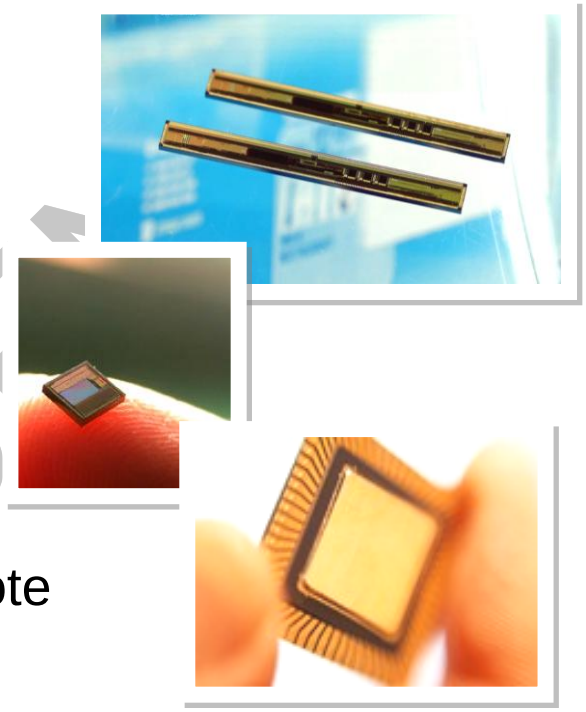
Raydium 瑞鼎科技股份有限公司
Raydium Semiconductor Corporation



WT030 Touch Application Note

Single Chip Driver and Touch Controller
with 16.7M Color
for 480RGBx480 AMOLED

Revision：2.0

Date：Aug.20, 2020

Revision History

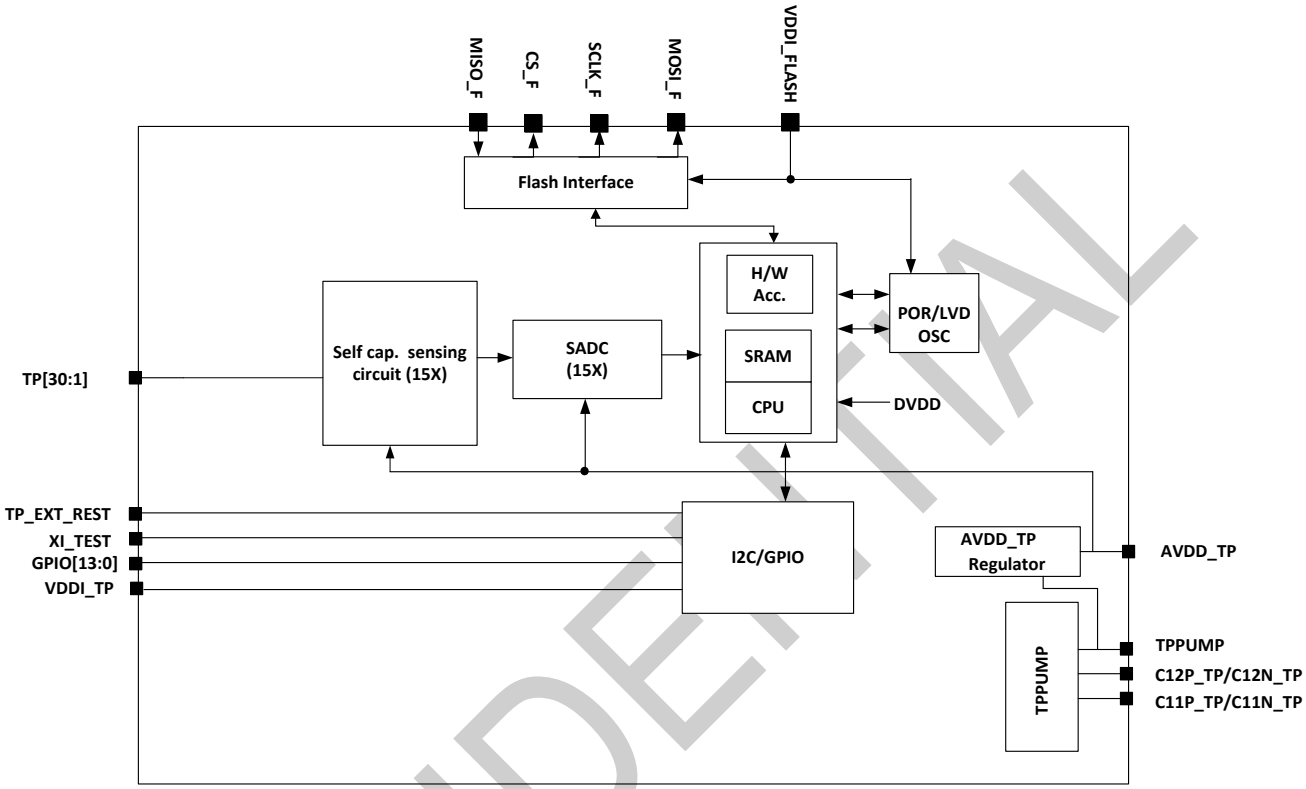
Version	Date	Prepared By	Checked By	Description	page
V0.1	2018/12/20	Eddie	CN Lin	First Release	
V0.2	2019/01/04	Eddie	CN Lin	Modify touch status command	18
V0.3	2019/03/13	Roger	CN Lin	Add I2C Device ID	12
V0.4	2019/04/17	Gary	CN Lin	Add Selftest	26
V0.5	2019/05/02	Gary	CN Lin	Modify touch report command, add i2c packet format	18
V0.6	2019/05/24	Gary	Roger	Modify touch report flow	24
V0.7	2019/07/04	Gary	Roger	Add touch record data chapter	29
V0.8	2019/07/11	Gary	Roger	Modify i2c packet format	27
V0.9	2019/07/15	Gary	Roger	Fix i2c packet format	27
V1.0	2019/07/19	Gary	Roger	Modify i2c packet format, support driver SDK1.3	27
V1.1	2019/09/05	Roger	CN Lin	Add XTX Ext. Flash	16
V1.2	2019/09/12	Eddie	Gary	Modify Touch firmware upgrade	23
V1.3	2019/10/30	Gary	Eddie	Add palm command	19
V1.4	2020/02/13	Eddie	Roger	Add ESMT Ext. Flash	16
V1.5	2020/04/10	Eddie	CN Lin	Add description about reset	8,9
V1.6	2020/04/16	Eddie	Roger	Add touch status command	19
V1.7	2020/06/16	Gary	Roger	Add touch manual mode command	19
V1.8	2020/06/29	Gary	Roger	Modify power state	17
V1.9	2020/07/08	Gary	Roger	Modify touch manual mode command	19
V2.0	2020/08/20	Eddie	Roger	Modify power state	17

Table of Contents

1 Features

- **Single chip AMOLED controller/driver/touch with display RAM**
- **Touch**
  - ARM Cortex-M0 32-bit micro-processor
  - Support 64KB External Flash
  - On chip 4KB data memory and 28KB program memory
  - On chip Sensing memory and Baseline memory
  - Support serial wire debug interface with 2 watch points and 4 break points
  - Support in-system-programming
  - I2C-compatible serial interface
  - Power management unit – Normal / IDLE mode / Wake up mode/ Sleep mode
  - Sensing mode – Node Self Scan for charge sensing / Parallel Scan(non driving)
  - Support Node base compensation
  - 30 Sensing channels for Self Mode
  - Low power consumption
  - Auto noise filter function
- **Logic / interface power supply voltage VDDI = 1.65V ~ 1.95V**
- **Analog power supply voltage VDD = 2.7V ~ 3.6V**
- **Low voltage detection (VDDA / VDDI) for abnormal power off**
- **Package: COF**

