RA6M2 Group

Capacitive Touch Low Power Guide

Introduction

This application note explains the electrostatic Capacity Touch measurement that uses the Asynchronous General-Purpose Timer (AGT) function and Low power mode (Deep Software standby mode) installed in RA6M2.

Target Device

RA6M2 Group

Contents

1. Specification

1.1 Project description

Sample code that this application note describes is confirmed to operate on RA6M2 Group Capacitive Touch Evaluation System (RTK0EG0021S01001BJ). The setting of this project is adjusted to R7FA6M2AF3CFB implemented on RA6M2 Group Capacitive Touch Evaluation System. Modify the device setting in the project when you use the other device.

1.2 Used Peripherals

Table 1.1 lists the used peripherals in the sample code.

Table 1.1 Peripherals in the sample code

Used Peripherals	Functions
Capacitive Touch Sensing Unit (CTSU)	- Measures electrostatic capacitance of the touch sensor.
Data Transfer Controller (DTC)	- Transfers CTSUSSC, CTSUSO0 and CTSUSO1 setting value in RAM to CTSU register. - Transfers CTSUSC and CTSURC counter in CTSU register to RAM.
Asynchronous General-Purpose Timer (AGT)	Timer used to cancel Deep Software Standby mode.
Event Link Controller (ELC)	- Starts CTSU measurement by Snooze entry.

1.3 CPU Operation Mode

Table 1.2 lists CPU mode used in this sample code.

Table 1.2 CPU operation mode

CPU operation mode	Transition condition
Normal mode (MCU startup and Touch measurement Start process)	Transition from Reset state after 100ms timer has elapsed in Deep Software Standby mode
Software Standby mode	Transition by API of r_lpm driver.
Snooze mode	Transition from AGT counter underflow in Software Standby mode
Normal mode (Touch measurement End process and Touch On/Off detection process)	Transition from CTSU measurement end interrupt in Snooze mode
Deep Software Standby mode	Transition by API of r_lpm driver.

1.4 CTSU Operation Status

Table 1.3 lists CTSU operation status used in this sample code.

Table 1.3 CTSU operation status

CTSU Operation Status	Transition condition
Stop	Reset status
Start (Suspended)	Enable power-saving function during wait state
Start	Start touch measurement
Stop	Stop CTSU module
Stop (Undefined)	Deep Software Standby mode

1.5 Register settings

Register settings changed from the reset value are shown below.

Table 1.4 Register settings

Function	Register	Value	Remarks
I/O Ports	P000PFS	0x00000000	
	P001PFS	0x00000000	
	P002PFS	0x00000000	
	P003PFS	0x00000000	
	P004PFS	0x00000000	
	P005PFS	0x00000000	
	P006PFS	0x00000000	
	P007PFS	0x00000000	
	P108PFS	0x00000000	
	P109PFS	0x00000000	
	P110PFS	0x00000000	
	P201PFS	0x00000000	
	P300PFS	0x00000000	
Battery Backup Function	VBTICTLR	0x00	
Realtime Clock (RTC)	RCR2	0x00	Set according to 26.6.7 “Initialization Procedure When RTC Is Not to Be Used” in RA6M2 Group User’s Manual: Hardware.
	RCR4	0x01	RCKSEL bit (Count Source Select in normal operation mode): 1 LOCO is selected
Low Power Modes	SBYCR	0x8000	The settings for each bit are as follows. OPE bit (Output Port Enable): 0 Set the address bus and bus control signals to the high-impedance state SSBY bit (Software Standby): 1 Deep Software Standby mode

	SNZCR	0x82	The settings for each bit are as follows. (In case of Snooze mode) RXDREQEN bit (RXD0 Snooze Request Enable) : 0 Ignore RXD0 falling edge in Software Standby mode SNZDTCEN bit (DTC Enable in Snooze mode) : 1 Enable DTC operation SNZE (Snooze mode Enable) : 1 Enable Snooze mode
	DPSBYCR	0x80	The settings for each bit are as follows. DEEPCUT[1:0] bits (Power-Supply Control): 0 Supply power to the Standby SRAM, low-speed on-chip oscillator, AGTn, and USBFS resume detecting unit in Deep Software Standby mode IOKEEP bit (I/O Port Retention): 0 When Deep Software Standby mode is canceled, clear the I/O ports to the reset state DPSBY bit (Deep Software Standby): 1 Deep Software Standby mode
Clock Generation Circuit	SOSCCR	0x01	SOSTP bit (Sub-Clock Oscillator Stop): 1 Stop the sub-clock oscillator

1.6 File Configuration

Table 1.5 lists the file added or changed in the sample code generated by RA configurator and QE for Capacitive Touch.

Table 1.5 Files added or changed in the sample code

Name	Outline	Remarks
qe_touch_sample.c	Main processing	Changed file
r_captouch_lpm.c	Standby RAM processing	Added file
r_captouch_lpm.h	Definition of Standby RAM processing function	Added file
r_user_warm_start.c	Reset source and Deep Software Standby cancel source detection processing	Added file
r_user_warm_start.h	Definition of reset source and Deep Software Standby cancel source detection processing	Added file

2. Operation Conditions

This application note confirms operation based on the items and conditions stated below.

Table 2.1 Operation Conditions

Item	Description
MCU	R7FA6M2AF3CFB (RA6M2 Group)
Operating frequency	20MHz Middle-speed on-chip oscillator (HOCO) 32kHz Low-speed on-chip oscillator (LOCO)
Operating voltage	5.0V
Target board	RA6M2 Group Capacitive Touch Evaluation System (RTK0EG0021S01001BJ)
Integrated development environment	e^2 studio (2021-10)
C compiler	GCC Arm Embedded (9.3.1.20200408)
Operation mode	Single chip mode
Debugger	E2 Emulator Lite
FSP Version	Ver.3.4.0
Sample code Version	Ver.1.00

3. Software Description

The sample code operates as follows by using the FSP driver and middleware functions.

1. After reset release by power on, the sample code opens rm_touch middleware and executes offset tuning and initial calibration to be able to detect Touch On/Off.
2. The sample code saves information about rm_touch middleware to Standby RAM.
3. Transits to Deep Software Standby mode.
4. After reset release from Deep Software Standby, the sample code loads the information about rm_touch middleware from Standby RAM.
5. Transits to Software Standby mode.
6. Transits to Snooze mode by AGT counter underflow and start CTSU measurement by Snooze entry.
7. Save result of Touch On/Off detection.
8. Repeat step 2 from step 7 and the sample code flashes user LED when Touch On is detected five times in a row.

3.1 Operation image

Figure 3-1 shows CPU operation mode and CTSU operation status according to the process in the sample code.

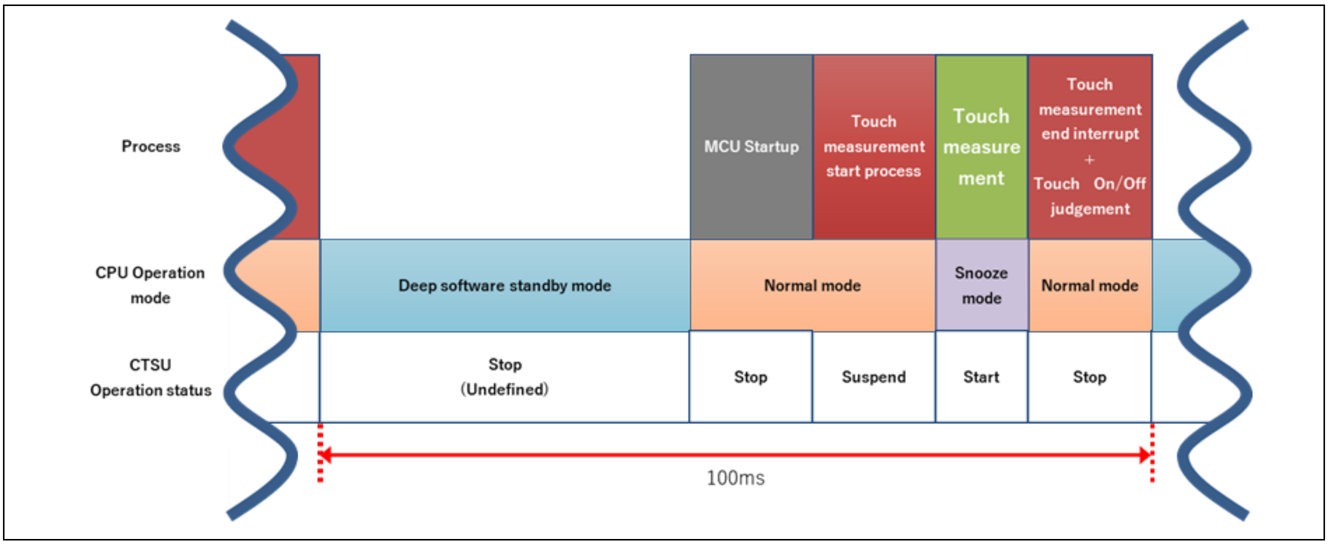


Figure 3-1 Operation image

3.2 FSP driver and middleware

Table 3.1 lists FSP driver and middleware used in the sample code.

Table 3.1 FSP driver and middleware used in the sample code

FSP driver, middleware	Version
Board Support Package (BSP)	3.4.0
CTSU Driver (r_ctsu)	3.4.0
Capacitive Touch Middleware (rm_touch)	3.4.0
I/O Port Driver (r_ioport)	3.4.0
Timer Driver (r_agt)	3.4.0
Low Power Mode Driver (r_lpm)	3.4.0
Event Link Controller (r_elc)	3.4.0
DTC Driver (r_dtc)	3.4.0

3.2.1 Standby SRAM Notes

To use Standby SRAM, add definition of Standby SRAM section to section information of linker script file according to the SRAM address and SRAM capacity of the Standby RAM described in the user’s manual.

In the sample code, “script\fsp.ld” the linker script file.

3.3 List of Constants

The modifiable constant is show in Table 3.2.

