RX Family

Capacitive Touch Sensor Correction for Accuracy Enhancement

R01AN3610EJ0100
Rev.1.00
Jan 18, 2017

Introduction

This application note details the procedure for enhancing capacitance detection accuracy of the Capacitive Touch Sensor Unit (CTSU herein) by correcting variations of the internal current-controlled oscillator (ICO herein). The objective of this method is to calculate a coefficient (factor), and multiply CTSU ICO values by it.

Target Device

RX113, RX130, RX231, RX230

Contents

1. Outline

Measurement of capacitance, using the CTSU, relies on the operation of a Current-Controlled Oscillator (ICO). Operation of the CTSU and the underlying component circuits is detailed in R30AN0218. The CTSU outputs counter values representing the measured capacitance for a TS Pin. However, due to variation in the characteristics of the current controlled oscillators, two MCUs operating under identical conditions may not produce identical outputs. Variations observed may be up to 20 percent of the threshold. This application note explains these variations and describes the procedure to correct the same.

Figure 1.1 shows an overview of the entire process for converting capacitance to the measurement count value.

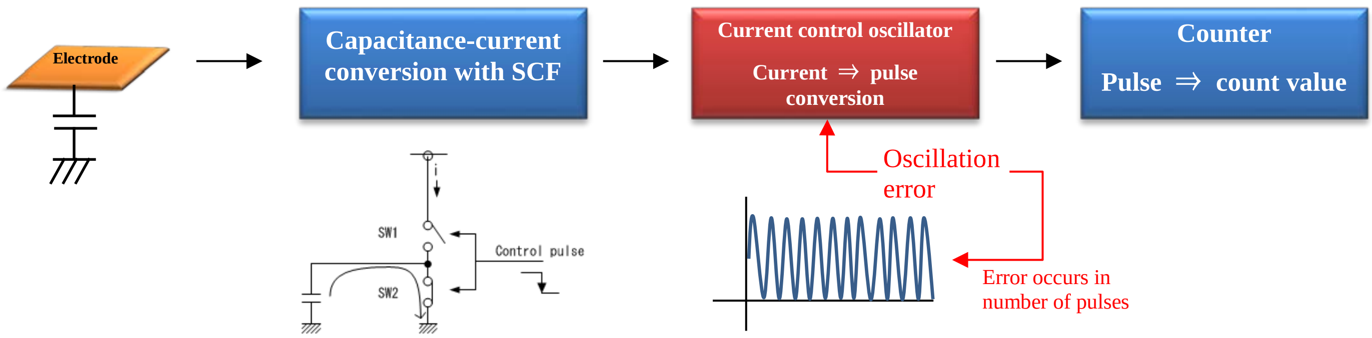

Figure 1.1 Capacitance Conversion Overview

In ICO oscillation error correction, as shown in Figure 1.1, the correction coefficient is determined by calculating two values from two different current values, and then correcting the measurement count value. Figure 1.2 and Figure 1.3 show the correction data graph.

Horizontal axis: current value   Vertical axis: count value

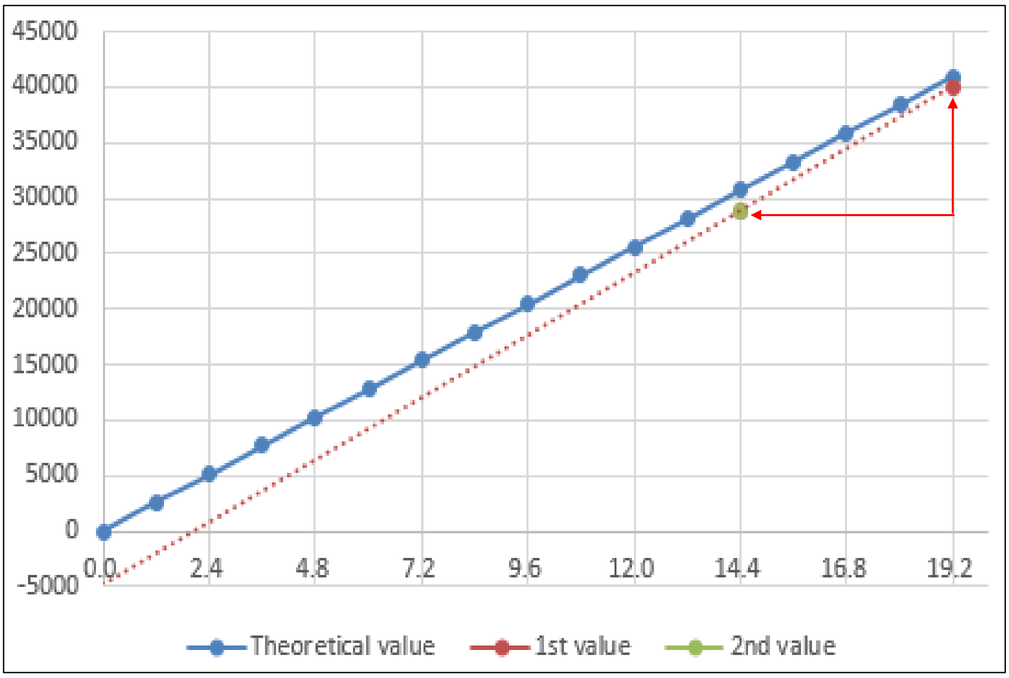

Figure 1.2 Correction Coefficient

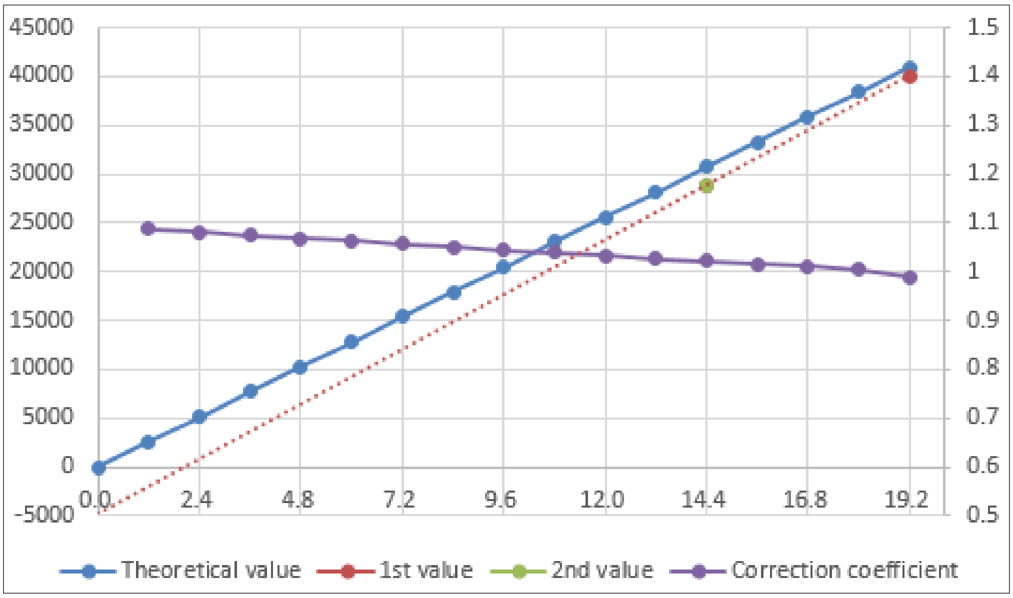

Figure 1.3 Correction Coefficient

The oscillation error is corrected by multiplying the measurement count value by the correction coefficient.

In conventional ICO correction processing, the current supplied to the ICO was matched with the reference ICO counter by the current offset function. In this method, the offset amount of the capacitance to be measured could be corrected, but the ICO gain characteristic (slope), which differs with each MCU, could not be corrected. Expressed in a linear function, offset value b in y ＝ax+b can be corrected, but gain characteristic coefficient a cannot.

y : Correction count value

a : Gain characteristic per MCU

x : Measurement count value

b : MCU measurement offset value

In the correction process, accuracy is improved by correcting the ICO oscillation error of approximately ±20% (correct *a* above) for each MCU before correcting the current offset and count error due to the difference in parasitic capacitance. Figure 1.4 shows an example in which the relationship between capacitance and measurement count value is measured for four samples. As you can see, the gain characteristic and offset value differ for each sample.

