EPSON
EXCEED YOUR VISION

# S1C17W22/S1C17W23
# Touch Key
# Application Note

SEIKO EPSON CORPORATION

Rev.1.0

## Evaluation board/kit and Development tool important notice

1. This evaluation board/kit or development tool is designed for use for engineering evaluation, demonstration, or development purposes only. Do not use it for other purposes. It is not intended to meet the requirements of design for finished products.
2. This evaluation board/kit or development tool is intended for use by an electronics engineer and is not a consumer product. The user should use it properly and in a safe manner. Seiko Epson dose not assume any responsibility or liability of any kind of damage and/or fire coursed by the use of it. The user should cease to use it when any abnormal issue occurs even during proper and safe use.
3. The part used for this evaluation board/kit or development tool may be changed without any notice.

## NOTICE

## Summary

This is reference material for the S1C17W22 or S1C17W23 control program, which detects variations of touch key capacitance using the I/O port (PPORT), universal port multiplexer (UPMUX), and 16-bit PWM timer (T16B) in an interval configured with the 16-bit timer (T16), and sends the equivalent value of the detected capacitance via the UART or displays the key status on the LCD panel.

## Operating Environment

- S5U1C17W23T (hereinafter referred to as SVT17W23: Software Evaluation Tool for S1C17W23)
  Two dedicated cables (4 pins to 4 pins) are required to connect with ICDmini.
- ICDmini (S5U1C17001H)
  A USB cable is required to connect with a PC.
- PC
  - With GNU17 (S5U1C17001C) development tool installed *
  - With ICDmini USB driver installed
- Latest version FLS17W22 (file name: fls17w22.elf) or FLS17W23 (file name: fls17w23.elf)
  This file is mandatory for programming the embedded FLASH.
- S1C17W22/S1C17W23 Touch Key Programming Package (this package)
  - Touch Key Capacitance Variation Visualization Programming Package (s1c17w22_w23_touch_oscillo), Excel file with VBA macro included (MeasTouch.xlsm), and Active-X control file (NonComSck.ocx)
  - Touch Key Status LCD Display Programming Package (s1c17w22_w23_touch_demo)

* GNU17 V2.3.0 is used for operation checking of this package.

# Table of Contents

# 1. Specifications

The program described in this application note detects variations of touch key capacitance using the I/O port (PPORT), universal port multiplexer (UPMUX), and 16-bit PWM timer (T16B) embedded in the S1C17W22/S1C17W23.

The sample program s1c17w22_w23_touch_oscillo samples the counter values equivalent to the touch key capacitance variations in the set intervals and sends them to the PC via the UART to write to an Excel sheet in conjunction with a VBA macro. The values in the array written to the Excel sheet are graphed, this makes it possible to visualize the changes of the capacitance equivalent values on the time series.

The sample program s1c17w22_w23_touch_demo scans the touch key status periodically using the 16-bit timer (T16) and displays it on the LCD panel.

## 2. Descriptions of the Functions Used

| | |
|---|---|
| UPMUX | Used to switch over the assignment of touch key pins between the PPORT outputs and the T16B Ch.0 capture inputs. |
| PPORT | P0[7:0], P1[2:0], P1[7:5], and P2[1:0]: These 16 ports are assigned for the touch keys and configured to L level output, H level output, or Hi-Z. |
| T16B Ch.0 | Used to determine the touch key capacitance equivalent values from the counter values in capture mode. |
| T16 Ch.0 | Used as an interval timer to scan the touch key status periodically. |

The peripheral circuit shown below is used in the s1c17w22_w23_touch_oscillo program.

| | |
|---|---|
| UART Ch.0 | USIN0 and USOUT0 are assigned to P36 and P37 of PPORT, respectively, using UPMUX. They are used to communicate with the Excel VBA macro executed on the PC. |

The two peripheral circuits shown below are used in the s1c17w22_w23_touch_demo program.

| | |
|---|---|
| LCD24 | Used to display touch key status on the LCD panel. |
| SNDA | Used to output the touch key click sound. |

Operating mode — T16B Ch.0: Configured to capture mode. The capture 0 and 1 interrupts are used to count the touch key capacitance equivalent value.

System clock — OSC3 (4 MHz internal oscillator) is used as the system clock.
OSC1 (32.768 kHz) is also used in s1c17w22_w23_touch_demo.