Table 3.2 The modifiable constant (qe_touch_sample.c)

Constants	Initial value	Setting
DETECTED_PRESSES	5U	Number of Touch On detection
TOUCH_BUTTON_00	1U	Bit position to check Touch On/Off of Touch Button
NO_DETECTED_PRESS	0U	Non-detection of Touch On
LED_LOOP_MAX	10U	Number of LED processing at Touch On detection
LED_CYCLE_MS	50U	LED lighting cycle at Touch On detection
RTC_STABLIZATION_TIME_US	190U	Time to wait for RTC clock to stabilize
DEEP_STBY_TIME	0xcc0	Time in deep software standby mode

Table 3.3 the modifiable constant (r_user_warm_start.h)

Constants	Initial value	Setting
IWDT_RESET_ENABLED	0	0: Disable Independent watchdog timer reset 1: Enable Independent watchdog timer reset
WDT_RESET_ENABLED	0	0: Disable Watchdog timer reset 1: Enable Watchdog timer reset
VOLTAGE_MONITOR_0_RESET_ENABLED	0	0: Disable Voltage monitor 0 reset 1: Enable Voltage monitor 0 reset
VOLTAGE_MONITOR_1_RESET_ENABLED	0	0: Disable Voltage monitor 1 reset 1: Enable Voltage monitor 1 reset
VOLTAGE_MONITOR_2_RESET_ENABLED	0	0: Disable Voltage monitor 2 reset 1: Enable Voltage monitor 2 reset
SRAM_PARITY_RESET_ENABLED	0	0: Disable SRAM parity error reset 1: Enable SRAM parity error reset
SRAM_ECC_RESET_ENABLED	0	0: Disable SRAM ECC error reset 1: Enable SRAM ECC error reset
BUS_MPU_SLAVE_RESET_ENABLED	0	0: Disable Bus slave MPU error reset 1: Enable Bus slave MPU error reset
BUS_MPU_MASTER_RESET_ENABLED	0	0: Disable Bus master MPU error reset 1: Enable Bus master MPU error reset
CPU_SP_RESET_ENABLED	0	0: Disable Stack pointer error reset 1: Enable Stack pointer error reset
CPU_DEEP_SW_STBY_RESET_ENABLED	1	0: Disable Deep Software Standby reset 1: Enable Deep Software Standby reset
CPU_SOFTWARE_RESET_ENABLED	1	0: Disable Software reset 1: Enable Software reset

3.4 List of Structures

Structures added in this sample code are shown below.

3.4.1 Reset source (r_user_warm_start.h)

```
/* Reset source */
typedef struct st_reset_source
{
    union
    {
        volatile uint16_t all;

        struct
        {
            volatile uint16_t power_on_reset_detect_flag : 1; /* Power-On
reset */
            volatile uint16_t voltage_monitor_0_reset_detect_flag : 1; /*
Voltage monitor 0 reset */
            volatile uint16_t voltage_monitor_1_reset_detect_flag : 1; /*
Voltage monitor 1 reset */
            volatile uint16_t voltage_monitor_2_reset_detect_flag : 1; /*
Voltage monitor 2 reset */
            volatile uint16_t deep_software_standby_reset_flag : 1; /* Deep
Software Standby reset */
            volatile uint16_t independent_watchdog_timer_reset_detect_flag :
1; /* IWDT underflow/refresh error reset */
            volatile uint16_t watchdog_timer_reset_detect_flag : 1; /* WDT
underflow/refresh error reset */
            volatile uint16_t software_reset_detect_flag : 1; /* Software
reset */
            volatile uint16_t sram_parity_error_reset_detect_flag : 1; /* SRAM
parity error reset */
            volatile uint16_t sram_ecc_error_reset_detect_flag : 1; /* SRAM
ECC error reset */
            volatile uint16_t bus_slave_mpu_error_reset_detect_flag : 1; /*
MPU bus slave error reset */
            volatile uint16_t bus_master_mpu_error_reset_detect_flag : 1; /*
MPU bus master error reset */
            volatile uint16_t sp_error_reset_detect_flag : 1; /* CPU stack
pointer monitor reset */
            volatile uint16_t cold_warm_start_determination_flag : 1; /*
Cold/Warm Start detection */
            volatile uint16_t : 2;
        } flag;
    };
} reset_source_t;
```

3.4.2 Deep Software Standby mode cancel source (r_user_warm_start.h)

```
/* Deep Software Standby cancel source */
typedef struct st_dp_sw_stby_source
{
    union
    {
        volatile uint32_t all;

        struct
        {
            volatile uint32_t irq0_ds : 1;    /* IRQ0 */
            volatile uint32_t irq1_ds : 1;    /* IRQ1 */
            volatile uint32_t irq2_ds : 1;    /* IRQ2 */
            volatile uint32_t irq3_ds : 1;    /* IRQ3 */
            volatile uint32_t irq4_ds : 1;    /* IRQ4 */
            volatile uint32_t irq5_ds : 1;    /* IRQ5 */
            volatile uint32_t irq6_ds : 1;    /* IRQ6 */
            volatile uint32_t irq7_ds : 1;    /* IRQ7 */
            volatile uint32_t irq8_ds : 1;    /* IRQ8 */
            volatile uint32_t irq9_ds : 1;    /* IRQ9 */
            volatile uint32_t irq10_ds : 1;   /* IRQ10 */
            volatile uint32_t irq11_ds : 1;   /* IRQ11 */
            volatile uint32_t irq12_ds : 1;   /* IRQ12 */
            volatile uint32_t irq13_ds : 1;   /* IRQ13 */
            volatile uint32_t irq14_ds : 1;   /* IRQ14 */
            volatile uint32_t irq15_ds : 1;   /* IRQ15 */
            volatile uint32_t lvd1_ds : 1;    /* LVD1 */
            volatile uint32_t lvd2_ds : 1;    /* LVD2 */
            volatile uint32_t rtcint_ds : 1;  /* RTC Interval */
            volatile uint32_t rtcalm_ds : 1;  /* RTC Alarm */
            volatile uint32_t nmi_ds : 1;     /* NMI */
            volatile uint32_t usbf_ds : 1;    /* USBFS */
            volatile uint32_t usbh_ds : 1;    /* USBHS */
            volatile uint32_t agt1_ds : 2;    /* AGT1 */
        } flag;
    };
} dp_sw_stby_source_t;
```

3.5 List of Variables

Variables added and changed in this sample code are shown below.

Table 3.4 Variables (qe_touch_sample.c)

Attribute	Variable name	Description
static uint8_t	g_button_status[DETECTED_PRESSES]	Store result of Touch On/Off detection in last 5 times.
static uint32_t	g_button_status_index	Index of buffer to store the result of Touch On/Off detection
static uint64_t	button_status	Result of Touch On/Off detection

Table 3.5 Variables (r_user_warm_start.h)

Attribute	Variable name	Description
volatile reset_source_t	g_reset_source	Reset source
volatile dp_sw_stby_source_t	g_dp_sw_stby_source	Deep Software Standby cancel source

Table 3.6 Variables (r_captouch_lpm.c)

Attribute	Variable name	Description
static uint16_t	g_power_on_reset_symbol	Power-on reset detection flag
static uint16_t	g_touch_button_threshold_ssram[TOUCH_CFG_NUM_BUTTONS]	Threshold for Touch Button
static uint16_t	g_touch_button_hysteresis_ssram[TOUCH_CFG_NUM_BUTTONS]	Hysteresis for Touch Button
static uint16_t	g_touch_button_reference_ssram[TOUCH_CFG_NUM_BUTTONS]	Reference Value for Touch Button
static uint16_t	g_touch_button_on_count_ssram[TOUCH_CFG_NUM_BUTTONS]	Touch On counter for Touch Button
static uint16_t	g_touch_button_off_count_ssram[TOUCH_CFG_NUM_BUTTONS]	Touch Off counter for Touch Button
static uint32_t	g_touch_button_drift_buf_ssram[TOUCH_CFG_NUM_BUTTONS]	Sum of Count Value for Touch Button until Drift Correction Execution
static uint16_t	g_touch_button_drift_count_ssram[TOUCH_CFG_NUM_BUTTONS]	Drift Correction Interval for Touch Button
static uint8_t	g_ctsu_tuning_complete_ssram[CTSU_CFG_NUM_SELF_ELEMENTS + CTSU_CFG_NUM_MUTUAL_ELEMENTS]	Result of Initial Calibration

static int32_t	g_ctsu_tuning_diff_ssram[CTSU_CFG_NUM_SELF_ELEMENTS + CTSU_CFG_NUM_MUTUAL_ELEMENTS]	Difference between Count Value and ideal Count Value in Initial Calibration
static ctsu_ctsuwr_t	g_ctsu_ctsuwr_ssram[(CTSU_CFG_NUM_SELF_ELEMENTS + CTSU_CFG_NUM_MUTUAL_ELEMENTS) * CTSU_MULTI_NUM]	Setting value of CTSUSSC, CTSUSO0, CTSUSO1 register for Touch Button
static ctsu_self_buf_t	g_ctsu_self_raw_ssram[CTSU_CFG_NUM_SELF_ELEMENTS * CTSU_MULTI_NUM]	Value of CTSUSC and CTSURC register for Touch Button
static uint16_t	g_ctsu_self_data_ssram[CTSU_CFG_NUM_SELF_ELEMENTS]	Count Value for Touch Button
static ctsu_instance_ctrl_t	g_qe_ctsu_ctrl_config_ssram	Configuration of CTSU driver control
static touch_instance_ctrl_t	g_qe_touch_ctrl_config_ssram	Configuration of Touch middleware control
static transfer_info_t	gp_dtc_vector_table_ssram[CTSU_VECTOR_TABLE_ENTRIES]	Vector table for DTC
static uint32_t	dtc_vector_table_address_ssram	Address of Vector table for DTC
static dtc_instance_ctrl_t	g_transfer0_ctrl_ssram	Configuration of DTC for CTSUWR interrupt
static dtc_instance_ctrl_t	g_transfer1_ctrl_ssram	Configuration of DTC for CTSURD interrupt

- Variables in r_captouch_lpm.c are assigned to Standby RAM address.

3.6 List of Functions

The specification for functions added in this sample code are shown below.

3.6.1 qe_touch_main ()

qe_touch_main ()		
Outline	Main processing	
Declaration	void qe_touch_main (void)	
Description	Controls Touch measurement and transition to Deep Software Standby mode. Processes the followings according to reset source. - Power-on reset, Software reset by debugger, Reset pin 1. Open Touch middleware 2. Initial calibration for Touch Button 3. Save data for Touch middleware to Standby SRAM 4. Transition to Deep Software Standby mode (cancel after 100ms) - Others 1. Load data for Touch middleware in Standby SRAM to RAM 2. Prepare Touch measurement 3. Start Touch measurement 4. Blink user LED with detection Touch-On of Touch Button in 5 times continuously 5. Save data for Touch middleware to Standby SRAM 6. Transition to Deep Software Standby mode (cancel after 100ms)	
Argument	-	-
Return value	-	

3.6.2 r_captouch_low_power_scan

r_captouch_low_power_scan ()		
Outline	Control touch measurement	
Declaration	void r_captouch_low_power_scan (void)	
Description	Controls touch measurement. 1. Start touch measurement. 2. Start count of AGT. 3. Transit to Software Standby mode. 4. Stop AGT count.	
Argument	-	-
Return value	-	

3.6.3 r_captouch_low_power_disable_rtc

r_captouch_low_power_disable_rtc ()		
Outline	Disable RTC	
Declaration	void r_captouch_low_power_disable_rtc (void)	
Description	Initialize RTC registers.	
Argument	-	-
Return value	-	

3.6.4 R_BSP_WarmStart

R_BSP_WarmStart()		
Outline	Detect reset source and cancel source of Deep Software Standby mode	
Declaration	void R_BSP_WarmStart (bsp_warm_start_event_t event)	
Description	Detects reset source and cancel source of Deep Software Standby mode	
Argument	bsp_warm_start_event_t event	Warm start event BSP_WARM_SRART_RESET: After reset BSP_WARM_START_POST_CLOCK: After clock setting BSP_WARM_START_POST_C: After initialization C runtime
Return value	-	

3.6.5 R_CAPTOUCH_LPM_Save

R_CAPTOUCH_LPM_Save()		
Outline	Save Standby SRAM	
Declaration	void R_CAPTOUCH_LPM_Save (void)	
Description	Save data for Touch middleware to Standby SRAM	
Argument	-	-
Return value	-	

3.6.6 R_CAPTOUCH_LPM_Load

R_CAPTOUCH_LPM_Load()		
Outline	Load Standby SRAM	
Declaration	void R_CAPTOUCH_LPM_Load (void)	
Description	Load data for Touch middleware from Standby SRAM.	
Argument	-	-
Return value	-	

3.6.7 R_CAPTOUCH_LPM_IsPowerOnReset

R_CAPTOUCH_LPM_IsPowerOnReset()		
Outline	Power-on reset detection	
Declaration	void R_CAPTOUCH_LPM_IsPowerOnReset (void)	
Description	Return result of Power-on reset detection.	
Argument	-	-
Return value	-	

4. Flowcharts

4.1 qe_touch_main ()

Flowchart is shown below.

