x805D	Mini_Filter	Reserved		Mini filter configuration during line drawing process, configured as 0 indicates 4	
0x805E	Stretch_R0	coefficient of Stretch space 1			
0x805F	Stretch_R1	coefficient of Stretch space 2			
0x8060	Stretch_R2	coefficient of Stretch space 3			
0x8061	Stretch_RM	The base of multiple stretch spaces			
0x8062	Drv_GroupA_Num	All_Driving	Reserved	Driver_Group_A_number	
0x8063	Drv_GroupB_Num	Reserved	Dual_Freq	Driver_Group_B_number	
0x8064	Sensor_Num	Sensor_Group_B_Number		Sensor_Group_A_Number	
0x8065	FreqA_factor	Clock Multiplier Factor of drive frequency of Driver Group A GroupA_Frequence = Clock Multiplier Factor * Fundamental Frequency			
0x8066	FreqB_factor	Clock Multiplier Factor of drive frequency of Driver Group B GroupB_Frequence = Clock Multiplier Factor * Fundamental Frequency			
0x8067	Pannel_BitFreqL	Fundamental Frequency of Driver Groups A and B (1526HZ< Fundamental Frequency <14600Hz)			
0x8068	Pannel_BitFreqH				
0x8069	Pannel_Sensor_TimeL	Output Interval between two adjacent drive signals (unit: us); Reserved (used in beta version; invalid in a Release)			
0x806A	Pannel_Sensor_TimeH				
0x806B	Pannel_Tx_Gain	Reserved		Pannel_Drv_output_R 4 gain values, configurable	Pannel_DAC_Gain 0: Gain max. 7: Gain min.
0x806C	Pannel_Rx_Gain	Pannel_PGA_C	Pannel_PGA_R	Pannel_Rx_Vcmi (4 gain values, configurable)	Pannel_PGA_Gain (8 gain values, configurable)
0x806D	Pannel_Dump_Shift	Amplification factor of raw data in Gesture Mode (2^N)		Amplification factor of raw data on the touch panel (2^N)	
0x806E	Drv_Frame_Control	Reserved	SubFrame_DrvNum (maximum setting is 17)		Repeat_Num (Accumulated sampling count)
0x806F	Charging_Level_Up	After the host issues Charge command, IC enters Charge mode and raises the Touch_Level and Leave_Level. The level applicable to Charge mode= original level+configuration level. When configuration level is 0, the charging level equals to the original level.			

0x8070	Module_Switch3	Reserved	Gesture_Hop_Dis	Strong_Smooth	Reserved				Shape_En
0x8071	GESTURE_DIS	Valid distance for slide-up/down wakeup				Valid distance for slide-left/right wakeup			
0x8072	Gesture_Long_Press_Time	The gesture recognizing processing aborting time period when long touching							
0x8073	X/Y_Slope_Adjust	The adjustment parameter of X direction slope when using "four point trigonometric approximation algorithm" to calculate the coordinates (0: algorithm disabled)				The adjustment parameter of Y direction slope when using "four point trigonometric approximation algorithm" to calculate the coordinates (0: algorithm disabled)			
0x8074	Gesture_Control	Invalid time for double-tap wakeup (unit:100 ms, defaults to 1.5s when configured as 0)				GestureDrv_PGA_Gain (8 gain values, configurable)			
0x8075	Gesture_Switch1	Swipe left	Swipe up	Swipe right	w	o	m	e	c
0x8076	Gesture_Switch2	Swipe is valid only at the bottom of the TP	z	s	^	>	V	Double-tap	Swipe down
0x8077	Gesture_Refresh_Rate	Report rate in Gesture mode (period is 5+ms)							
0x8078	Gesture_Touch_Level	Touch threshold in Gesture mode							
0x8079	NewGreenWakeUpLevel	Threshold for NewGreen wakeup of Gesture wakeup function							
0x807A	Freq_Hopping_Start	Start frequency for frequency hopping(when Range_Ext=0，the unit is 2KHz，for example, 50 indicates100KHz; When Range_Ext=1，the unit is BitFreq)							
0x807B	Freq_Hopping_End	End frequency for frequency hopping(when Range_Ext=0，the unit is 2KHz，for example, 150 indicates 300KHz; when Range_Ext=1，the unit is BitFreq)							
0x807C	Noise_Detect_Times	Detect_Stay_Times (Number of tests taken on each frequency point in each noise test; 2 is recommended)			Detect_Confirm_Times (Confirmed noise level after repeated noise tests, 1-63 valid; 20 is recommended)				
0x807D	Hopping_Flag	Hopping_En	Range_Ext	Dis_Force_Ref	Delay_Hopping		Detect_Time_Out (timeout for noise detection, unit: second), Reserved		
0x807E	Hopping_Threshold	Fast_Hopping_Limit: fast hopping is enabled only when the interference value of current frequency is greater than Fast_Hopping_Limit*4. The minimum setting of this limit is 5.				Hopping_Hit_Threshold (Conditions for selecting optimal frequency: Current operating frequency interference- Minimum interference>Set			

			valuex4, then optimal frequency is selected and frequency hopping is enabled)
0x807F	Noise_ Threshold	Threshold to distinguish if there is interference (if the interference on all frequency point is less than this threshold, it is regarded as no interference), Reserved	
0x8080	Noise_Min_Threshold	When ESD causes the minimum interference point to be greater than the threshold value, it will initiate fast reduction treatment. Configured to 0 means this function is disabled and configured to high value (such as 200 or higher) has the equivalent effect. To enable this function, it is recommended to set the value 5 to 20 higher than the minimum frequency point (LCD interference and common-mode interference, whichever is greater) in normal interference.	
0x8081	NC	Reserved	
0x8082	Hopping_Sensor_Group	Sections for Hopping Frequency Noise Detection (4 sections recommended)	
0x8083	Hopping_seg1_Normalize	Seg1 Normalize coefficient (sampling value *N / 128= Raw data)	
0x8084	Hopping_seg1_Factor	Seg1 Central point Factor	
0x8085	Main_Clock_Ajdust	Fine adjustment of IC main clock Frequency, within the range of -7 to +8	
0x8086	Hopping_seg2_Normalize	Seg2 Normalize coefficient (sampling value *N / 128= Raw data)	
0x8087	Hopping_seg2_Factor	Seg2 Central point Factor	
0x8088	NC	Reserved	
0x8089	Hopping_seg3_Normalize	Seg3 Normalize coefficient (sampling value *N / 128= Raw data)	
0x808A	Hopping_seg3_Factor	Seg3 Central point Factor	
0x808B	NC	Reserved	
0x808C	Hopping_seg4_Normalize	Seg4 Normalize coefficient (sampling value *N / 128= Raw data)	
0x808D	Hopping_seg4_Factor	Seg4 Central point Factor	
0x808E	NC	Reserved	
0x808F	Hopping_seg5_Normalize	Seg5 Normalize coefficient (sampling value *N / 128= Raw data)	
0x8090	Hopping_seg5_Factor	Seg5 Central point Factor	
0x8091	NC	Reserved	
0x8092	Hopping_seg6_Normalize	Seg6 Normalize coefficient (sampling value *N / 128= Raw data)	
0x8093	Key 1	Key 1 address: 0-255 valid	