2 Block Diagram



Touch Circuit

The WT030 supports 30CH touch-controller IC with I2C serial interface for capacitive touch panel applications

3 Pin Description

3.1 Power Supply Pins

Signal	I/O	Function
VDDB	P	Power supply for DC/DC converter VDDB, VDDA , VDDA_TP and VDDR should be the same input voltage level
VDDA	P	Power supply for analog system VDDB, VDDA , VDDA_TP and VDDR should be the same input voltage level
VDDR	P	Power supply for regulator system VDDB, VDDA , VDDA_TP and VDDR should be the same input voltage level
VDDA_TP	P	Power supply for TP circuit. VDDB, VDDA , VDDA_TP and VDDR should be the same input voltage level
VDDI	P	Power supply for interface system except MIPI interface
VDDI_TP	P	Power supply for TP interface circuit
VDDI_FLASH	P	Power supply for TP FLASH circuit (1.65~1.95v)
VCC	P	Power supply for DVDD regulator
VSSB	P	System ground for DC/DC converter
VSSA	P	System ground for analog system
VSSR	P	System ground for regulator system
VSSAM	P	System ground for internal MIPI analog system
VSSA_TP	P	System ground for TP circuit
VSSI	P	System ground for interface system except MIPI interface
DVSS	P	System ground for internal digital system
AVSS	P	System ground for source OP system.
MTP_PWR	P	MTP programming power supply pin (7.5V typical) Must be left open or connected to DVSS in normal condition.
AVDD_SD	P	Power supply for source circuit. If not use, please leave it open.

3.2 TP Interface Pins (PWR/GND=VDDI_TP / VSSI)

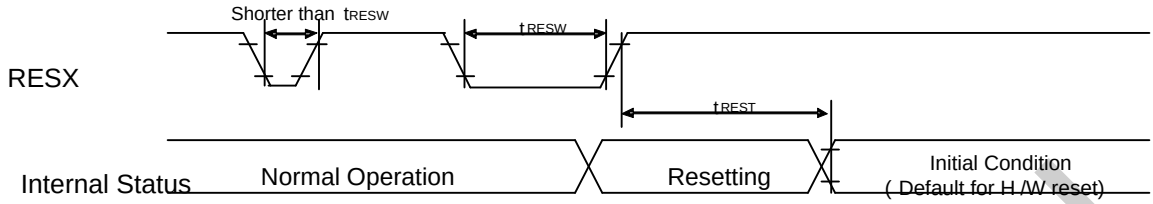
Signal	I/O	Function
TP_EXT_RESET	I	System reset (default: H), (Default : Internal Pull-High), If no use, please leave it open. TP_EXT_RESET = “L”: System reset TP_EXT_RESET = “H”: Normal operation
XI_TEST	I	TEST mode (Default : Internal Pull-Low), If no use, please leave it open. XI_TEST = “H”: Enable test mode XI_TEST= “L”: Normal operation
TP_SCL	I/O	I^2C clock (I2C_SCL) (internal pull -High), If no use, please leave it open. For I2C interface, this line must be connected to a positive supply via a pull-up resistor. The value of the resistor should be chosen to ensure that the rise time is within the limits set by the I2C specification. The value typically would fall within the ranges of 1.5k~10kohm.
TP_SDA	I/O	I^2C data(I2C_SDA) (internal pull -High) , If no use, please leave it open. For I2C interface, this line must be connected to a positive supply via a pull-up resistor. The value of the resistor should be chosen to ensure that the rise time is within the limits set by the I2C specification. The value typically would fall within the ranges of 1.5k~10kohm.
TP_INT	O	Touch INT output , If no use, please leave it open.
GPIO3	I/O	RS232_TX/SWDIO (internal pull-High) , If no use, please leave it open.
GPIO4	I/O	SWCLK(internal pull-High) , If no use, please leave it open.
GPIO5	I/O	HOLD(internal pull-High) , If no use, please leave it open.
GPIO6	I/O	WP(internal pull-High) , If no use, please leave it open.
GPIO7	I/O	BOOT_DEVICE(internal pull -High) , If no use, please leave it open.
GPIO8 ~ GPIO13	I/O	GPIO pins(internal pull-High) , If no use, please leave it open.

3.3 TP Interface Pins (PWR/GND=VDDI_FLASH / VSSI)

Signal	I/O	Function
CS	O	SPI Chip select to FLASH , If no use, please leave it open.
SCLK	O	SPI serial clock to FLASH, If no use, please leave it open.
MOSI	O	SPI data output to FLASH, If no use, please leave it open.
MISO	I	SPI data input from FLASH (Internal Pull-Low) , If no use, please leave it open.

4 Reset, INT and I2C Interface

4.1 Reset Timing(RESX)



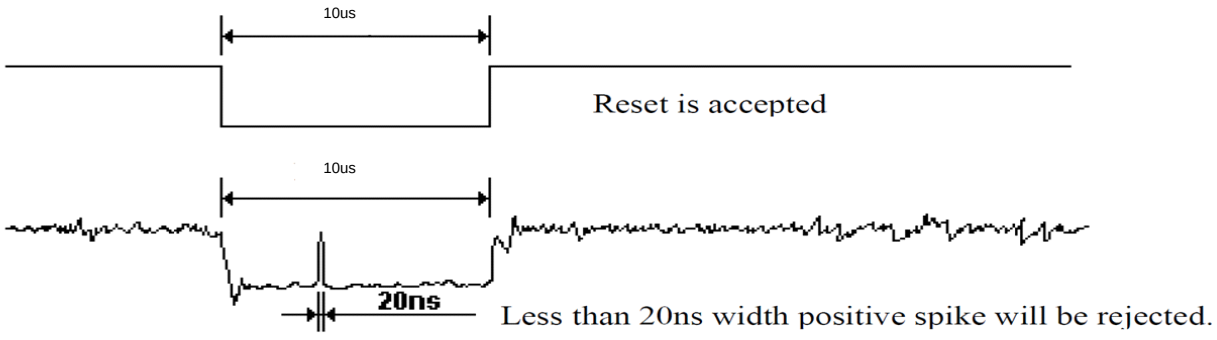
Reset input timing:
VDDI=1.65 to 1.95V, VDD=2.7 to 3.6V, AGND=DGND=0V, Ta=-30 to 85℃

Symbol	Parameter	Related Pins	MIN	TYP	MAX	Note	Unit
t_{RESW}	*1) Reset low pulse width	RESX	10	-	-	-	μs
t_{REST}	*2) Reset complete time	-	-	-	5	When reset applied during Sleep in mode	ms
		-		-	120	When reset applied during Sleep out mode	ms

Note 1. RESX is whole chip reset, if setting RESX, display and touch will be resetting in the same time.
Note 2. Spike due to an electrostatic discharge on RESX line does not cause irregular system reset according to the table below.

RESX Pulse	Action
Shorter than 5μs	Reset Rejected
Longer than 10μs	Reset
Between 5μs and 10μs	Reset starts (It depends on voltage and temperature condition.)