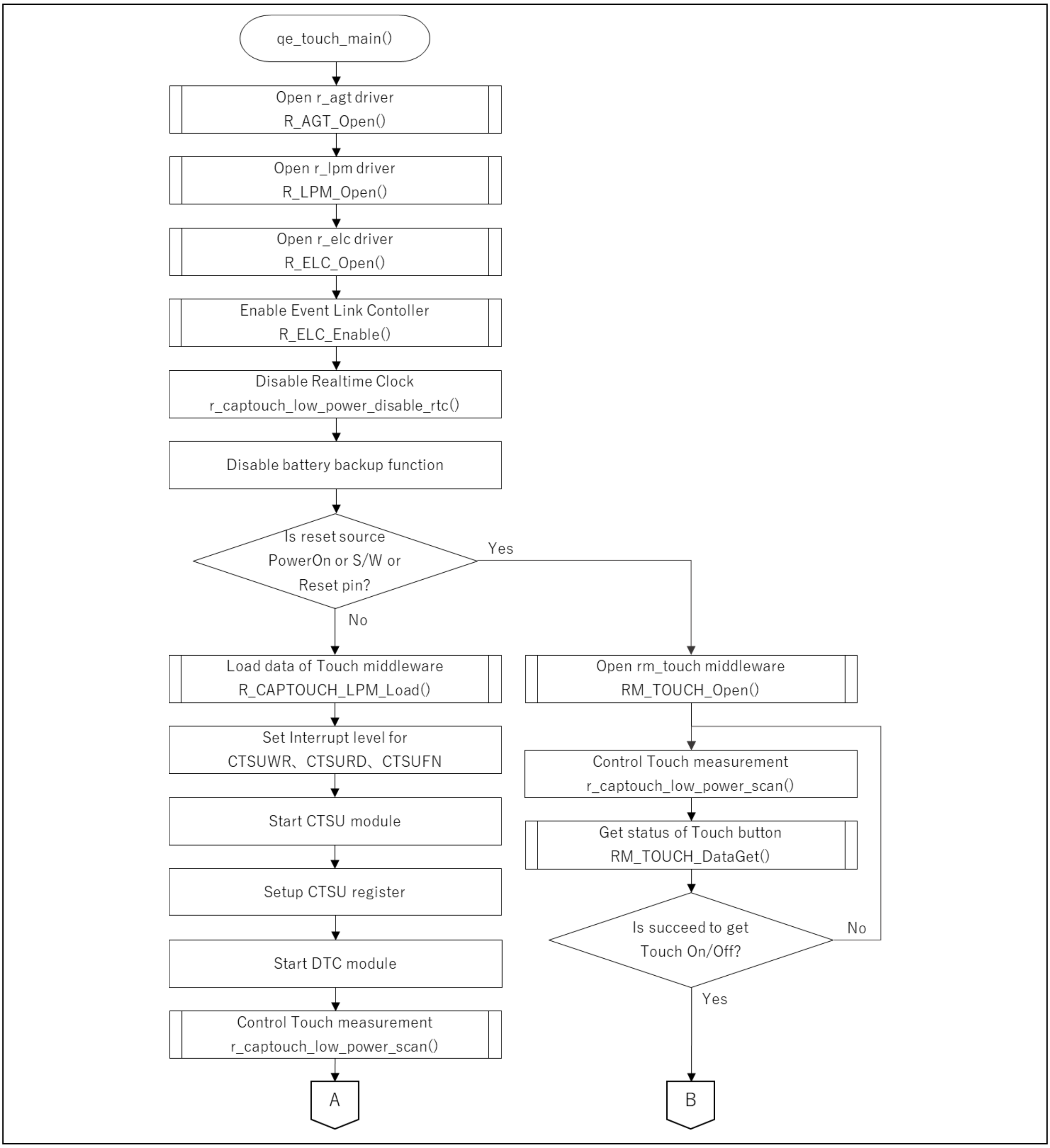


Figure 4-1 qe_touch_main () (1/2)

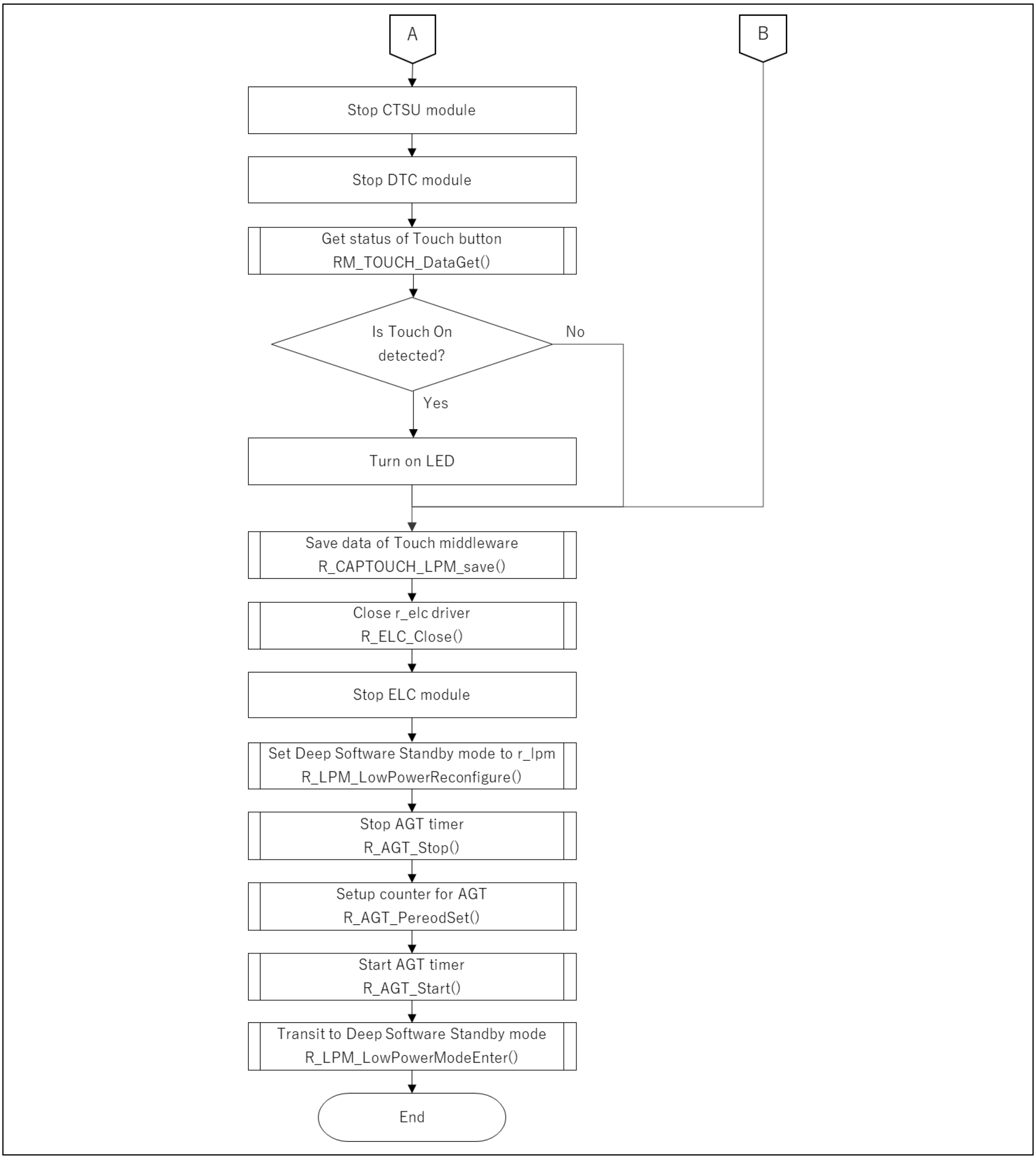


Figure 4-2 qe_touch_main () (2/2)

4.2 r_captouch_low_power_scan ()

Flowchart is shown below.

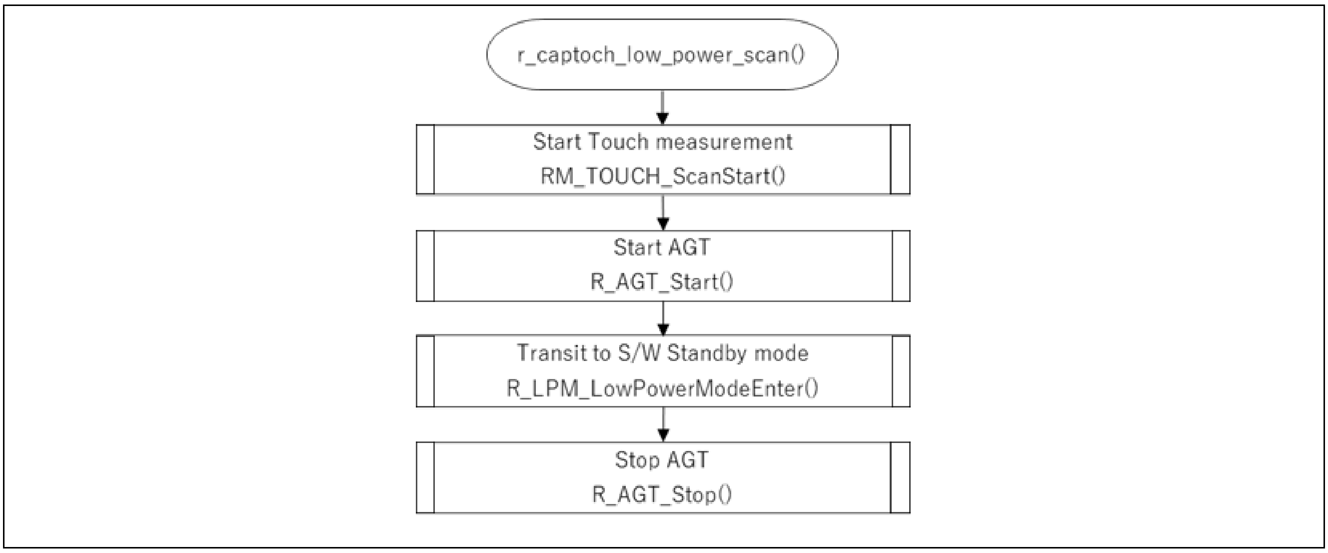


Figure 4-3 r_captouch_low_power_scan ()

4.3 r_captouch_low_power_disable_rtc ()

Flowchart is shown below.