		(0 indicates no key is available. When the addresses of all four keys are the multiples of 8, it means independent key design manner.)							
0x8094	Key 2	Key 2 address: 0-255 valid (0 indicates no key is available. When the address of all four keys is the multiples of 8, it means independent key design manner)							
0x8095	Key 3	Key 3 address: 0-255 valid (0 indicates no key is available. When the address of all four keys is the multiples of 8, it means independent key design manner)							
0x8096	Key 4	Key 4 address: 0-255 valid (0 indicates no key is available. When the address of all four keys is the multiples of 8, it means independent key design manner)							
0x8097	Key_Area	Time limit for long-press update (1s to 15s). Long-press update is disabled when configured to 0.				Key active area configuration (single side): 0-15 valid			
0x8098	Key_Touch_Level	Touch key touch threshold							
0x8099	Key_Leave_Level	Touch key release threshold							
0x809A	Key_Sens	KeySens_1(sensitivity coefficient of Key 1)				KeySens_2 (sensitivity coefficient of Key 2)			
0x809B	Key_Sens	KeySens_3(sensitivity coefficient of Key 3)				KeySens_4 (sensitivity coefficient of Key 4)			
0x809C	Key_Restrain	The key restrain interval after finger leaves screen (unit: 100ms), 0 means the key suppression interval is 600ms.				Independent adjacent key restrain parameter			
0x809D	Key_Restrain_ Time	Reserved				Adjacent key restrain time internal after the finger slides to leave at the bottom of the TP (unit: 100 ms). Timing starts from the moment that finger leaves the TP. If there is touch key event within this time interval, the touch key will be restrained until the touch key is released and touched down again. (configured as 0, this function is disabled)			
0x809E	GESTURE_ LARGE_TOUCH	Large-area touch processing in Gesture mode (the size of the touch rectangle). Configured as 0, this function is disabled.							
0x809F	NC	Reserved							
0x80A0	NC	Reserved							
0x80A1	Hotknot_Noise_ Map	Reserved		200K	250K	300K	350K	400K	450K
0x80A2	Link_Threshold	Link_NoiseThreshold							
0x80A3	Pxy_Threshold	Pxy_NoiseThreshold							
0x80A4	GHot_Dump_ Shift	Reserved			Rx_Self	Amplification factor of raw Data (2^N)			
0x80A5	GHot_Rx_Gain	PGA_C	PGA_R		Reserved		PGA_Gain (8 levels to be configured）		
0x80A6	Freq_Gain0	400K signal gain calibration, calibration volume is N/16. Invalid when N=0.				450K signal gain calibration, calibration volume is N/16. Invalid when N=0.			

0x80A7	Freq_Gain1	300K signal gain calibration, calibration volume is N/16. Invalid when N=0.	350K signal gain calibration, calibration volume is N/16. Invalid when N=0.
0x80A8	Freq_Gain2	200K signal gain calibration, calibration volume is N/16. Invalid when N=0.	250K signal gain calibration, calibration volume is N/16. Invalid when N=0.
0x80A9	Freq_Gain3	Reserved	150K signal gain calibration, calibration volume is N/16. Invalid when N=0.
0x80AA	NC	Reserved	
0x80AB	NC	Reserved	
0x80AC	NC	Reserved	
0x80AD	NC	Reserved	
0x80AE	NC	Reserved	
0x80AF	NC	Reserved	
0x80B0	NC	Reserved	
0x80B1	NC	Reserved	
0x80B2	NC	Reserved	
0x80B3	Combine_Dis	Distance for adjacent rectangles to be combined in Gesture mode	Distance for adjacent rectangles to be combined
0x80B4	Split_Set	Distance for a large-area rectangle to be split	Distance for a normal-size rectangle to be split
0x80B5	NC	Reserved	
0x80B6	NC	Reserved	
0x80B7 to 0x80C4	Sensor_CH0 to Sensor_CH13	Channel number on chip corresponding to ITO Sensor	
0x80C5 to 0x80D4	NC	Reserved	
0x80D5 to 0x80EE	Driver_CH0 to Driver_CH25	Channel number on chip corresponding to ITO Driver	
0x80EF to 0x80FE	NC	Reserved	
0x80FF	Config_Chksum	Configuration verification (checksum value of the bytes from 0x8047 to 0x80FE)	
0x8100	Config_Fresh	Configuration updated flag (the flag is written by the host)	

Supplementary description on some registers:

[0x804D] Module_Switch1

Bit7: Driver_Resersal(Y2Y), configured as 1 indicates Y axis reversal.

Bit6: Sensor_Resersal(X2X), configured as 1 indicates X axis reversal.

Bit5-bit4：Stretch_rank, stretching method

00,01,02：Weak stretch 0.4P

03：User-defined stretch

[0x804E] Module_Switch2

Bit5: FirstFilter_Dis, enlarge the first de-bouncing value. 0: enabled; 1: disabled.

Bit2：Approch_En，hotknot proximity sensing on/off.

Bit1：Approch_En，hotknot function on/off.

[0x8056] Refresh_Rate

Bit7~Bit4: pulse width setting for Gesture wakeup, unit: 250us. Configured as "f" indicates INT should be driven high if the host does not succeed in reading the data.

[0x805B-0x805C] Space

Configuration of the reported area, which is used to adjust the reported borders when the ITO exceeds the viewing area, 0-15 configurable（indicates cutting N×32 original coordinates）where as 0 indicates no cutting. The maximum cutting area is 15×32=480 original coordinates（one Pitch consists of 512 original coordinates，if the cutting exceeds one Pitch, it is allowed to subtract one Pitch from the configuration.）

[0x8070] Module_Switch3

Bit6：Gesture_Hop_Dis, whether to disable the frequency hopping function in Gesture Mode. Default setting is 0: enabled; configured to 1: disabled.

Bit5：Strong_Smooth：5-stage moving average smoothing, default setting is 0 (disabled). It is recommended not to enable this function unless the pitch is comparatively large and linearity is poor.

Bit0：Shape_En：Deformation processing, configured as 1: enabled; reset to 0: disabled.

[0x8071] GESTURE_DIS

Bit7~4：valid distance setting for swipe-up/down wakeup. The minimum valid swipe distance is the N/16 of the TP length. Configured to 0 indicates 8.

Bit3~0：valid distance setting for swipe-left/right wakeup. The minimum valid swipe distance is the N/16 of the TP length. Configured to 0 indicates 8.

[0x807C] Noise_Detect_Times

Bit7~6: Detect_Stay_Times, Number of tests taken on each frequency point in each noise test; recommended setting is 2.

Bit5~0: Detect_Confirm_Times, Confirmed noise level after repeated noise tests, recommended setting is 15~20.

[0x807D] Hopping_Flag

Bit7: Hopping_En, frequency hopping enable bit (1: enabled, 0: disabled).

Bit6: Range_Ext, frequency hopping range extension flag. For V1040, please configure to 1.

Bit5: Dis_Force_Ref, configured to 0: update the baseline compulsively after frequency hopping; configured to 1: do not update the baseline compulsively after frequency hopping.

Bit4: Delay_Hopping, configured to 1: frequency hopping occurs after finger touch leaves; this bit configured to 0 or Dis_Force_Ref configured to 1: Delay_Hopping is disabled.

Bit3~0: Detect_Time_Out， timeout for noise detection, unit: s.

[0x807E] Hoppging_Threshold

Bit3~0: Hopping_Hit_Threshold， condition for optical frequency selection; when the interference of the current operating frequency —the minimum interference >set value x4，then the optical frequency is selected and frequency hopping is enabled.

[0x809A-0x809B] Key_Sens

The sensitivity coefficient configuration of 4 independent touch keys, can be configured to 0 ~15 (16 grades in total). Higher grade indicates higher sensitivity. This is valid only for independent touch keys, which is to avoid sensitivity inconsistency between touch keys arising out of the nodal capacitance deviation during the independent touch key design process.

[0x809C] Key_Restrain

Bit3~0: Adjacent key restrain parameter. When the second largest value is greater than the largest value Key_Restrain/16, no touch key is reported. Recommended setting is 7±2.