Note 3. During the resetting period, the display will be blanked (The display is entering blanking sequence, which maximum time is 120 ms, when Reset Starts in Sleep Out –mode. The display remains the blank state in Sleep In –mode) and then return to Default condition for H/W reset.
Note 4. During Reset Complete Time, data in OTP will be latched to internal register during this period. This loading is done every time when there is H/W reset complete time (tREST) within 5ms after a rising edge of RESX.
Note 5. Spike Rejection also applies during a valid reset pulse as shown below:



Note 6. It is necessary to wait 5msec after releasing RESX before sending commands. Also Sleep Out command cannot be sent for 120msec.

4.2 Reset Timing(TP_EXT_RESET)

The TP_EXT_RESET default is internal pull-high. If no use, please leave it open.

(VDDI=1.65 to 1.95V, VDD=2.7 to 3.6V, AGND=DGND=0V, Ta=-30 to 85℃)

Symbol	Item	Min	Typ	Max	Units	Conditions
tRES	Reset low-level width	1	-	-	ms	
trRES	Reset rise time	-	-	10	us	

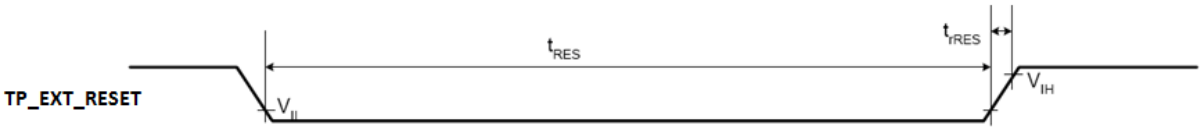


Figure 1 Reset Operation

Note1: TP_EXT_RESET is touch reset signal, if setting TP_EXT_RESET, only touch function will be resetting.

4.3 INT Timing(TP_INT)

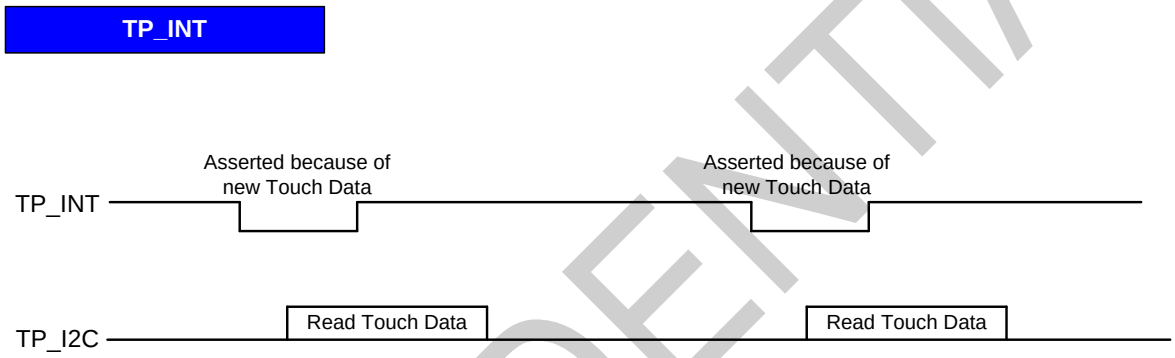
The TP_INT default is push-pull output pin and do not need pull-up resistor.

The touch INT signal is designed to be connected to a host processor's interrupt request and indicates when new touch data is available to be read from the touch I2C interface.

TP_INT are compatible with both level and edge sensitive host interrupt behavior, and it can be either active-high or active-low, depending on the product request.

When TP_INT is asserted, the host most read touch data register to acknowledge the TP_INT.

The contents of touch data register are valid and can be read any time, when the TP_INT asserted.



4.4 I2C Timing Characteristics

The I2C is always configured in the Slave mode. The definition of I2C timing is as following.

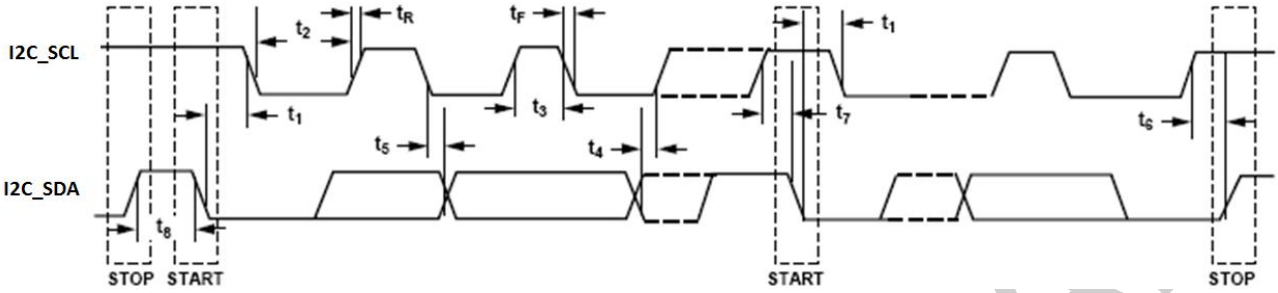


Figure 2 I2C Timing Diagram

I2C AC Timing Specification

(VDDI=1.65 to 1.95V, VDD=2.7 to 3.6V, AGND=DGND=0V, Ta=-30 to 85℃)

Symbol	Item	Min	Typ	Max	Units	Conditions
fSCL	I2C_SCL frequency	-	-	400	KHz	
t1	Start condition hold time, tHD; STA	0.6			us	
t2	Clock low period, tLOW	1.3	-	-	us	
t3	Clock high period, tHIGH	0.6	-	-	us	
t4	Data setup time, tSU; DAT	100	-	-	ns	
t5	Data hold time, tHD; DAT	120	-	-	ns	
t6	Stop condition setup time, tSU; STO	0.6	-	-	us	
t7	Start condition setup time, tSU; STA	0.6	-	-	us	
t8	Bus-free time between stop and start conditions, tBUF	1.3	-	-	us	
tR	Clock/data rise time	-	-	300	ns	
tF	Clock/data fall time	-	-	300	ns	
Cb	Capacitive load for each bus line			400	pF	

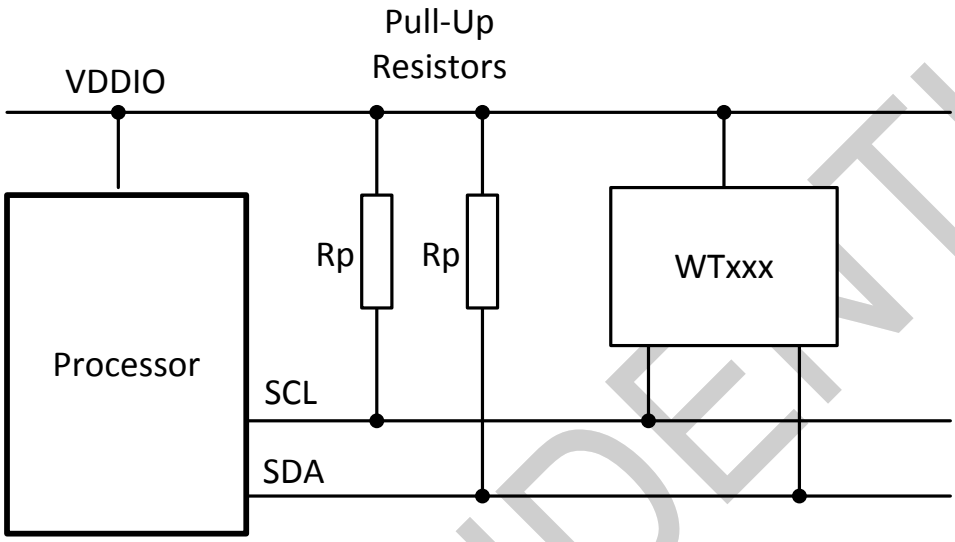
NOTE: All values are referred to VIH and VIL level of VDDI

NOTE: WT030 support I2C interface and the slave address is 39h.