This application explains how to improve the difference in ICO characteristics of each MCU using the calibration function and software-based correction.

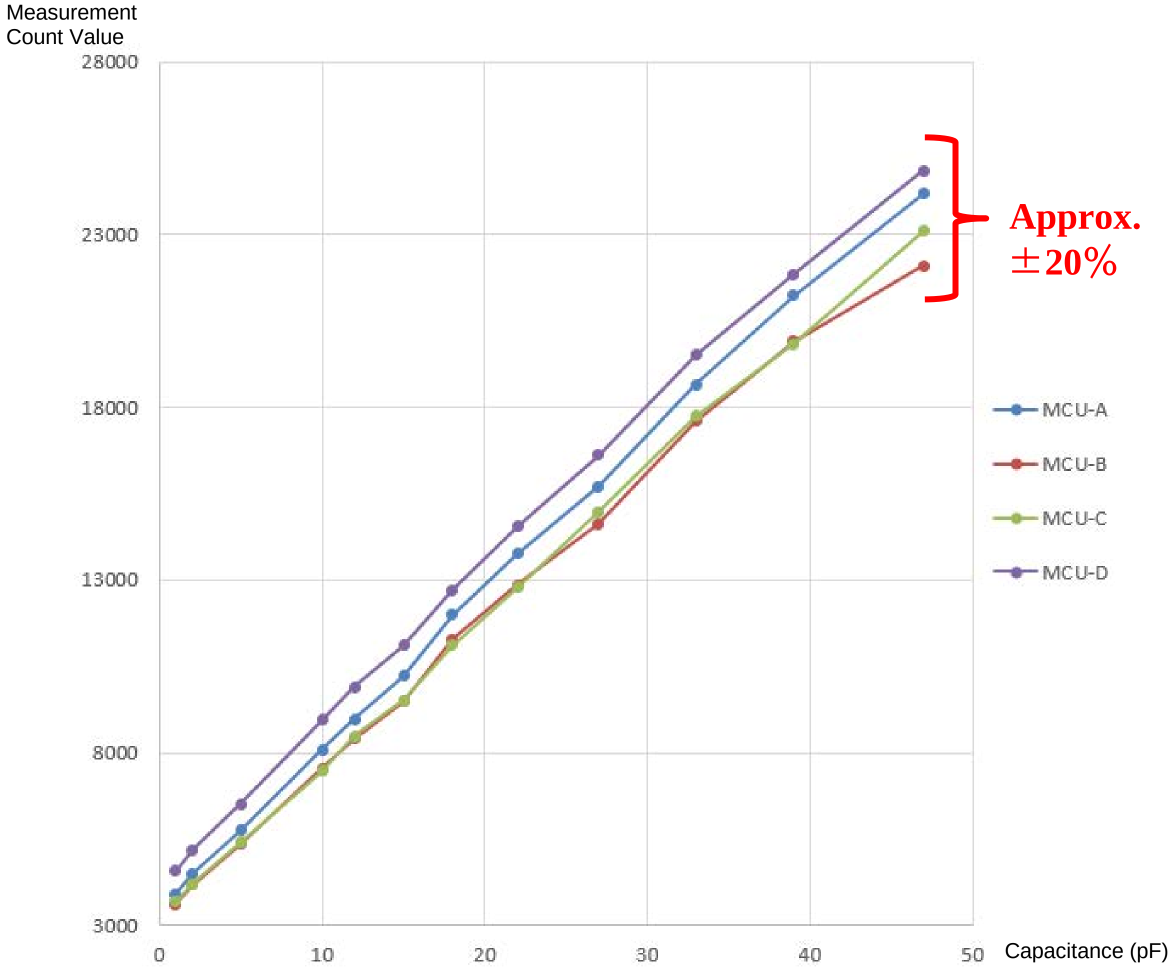

Figure 1.4 Variation between MCUs

2. ICO Gain Correction

Actions needed to perform ICO correction are shown in Figure2.1. ICO gain correction requires the CTSU to be operated in “calibration mode” during initialization. Once correction coefficients are calculated, the CTSU is placed back in normal operating mode and all measured counter values are multiplied with a correction coefficient.

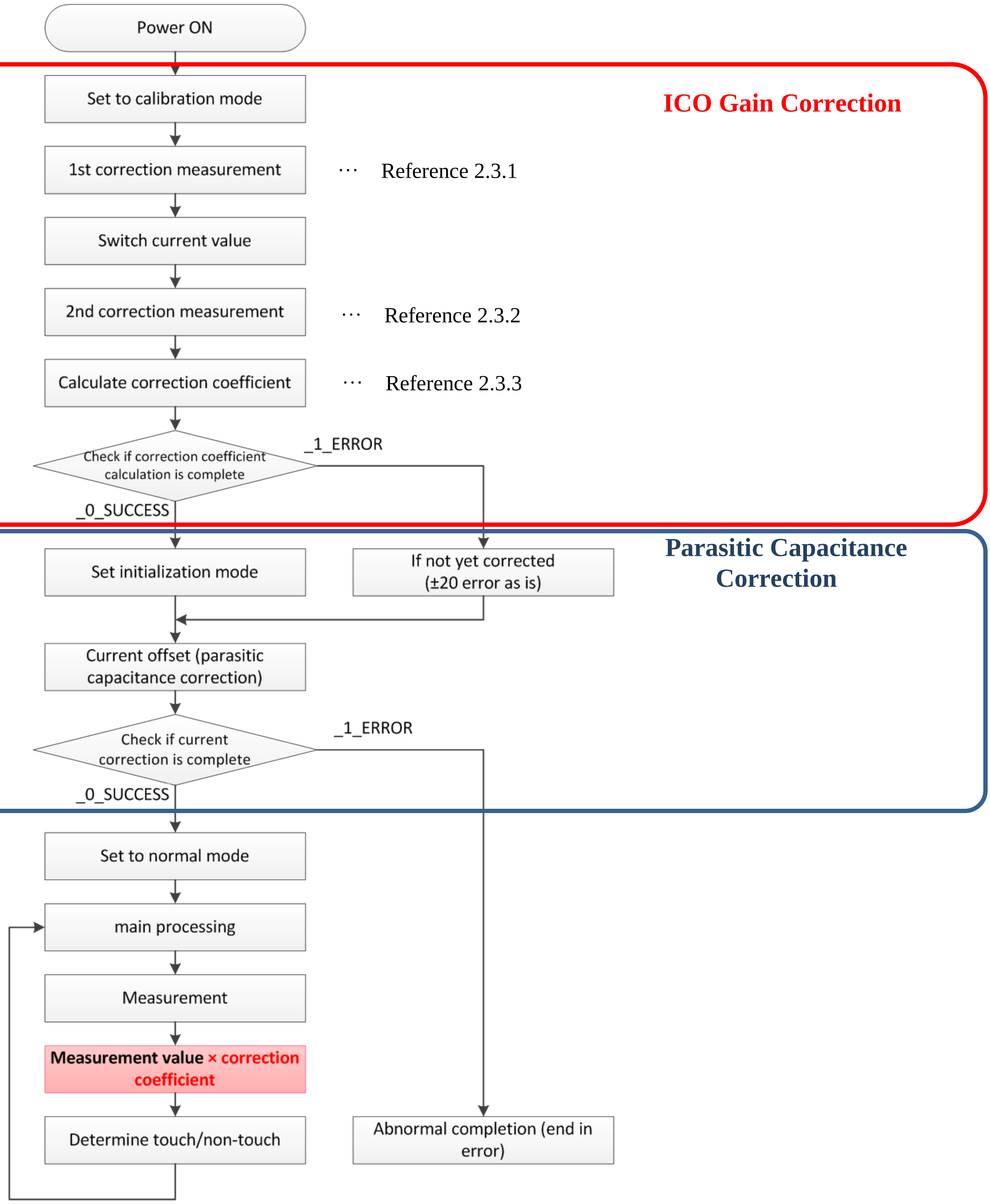

Figure2.1 Correction Overview Flowchart

2.1 Theoretical Value

ICO gain correction is carried out by making reference to the theoretical values output by an ideal ICO. Table 2.1 below shows the theoretical count value output when a corresponding input current (I) value is used. The count value output is determined by counting the number of pulses output from the ICO every 0.526ms (the measurement time unit). Figure 2.2 is a graphical representation of the theoretical output shown in Table 2.1.