[0x80A2] Data_Threshold

HotKnot technology uses frequency to indicate data. Two hotknot terminals perform data transmission by issuing signals of specified frequency. Data_Threshold is a threshold to distinguish whether there is signal or not. When LCD is turned off and no signal is present, the white noise received by HotKnot terminal is within 5. It is recommended that the Data_Threshold be greater than 5 and no less than 10.

[0x80A3] Pxy_Threshold

If the HotKnot proximity sensing function is enabled when LCD is on, employ differential method to filter interference. The threshold is differencial threshold. When the noise changes greatly, configure the differential threshold to a larger value. The threshold can be adjusted according to the touching area and distance of two hotknot terminals. It is suggested that this threshold be greater than 15 and 20 is highly recommended.

[0x80A4] Dump_Shift

The Dump_Shift is applicable to the amplification of HotKnot raw data. Normal configuration is 2~4.

[0x80A5] Rx_Gain

The Rx_Gain is applicable to the receiving hardware configuration of HotKnot. The mechanism is the same as that of configuration of TP Rx_Gain.

[0x80A6-0x80A9] Freq_Gain0~3

Software gain coefficient of HotKnot signals. The 7 frequencies used by HotKnot are ranged from 150KHz to 450KHz，one step is 50KHz。Adjust the software gain value according to the actual sampled raw data of frequency points to improve the data consistency between frequency points and minimize the signal diversity between different frequencies.

[0x80B3] Combine_Dis

Bit7~4: Distance for adjacent rectangles to be combined in Gesture mode, 0 to 15 configurable; the distance for adjacent touch points to be combined is calculated as: sqrt (2*(Combin_Dis)) pitch. For backward compatibility, being configured to 0 means the distance for adjacent touch points to be combined is 2 pitches.

Bit3~0: Distance for adjacent rectangles to be combined, 0 to 15 configurable; the distance for adjacent touch points to be combined is calculated as: sqrt (2*(Combin_Dis)) pitch. For backward compatibility, being configured as 0 means the distance for adjacent touch points to be combined is 2 pitches.

[0x80B4] Split_Set

Bit7~4: Distance for a large-area rectangle to be split, 0 to 15 configurable; distance for a large-area touch point to be split is calculated as: sqrt(2*(Split_Set)) pitch. For backward compatibility, being configured as 0 means defaulting as previous setting, and the distance for an adjacent touch point to be split is sqrt (12) pitch.

Bit3~0: Distance for a normal-size rectangle to be split, 0 to 15 configurable; distance for a normal-size touch point to be split is calculated as: sqrt (2*(Split_Set)) pitch. For backward compatibility, being configured as 0 means defaulting as previous setting, and the distance for a normal-size touch point to be split is sqrt (7) pitch.

3.3 Coordinate information

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

<table>
<tr><td>0x8140</td><td>R</td><td colspan="5">Product ID (first byte， ASCII)</td></tr>
<tr><td>0x8141</td><td>R</td><td colspan="5">Product ID (second byte， ASCII)</td></tr>
<tr><td>0x8142</td><td>R</td><td colspan="5">Product ID (third byte， ASCII)</td></tr>
<tr><td>0x8143</td><td>R</td><td colspan="5">Product ID (forth byte， ASCII)</td></tr>
<tr><td>0x8144</td><td>R</td><td colspan="5">Firmware version (HEX. low byte)</td></tr>
<tr><td>0x8145</td><td>R</td><td colspan="5">Firmware version (HEX. high byte)</td></tr>
<tr><td>0x8146</td><td>R</td><td colspan="5">x coordinate resolution (low byte)</td></tr>
<tr><td>0x8147</td><td>R</td><td colspan="5">x coordinate resolution (high byte)</td></tr>
<tr><td>0x8148</td><td>R</td><td colspan="5">y coordinate resolution (low byte)</td></tr>
<tr><td>0x8149</td><td>R</td><td colspan="5">y coordinate resolution (high byte)</td></tr>
<tr><td>0x814A</td><td>R</td><td colspan="5">Vendor_id (ID of the current module)</td></tr>
<tr><td>0x814B</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x814C</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x814D</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x814E</td><td>R/W</td><td>buffer status</td><td>large detect</td><td>Proximity Valid</td><td>HaveKey</td><td>number of touch points</td></tr>
<tr><td>0x814F</td><td>R</td><td colspan="5">track id is 32, indicates the signal is proximity sensing signal</td></tr>
<tr><td>0x8150</td><td>R</td><td>PxyOk</td><td colspan="4">Reserved</td></tr>
<tr><td>0x8151</td><td>R</td><td>PxyOk</td><td colspan="4">Reserved</td></tr>
<tr><td>0x8152</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x8153</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x8154</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x8155</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x8156</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x8157</td><td>R</td><td colspan="5">track id</td></tr>
<tr><td>0x8158</td><td>R</td><td colspan="5">point 1 x coordinate (low byte)</td></tr>
<tr><td>0x8159</td><td>R</td><td colspan="5">point 1 x coordinate (high byte)</td></tr>
<tr><td>0x815A</td><td>R</td><td colspan="5">point 1 y coordinate (low byte)</td></tr>
<tr><td>0x815B</td><td>R</td><td colspan="5">point 1 y coordinate (high byte)</td></tr>
<tr><td>0x815C</td><td>R</td><td colspan="5">Point 1 size (low byte)</td></tr>
<tr><td>0x815D</td><td>R</td><td colspan="5">point 1 size (high byte)</td></tr>
<tr><td>0x815E</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x815F</td><td>R</td><td colspan="5">track id</td></tr>
<tr><td>0x8160</td><td>R</td><td colspan="5">point 2 x coordinate (low byte)</td></tr>
<tr><td>0x8161</td><td>R</td><td colspan="5">point 2 x coordinate (high byte)</td></tr>
<tr><td>0x8162</td><td>R</td><td colspan="5">point 2 y coordinate (low byte)</td></tr>
<tr><td>0x8163</td><td>R</td><td colspan="5">point 2 y coordinate (high byte)</td></tr>
<tr><td>0x8164</td><td>R</td><td colspan="5">point 2 size (low byte)</td></tr>
<tr><td>0x8165</td><td>R</td><td colspan="5">point 2 size (high byte)</td></tr>
<tr><td>0x8166</td><td>R</td><td colspan="5">Reserved</td></tr>
<tr><td>0x8167</td><td>R</td><td colspan="5">track id</td></tr>
</table>

0x8168	R	point 3 x coordinate (low byte)
0x8169	R	point 3 x coordinate (high byte)
0x816A	R	point 3 y coordinate (low byte)
0x816B	R	point 3 y coordinate (high byte)
0x816C	R	point 3 size (low byte)
0x816D	R	point 3 size (high byte)
0x816E	R	Reserved
0x816F	R	track id
0x8170	R	point 4 x coordinate (low byte)
0x8171	R	point 4 x coordinate (high byte)
0x8172	R	point 4 y coordinate (low byte)
0x8173	R	point 4 y coordinate (high byte)
0x8174	R	point 4 size (low byte)
0x8175	R	point 4 size (high byte)
0x8176	R	Reserved
0x8177	R	track id
0x8178	R	point 5 x coordinate (low byte)
0x8179	R	point 5 x coordinate (high byte)
0x817A	R	point 5 y coordinate (low byte)
0x817B	R	point 5 y coordinate (high byte)
0x817C	R	point 5 size (low byte)
0x817D	R	point 5 size (high byte)
0x817E	R	Reserved
0x817F	R	KeyValue

Supplementary description on some registers:

[0x814A] Vendor_id

The ID of the current module is codetermined by pins *sensor_opt1* and *sensor_opt2* on the circuit. When the two pins are connected to different level status, there comes in 6 sensor IDs as shown below:

sensor_opt1	sensor_opt2	Vendor_id
GND	GND	0
VDDIO	GND	1
NC	GND	2
GND	300K	3
VDDIO	300K	4
NC	300K	5

[0x814E]:

Bit7: Buffer status, 1 = coordinate (or key) is ready for host to read; 0 = coordinate (or key) is not ready and data is not valid. After reading coordinates, host should configure this flag (or the entire byte) to 0 via I^2C.