4.5 I2C Pull-Up Resistor

The appropriate value for the pull-up resistor is limited by two factors. The first factor is depend on bus capacitance due to the specified rise time of I2C frequency, the second factor is the minimum sink current of all devices on the bus.

The typical pull-up resistor value is 1.7-10K Ohm for I2C standard mode and fast mode



4.6 I2C Device ID

This chip has three I2C device ID, and the function of each ID is described as following

RID: Use for chip production test in the foundry
EID: Use to switch I2C operation mode, generally used to do firmware update and check IC internal status
NID: General I2C device ID

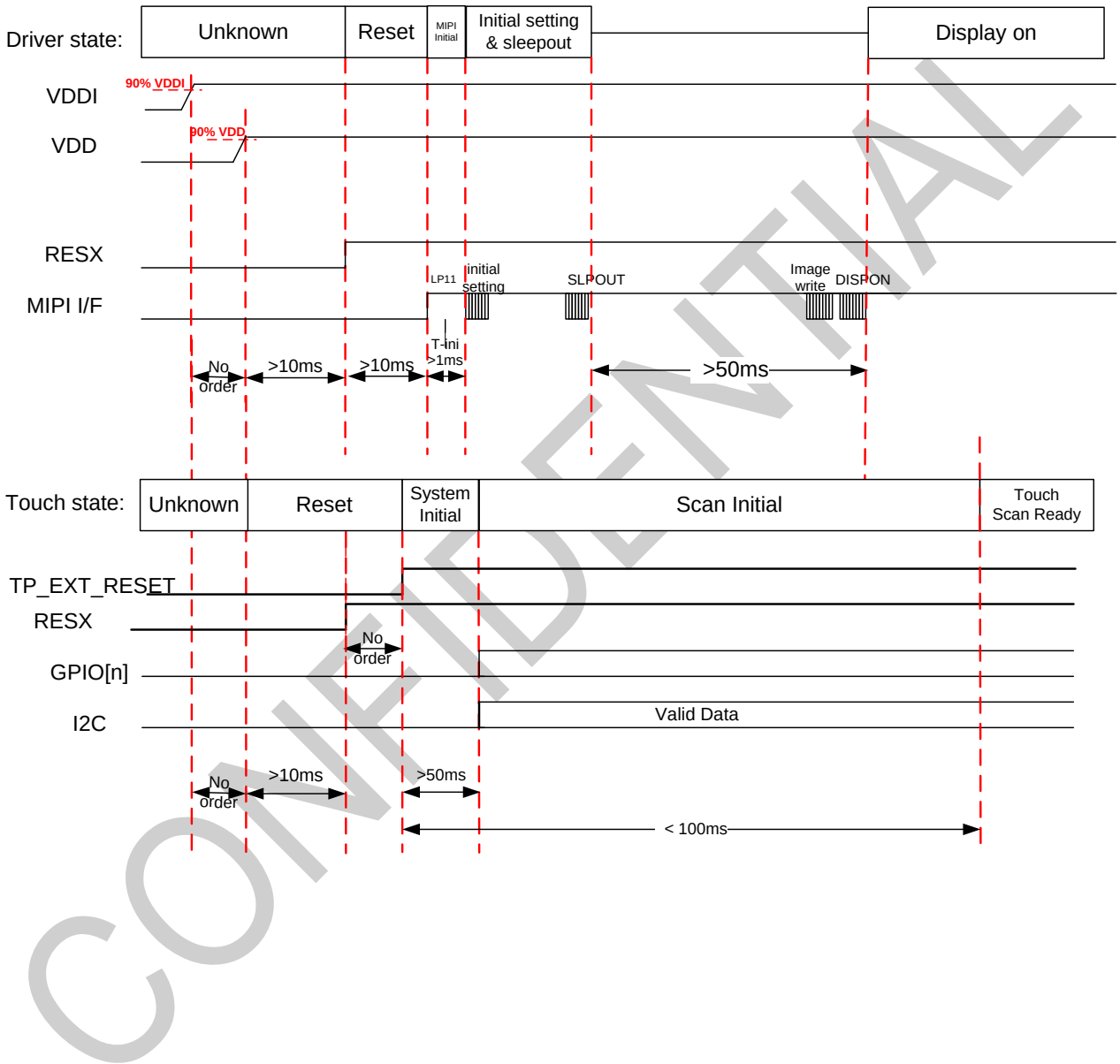
Symbol	Description	Default
I2C_RID	I2C Real Time Debug ID	0x5B
I2C_EID	I2C Engineer ID	0x5A
I2C_NID	I2C Normal ID	0x39

Note: The default value of three I2C device ID are stored in the IC's OTP and can be support one-time reprogram.