Interrupts — The following shows the T16B Ch.0 vector number and vector address:

T16B Ch.0 Vector number: 14 (0x0e)
Vector address: 0x8038

The sample program uses the following two interrupts:

Capture 0 interrupt
Capture 1 interrupt

The following shows the T16 Ch.0 vector number and vector address:

T16 Ch.0 Vector number: 9 (0x09)
Vector address: 0x8024

The sample program uses the following interrupt:

Underflow interrupt

The following description is applied to s1c17w22_w23_touch_oscillo.
The following shows the UART Ch.0 vector number and vector address:

UART Ch.0 Vector number: 10 (0x0a)
Vector address: 0x8028

The sample program uses the following interrupt:

Receive buffer one byte full interrupt

Figures 2-1 and 2-2 show the configurations of UPMUX and T16B, respectively.

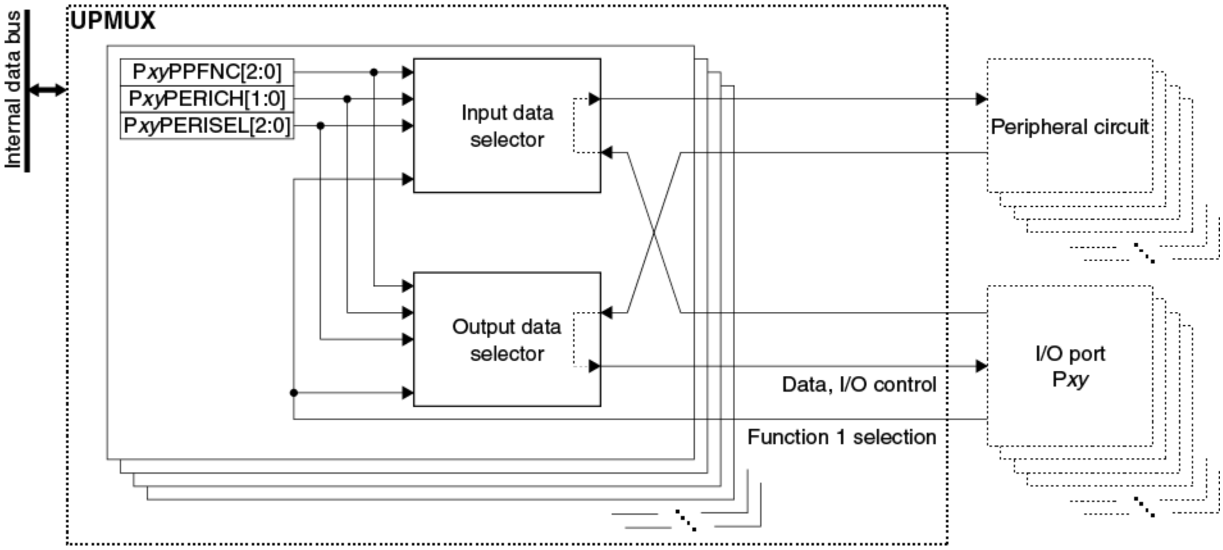



Figure 2-1 UPMUX Configuration

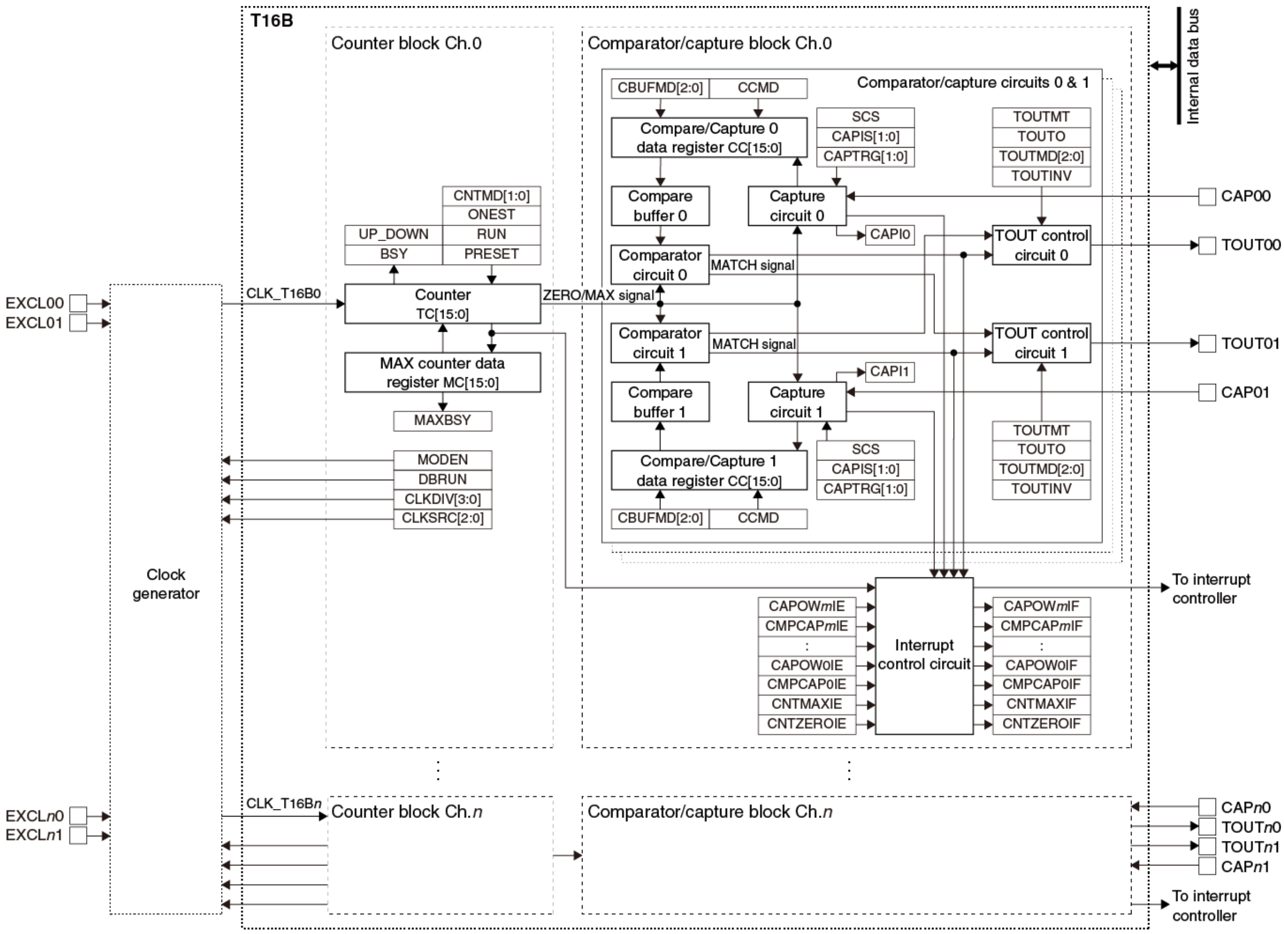



Figure 2-2 T16B Configuration

# 3. Principle of Operation

## 3.1 Principle of Detection

A capacitor with a certain capacitance is formed between two conductors isolated from each other with an insulator (dielectric) such as air or plastic. When an electrode for a sensor pad and ground patterns are placed on a printed circuit board, for example, as the structure shown in Figure 3-1, two capacitors C1 and C2 are formed. In this case, capacitance (C1 + C2) when both capacitors are connected in parallel is obtained between the sensor pad and the ground patterns.

Furthermore, the human body is a good conductor and a large capacitance is generated between the body and ground even if it is isolated from the ground with a high insulative material such as shoes as shown by the fact that static electricity is charged on the body. Therefore, a proximity effect occurs when the body comes near the sensor pad. In other words, touching the dielectric surface of the sensor pad with a finger forms the capacitor C3 on the sensor pad anew and the sensor pad capacitance is increased to (C1 + C2 + C3).

By detecting this capacitance variation, a touch sensor can be configured.