Measurement time is determined by the measurement frequency and number of measurements.

Table 2.1 Theoretical Value

Current Value (uA)	Count Value	Current Value (uA)	Count Value
0.0	0	10.8	23040
1.2	2560	12.0	25600
2.4	5120	13.2	28160
3.6	7680	14.4	30720
4.8	10240	15.6	33280
6.0	12800	16.8	35840
7.2	15360	18.0	38400
8.4	17920	19.2	40960
9.6	20480	-	-

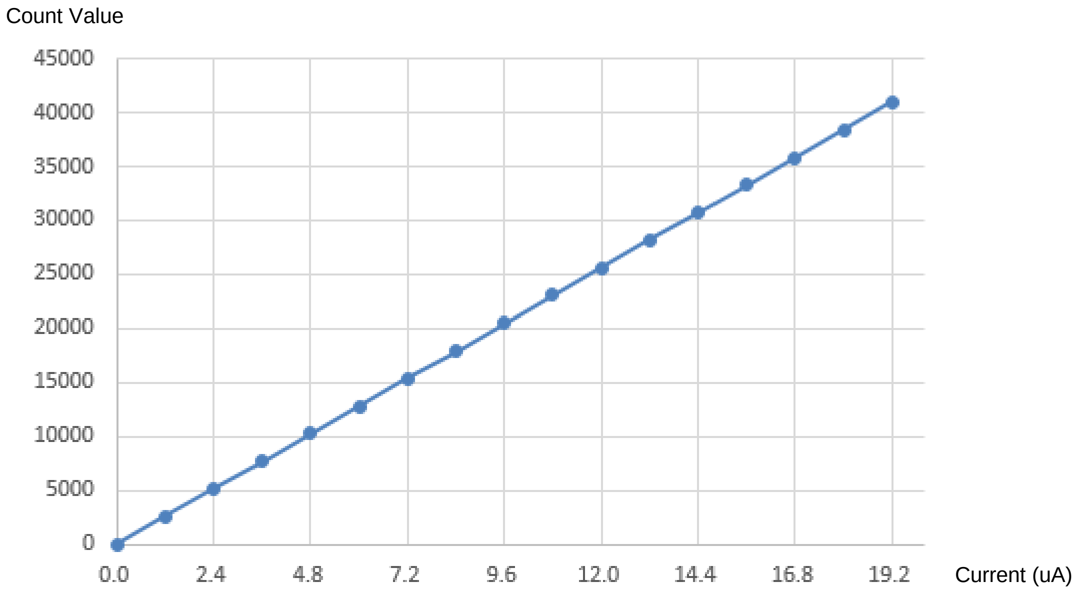

Figure 2.2 Theoretical Value

2.2 ICO Correction Register

The CTSU reference current adjustment register (CTSUTRMR) and the CTSU error status register (CTSUERRS) are two additional SFRs used for ICO correction measurement. Registers structures for CTSUTRMR and CTSUERRS are shown below.

CTSU Reference Current Adjustment Register **(CTSUTRMR)**

Address 007F FFBEh

	b7	b6	b5	b4	b3	b2	b1	b0
Reset Value	Value for Each Chip							

Note : The CTSUTRMR register is programmed at the time of factory shipment with the reference current value adjusted to certain conditions pertaining to each chip. This value is accessed during the ICO gain correction.

CTSU Error Status Register (CTSUERRS)

Address 000A 091Ch

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CTSUICOMP	-	-	-	-	-	-	-	CTSUTSOC	-	-	-	CTSUDRV	CTSUTSOD	CTSUSPMD	
Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit Name	Function	Read/Write
b1-b0	CTSUSPMD	Calibration mode bit	b1 b0 0 0 : Capacitance measurement mode 0 1 : Setting disabled **1 0 : Calibration mode** 1 1 : Setting disabled	R/W
b2	CTSUTSOD	TS pin fixed output bit	0 : Capacitance measurement mode 1 : TS pin fixed output (High/Low output)	R/W
b3	CTSUDRV	Calibration mode bit setting 3	0 : Capacitance measurement mode 1 : Calibration setting 3	R/W
b6-b4	-	Reserved bit	Reads out 0. When writing, specify 0.	R/W
b7	CTSUTSOC	Calibration mode bit setting 7	0 : Capacitance measurement mode 1 : Calibration setting 7	R/W
b14-b8	-	Reserved bit	Reads out 0. When writing, specify 0.	R/W
b15	CTSUICOMP	TSCAP voltage abnormal monitoring bit	0 : TSCAP voltage normal 1 : TSCAP voltage abnormal	R

The ICO gain correction method requires operating the CTSU in “Calibration mode” by setting the CTSUERRS SFR bits CTSUSPMD (b1:b0) to 0b10. Then, it measures two current (I) values; first with CTSUTRMR set to the factory programmed value (19.2uA) then with CTSUTRMR set to 0xFF. The sensor counter values obtained are used to calculate the calibration coefficients.

2.3 ICO Correction Measurements

2.3.1 1st Correction Measurement

The 1st correction measurement is started when the CTSUTRMR register is set to the factory programmed value which is configured to always supply 19.2uA to the ICO. The measured sensor counter value is designated as the “1st value,” as shown in Figure2.3.

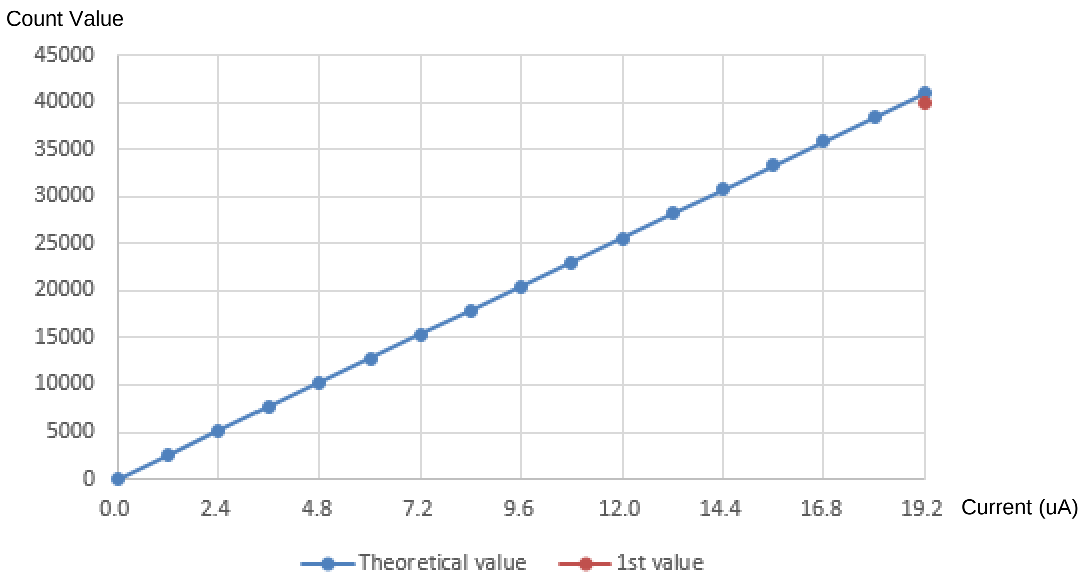

Figure2.3 1st Value

The 1st value shown in Figure2.3 is an example of an ICO with an oscillation frequency lower than that of the theoretical value of 19.2uA. Some ICOs have higher oscillation frequencies, but we will base our explanation here on an ICO with a lower than ideal oscillation frequency.

2.3.2 2nd Correction Measurement

The 2nd correction measurement is taken after changing the supply current to the ICO, which is done by writing 0xFF to the CTSUTRMR register.

CTSUTRMR is trimmed before shipping to ensure that a 19.2uA current is supplied to the CTSU. Due to variations between MCUs, due to the manufacturing process, the set value of each MCU differs.