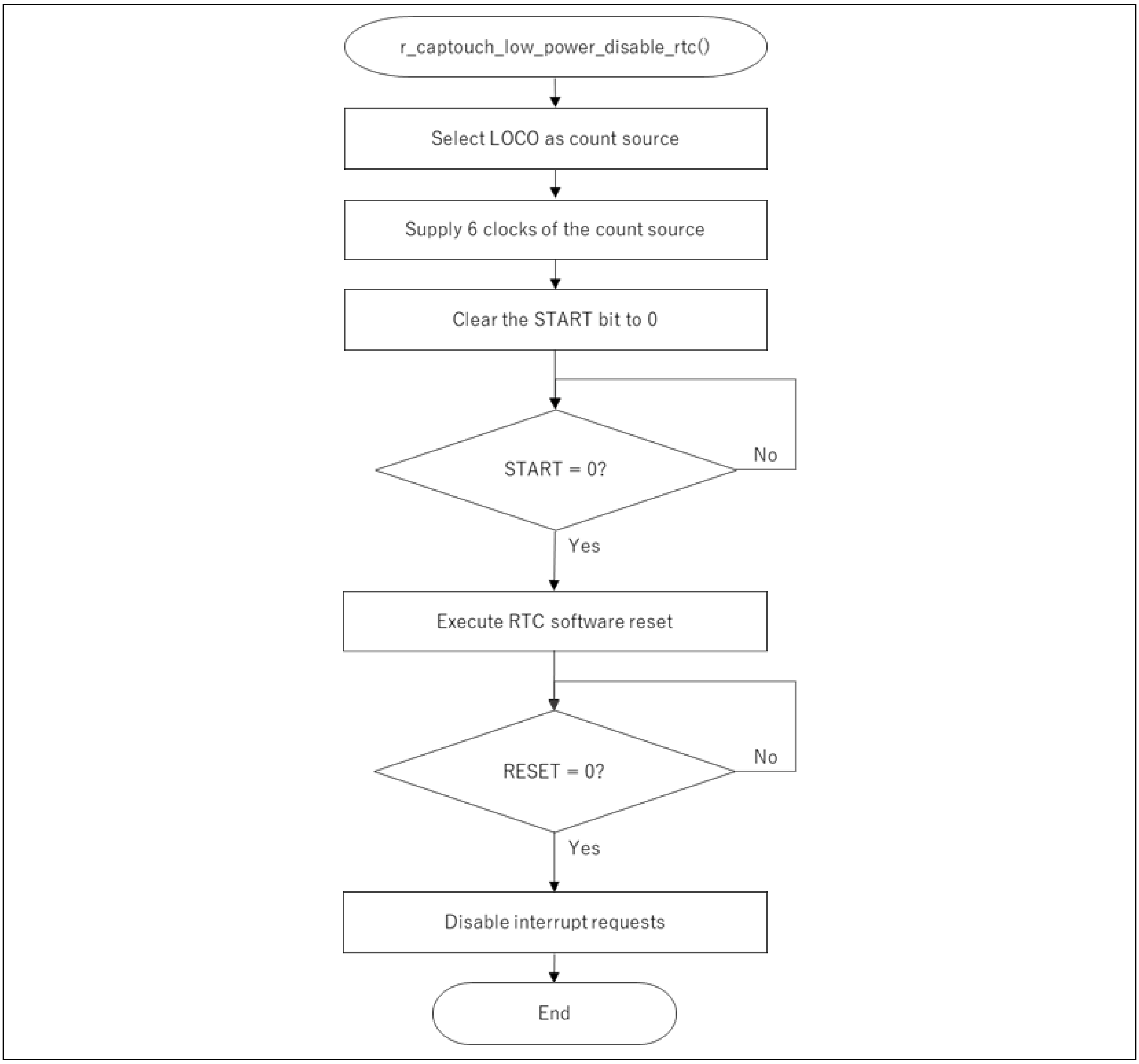


Figure 4-4 r_captouch_low_power_disable_rtc ()

4.4 R_BSP_WarmStart ()

Flowchart is shown below.

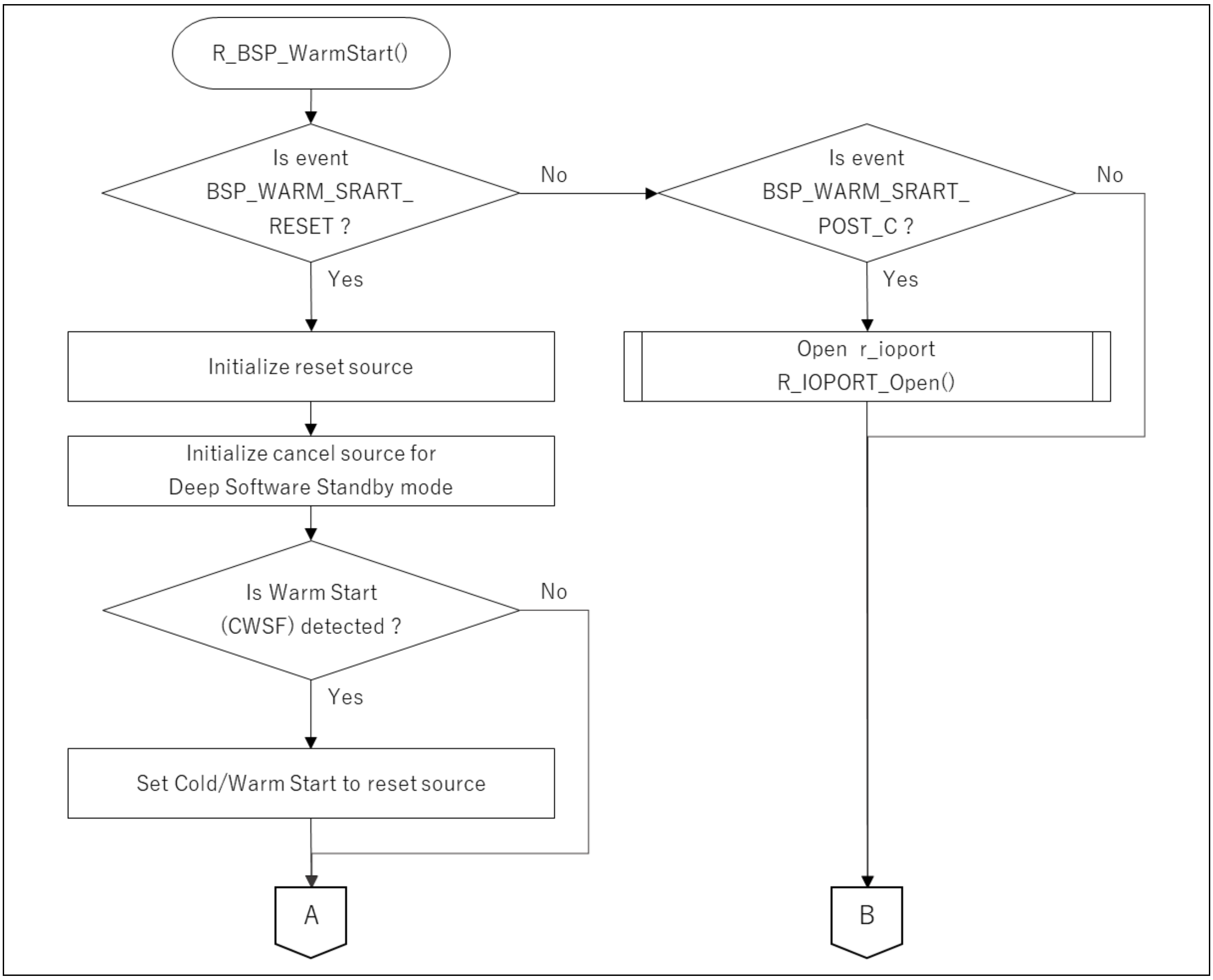


Figure 4-5 R_BSP_WarmStart () (1/4)