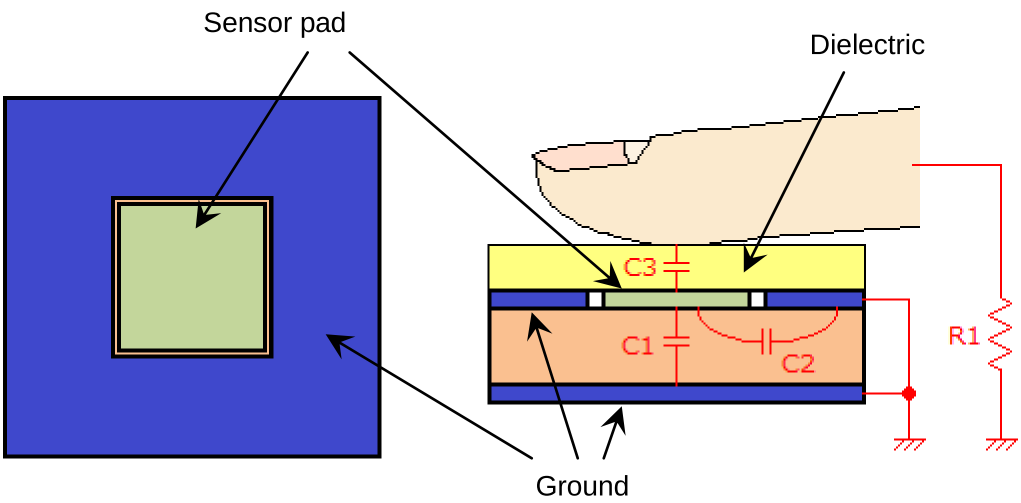



Figure 3-1 Sensor Pad Capacitances (Top View and Section View)

## 3.2 Method of Detection

This section explains the method of detection using Figure 3-2, Capacitance Detection Circuit Conceptual Diagram, and Figure 3-3, Temporal Change of Detection Waveform.