In this sample, correction is implemented by creating a slope from the current values measured at two different points. The correction coefficient accuracy can be improved by taking the 1st correction measurement at 19.2uA and the 2nd correction measurement with a large current difference. In order to accomplish this, the measurement is taken at the maximum current difference by inputting the maximum 0xFF value that can be set in CTSUTRMR.

*CTSU reference current adjustment register value: as the setting value is increased, the current providing to ICO decreases.

After the CTSUTRMR supply current has been changed, a measurement is taken in the same manner as the 1st measurement and the count value is obtained. This becomes the 2nd value, as shown in Figure 2.4.

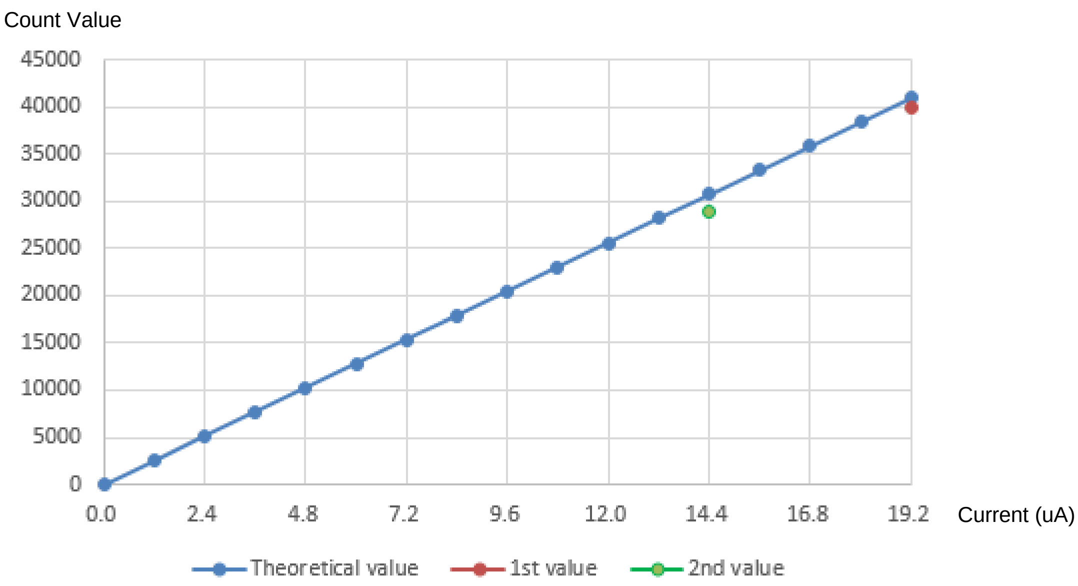

Figure 2.4 2nd Value

The 16-point correction coefficient for 1.2uA to 19.2uA is calculated from the count of the 1st and 2nd values, then IOC correction is implemented. Figure 2.5 shows the correction coefficients.

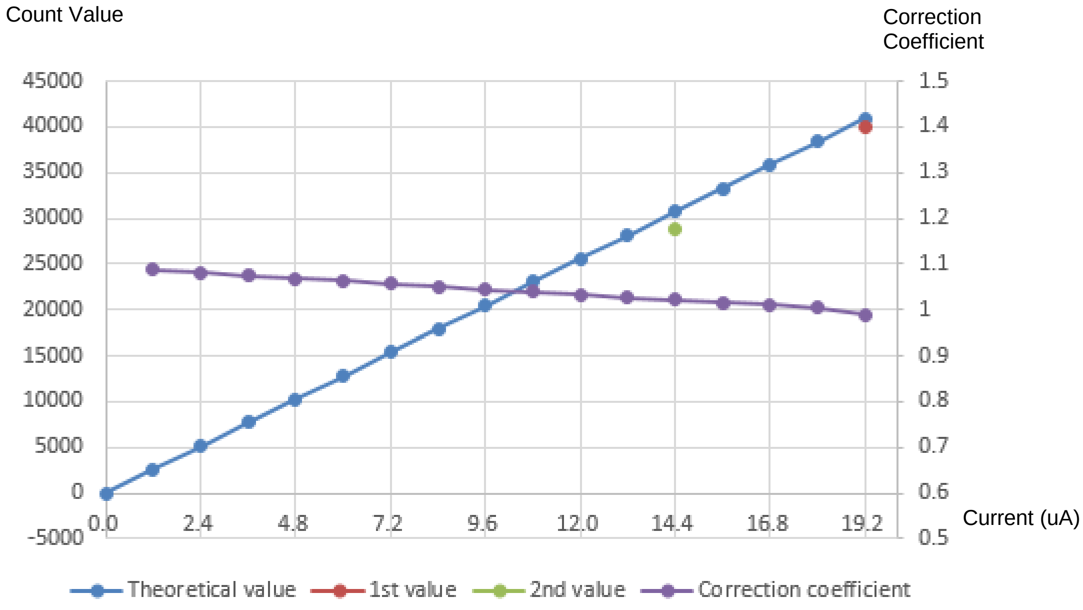

Figure 2.5 Correction Coefficient

2.3.3 Correction Coefficient Calculation

The steps for calculating the 16-point correction coefficient for 1.2uA to 19.2uA are listed below.

- Magnification calculation of 19.2uA theoretical value

$$\text{1st magnification} = \frac{\text{1st value}}{\text{19.2uA theoretical value}} = \frac{40040}{40960} = 0.978$$

Refer to Table

- 2nd correction reference data calculation

$$\text{2nd correction reference data} = \frac{(\text{1stCTSUTRMR value} + 273)}{528} \times \text{19.2uA theoretical value}$$

$$= \frac{(116+273) \times 40960}{528} = 30176$$

- Magnification calculation of 2nd correction reference data

$$\text{2nd magnification} = \frac{\text{2nd value}}{\text{2nd correction reference data}} = \frac{28807}{30176} = 0.954$$

- Calculation of 16-point coefficient for 1.2uA to 19.2uA

Difference of 1st and 2nd magnification = 1st magnification – 2nd magnification (or 2nd magnification – 1st magnification)

= 1001－978 = 23

$$\text{Change rate per 1 point} = \frac{\text{Difference of 1st and 2nd magnification} \times \text{19.2uA theorectical value}}{\text{19.2uA theoretcial value} - \text{2nd correction reference data}} = \frac{23 \times 40960}{40960 - 30176} = 87$$

$$\text{16-point magnification} = (\text{1st magnification} + \frac{\text{Change rate per 1 point} \times N}{16}) / 1024$$

*N: point number

$$= (1001 + \frac{87 \times N}{16}) / 1024$$

Note : In the sample code, when describing the integer operation of division processing for the MCU, in order to reduce calculation error due to fractional truncation, a 10-bit shift to the left is implemented before the division in consideration of the third decimal place and to reduce processing error. In addition, a process to add the denominator divided by 2 is implemented as a workaround for quantization error due to further truncation of decimal points.

Figure 2.6 shows the factor by which the gain has changed at 16 intervals of 1.2µA. This “factor of change” is referred as the “magnification value”. The inverse of the magnification value is used as a multiplier to correct the measured sensor count value to the expected ideal sensor count value. The inverse of the magnification value is the correction coefficient. Table 2.2 shows the results