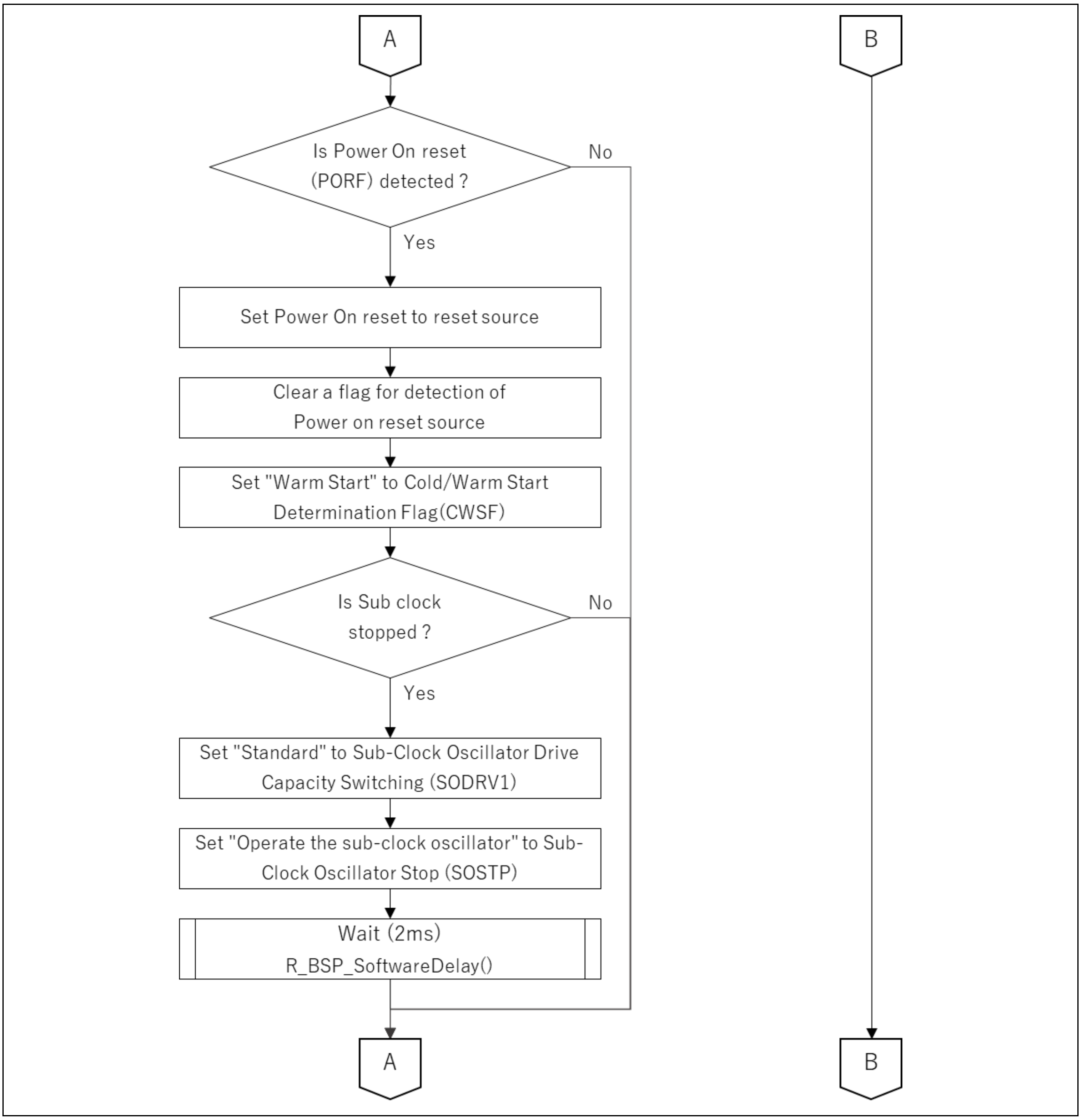


Figure 4-6 R_BSP_WarmStart () (2/4)

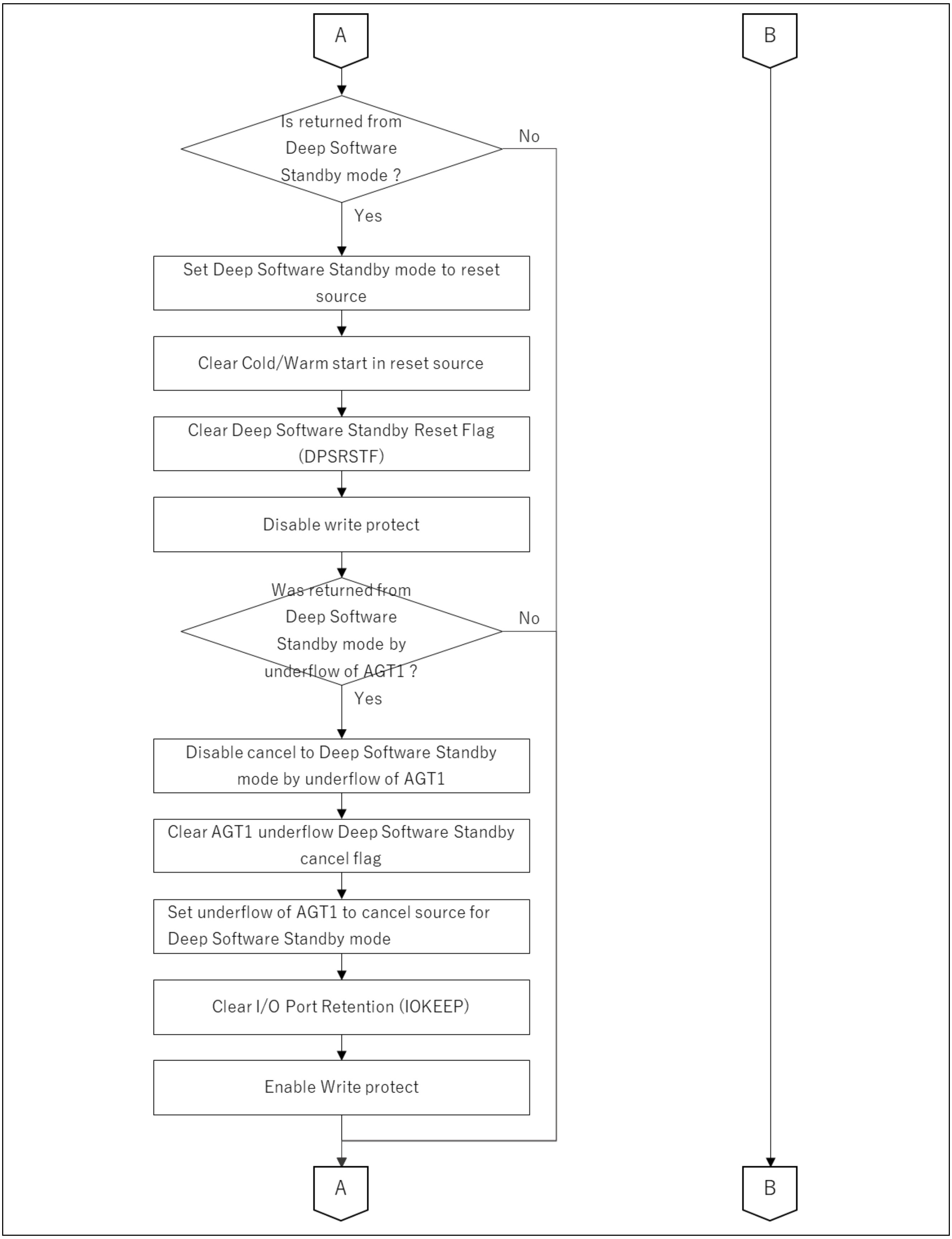


Figure 4-7 R_BSP_WarmStart () (3/4)

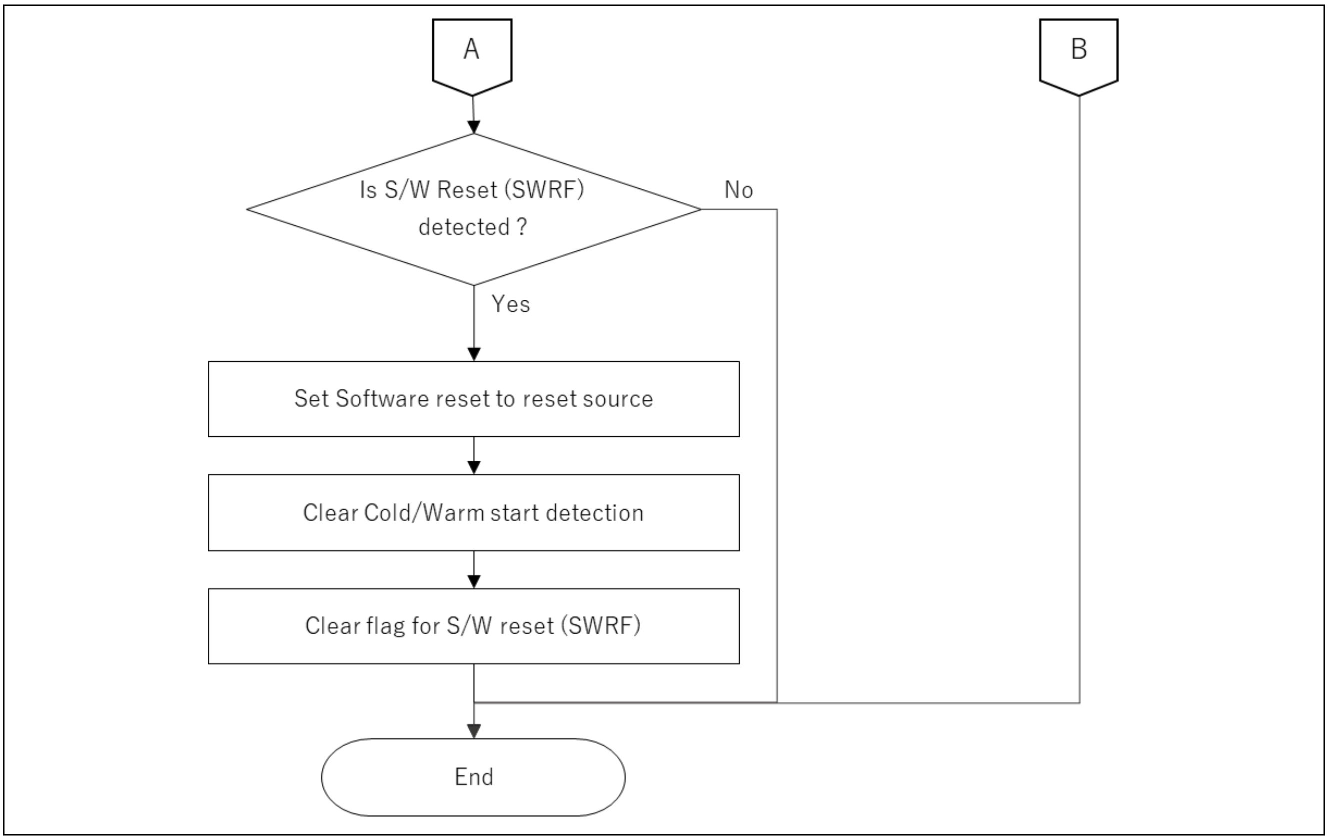


Figure 4-8 R_BSP_WarmStart () (4/4)

4.5 R_CAPTOUCH_LPM_Save ()

Flowchart is shown below.