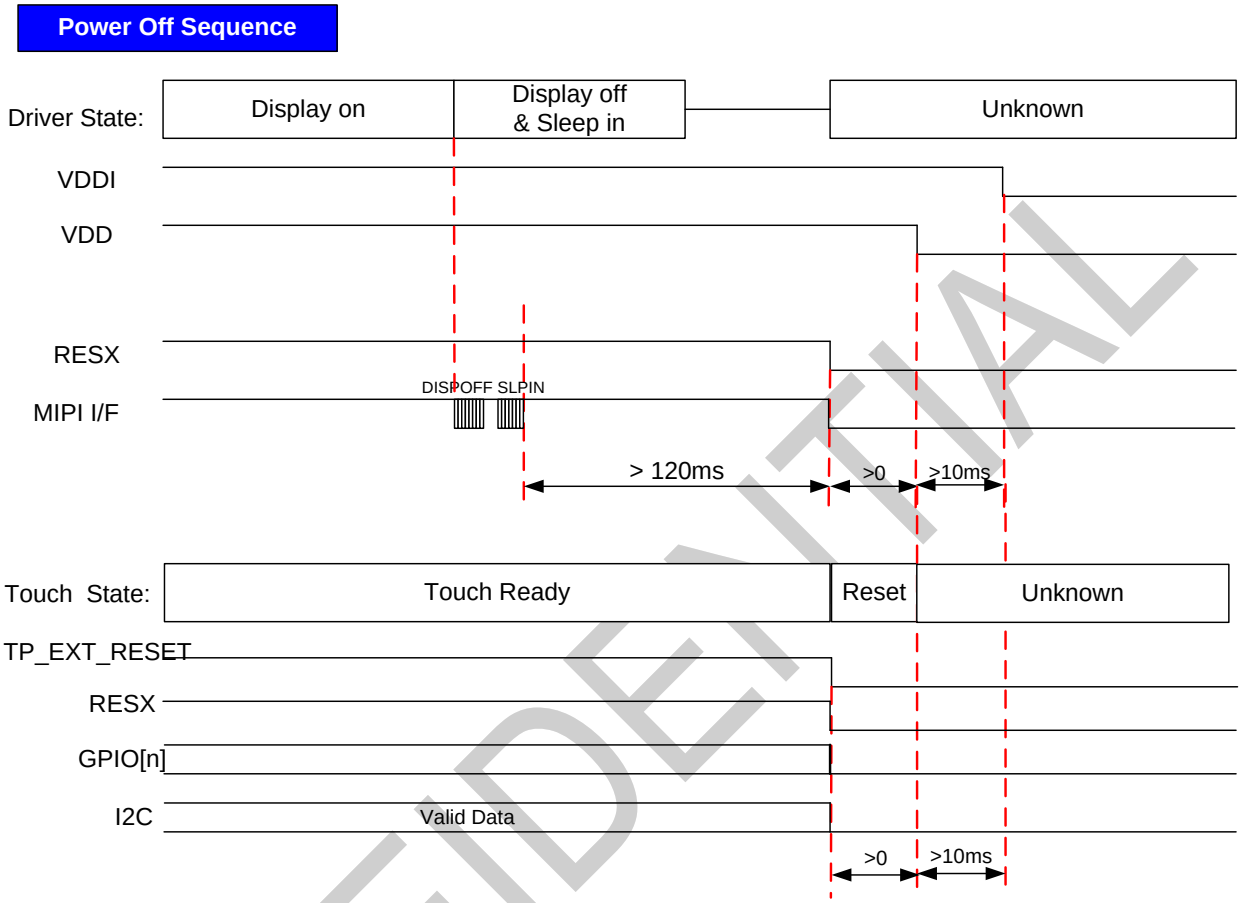
5 Power On/Off/Reset Sequence

5.1 Power on sequence of driver & touch

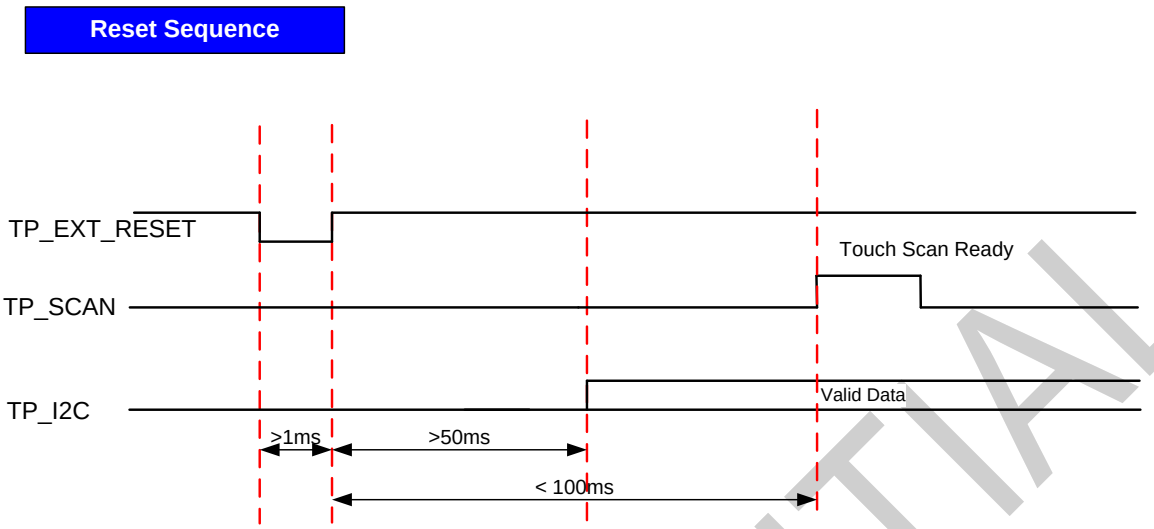
Power On Sequence



5.2 Power off sequence of driver & touch



5.3 Reset sequence of touch



6 External Flash Requirements

WT030 support touch firmware booting with external flash. The touch controller comes out of reset and reads the firmware out of the external flash device, and stores it in internal SRAM and boots the embedded ARM processor to execute the firmware.

External flash that meets criteria and can be used with WT030

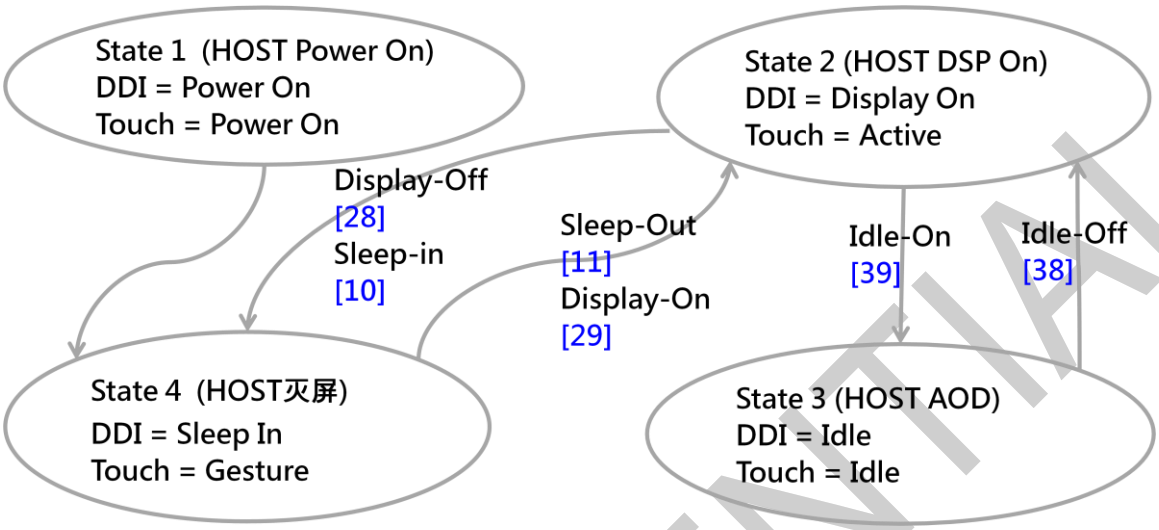
- Standard JEDEC Flash Command
- Supply Voltage: 1.8V
- Erasable sector size: 4kB
- Density: 4Mbits~512Kbits

List of suggested flash devices of the WT030 , these are simulated examples of the WT030 external flash control.

Part Number	Manufacturer	Density	Voltage
W25Q20EW	Winbond	2Mb	1.8V
W25Q10EW	Winbond	1Mb	1.8V
MX25R1035F	MXIC	1Mb	1.8V
MX25R512F	MXIC	512Kb	1.8V
XT25Q01D	XTX	1Mb	1.8V
XT25Q02D	XTX	2Mb	1.8V
GD25LD40	GD	4Mb	1.8V
EN25S40A	ESMT	4Mb	1.8V
EN25S20A (2SF)	ESMT	2Mb	1.8V

7 Touch Power Mode Operation

7.1 Power Mode Behavior between DDI and Touch



Note: If gesture mode is used, keep VCI / VDDIO / TP_EXT_RESET / RESX at high voltage

8 Touch Command

8.1 I2C Command description

8.1.1 Touch status command

Register name	Touch status	
Page	0x00	
Register addr	0x00	
	Read/Write	Description
Byte1	Read Only (RO)	SEQ_NUM
Byte2	Read Only (RO)	TOUCH_POINTS
Byte3	Read Only (RO)	GESTURE_STATE
Byte4	Read Only (RO)	RESERVED