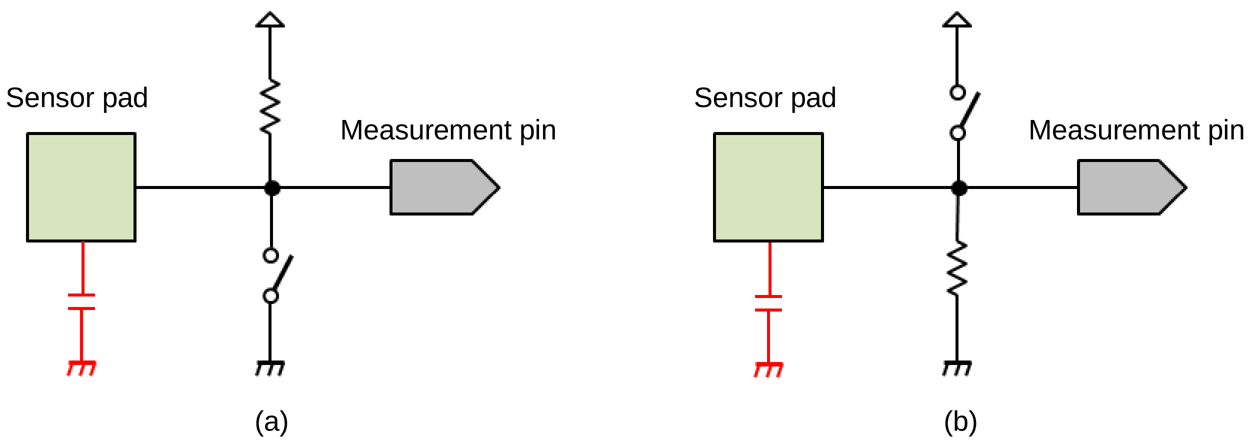



Figure 3-2 Capacitance Detection Circuit Conceptual Diagram

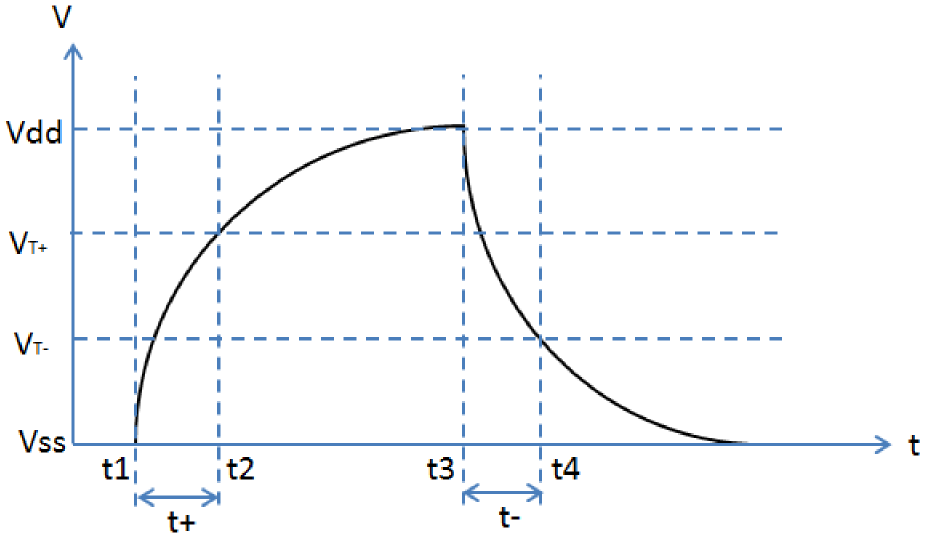



Figure 3-3 Temporal Change of Detection Waveform

There is a capacitive component between the sensor pad and ground shown in Figure 3-2. First, the switch (a) in Figure 3-2 is turned on for a fixed time to discharge the capacitive component completely, and then the switch is turned off. Time t1 in Figure 3-3 indicates that the switch is turned off. After that, the capacitive component is charged through the resistor and the potential at the measurement pin is increased with the time constant $\tau$ = CR. A CMOS Schmitt input circuit, in which a voltage equal to or lower than $V_{T-}$ is assumed as L level and a voltage equal to or higher than $V_{T+}$ as H level, is used as the measurement input port in this example. Therefore, the CMOS Schmitt input circuit determines that the input signal is L level until t2 at which the voltage reaches $V_{T+}$. After that, it determines that the input signal is H level. By measuring the time t+ between t1 and t2, the sensor pad capacitance value can be obtained.

The capacitance can also be measured by making a change to reverse potential.

The switch (b) in Figure 3-2 is turned on for a fixed time to charge the sensor pad capacitive component until the potential reaches Vdd, and then the switch is turned off. Time t3 in Figure 3-3 indicates that the switch is turned off. After that, the capacitive component is discharged through the resistor and the potential at the measurement pin is decreased with the time constant $\tau$ = CR. The CMOS Schmitt input circuit determines that the input signal is H level until t4 at which the voltage reaches $V_{T-}$. After that, it determines that the input signal is L level. By measuring the time t- between t3 and t4, the sensor pad capacitance value can also be obtained.

In this application note, T16B is used to count the times t+ and t-, and PPORT as the switches in Figure 3-2. Turning the switch (a) in Figure 3-2 on and off is realized by setting the PPORT to L level output and Hi-Z, respectively. Similarly, turning the switch (b) on and off is realized by setting the PPORT to H level output and Hi-Z, respectively. T16B is used to measure the capacitance equivalent values by assigning a sensor pad output to the T16B capture trigger input using UPMUX and capturing the counter value.

Refer to Figure 3-4 in the next section that shows the PPORT, which becomes active while the sensor pad being touched with a finger is detected, with double line frame, and the T16B assignment status by UPMUX as a switch on symbol. It will help you to understand these operations.

## 3.3 Detection Circuit

Figure 3-4 shows an example of the detection circuit that consists of the PPORT, UPMUX, and T16B embedded in the S1C17W22/S1C17W23.