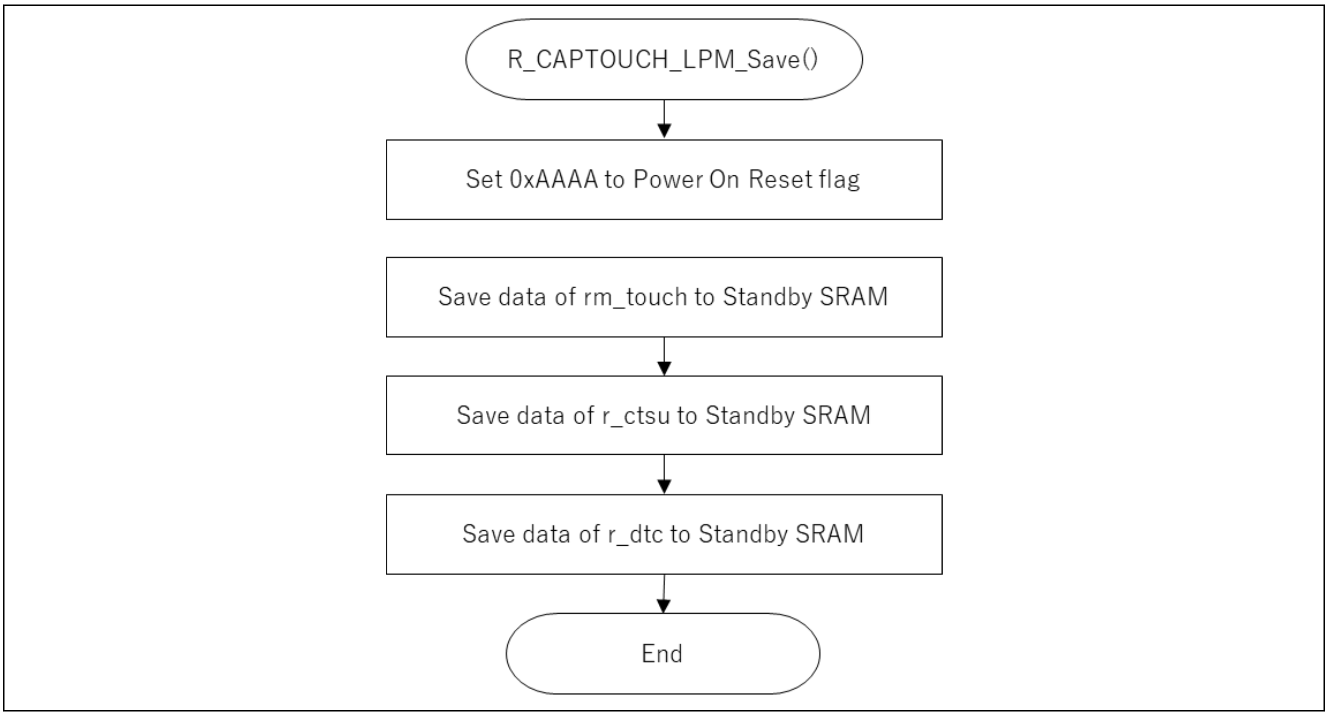


Figure 4-9 R_CAPTOUCH_LPM_Save ()

4.6 R_CAPTOUCH_LPM_Load ()

Flowchart is shown below.

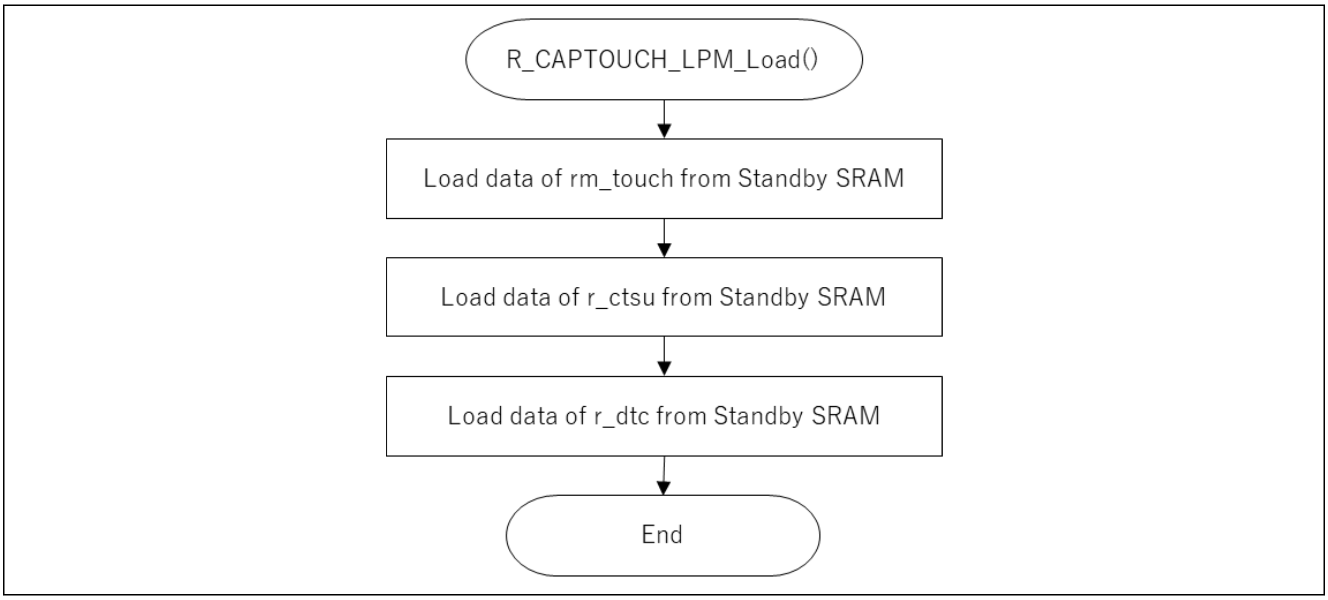


Figure 4-10 R_CAPTOUCH_LPM_Load ()

4.7 R_CAPTOUCH_LPM_IsPowerOnReset ()

Flowchart is shown below.

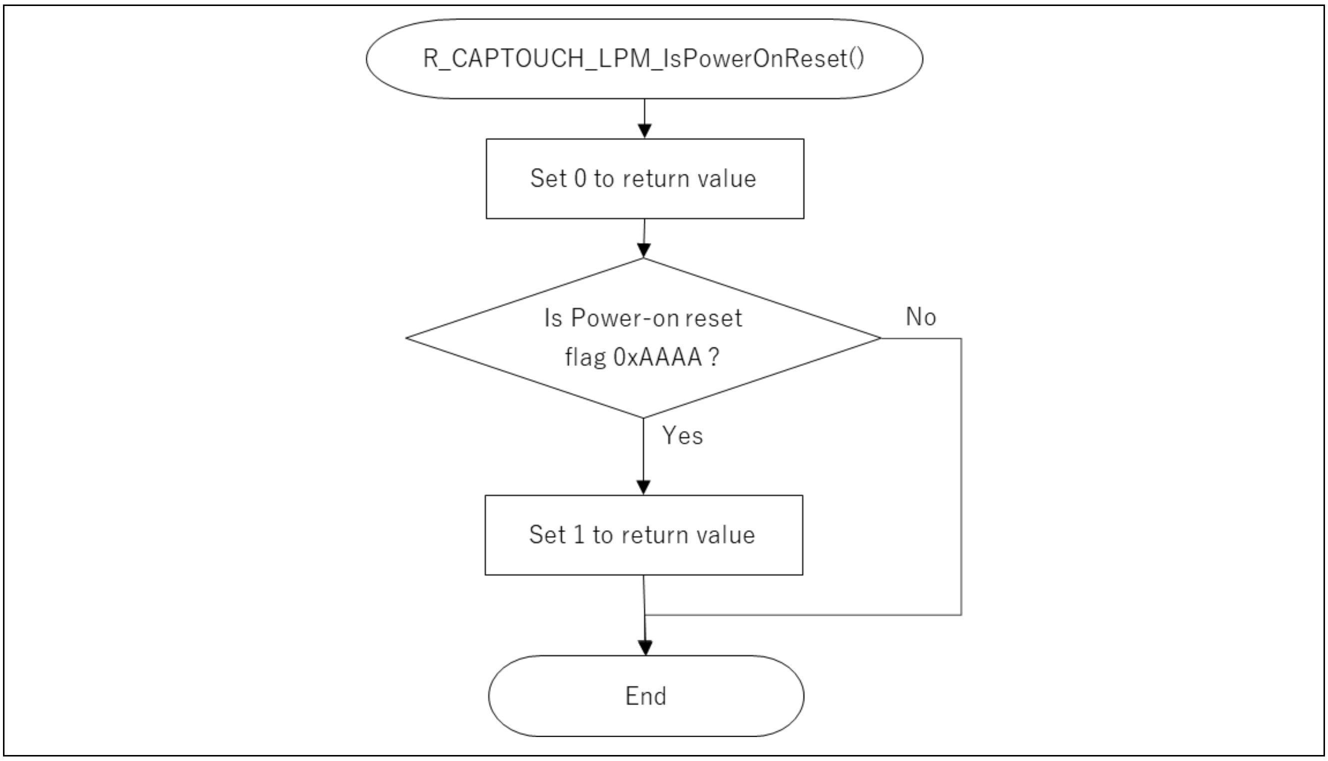


Figure 4-11 R_CAPTOUCH_LPM_IsPowerOnReset ()

5. Current consumption

The system configuration in the following red box shows a model of the electrostatic capacitive touch low power consumption operation described in this application note.

The system is in low power mode and only the capacitive touch button (power button) is being measured in 100ms cycles.

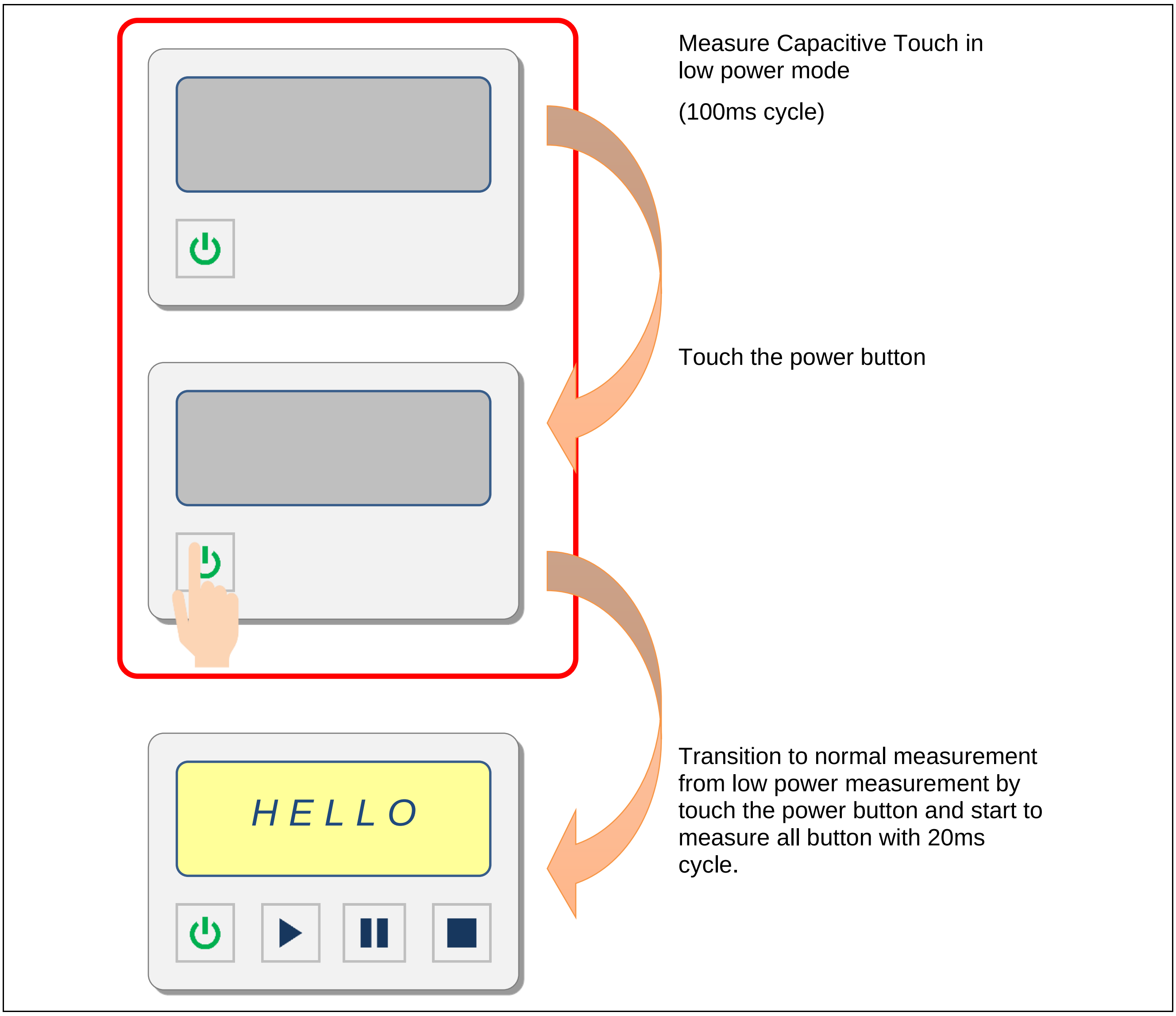


Figure 5-1 Model of the electrostatic capacitive touch low power consumption operation

5.1 Operation Conditions

Table 5.1 shows operation condition.