8.1.1.1 Gesture state

Gesture type	Define
GES_PALM	0x01
GES_WAKEUP	0x02

8.1.2 Touch report command

Register name	Touch report	
Page	0x00	
Register addr	0x01	
	Read/Write	Description
Byte1	Read Only (RO)	TOUCH1_ID
Byte2	Read Only (RO)	TOUCH1_XL_POS
Byte3	Read Only (RO)	TOUCH1_XH_POS
Byte4	Read Only (RO)	TOUCH1_YL_POS
Byte5	Read Only (RO)	TOUCH1_YH_POS
Byte6	Read Only (RO)	TOUCH1_ZL_W
Byte7	Read Only (RO)	TOUCH1_ZH_W
Byte8	Read Only (RO)	TOUCH1_XL_SHAPE
Byte9	Read Only (RO)	TOUCH1_XH_SHAPE
Byte10	Read Only (RO)	TOUCH1_YL_SHAPE
Byte11	Read Only (RO)	TOUCH1_YH_SHAPE
Byte12	Read Only (RO)	TOUCH2_ID
Byte13	Read Only (RO)	TOUCH2_XL_POS
Byte14	Read Only (RO)	TOUCH2_XH_POS
Byte15	Read Only (RO)	TOUCH2_YL_POS
Byte16	Read Only (RO)	TOUCH2_YH_POS
Byte17	Read Only (RO)	TOUCH2_ZL_W
Byte18	Read Only (RO)	TOUCH2_ZH_W
Byte19	Read Only (RO)	TOUCH2_XL_SHAPE
Byte20	Read Only (RO)	TOUCH2_XH_SHAPE
Byte21	Read Only (RO)	TOUCH2_YL_SHAPE
Byte22	Read Only (RO)	TOUCH2_YH_SHAPE

8.1.3 Touch host command

8.1.3.1 Touch

Register name	Host command	
Page	0x00	
Register addr	0x02	
	Read/Write	Description
Byte1	Read/Write (RW)	Command 0x30 : Enter touch sleep mode

8.1.3.2 Touch manual mode (*need FW support)

Register name	Host command	
Page	0x00	
Register addr	0x02	
	Read/Write	Description
Byte1	Write (W)	Command 0x34 : Enter TP manual mode
Byte2	Reserved	
Byte3	Write (W)	Command 0x00 : TP normal mode 0x01 : TP speed up mode (90Hz) 0x02 : TP speed down mode (15Hz)
Byte4	Reserved	

Note: Only for power measurement and does not support touch point reporting

8.1.3.3 Palm area

Register name	Gesture status	
Page	0x00	
Register addr	0x03	
	Read/Write	Description
Byte1	Read/Write (RW)	Palm threshold

8.1.3.4 TP status

Register name	TP status	
Page	0x00	
Register addr	0x05	
	Read/Write	Description
Byte1	Read Only (RO)	Touch mode

	Description
Touch mode[7]	Touch mode state 1 : Active ; 0 : Idle

8.1.4 Others command

8.1.4.1 FW version

Register name	FW version	
Page	0x00	
Register addr	0x06	
	Read/Write	Description
Byte1	Read (RO)	FW version [0]
Byte2	Read (RO)	FW version [1]
Byte3	Read (RO)	FW version [2]
Byte4	Read (RO)	FW version [3]

8.1.4.2 Panel version

Register name	Panel version	
Page	0x00	
Register addr	0x07	
	Read/Write	Description
Byte1	Read (RO)	Panel version [0]
Byte2	Read (RO)	Panel version [1]
Byte3	Read (RO)	Panel version [2]
Byte4	Read (RO)	Panel version [3]
Byte5	Read (RO)	Panel version [4]
Byte6	Read (RO)	Panel version [5]

8.2 System Command Introduction

8.2.1 System command description

8.2.1.1 Check I2C ready

shell@watch: $ cat raydium_check_i2c

8.2.1.2 Enable Touch lock

shell@watch: $ echo 1 > raydium_i2c_touch_lock

8.2.1.3 Disable Touch lock

shell@watch: $ echo 0 > raydium_i2c_touch_lock

8.2.1.4 Check FW version

shell@watch: $ cat raydium_check_fw_version

8.2.1.5 Check panel version

shell@watch: $ cat raydium_check_panel_version

8.2.1.6 Check driver version

shell@watch: $ cat raydium_driver_version

8.2.1.7 Do HW reset

shell@watch: $ cat raydium_hw_reset

8.3 I2C Packet Format

8.3.1 I2C Write Format

S	I2C_NID	W	A	PAGE CMD	A	PAGE ADDR	A	P

S	I2C_NID	W	A	REG ADDR	A	Data Byte	A	Data Byte	A	Data Byte	A	P

Ex: clear sequence number

S	0x39	W	A	0x0A	A	0x00	A	P

S	0x39	W	A	0x00	A	0x00	A	P

8.3.2 I2C Read Format

S	I2C_NID	W	A	PAGE CMD	A	PAGE ADDR	A	P

S	I2C_NID	W	A	REG ADDR	A	RS	I2C_NID	R	A	Data Byte	A	Data Byte	A	P

Ex: Read FW version

S	0x39	W	A	0x0A	A	0x00	A	P

S	0x39	W	A	0x06	A	RS	0x39	R	A	Version[0]	A	Version[1]	A	Version[2]	A	Version[3]	A	P

Note: If the I2C command is in the same page and system already do the set page command first, you could skip the set page command in the read/write command flow.

9 Touch firmware upgrade

9.1 Touch firmware upgrade description

The chapter descripts how to burn touch firmware by script.

1. Please implement the following code for sysfs.

Ex.

raydium_i2c_pda_access
raydium_i2c_pda2_mode
raydium_receive_fw_control
raydium_fw_upgrade
raydium_hw_reset
raydium_check_fw_version

After the code have been implemented, please check device file node whether exist.
(Note. “1-0039” may be differences in your system)

Ex.

/sys/bus/i2c/devices/1-0039/ raydium_i2c_pda_access
/sys/bus/i2c/devices/1-0039/ raydium_i2c_pda2_mode
/sys/bus/i2c/devices/1-0039/ raydium_ receive_fw_control
/sys/bus/i2c/devices/1-0039/ raydium_ fw_upgrade
/sys/bus/i2c/devices/1-0039/ raydium_hw_reset
/sys/bus/i2c/devices/1-0039/ raydium_check_fw_version