Bit6: large detect, 1 indicates there is large-area touch on TP.

Bit4: HaveKey, 1 : Have touch key; 0 : No touch key (released).

Bit3~0: Number of touch points.

[0x814F]

When HotKnot proximity sensing function is enabled and another HotKnot-featured terminal is detected, GT911 will report the detection result to the host in coordinates. Therefore, the Number of touch points will add 1. The track id of the added touch point is fixed to 32, and Pxyos is set to 1. Please note that the address of the added touch point is fixed to the address of the first coordinate.

[0x8177]: KeyValue

The address of KeyValue is not fixed. Instead, it stays behind valid coordinates. For example, 0x817F is the address of the key when there are 5 coordinates on TP. However, if there are 4 coordinates on TP, the address of the key will be 0x8177.

（Gesture Features: share the addresses with the coordinate information）

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x8140	R	Gesture ID (first Byte，ASCII G)							
0x8141	R	Gesture ID (second Byte，ASCII E)							
0x8142	R	Gesture ID (third Byte，ASCII S)							
0x8143	R	Gesture ID (forth Byte，ASCII T)							
0x8144	R	Gesture Firmware version (HEX.low byte)							
0x8145	R	Gesture Firmware version (HEX.high byte)							
0x8146	R	x coordinate resolution (low byte)							
0x8147	R	x coordinate resolution (high byte)							
0x8148	R	y coordinate resolution (low byte)							
0x8149	R	y coordinate resolution (high byte)							
0x814A	R	Reserved							
0x814B	R/W	Gesture types（character ASCII indicates 0x21-0x7F），swipe right（0xAA），swipe left（0xBB），swipe down（0xAB），swipe up（0xBA），double-tap（0xCC）, double-tap on touch key (0xCC, key value is stored at coordinate address)							
0x814C	R	The number of gesture touch point（coordinates stored at 0x9420）							
0x814D	R	Gesture start point x coordinate（low byte）							
0x814E	R	Gesture start point x coordinate（high byte）							
0x814F	R	Gesture start point y coordinate（low byte）							
0x8150	R	Gesture start point y coordinate（high byte）							
0x8151	R	Gesture end point x coordinate（low byte）							

0x8152	R	Gesture end point x coordinate（high byte）
0x8153	R	Gesture end point y coordinate（low byte）
0x8154	R	Gesture end point y coordinate（high byte）
0x8155	R	Gesture Width（low byte）
0x8156	R	Gesture Width（high byte）
0x8157	R	Gesture Height（low byte）
0x8158	R	Gesture Height（high byte）
0x8159	R	Gesture Mid X coor(low byte)
0x815A	R	Gesture Mid X coor(high byte)
0x815B	R	Gesture Mid Y coor(low byte)
0x815C	R	Gesture Mid Y coor(high byte)
0x815D	R	Gesture P1 X coor(low byte)
0x815E	R	Gesture P1 X coor(high byte)
0x815F	R	Gesture P1 Y coor(low byte)
0x8160	R	Gesture P1 Y coor(high byte)
0x8161	R	Gesture P2 X coor(low byte)
0x8162	R	Gesture P2 X coor(high byte)
0x8163	R	Gesture P2 Y coor(low byte)
0x8164	R	Gesture P2 Y coor(high byte)
0x8165	R	Gesture P3 X coor(low byte)
0x8166	R	Gesture P3 X coor(high byte)
0x8167	R	Gesture P3 Y coor(low byte)
0x8168	R	Gesture P3 Y coor(high byte)
0x8169	R	Gesture P4 X coor(low byte)
0x816A	R	Gesture P4 X coor(high byte)
0x816B	R	Gesture P4 Y coor(low byte)
0x816C	R	Gesture P4 Y coor(high byte)

（Gesture coordinate information）

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x9420	R	Gesture point 1 x coordinate（low byte）							
0x9421	R	Gesture point 1 x coordinate（high byte）							
0x9422	R	Gesture point 1 y coordinate（low byte）							
0x9423	R	Gesture point 1 y coordinate（high byte）							
0x9424~0x951F	R	Gesture point 2~64 coordinate（the number of coordinate is the value of 0x814C）							

3.4 Command status registers of GT911

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x81A8	R	GT911_Status：0x00：touch detection mode；0x88：slave approach mode；0x99：master approach mode；0xAA：Receive mode；							

		0xBB：Send mode, indicates the Transmit Buffer is flushed correctly.
0x 81A9	R	GT911_Status_Bak：GT911_Status backup

[0x81A8] GT911_Status

0x00：indicates GT911 will only perform touch detection, no HotKnot-relevant operations implemented.

0x88：When HotKnot proximity sensing function is enabled, the host sends command 0X20 to enable GT911 to enter slave approach mode (works as receiving terminal). In this mode, hotkont proximity sensing and touch detection alternate. When successfully detect the transmitting terminal, GT911 will report the detection result to the host in coordinates (track id is 32). The host can issue command 0X28 to enable GT911 to exit slave approach mode.

0x99：When HotKnot proximity sensing function is enabled, the host sends command 0X21 to enable GT911 to enter master approach mode. In this mode, hotkont proximity sensing and touch detection alternate. When successfully detect the slave/receiving terminal, GT911 will report the detection result to the host in coordinates (track id is 32). The host can issue command 0X29 to enable GT911 to exit master approach mode.

0xAA： When GT911 successfully detect another hotknot-featured terminal, the host downloads and sends the hotknot transmission firmware to GT911. When the firmware operates, GT911 enters Receive mode by default. In this mode, GT911 will not implement any operation related to touch detection, and it will keep detecting data from the transmitting terminal. Once a data frame is received, GT911 will notify the host to process the data via INT.

0xBB：When GT911 successfully detect another hotknot-featured terminal, the host downloads and sends the hotknot transmission firmware to GT911. When the firmware operates, GT911 enters Receive mode by default. In this mode, when the Transmit Buffer is flushed correctly, GT911 switches to Send mode. When the data in the Transmit Buffer is transmitted successfully, GT911 will notify the host to process the data via INT. When data processing is completed, GT911 switches to Receive mode and perform Leave Detection, until the Transmit Buffer is correctly flushed again.

When GT911 is implementing hotknot-related operations, the host can distinguish whether the previously-issued command is transmitted successfully and whether resending is needed, or decide which hotknot command to be sent by querying GT911_Status.

[0x81A9] GT911_Status_Bak

Backup of GT911_Status. It is suggested that the host reads the GT911_Status and GT911_Status_Bak simultaneously. Only when these two values are the same, can the status be considered valid, thus reducing the interference to I2C bus which causes data errors.

3.5 Hotknot status registers

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xAB10	R	SendStatus: Send status register							
0xAB11	R	RevStatus: Receive status register							
0xAB12	R	SendStatusBak: Send status register backup							
0xAB13	R	RevStatusBak: Receive status register backup							
…	R	NC (11 bytes reserved)							
0xAB1F	R/W	When there is work needs to be handled by the host, GT911 will write 0xAA to this address and notify the host to handle the work via INT. After the work is finished, the host sends another command other than 0xAA. And GT911 will implement the command. Otherwise, it will wait for 2.5s.							