Table 5.1 Operation Conditions

Item	Description
Operating frequency	20MHz High-speed on-chip oscillator（HOCO） 32kHz Low-speed on-chip oscillator（LOCO）
System clock (ICLK)	20MHz
Peripheral module clock A（PCLKA）	20MHz
Peripheral module clock B (PCLKB)	20MHz
Peripheral module clock C（PCLKC）	20MHz
Peripheral module clock D（PCLKD）	20MHz
Capacitive Touch measurement cycle	100ms
Sensor drive pulse frequency	2MHz
Touch Sensor	TS09
CTSU Measurement Mode	Self-capacitance multi-scan mode
CTSU Power Supply Capacity Adjustment	Normal output
CTSU High-Pass Noise Reduction Function	Turned on
CTSU Spectrum Diffusion Frequency Division	2MHz (0001b)
CTSU Measurement Count	3

5.2 Measurement Environment

Table 5-2 shows equipment and software used in current consumption measurement.

Table 5-2 Equipment and software

Type	Name	Use
Digital multi meter	Keithley DMM7510	Measure current consumption
Power supply	KENWOOD PA18-1.2A	Supply power to RA6M2 Cap Touch CPU board
Software	Keithley KickStart Software	Get result of current consumption measurement from Keithley DMM7510 and output the result to log-file.

5.3 RA6M2 Cap Touch CPU board jumper settings

Table 5-3 shows jumper settings of RA6M2 Cap Touch CPU board to measure current consumption.

Table 5-3 Jumper settings

Position	Circuit group	Jumper	Use
JP1	Power	Open	Measure current consumption
JP2	Power	Close 1-2 pin	Power supply from DC jack

5.4 RA6M2 Cap Touch CPU board

The front of RA6M2 Cap Touch CPU board are as follows.

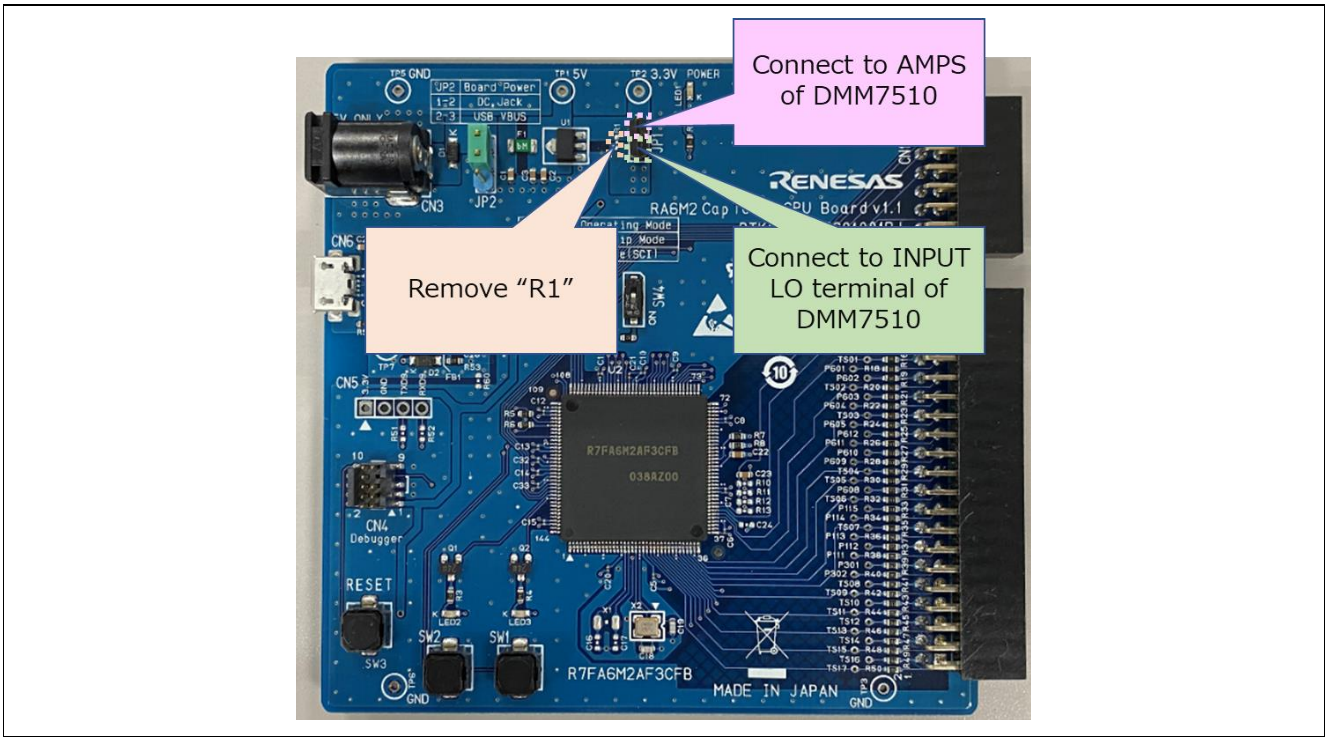


Figure 5-2 RA6M2 Cap Touch CPU board - Front part

5.5 Environment to measure current consumption

Figure 5-3 shows environment to measure current consumption.

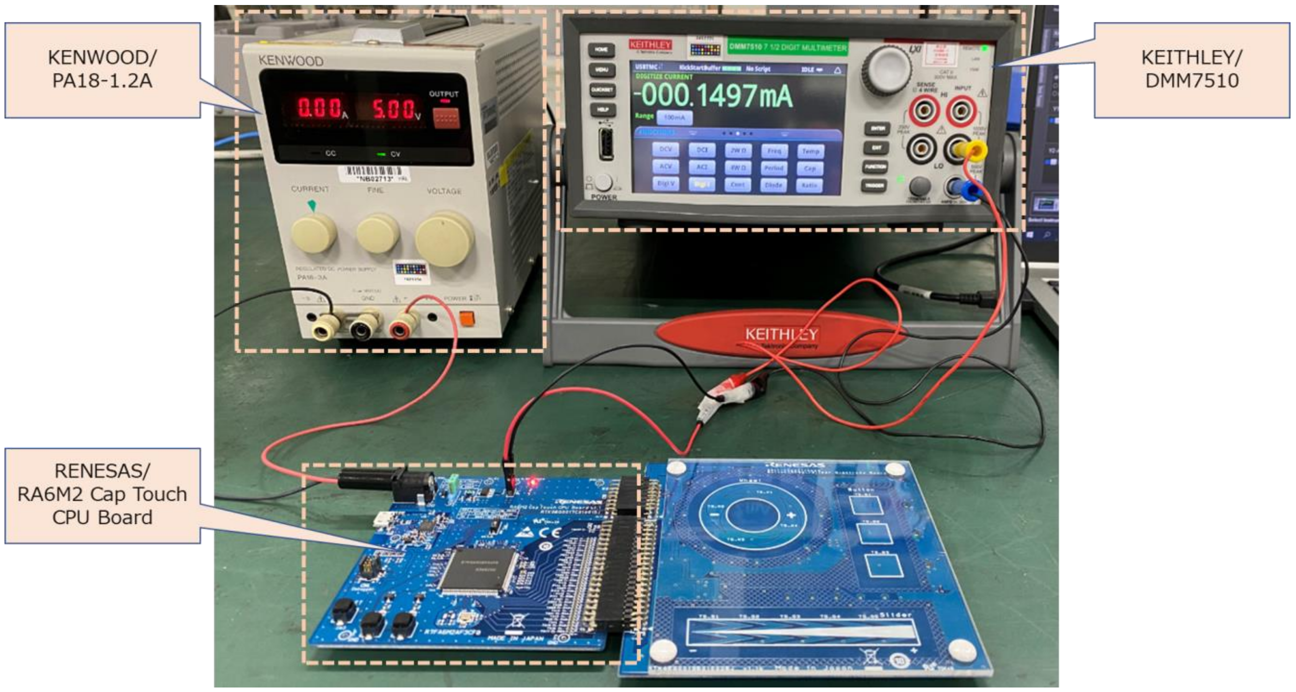
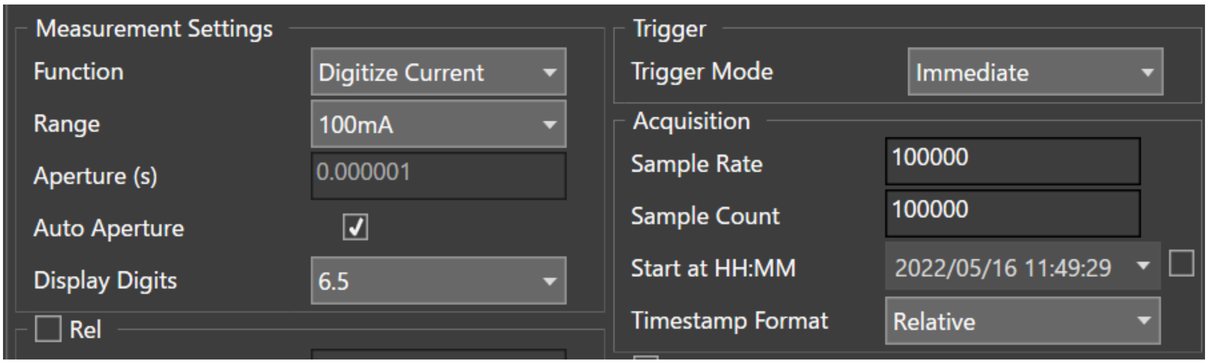


Figure 5-3 Environment to measure current consumption

5.6 Setting to measure current consumption

Table 5- shows settings of Keithley KickStart to measure current consumption.