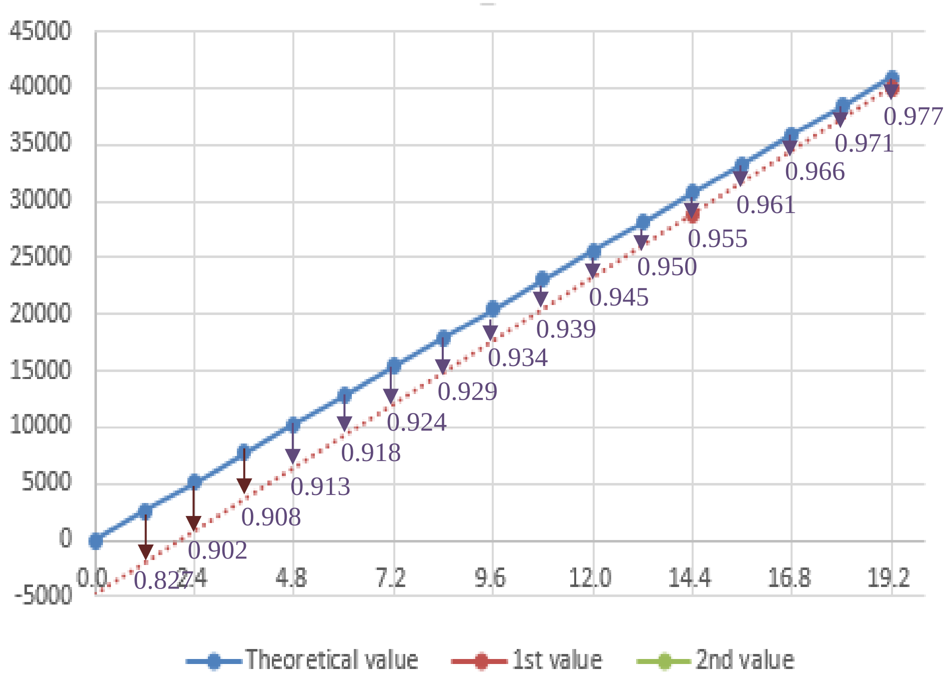

Figure 2.6 Magnification

Table 2.2 Correction Coefficient

N	Magnification	Correction Coefficient (1/magnification)
0 (19.2uA)	0.977	1.023
1 (18.0uA)	0.971	1.029
2 (16.8uA)	0.966	1.034
3 (15.6uA)	0.961	1.040
4 (14.4uA)	0.955	1.046
5 (13.2uA)	0.950	1.052
6 (12.0uA)	0.945	1.058
7 (10.8u)	0.939	1.064
8 (9.6uA)	0.934	1.070
9 (8.4uA)	0.929	1.076
10 (7.2uA)	0.924	1.082
11 (6.0uA)	0.918	1.088
12 (4.8uA)	0.913	1.094
13 (3.6uA)	0.908	1.101
14 (2.4uA.)	0.902	1.107
15 (1.2uA)	0.897	1.114

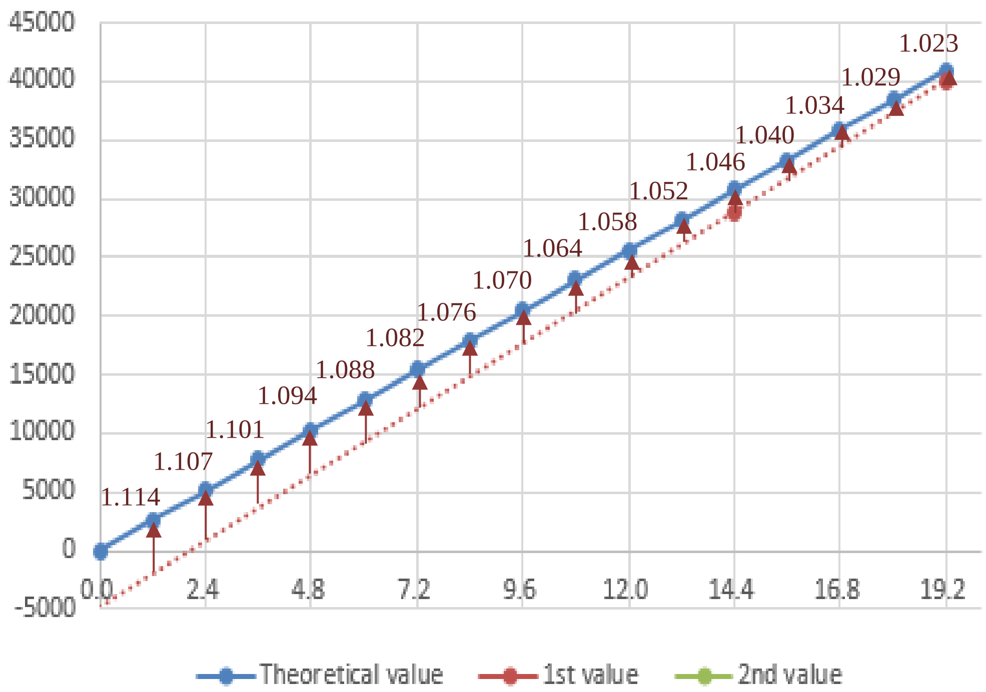

Figure 2.7 Correction Coefficient

3. ICO Correction Software Explanation

3.1 File Configuration

Table 3.1 Lists the ICO correction processing files. Files automatically generated by the integrated development environment are not included in the list.

Table 3.1 Files Used by ICO Software

File Name	Description	Notes
r_main.c	main processing source file	
r_ctsu.c	CTSU control source file	
r_ctsu_physical_driver.c	CTSU physical driver source file	
r_dtc.c	DTC control source file	
r_touch_API.c	Touch API source file	
r_ctsu.h	CTSU control header file	
r_dtc.h	DTC control header file	
r_touch_API.h	Touch API header file	

3.2 Option Setting Memory

Table 3.2 lists the status of each option setting memory. Please set the optimal value for your system as needed.

Table 3.2 Option Setting Memory

<table>
<tr><th>MCU</th><th>Symbol</th><th>Address</th><th>Setting Value</th><th>Description</th></tr>
<tr><td>RX113</td><td rowspan="4">DTC.DTCVBR</td><td rowspan="3">0x00007C00</td><td rowspan="3">dtc_vector60 (0x00007CF0)
dtc_vector61 (0x00007CF4)</td><td rowspan="4">DTC transfer address</td></tr>
<tr><td>RX231</td></tr>
<tr><td>RX230</td></tr>
<tr><td>RX130</td><td>0x00002400</td><td>dtc_vector60 (0x000024F0)
dtc_vector61 (0x000024F4)</td></tr>
</table>

3.3 Constants

Table 3.3 shows the constants used for ICO correction.

Table 3.3 Constants (**r_ctsu.h**)

r_ctus.h		
Constant Name	Setting Value	Description
CTSU_INPUT_FREQUENCY	32000000 (Default)	PCLKB operation clock (Hz) 32000000 (=32MHz) 27000000 (=27MH) 24000000 (=24MHz) 16000000 (=16MHz) 8000000 (=8MHz)
CTSU_INPUT_FREQUENCY_DIV	32 (Default)	CTSU_INPUT_FREQUENCY/ 1000000
CORRECTION_AVERAGE	32	Average number of calibration measurements
_0_NORMAL	0	Normal measurement mode
_1_CORRECTION	1	Calibration measurement mode
_0_1ST	0	1st measurement flag
_1_2ND	1	2nd measurement flag
_19_2UA	40960	Correction reference data when supplying 19.2uA
_18_0UA	38400	Correction reference data when supplying 18.0uA
_16_8UA	35840	Correction reference data when supplying 16.8uA
_15_6UA	33280	Correction reference data when supplying 15.6uA
_14_4UA	30720	Correction reference data when supplying 14.4uA
_13_2UA	28160	Correction reference data when supplying 13.2uA
_12_0UA	25600	Correction reference data when supplying 12.0uA
_10_8UA	23040	Correction reference data when supplying 10.8uA
_09_6UA	20480	Correction reference data when supplying 9.6uA
_08_4UA	17920	Correction reference data when supplying 8.4uA
_07_2UA	15360	Correction reference data when supplying 7.2uA
_06_0UA	12800	Correction reference data when supplying 6.0uA
_04_8UA	10240	Correction reference data when supplying 4.8uA
_03_6UA	7680	Correction reference data when supplying 3.6uA
_02_4UA	5120	Correction reference data when supplying 2.4uA
_01_2UA	2560	Correction reference data when supplying 1.2uA

3.4 Functions

Table 3.4 lists the functions used for ICO correction.