2. Push touch firmware to the path(/data/selftest/fw)

$ ls

bootloader.bin fw.bin fw.para Init_code.bin
$ adb push * /data/selftest/fw

3. Push raydium_utility tool to the path(/data/selftest/)

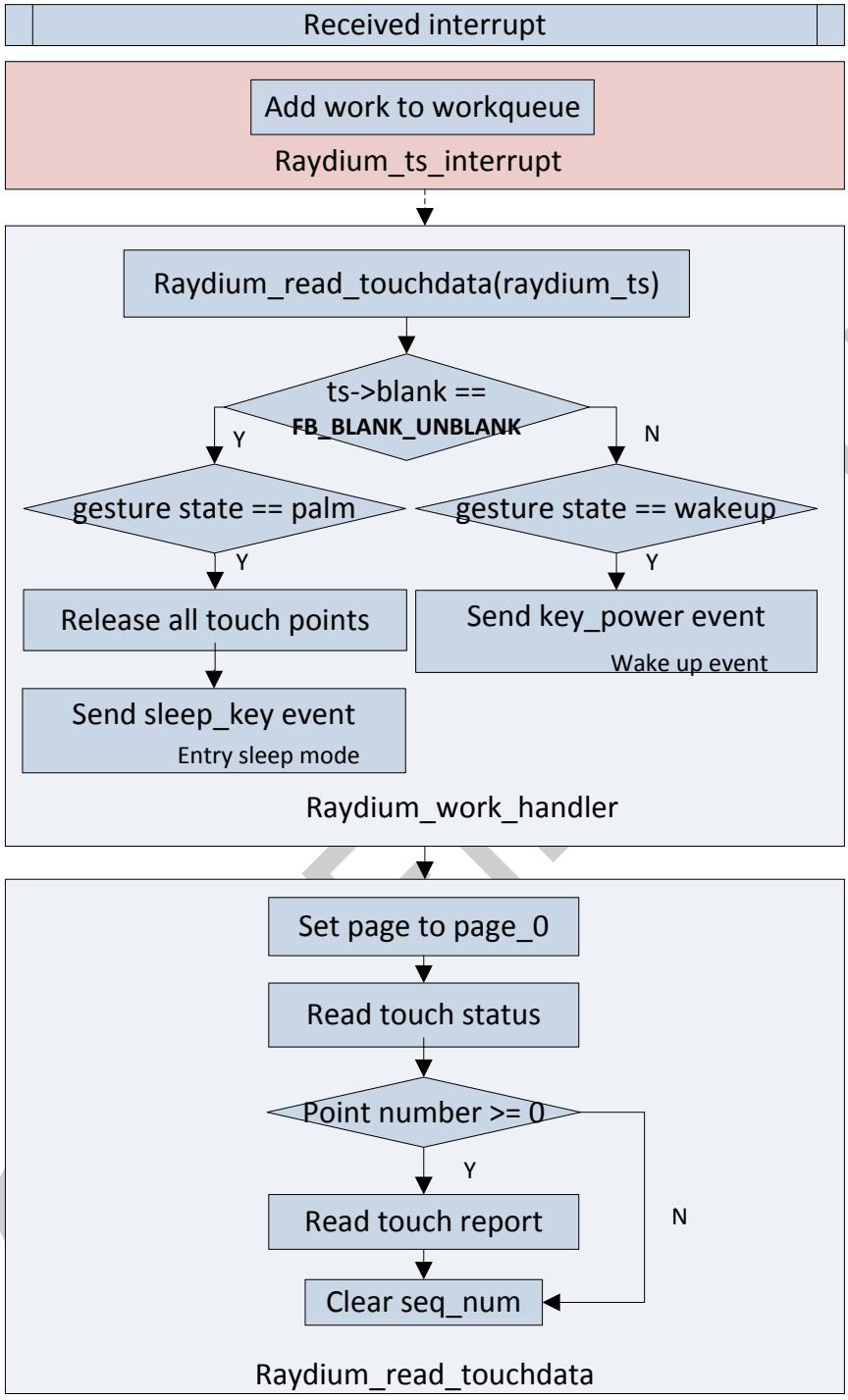
$ adb push raydium_utility /data/selftest/

4. Start to burn touch firmware by raydium_utility

$ adb shell “./data/selftest/raydium_utility”

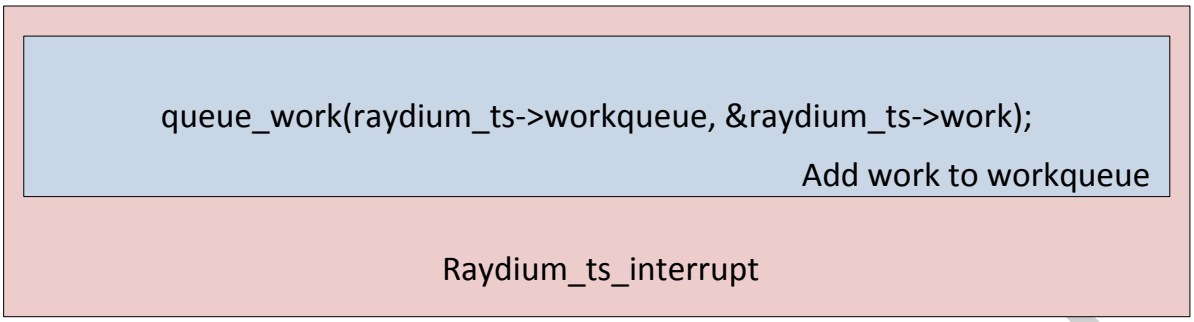
10 Touch report Sample code

10.1 Touch report flow

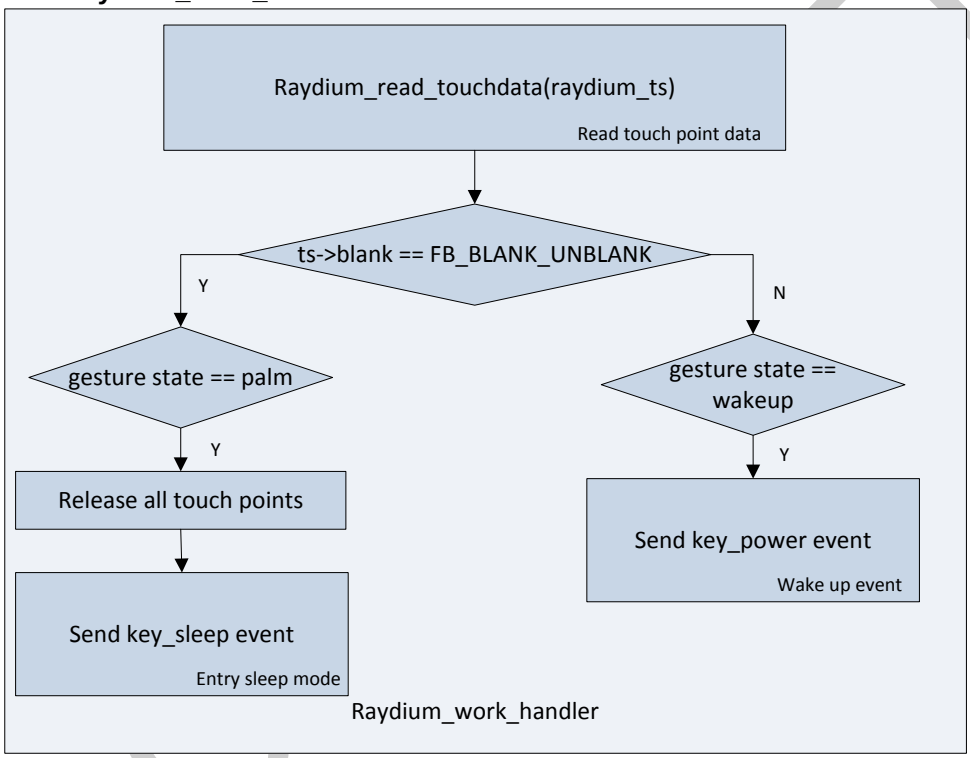


Note: The set page function can set base address of IC and clear interrupt pin to initial state.