Supplementary description on some registers:

The data read from this area is valid only when GT911 operates in Receive Mode or Send Mode, that is to say, only when GT911_Status is 0xAA or 0xBB.

[0xAB10] SendStatus

This register indicates the Send status in Send Mode.

0x01: indicates GT911 is in idle state. When a data frame is transmitted successfully and there is no data needs to be sent, GT911 will automatically switch to Receive Mode and perform Leave Detection. The host sends the outgoing data to the hotknot Transmit Buffer in this state.

0x02: indicates GT911 is transmitting data. The host cannot modify the data in the Transmit Buffer in this state.

0x03: all data in Transmit Buffer is transmitted successfully. GT911 notifies the host to process the data via INT. After reading the status, the host writes a number other than 0xAA to 0xAB1F, then GT911 will automatically switch to idle state and enter Receive Mode and perform Leave Detection.

0x04: The data sent by the host to the Transmit Buffer fails to pass the verification (incorrect or byte length does not match). GT911 notifies the host to handle via INT. After reading the status, the host writes a number other than 0xAA to 0xAB1F, and then resends the previously-issued data again.

0x05: GT911 has finished transmitting a data frame but the transmission fails. Instead of notifying the host via INT, GT911 will resend the data frame automatically.

Please note that, if the transmission is unsuccessful, GT911 will not stop resending until the data is transmitted successfully. Therefore, the Send Failure status needs to be defined as “timeout setting of the host” or “Leave Detection”.

0x07: GT911 has detected that the receiving terminal has left. After the host capturing this status, GT911 may exit Send mode. When transmitting a data frame successfully, GT911will enable Leave detection. GT911 can distinguish whether the receiving terminal exists or not by detecting the feedback signal on the frequency sweep signal sequence. If no feedback signal is detected for 1s, it is regarded that the receiving terminal has left.

[0xAB11] RevStatus

This register indicates the Receive status in Receive Mode.

0x01: indicates GT911 is in idle state. It is detecting data from the transmitting terminal but no valid signal has been detected yet.

0x02: indicates GT911 has detected the start signal and is receiving data.

0x03: indicates GT911 has received a data frame successfully and has sent it to the Receive Buffer. GT911 will notify the host to process the data via INT. After reading the data in Receive Buffer, the host has to write a number other than AA to 0xAB1F.

0x04: indicates GT911 has received a data frame but does not pass the CRC16 verification. Instead of notifying the host to handle via INT, GT911 will automatically detect the start signal again.

Please note that if the CRC verification fails or an overlong void signal is received, it will not stop detecting the start signal until the data frame is received successfully. Therefore, Receive Failure should be implemented by the host through timeout setting.

0x07:GT911 detected that the transmitting terminal has left. After the host capturing this status, GT911 may exit receive mode. When receiving a data frame successfully, GT911 will start Leave detection. GT911 can distinguish whether the transmitting terminal exists or not by detecting the frequency sweep signal sequence sent by the transmitting terminal. If no signal is detected for 1s, it is regarded that the transmitting terminal has left.

[0xAB12] SendStatusBak

Backup for SendStatus. Before GT911 notifies the host via INT, SendStatus will assign the value to SendStatusBak. The SendStatus is valid only when the host reads the same value in SendStatus and SendStatusBak. If the values are different, the host will read the values again 2ms later, thus improving the anti-ESD capability.

[0xAB13] RevStatusBak

Backup for SendStatus. Before GT911 notifies the host via INT, RevStatus will assign the value to RevStatusBak. The RevStatus is valid only when the host reads the same value in SendStatus and SendStatusBak. If the values are different, the host will read the values again 2ms later.

3.6 HotKnot Transmit Buffer

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xAC90	W	DataLength: valid data length , <129 bytes							
0x AC91	W	Data0							
0xAC92	W	Data1							
…	W	…							
0xAD10	W	Data127							
0xAD11	W	CheckSum							
…	NC	Reserved							
0xAD91	W	Data_Fresh data updated flag (0xAA written by the host)							

Supplementary description on some registers:

This area can be written only when GT911 operates in Receive Mode, that is to say, GT911_Status is 0xAA. Otherwise, unpredictable results will occur.

[0xAC90] DataLength

The maximum capacity of a data frame supported by HotKnot is 128 Bytes; DataLength must be shorter or equal to 128 and must be even number.

[0xAD11] CheckSum

The address of CheckSum is not fixed. It stays behind the valid data. Checksum check starts from 0xAC90. For example, if there are 2 data bytes, the address of Checksum is 0xAC93. The value is the complement of the sum.

[0xAD91] Data_Fresh

The host writes data to other addresses before writing 0xAA to 0xAD91; that is to say, the host sets the Transmit Buffer flushed flag; after GT911 finds the flag, it will check whether the data in the Transmit Buffer passes the verification. If the data passes the verification, GT911 will switch to Send Mode and start transmission immediately; if the data does not pass the verification, GT911 will notify the host to handle via INT.

3.7 Hotknot Receive Buffer

Addr	Access	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xAE10	R/W	buffer status							
0x AE11	R/W	DataLength valid data length , <129 bytes							
0xAE12	R	Data0: the first data byte							
0xAE13	R	Data1: the second data byte							
…	R	…							
0xAE91	R	Data127 : the 128th data byte							
0xAE92~ 0xAE93	R	Crc16Check data CRC16 verification. Please note that it should stay behind the data, not fixed in this address. big-edian mode							

Supplementary description on some registers:

The data in this area is valid only when GT911 operates in Receive Mode, GT911_Status is 0xAA, and RevStatus is 0x03.

[0xAE10]buffer status

bit7: buffer status as 1 indicates data in Receive Buffer is ready to be read.

[0x AE11]DataLength

Valid data length, < 128 bytes.

[0xAE92~0xAE93] Crc16Check

Data CRC-CICTT verification, big-edian mode。

Instruction on check mechanism:

As for data frame whose length is n, the result of CRC check is：the check of the n data bytes+length. For example：the data frame length is 112 bytes; the host needs to read 114 bytes (112 data bytes+2 bytes CRC16 check) from the address 0xAE12. The host figures out the CRC of "112 data bytes+length", and compares it with the CRC at (0xAE12+112). If the two CRCs are the same, the data passes verification; otherwise, verification fails. Please note that, the calculation of CRC and length is performed in the end of the process, not at the beginning.