Table 3.4 Functions

Function Name	Description
main()	System main function
correction_CTSU_sensor_ico()	Sensor ICO correction function
correction_CTSU_sensor_selection()	Sensor correction sensor selection function
correction_CTSU_register_parameter_set()	ICO correction CTSU register value setting function
correction_CTSU_register_txd_set()	ICO correction transmission measurement parameter setting function
DTC_Set_Initial_of_CTSU()	CTSU DTC initialization function
correction_CTSU_register_set()	ICO correction CTSU register setting function
correction_CTSU_measurement_start()	ICO correction measurement start function
correction_CTSU_1st_coefficient_create()	1st magnification correction reference data calculation function
correction_CTSU_2nd_standard_val_create()	2nd correction reference data calculation function
correction_CTSU_2nd_coefficient_create()	2nd coefficient correction reference data calculation function
correction_CTSU_16point_coefficient_create()	1.2uA - 19.2uA correction coefficient calculation function
correction_sensor_magnification_set()	Measurement time-based correction coefficient magnification setting function
correction_sensor_cnt_create()	Measurement value correction creation function
R_Set_Cap_Touch_Create()	Capacitance touch creation function
R_Set_CTSU_Correction_Mode()	ICO correction measurement mode setting function

3.5 Function Specifications

Function specifications for ICO correction are described below.

3.5.1 main

main()

Outline	System main function
Declaration	void main(void)
Description	This function controls the overall system.
Call Function	R_Set_Cap_Touch_Create() R_CMT0_Create() R_CMT0_Start() R_Set_Cap_Touch_Measurement_Start() *1 R_Get_Cap_Touch_Data_Check() *1 R_Get_Cap_Touch_Initial_Status() *1 R_Set_Cap_Touch_Result_Create() *1 R_Get_Cap_Touch_Result() *1 R_Set_Cap_Touch_Initial_Tuning() *1 R_Set_Cap_Touch_Next_Method_Change() *1
Argument	- -
Return Value	-

*1 Refer to Touch API Reference Guide.

3.5.2 R_Set_Cap_Touch_Create

R_Set_Cap_Touch_Create()

Outline	Capacitance touch creation function
Declaration	void R_Set_Cap_Touch_Create(void)
Description	This function is called in the main function to implement settings to measure capacitance touch.
Call Function	correction_CTSU_sensor_ico() touch_parameter_address_set() CTSU_register_initial_value_ram_set() DTC_Set_Initial_of_CTSU() initial_port_set() R_Set_CTSU_All_Register() DTC_transmit_data_set() touch_parameter_set() R_Set_Cap_Touch_Tuning_Cntrol() R_Set_Cap_Touch_Offset_Timing()
Argument	- -
Return Value	-

3.5.3 correcton_CTSU_sensor_ico

correcton_CTSU_sensor_ico()

Outline	Sensor ICO correction function
Declaration	uint8_t correcton_CTSU_sensor_ico(void)
Description	This function is called in the R_Set_Cap_Touch_Create function to implement sensor ICO correction.
Call Function	correction_CTSU_sensor_selection() correction_CTSU_register_parameter_set() correction_CTSU_register_txd_set() DTC_Set_Initial_of_CTSU() correction_CTSU_register_set() R_Set_CTSU_Correction_Mode() correction_CTSU_measurement_start() correction_CTSU_1st_coefficient_create() correction_CTSU_2nd_standard_val_create() correction_CTSU_2nd_coefficient_create() correction_CTSU_16point_coefficient_create()
Argument	- -
Return Value	0x00: Normal completion (success) 0x01: Abnormal completion (error)

3.5.4 correction_CTSU_sensor_selection

correction_CTSU_sensor_selection()

Outline	ICO correction sensor selection function
Declaration	uint8_t correction_CTSU_sensor_selection(void)
Description	This function is called in the correction_CTSU_sensor_ico function and selects the sensor for correction. The function selects the sensor with the lowest number among all sensors used by the application.
Call Function	-
Argument	- -
Return Value	0x00: Normal completion (Sensor number for correction selected) 0x01: Abnormal completion (Sensor number for correction not selected)

3.5.5 correction_CTSU_register_parameter_set

correction_CTSU_register_parameter_set()

Outline: ICO correction CTSU register value setting function

Declaration: void correction_CTSU_register_parameter_set(void)

Description: This function is called in the correction_CTSU_sensor_ico to implement the correction CTSU register setting. The contents of the setting are lists below.

CTSU Control Register 0 (CTSUCR0)		
Symbol	Set Value	Description
CTSUSNZ	0	Wait period low power consumption function disabled.
CTSUTXVSEL	0	Selects Vcc

CTSU Control Register 1 (CTSUCR1)		
Symbol	Set Value	Description
CTSUPON	1	Power ON
CTSUCSW	1	Capacitance switch ON
CTSUATUNE0	0	Normal operation mode
CTSUATUNE1	0	Normal output
CTSUCLK	0	PCLK (no division)
CTSUMD	1	Auto-capacitance multi-scan mode

CTSU Synchronous Noise Reduction Setting Register (CTSUSDPRS)		
Symbol	Set Value	Description
CTSUPRRATIO	3	Recommended value
CTSUPRMODE	2	62 pulse (recommended value)
CTSUOFF	0	High-frequency noise reduction function ON

CTSU Sensor Stabilization Wait Period Register (CTSUSST)		
Symbol	Set Value	Description
CTSUSST	16	Fixed value

CTSU High-Pass Noise Reduction Control Register (CTSUDCLKC)		
Symbol	Set Value	Description
CTSUSSMOD	0	Fixed value
CTSUSSCNT	3	Fixed value

Call Function: -

Argument: - -

Return Value: -

3.5.6 correction_CTSU_register_txd_set

correction_CTSU_register_txd_set()

Outline ICO correction transfer measurement parameter setting function

Declaration `void correction_CTSU_register_txd_set(void)`

Description This function is called by the correction_CTSU_sensor_ico function, and based on the operation clock, it sets the capacitance measurement frequency (switched capacity filter frequency) to 0.5MHz and the measurement time to 0.5ms according to the operation clock. It also sets the current offset setting to 0.

CTSU High-frequency noise reduction spectrum dispersion control register (CTSUSSC) ＝ 0x0700
CTSU sensor offset register 0 (CTSUSO0) = 0x0000

Operation Clock	CTSU Sensor Offset Register 1 (CTSUSO1)		
	CTSURICOA	CTSUSDPA	CTSUICOG
32MHz	15	31 (divide by 64)	1
27MHz		26 (divide by 54)	
24MHz		23 (divide by 48)	
16MHz		15 (divide by 32)	
8MHz		7 (divide by 16)	

Call Function -
Argument - -
Return Value -

3.5.7 DTC_Set_Initial_of_CTSU

DTC_Set_Initial_of_CTSU()

Outline CTSU DTC initialization function

Declaration `void DTC_Set_Initial_of_CTSU(void)`

Description This function is called by the correction_CTSU_sensor_ico function to initialize the CTSU data transmission.

Call Function -
Argument - -
Return Value -

3.5.8 correction_CTSU_register_set

correction_CTSU_register_set()

Outline	ICO correction CTSU register setting function
Declaration	uint16_t correction_CTSU_register_set(void)
Description	This function is called in the correction_CTSU_sensor_ico to implement the CTSU register correction settings. Based on the correction settings, the following register setting functions are called.
Call Function	R_Set_CTSU_TSCAP_Discharge() MPC_CTSU_set() R_Set_CTSU_Module_Operation() R_Set_CTSU_Transmit_Power_Supply() R_Set_CTSU_Power_Operation_Mode() R_Set_CTSU_Power_Capacity_Adjustment() R_Set_CTSU_Operation_Clock() R_Set_CTSU_Sensor_Stabilization_Wait_Time() R_Set_CTSU_Power_Supply() R_Set_CTSU_Measurement_Mode() R_Set_CTSU_Synchronous_Noise_Reduction() R_Set_CTSU_Measurement_Channel() R_Set_CTSU_Channel_Enable0() R_Set_CTSU_Channel_Transmit_Receive0() R_Set_CTSU_High_Pass_Noise_Reduction()
Argument	- -
Return Value	0x0000: Normal completion 0x0001-0xFFFF: Abnormal completion

3.5.9 R_Set_CTSU_Correction_Mode

R_Set_CTSU_Correction_Mode()

Outline	ICO correction measurement mode setting function
Declaration	void R_Set_CTSU_Correction_Mode(void)
Description	This function is called in the correction_CTSU_sensor_ico function to set the following, which in turns implements the correction measurement mode setting.