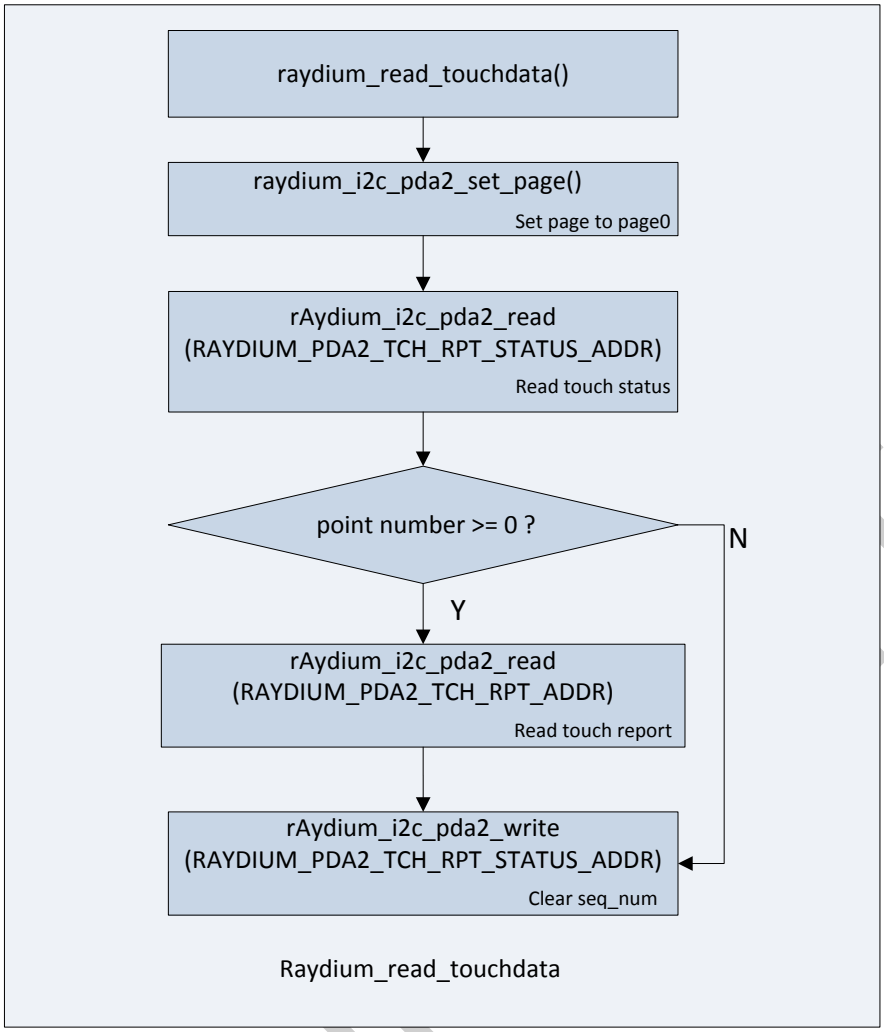
10.2 Raydium_ts_interrupt



10.3 Raydium_work_handler

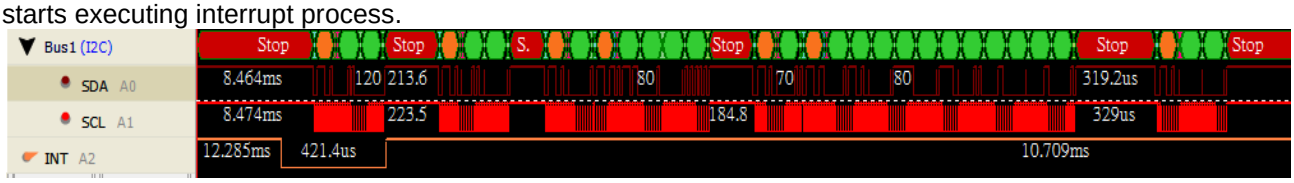


10.4 Raydium_read_touchdata



10.5 Example for touch INT/SDA/SCL waveform

The following picture describes touch waveform. When touch INT notify the host device. The host device starts executing interrupt process.



1. First, host send 2 bytes packages to set page0 and notify chip that the interrupt occurs in host. (In driver code, please refer read_touchdata)

Start	Address	Write	A-ACK	Data	D-ACK	Data	D-ACK	Stop
Start	39	Write	A-ACK	0A	D-ACK	00	D-ACK	Stop

2. Host sent package and receive 4 bytes packages for touch status on TCH_PRT_STATUS_ADDR. (In driver code, please refer read_touchdata)

Start	Address	Write	A-ACK	Data	D-ACK
Start	39	Write	A-ACK	00	D-ACK

Start	Address	Read	A-ACK	Data	D-ACK	Data	D-ACK	Data	D-ACK	Data	D-NACK	Stop
Start	39	Read	A-ACK	BB	D-ACK	01	D-ACK	00	D-ACK	AA	D-NACK	Stop

3. Host start receives 11 bytes packages on TCH_PRT_ADDR. (In driver code, please refer read_touchdata)

Start	Address	Write	A-ACK	Data	D-ACK
Start	39	Write	A-ACK	01	D-ACK

Start	Address	Read	A-ACK	Data	D-ACK	Data	D-ACK	Data	D-ACK	Data	D-ACK	Data	D-ACK	Data	D-ACK	Data	D-ACK	Data
Start	39	Read	A-ACK	01	D-ACK	E4	D-ACK	00	D-ACK	FF	D-ACK	00	D-ACK	FC	D-ACK	05	D-ACK	03

D-ACK	Data	D-ACK	Data	D-ACK	Data	D-NACK	Stop
D-ACK	00	D-ACK	03	D-ACK	00	D-NACK	Stop

4. Finally, host need clear seq num on TCH_PRT_STATUS address.

Start	Address	Write	A-ACK	Data	D-ACK	Data	D-ACK	Stop
Start	39	Write	A-ACK	00	D-ACK	00	D-ACK	Stop

11 Touch selftest

11.1 Raydium selftest description

The raydium_selftest is a bin file which is used to check the touch function in the system. It could check the Hardware function, like I2C and INT pin, and the touch test, like open, short and uniformity, etc.

11.2 Steps for usage

Step1: create the file folder
shell@linux: $ adb shell mkdir /data/selftest

Step2: push the raydium_selftest to the folder
shell@linux: $ adb push raydium_selftest /data/selftest/

Step3: execute the raydium_selftest to the folder
shell@linux: $ adb shell /data/selftest/raydium_selftest

Step4: get the result log
shell@linux: $ adb pull /sdcard/selftest.csv

11.3 Touch driver implement

Please implement the following code for sysfs.
Ex.
raydium_i2c_pda2_mode
raydium_int_flag
raydium_i2c_pda_access
raydium_touch_calibration
raydium_fw_upgrade
raydium_receive_fw_control
raydium_reset_control

After the code have been implemented, please check device file node whether exist.
(Note. “1-0039” may be differences in your system)
Ex.
/sys/bus/i2c/devices/1-0039/ raydium_i2c_pda2_mode
/sys/bus/i2c/devices/1-0039/ raydium_int_flag
/sys/bus/i2c/devices/1-0039/ raydium_i2c_pda_access
/sys/bus/i2c/devices/1-0039/ raydium_touch_calibration
/sys/bus/i2c/devices/1-0039/ raydium_fw_upgrade
/sys/bus/i2c/devices/1-0039/ raydium_receive_fw_control
/sys/bus/i2c/devices/1-0039/ raydium_reset_control

12 Touch data record

12.1 Raydium Logger data script description

This chapter describes how to record the data by script in linux system.

12.2 Steps for usage

Step1: create the file folder
shell@linux: $ adb shell mkdir /data/selftest

Step2: execute the script and input the parameter.
shell@linux: $ ./Rad_Logger_DeltaRawCount $1 $2 $3
ex: ./Rad_Logger_DeltaRawCount 6 10 20

$1: size. The channel dimension of your panel, and you could get the number from FW parameter.
$2: type. Reference Chapter 12.4
$3: count. The number of frame count you want to record.

Step3: open the result log to check result
After executing the script, there would be a Rad_Log_Delta.csv file. You could using the excel to check the data.

12.3 Touch driver implement

Please implement the following code for sysfs.
Ex.
raydium_i2c_raw_data
After the code have been implemented, please check device file node whether exist.
(Note. “1-0039” may be differences in your system)
Ex.
/sys/bus/i2c/devices/1-0039/raydium_i2c_raw_data

12.4 Touch data type table

Data	Type
Raw Data	0x08
Strength Data	0x10
Baseline Data	0x02
Compensation Cap Data	0x04