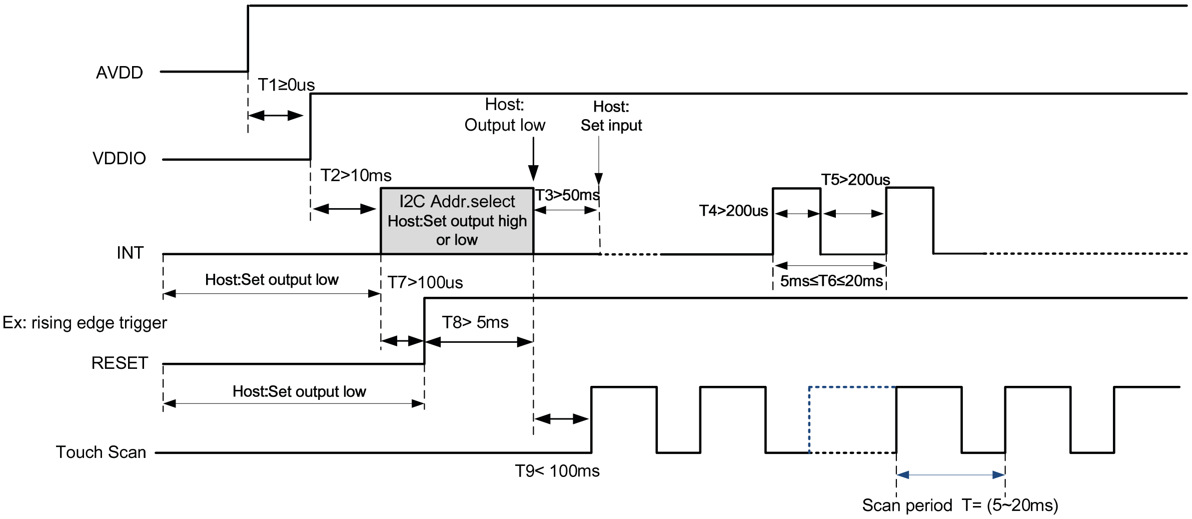
Reference Code for Crc16 calculation (note: big-edian mode)：

```
#define FREQ_CRC_SEED        0x1021

//calculate the CRC16 value of the defined length of data in SrcData

unsigned short    Crc16(unsigned char *SrcData,unsigned char length)
{
    unsigned short    crc=0xFFFF;
    unsigned char    i,j;
    unsigned char    value;
    bit flag;
    bit c15;

    for (i= 0; i < length; i++)
    {
     value=SrcData[i];
     for (j= 0; j < 8; j++)
     {
        flag = (value & 0x80);
        c15 = (crc & 0x8000);
        value <<= 1;
        crc <<= 1;
        if(c15^flag)
        crc ^= FREQ_CRC_SEED;
     }
  }
  return crc;
}
```

4. Power-on Initialization and Modification on Register Value

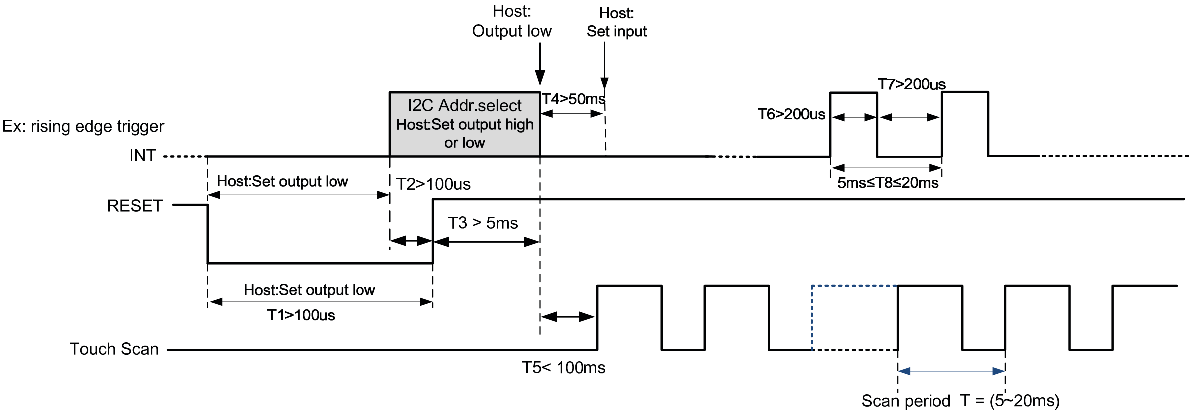
4.1 Power-On Timing of GT911

After power-on, the host needs to control such GT911 pins as AVDD, VDDIO, INT and Reset according to the timing sequence shown below:



Whether host outputs high or low after INT T2 depends on which I2C slave device address the host employs to communicate with GT911. If the address is 0x28/0x29, host outputs high; if the address is 0xBA/0xBB, host outputs low.

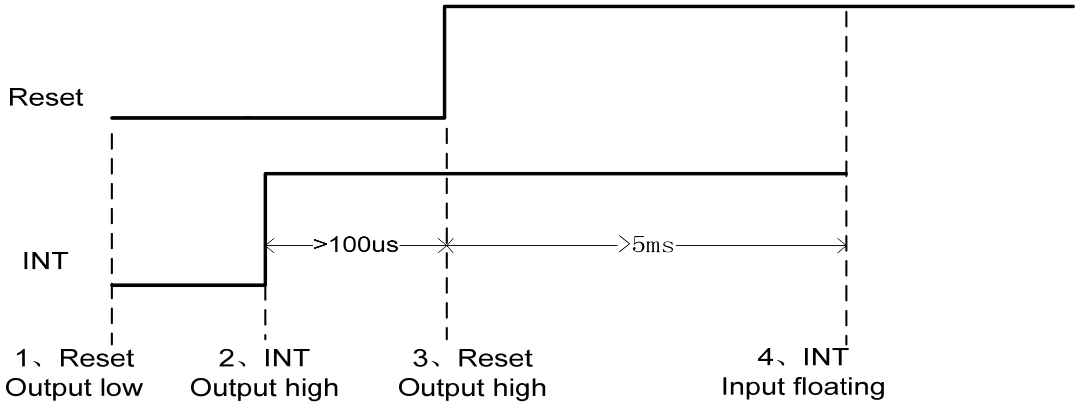
Timing for host resetting GT911:



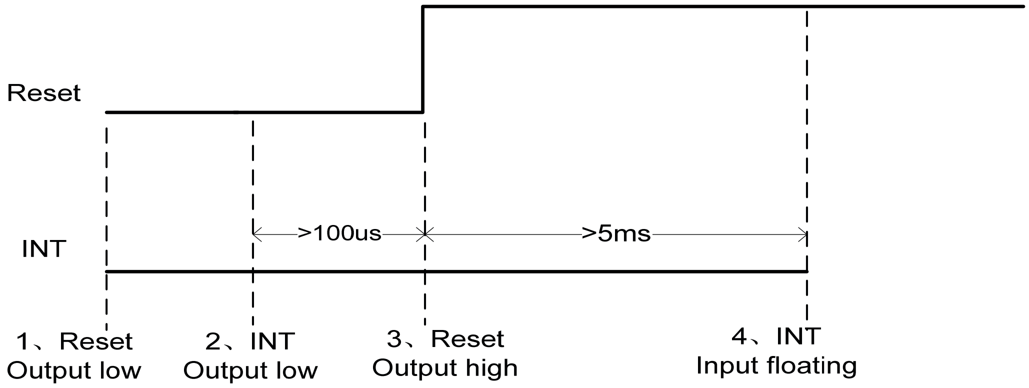
4.2 I^{2}C address selection during power-on or reset process

GT911 supports two I^{2}C slave device addresses: 0xBA/0xBB and 0x28/0x29. Host needs to select the I^{2}C slave device address during power-on initialization or Reset process via Reset pin. Host can select the I^{2}C address by controlling Reset and INT timing sequence. Diagram below provides details:

Timing sequence for setting address to 0x28/0x29:



Timing sequence of setting address to 0xBA/0xBB:



4.3 Send Configuration after Power-on

During the power-on process, after host converts its INT to input floating, it is required to wait for 50ms before sending configuration information.

4.4 Register Value Modification

GT911 supports Register Value Modification. When modifying any register in the configuration area (0x8047－0x80FE) based on the timing sequence as specified in section 2, it is required to update Config_Chksum (0x80FF) and eventually set Config_Fresh (0x8100) to 1. Otherwise, the modification is invalid; when modifying any registers outside configuration area, it is unnecessary to modify Config_Chksum and Config_Fresh.

5. Coordinates Reading

The host reads coordinates by periodic polling or interrupt request.

When periodic polling is adopted, the host reads coordinates through the following steps:

1. Based on the time sequence indicated in section 2, the host first reads register 0x814E. If the data in buffer is

ready (butter status: 1), it reads coordinate and touch key information based on finger touch number and touch key status.