CTSU Error Status Register (CTSUERRS)		
Symbol	Setting Value	Setting Description
CTSUTSOC	1	Calibration setting
CTSUSPMD	2	Calibration mode

Call Function	-
Argument	- -
Return Value	-

3.5.10 correction_CTSU_measurement_start

correction_CTSU_measurement_start()

Outline	ICO correction measurement start function	
Declaration	void correction_CTSU_measurement_start(void)	
Description	This function is called in the correcton_CTSU_sensor_ico function and takes correction measurements 32 times. The 32 measurement values are stored in global variables g_correction_1st_val and g_correction_2nd_val.	
Call Function	-	
Argument	-	-
Return Value	-	

3.5.11 correction_CTSU_1st_coefficient_create

correction_CTSU_1st_coefficient_create()

Outline	1st magnification correction reference data calculation function	
Declaration	void correction_CTSU_1st_coefficient_create(void)	
Description	This function is called in the correction_CTSU_sensor_ico function and calculates the 1st (19.2uA) magnification corresponding to the correction reference data. 1st magnification (g_correction_1st_coefficient) = $\frac{\text{1st value (g\_correction\_1st\_val)}}{\text{19.2uA correction reference data}}$ The results of the calculation are stored in global variable g_correction_1st_coefficient.	
Call Function	-	
Argument	-	-
Return Value	-	

3.5.12 correction_CTSU_2nd_standard_val_create

correction_CTSU_2nd_standard_val_create()

Outline	2nd correction reference data calculation function
Declaration	void correction_CTSU_2nd_standard_val_create(void)
Description	This function is called in the correction_CTSU_sensor_ico function and calculates the 2nd correction reference data. 2nd correction reference data = $\frac{(1stCTSUTRMR\ value+273)}{528} \times$ 19.2uA correction reference data The results of the calculation are stored in global variable g_correction_2nd_std_val.
Call Function	-
Argument	- -
Return Value	-

3.5.13 correction_CTSU_2nd_coefficient_create

correction_CTSU_2nd_coefficient_create()

Outline	Correction reference data 2nd magnification calculation function
Declaration	void correction_CTSU_2nd_coefficient_create(void)
Description	This function is called in the correction_CTSU_sensor_ico function and calculates the 2nd magnification corresponding to the 2nd correction reference data. 2nd magnification (g_correction_2nd_coefficient) = $\frac{2nd\ value}{g\_correction\_2nd\_std\_val}$ The results of the calculation are stored in global variable g_correction_2nd_coefficient.
Call Function	-
Argument	- -
Return Value	-

3.5.14 correction_CTSU_16point_coefficient_create

correction_CTSU_16point_coefficient_create()

Outline	1.2uA - 19.2uA correction coefficient calculation function
Declaration	void correction_CTSU_16point_coefficient_create(void)
Description	This function is called in the correction_CTSU_sensor_ico function and calculates the correction coefficient for 1.2uA to 19.2uA. The results of the calculation are stored in global variable g_correction_16coefficient[ ].
Call Function	-
Argument	- -
Return Value	-

3.5.15 correction_sensor_magnification_set

correction_sensor_magnification_set()

Outline	Measurement time-based correction coefficient magnification setting function
Declaration	void correction_sensor_magnification_set(void)
Description	This function is called in the R_Set_CTSU_Measurement_Start to set the magnification corresponding to the current touch measurement period as the touch measurement period 0.5ms correction coefficient. The current touch measurement period is determined based on the PCLKB supply clock, measurement frequency, and number of measurements, and the magnification is set accordingly.
Call Function	–
Argument	- -
Return Value	-

3.5.16 correction_sensor_cnt_create

correction_sensor_cnt_create()

Outline	Correction sensor count create function		
Declaration	uint8_t correction_sensor_cnt_create(uint8_t method, uint8_t ts_num, uint8_t number)		
Description	This function is called in the touch_data_moving_average function and multiplies the measured value by the correction coefficient and sets the corrected measurement value.		
Call Function	–		
Argument	uint8_t	method	0-9: default measurement method number
Return Value	uint8_t	ts_num	Measurement TS number
Outline	uint8_t	number	Measurement pointer number
Declaration	Correction measurement value : 0-65535		

4. Flowchart

4.1 Main function: main()

Figure4.1 shows the flowchart for the main function.

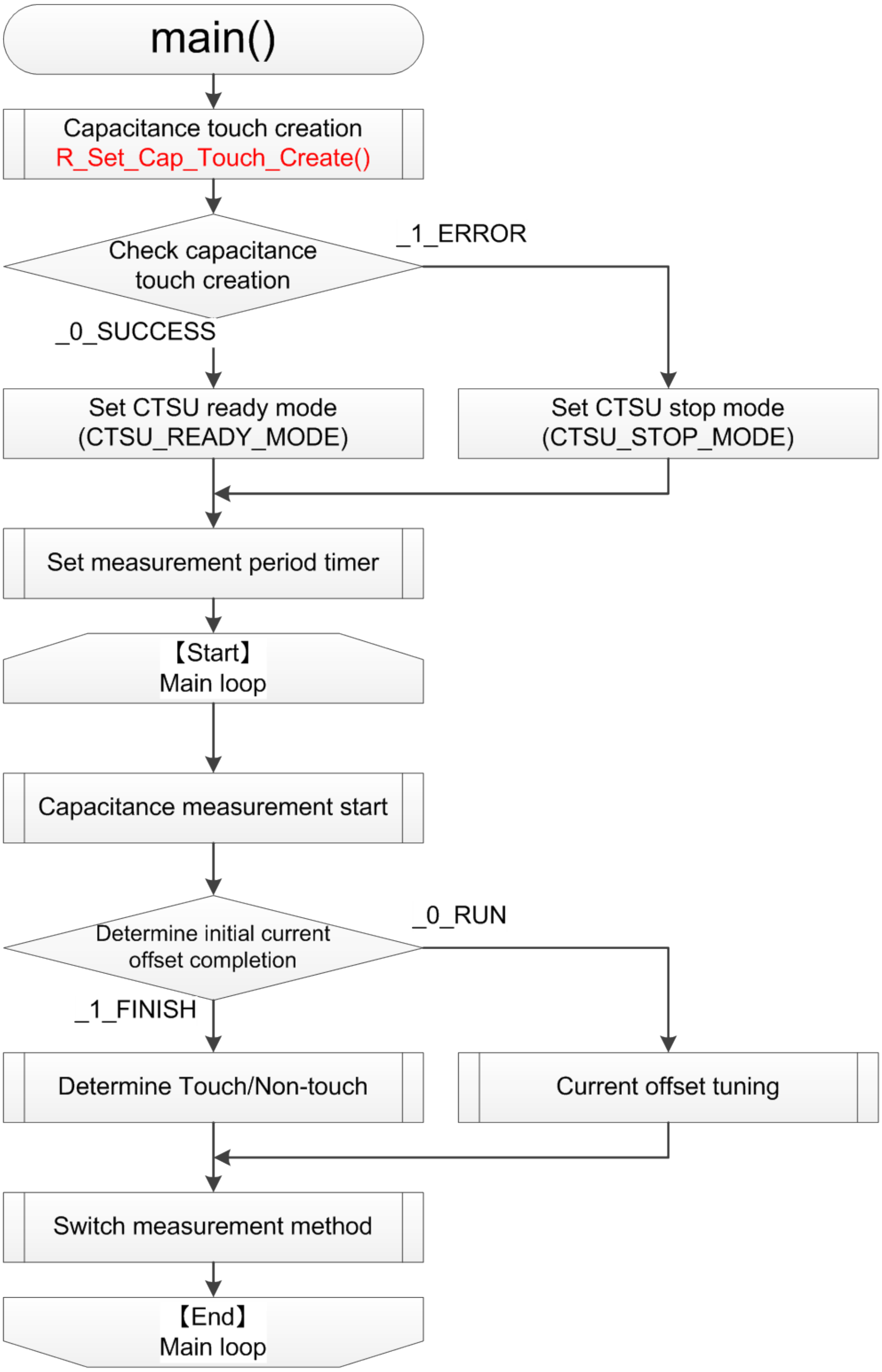

Figure4.1 main()

4.2 Capacity Touch Creation Function: R_Set_Cap_Touch_Create()

Figure 4.2 shows the flowchart for the capacity touch creation function.