Measurement Settings

Function: Digitize Current

Range: 100mA

Aperture (s): 0.000001

Auto Aperture: ☑

Display Digits: 6.5

☐ Rel

Trigger

Trigger Mode: Immediate

Acquisition

Sample Rate: 100000

Sample Count: 100000

Start at HH:MM: 2022/05/16 11:49:29 ☐

Timestamp Format: Relative

Table 5-4 Settings of Keithley KickStart to measure current consumption

5.7 Current Consumption Results

Figure 5-5 to Figure 5-6 show the current consumption waveforms for a series of operations in which the CPU operation mode transitions to Deep Standby mode, Normal mode(MCU wake up, Touch measurement start process), Snooze mode (touch measurement processing) and Normal mode (Touch measurement end processing and Touch On/Off judgment processing).

Figure 5-5 and Figure 5-6 show the touch measurement at TS pin 1 channel.

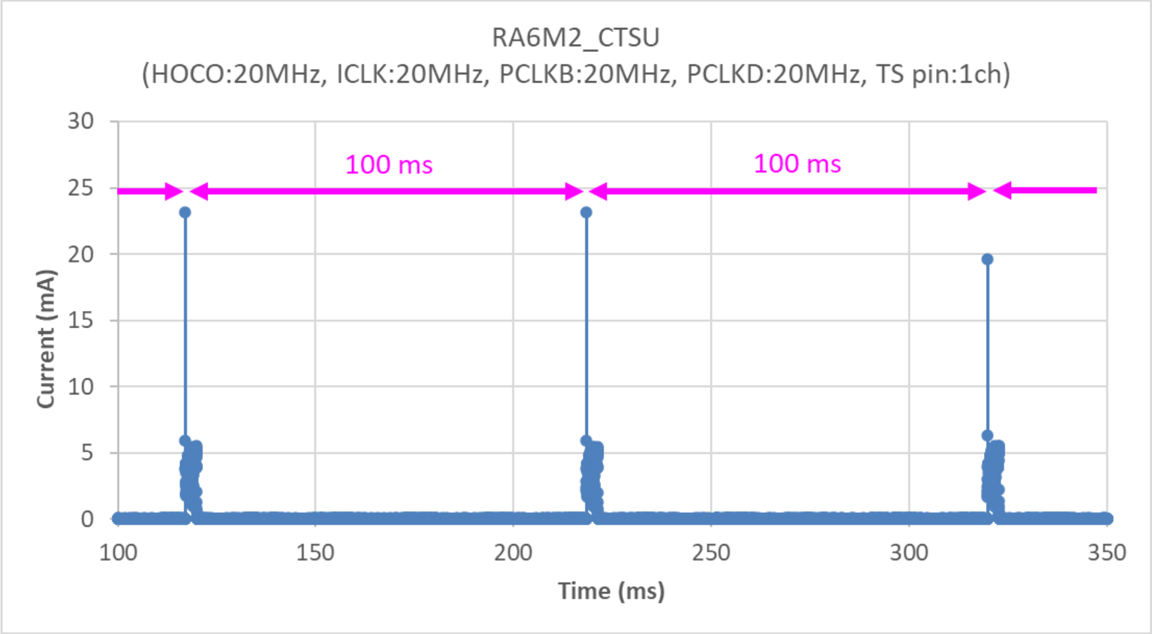


Figure 5-5 Current consumption waveform（1/2）

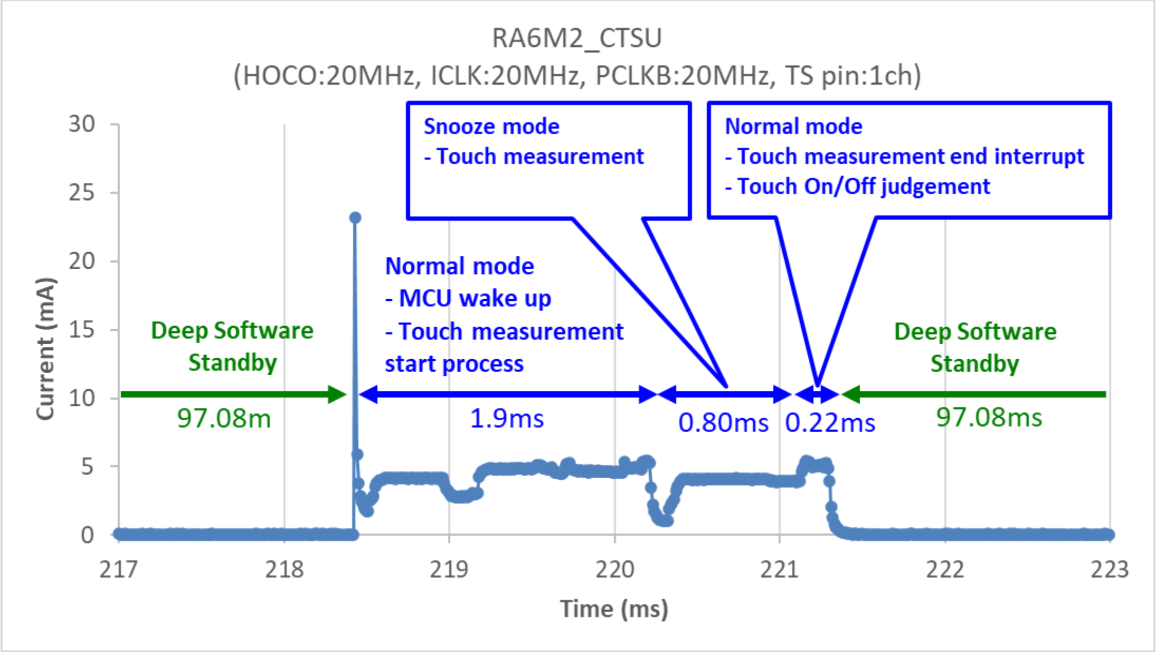
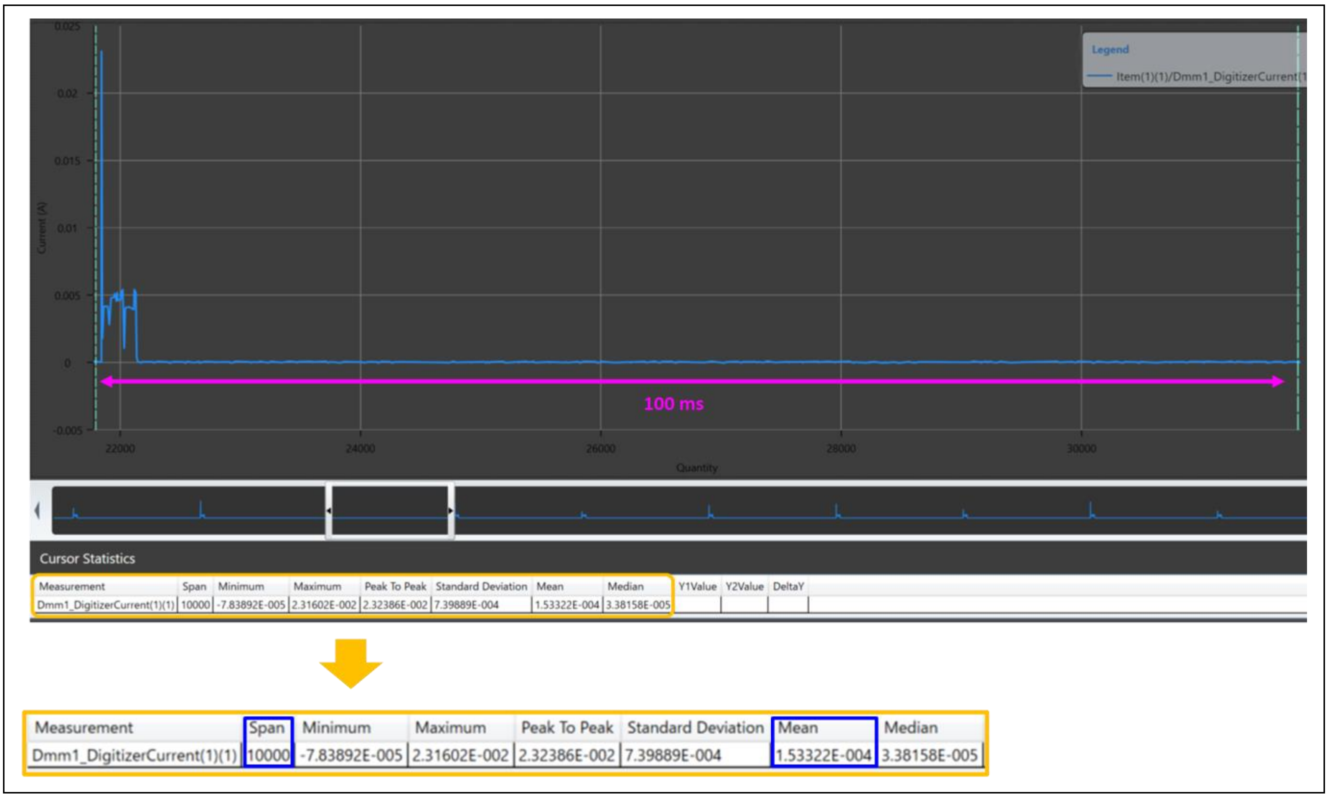


Figure 5-6 Current consumption waveform（2/2）

5.8 Current Consumption Calculation Resalts

The average current consumption of TS pin 1 channel measured with a touch measurement cycle of 100ms is shown below (Figure 5-7).



Measurement	Span	Minimum	Maximum	Peak To Peak	Standard Deviation	Mean	Median	Y1Value	Y2Value	DeltaY
Dmm1_DigitizerCurrent(1)(1)	10000	-7.83892E-005	2.31602E-002	2.32386E-002	7.39889E-004	1.53322E-004	3.38158E-005			

Measurement	Span	Minimum	Maximum	Peak To Peak	Standard Deviation	Mean	Median
Dmm1_DigitizerCurrent(1)(1)	10000	-7.83892E-005	2.31602E-002	2.32386E-002	7.39889E-004	1.53322E-004	3.38158E-005

Figure 5-7 Timing of actual program

Current consumption (touch measurement cycle of 100ms) = **153.322 µA**

6. References

User's Manual: Hardware

RA6M2 User's Manual Hardware R01UH0885

(The latest version can be downloaded from the Renesas Electronics website.)

Technical Update/Technical News

(The latest version can be downloaded from the Renesas Electronics website.)

User's Manual: Development Tools

(The latest version can be downloaded from the Renesas Electronics website.)

User's Manual: RA6M2 MCU Group Capacitive Touch Evaluation System

(The latest version can be downloaded from the Renesas Electronics website.)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jun.30.22	-	First edition issued

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.