2. If it is found out in step 1 that data in buffer is not ready (buffer status: 0), it will read again 1ms later.

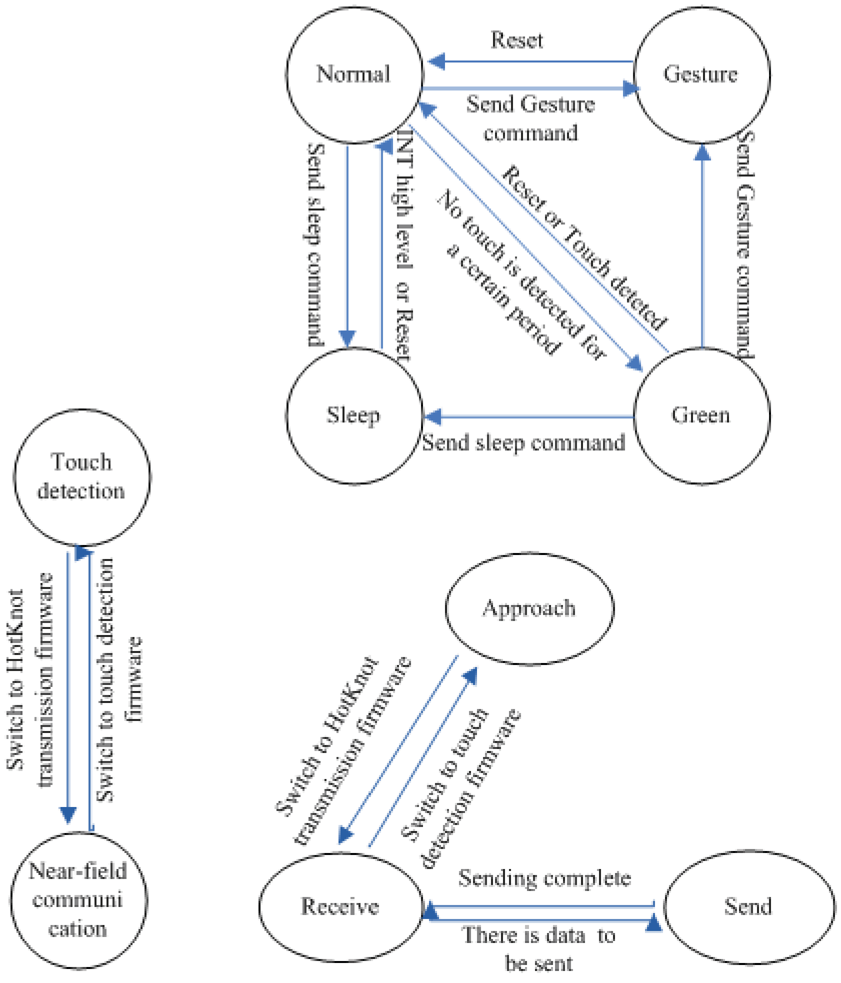
When interrupt request is used for reading, the host will read coordinates through the above polling procedure after interrupt is triggered.

The timing sequence for GT911 interrupt signal output (take the rising-edge triggered interrupt for example. The timing for falling-edge triggered interrupt is similar to this one):

1. In standby mode, INT outputs low level.
2. Output rising edge when any coordinate is updated.
3. After output rising edge in step 2, INT will keep outing high level until next cycle (the cycle is configurable by setting Refresh_Rate). The host is supposed to finish the reading within one cycle and reset buffer status (0x814E) to 0.
4. After output rising-edge in step 2, if the host fails to finish reading coordinates within one cycle, GT911 will output one INT pulse again instead of update coordinates even if it detects that the coordinate is updated.
5. If the host still fails to read coordinate, GT911 will keep outputting INT pulse.

6. Operation Modes Switchover

6.1 Operation Modes



GT911 can switch between Normal mode and Low Power mode automatically by default. When touch is pressing down or after touch is released for a certain period (0s ~ 15s, configurable), GT911 operates in Normal mode. If no touch is detected within that period, GT911 enters Low Power mode (low-speed scan).

a) Normal Mode

When GT911 is operating in Normal mode, its fastest coordinates refreshing cycle is 5ms-20ms (subject to configuration. One step is 1ms).

When no touch is detected for a certain period (0s~15s, subject to configuration, one step is 1s) in Normal mode, GT911 will enter Green mode to reduce power consumption.

b) Green(Low Power) mode

In Green mode, the scanning cycle for GT911 is about 40ms. It automatically enters Normal mode if any touch is detected.

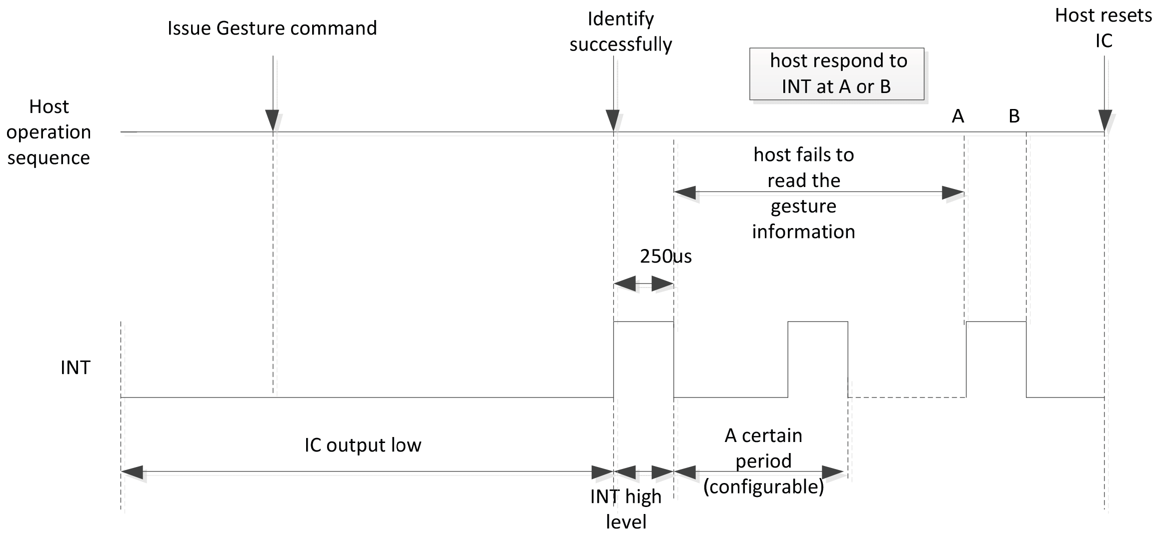
c) Gesture mode

After the host enables GT911 to enter Gesture mode by sending I2C command 8 to 0x8046, then to 0x8040, wake-up can be achieved by swipe, double-tap, or writing specified lower-case letters on TP.

In Gesture mode, when GT911 detects any finger swipe on TP for a sufficiently long distance, INT will output high level or a pulse that is greater than 250us. The host wakes up and turns on the screen after receiving such high level or pulse.

In Gesture mode, when GT911 detects any double-tap on TP, INT will output high level or a pulse that is greater than 250us. The host wakes up and turns on the screen after receiving such high level or pulse.