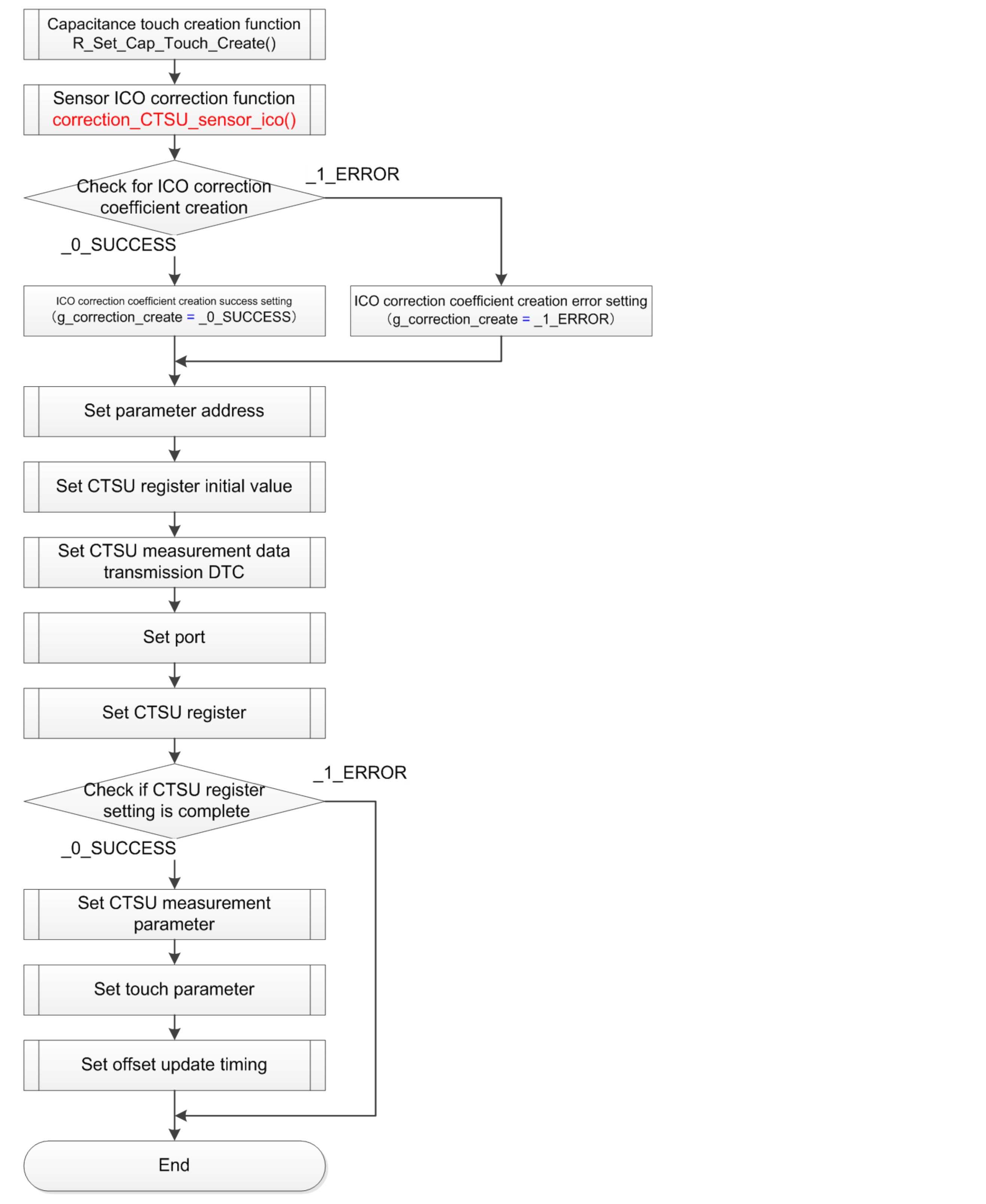

Figure 4.2 R_Set_Cap_Touch_Create()

4.3 Sensor ICO Correction Function: correction_CTSU_sensor_ico()

Figure 4.3 shows the flowchart for the capacity touch creation function.

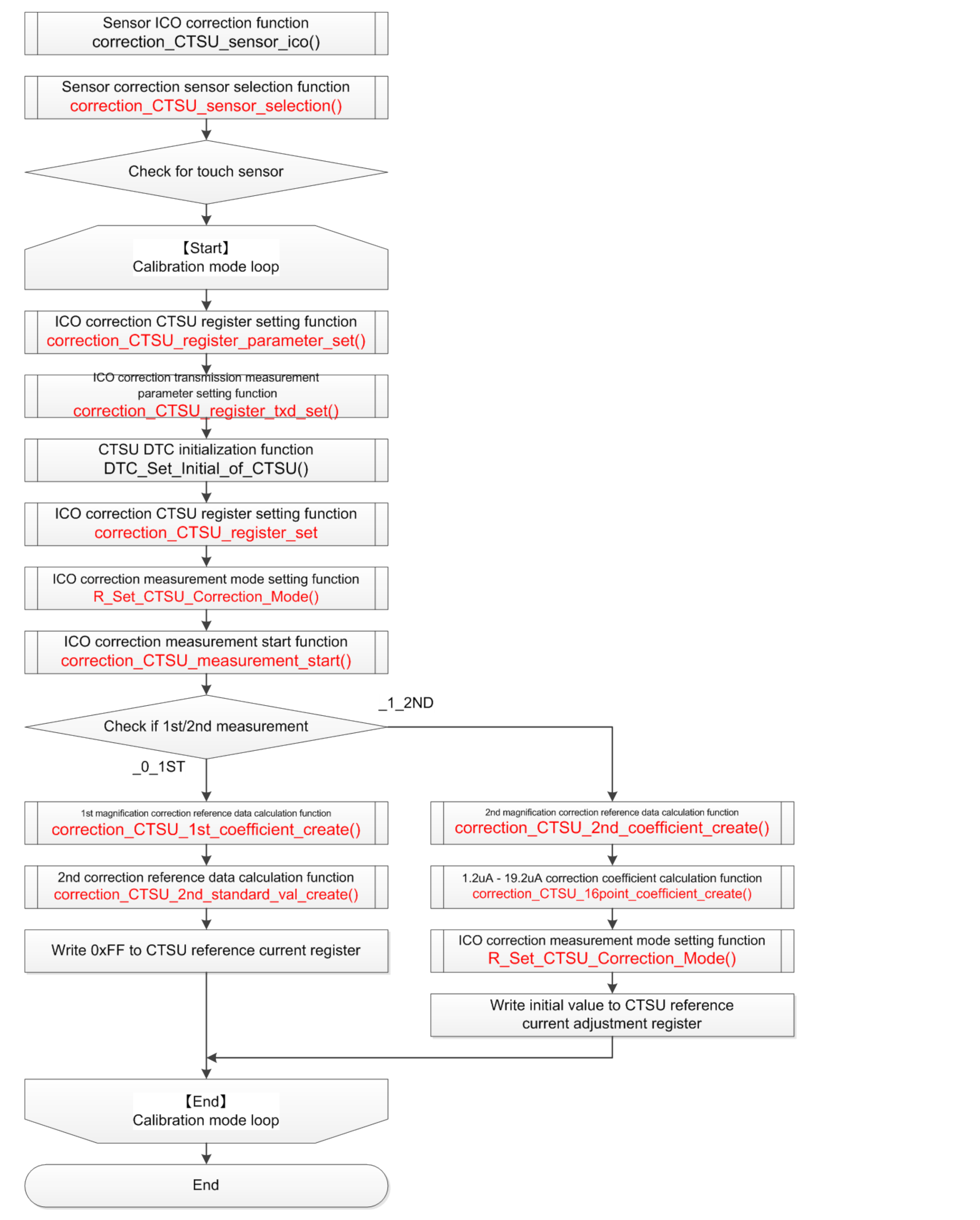

Figure 4.3 correction_CTSU_sensor_ico()

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jan. 18, 2017		Initial version

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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 6.0