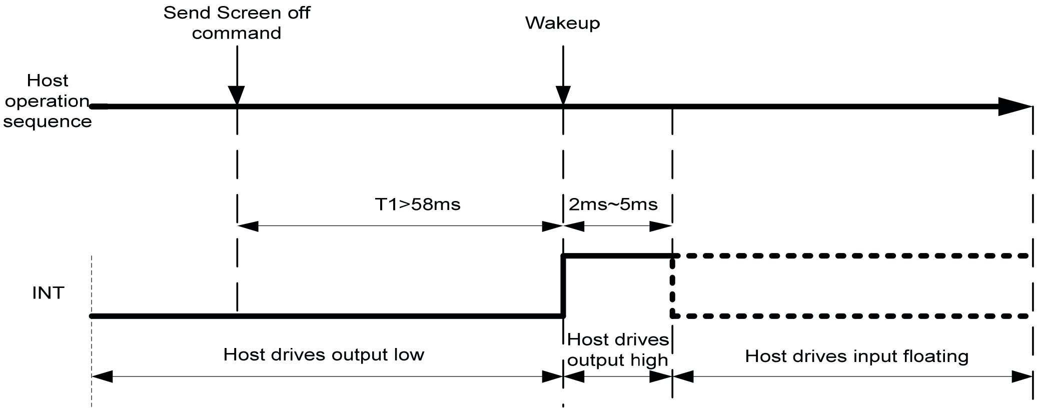
In Gesture mode, when GT911 detects any writing of specified lower-case letters on the TP, INT will output high level or a pulse that is greater than 250us. The host wakes up and turns on the screen after receiving such high level or pulse.



d) Sleep mode and wakeup

The host enables GT911 to enter Sleep mode by sending I2C command 0x05 to 0x8040 (requires INT to output low before the command). GT911 needs to exits Sleep mode, the host can employ INT high-level wakeup or reset wakeup. If the host employs INT high-level wakeup, the operation sequence is: host drives INT output high for 2ms~5ms, and then drives INT input floating. GT911 enters Normal mode after being woken up and it will output a touch-release pulse in every cycle. The host must read the interrupts of three cycles. Otherwise, GT911 will not stop outputting such pulse. The time interval between issuing the screen-off command and wakeup should be longer than 58ms. If the host employs reset wakeup, it is required to control the INT pin and Reset pin during the power-on initialization as mentioned in section 4.1.

Timing for INT high-level wakeup:



e) Approach Mode

When the HotKnot function is enabled, GT911 is operating in Approach Mode. When GT911 exits Approach Mode, the host can send command 0x20 or 0x21 to enable GT911 to enter Approach mode again. In this mode, touch detection and near-field proximity detection alternate. If the host sends 0x21 to GT911, GT911 will work as a transmitting terminal and transmit signals with a specified pattern and frequency via driving and sensing channels. Then, GT911 detects whether there are feedback signals with the same specified pattern and frequency from the receiving terminal. This helps to determine whether any receiving terminal exists. If the host sends 0x20 to GT911, GT911 will work as a receiving terminal and detect signals with a specified pattern and frequency from the transmitting terminal. If such a signal is detected, GT911 responds using signals with the specified pattern and frequency to the transmitting terminal. In Approach mode, when detecting any communicable terminal within the near-field range, GT911 will notify the host via INT to capture status. To ensure reliable detection between the transmitting terminal and the receiving terminal, it is required to keep detecting for a minimum of 150ms after the two terminals have detected each other. Then the host downloads and sends HotKnot transmission firmware to enable GT911 to enter Receive mode.

f) Receive Mode

When GT911 operates in Approach mode, after notified that GT911 has successfully detected another HotKnot terminal, the host downloads and sends HotKnot transmission firmware to enable GT911 to enter Receive mode. In Receive mode, GT911 continues to detect frame start signal, once the signal is detected, GT911 begins to detect and receive data. When the receiving process is complete, GT911 verifies the data. If GT911 finds erroneous data, the receiving process begins again. If the data is found to be correct, GT911 notifies the host via INT to read data in the Receive Buffer.

g) Send Mode

When GT911 works in Receive mode, the host sends outgoing data to the Transmit Buffer. When detecting that the Transmit Buffer is flushed and there is data to be sent, GT911 automatically switches from Receive mode to Send mode. In Send mode, GT911 sends a frame start signal. If it detects ACK fed back from the receiving terminal, it continues to send the data signal. After sending a data chunk, GT911 begins to detect ACK. If it does not detect any ACK or if it detects an erroneous ACK, GT911 will resend the data chunk. If this resending fails over 5 times, it will resend the current data frame another time to the receiving terminal until the host enables GT911 to exit Send mode due to timeout. If GT911 detects ACK and sends the data successfully, it will automatically switch to Receive mode after the host completes the data processing or due to timeout.

7. Host System Driver Modification in Gesture Mode

7.1 Enter Gesture mode after screen-off

a) If screen-off is achieved by pressing Power key (or any other key), send Command 8 to 0x8046, then to 0x8040;

b) If screen-off is achieved due to timeout, send Command 8 to 0x8046, then to 0x8040;

c) When the screen is off, if there is swipe, double-tap or writing of specified lower-case letters on TP, the INT pin will output a high level or a pulse that is greater than 250us to notify the host. The host reads the data at 0x814B after receiving such pulse. If the data meets wake-up conditions, the host wakes up, then resets GT911 and turns on the screen. Otherwise, the host resets 0x814B and waits for the next pulse or high level.

7.2 Enter Sleep Mode after screen-off

a) If screen-off is achieved by pressing Power key (or any other key), send Command 5 to 0x8040 ;

b) If screen-off is achieved due to timeout, send Command 5 to 0x8040 ;

c) In Sleep mode, host can be awakened only by pressing Power key (or Home key).

7.3 Press Power (or Home) key to wake up host

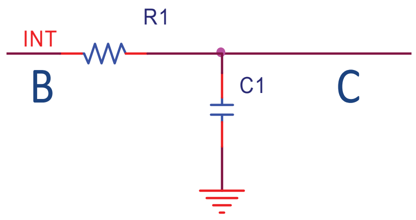
Press power key (or Home key) in any mode to wake up the host. And then, host resets GT911 based on reset timing sequence and executes reset process.

7.4 Recommended to apply in conjunction with IR

If gesture wake-up function is applied in conjunction with IR, the host can enable GT911 to enter Sleep mode to reduce power consumption when IR detects shielding object while screen-off. Otherwise, GT911 enters Gesture mode. To enter different modes, use the methods listed above (reset is required before sending command).

7.5 Hardware circuit modification

When debugging, connect RC circuit to INT pin in series (R: 680 , C: 1nF) as shown below:



Connect B to GT911 INT and C to host INT; pull-up resistor connected to host INT is not required.

8. Reading Coordinate in Gesture Mode

In Gesture mode, when 0x814B is not 0, the host can describe the wakeup trajectory of user by reading the gesture features and gesture coordinates.

Gesture features: the host reads registers ranged from 0x814D to 0x816C and captures the following Gesture features: start coordinate, end coordinate, trajectory width, trajectory height, trajectory central point and the four extreme points of the trajectory. The host can sketchily describe the wakeup trajectory of user by these features and the gesture type indicated by 0x814B.

Gesture coordinates: the host obtains the number of touch points of the trajectory by reading the register 0x814C. And then it reads the registers ranged from 0x9420 to 0x951F based on the principle that every four register correspond to one touch point. Finally, the host can describe the real touch trajectory of user by synthesizing these information.

9. Time Limit for Downloading HotKnot Firmware

In HotKnot mode, to ensure the I2C transmission rate and considering factors such as time expended by system calls and user experience, the time limit for transmitting HotKnot firmware should be within 800ms, that is to say, the I2C transmission rate should be no less than 200Kbps. FAEs should make sure this requirement is fulfilled when debugging for customers.

Since HotKnot may employ the frequencies such as 200K, 250K, 300K, 350K, 400K, 450K and etc, in order to avoid interference caused by trace routing, it is recommended that the I2C transmission rate should be different from the above frequencies, with a deviation greater than 10KHz, for example, 325Khz.

10. Revision History

Revision	Date	Description
Rev.00	2014-08-04	Preliminary version