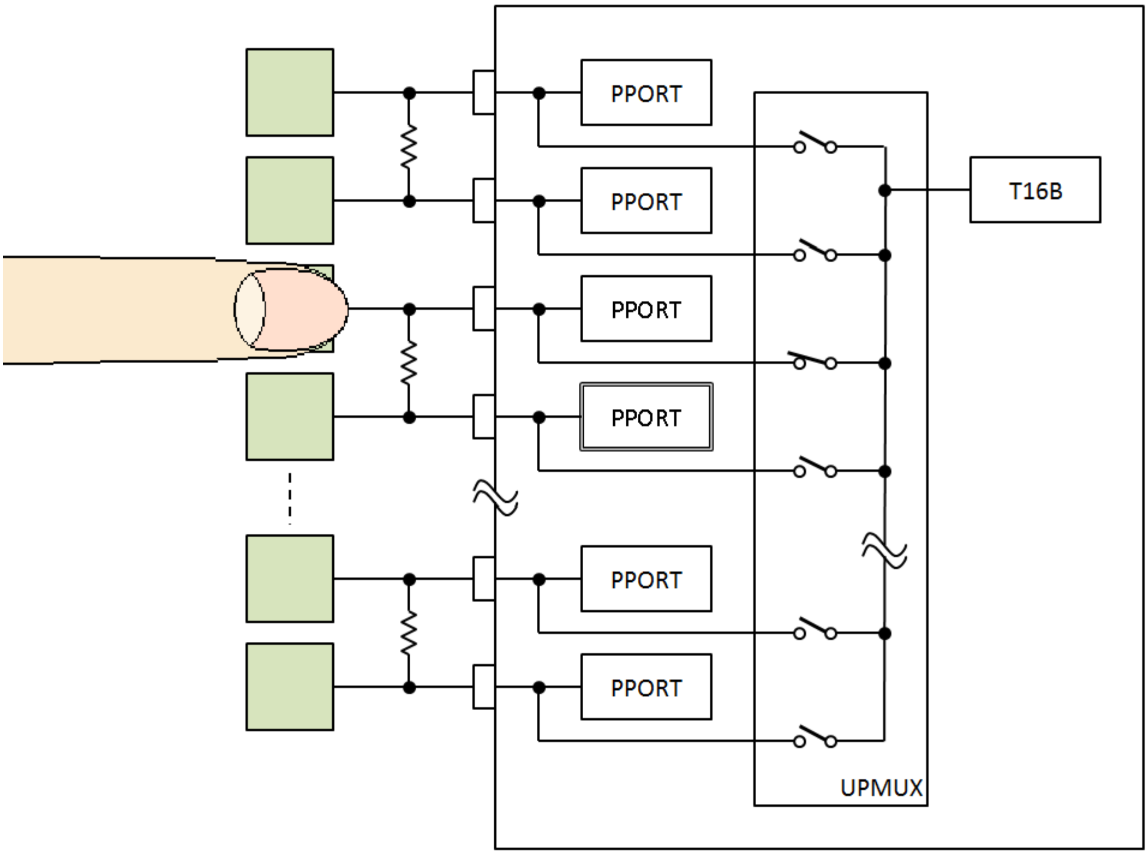



Figure 3-4 Detection Circuit Example

The touch key is formed in a pair and a resistor is attached between the keys. The resistance value can be determined as shown below by taking the PPORT leakage current range ($I_{LEAK}$ = ±150 nA) of the S1C17 Series microcontrollers and the average Schmitt input threshold voltage ($V_{T+}$ = 0.7Vdd, $V_{T-}$ = 0.3Vdd) into account.

If assuming that the touch key has a 12 mm x 12 mm square shape, an FR-4 glass epoxy board having the thickness d = 1.2 mm is used as the printed circuit board, and the touch key is a simple parallel plate capacitor, the capacitance C is calculated in the equation shown below by setting the parameters as footprint S = 144 mm$^2$, specific inductive capacity $\varepsilon_r$ = 4.7, dielectric constant under a vacuum $\varepsilon_0$ = 8.855 x $10^{-12}$ F/m.

$$C = \frac{\varepsilon_r \varepsilon_0 S}{d} = \frac{4.7 \times 8.855 \times 10^{-12} \times 144 \times 10^{-6}}{1.2 \times 10^{-3}} = 5.0\text{pF}$$

The following calculations are performed by focusing that the touch key voltage changes from Vss to $V_{T+}$ within the period t+ between t1 and t2 as shown in Figure 3-3.

As the equation below shows, the voltage V of the detection waveform changes according to the elapsed time t from t1 due to the effect of the time constant CR found from the touch key capacitance C and the resistance value R.

$$V = Vdd \times \left(1 - \exp\left(-\frac{1}{CR} \cdot t\right)\right)$$

The above equation can be converted as follows:

$$R = -\frac{t}{\ln\left(1 - \frac{V}{Vdd}\right)} \cdot \frac{1}{C}$$

When time t+ has elapsed, the voltage changes to $V = V_{T+} = 0.7Vdd$.

$$R = -\frac{t+}{\ln\left(1 - \frac{0.7 \cdot Vdd}{Vdd}\right)} \cdot \frac{1}{5.0 \times 10^{-12}}$$

$$= 1.66 \times 10^{11} \cdot t+$$

If t+ is substituted with 100 cycles of the 4 MHz clock input to T16B,

$$R = 1.66 \times 10^{11} \cdot t+$$

$$= 1.66 \times 10^{11} \cdot \frac{100}{4 \times 10^{6}}$$

$$= 4.15M\Omega$$

Leakage current of the S1C17 Series microcontrollers is within the range of $I_{LEAK} = \pm 150$ nA. If this value is converted into a resistance value at Vdd = 3.3 V,

$$R = \frac{V}{I} = \frac{3.3}{150 \times 10^{-9}}$$

$$= 22M\Omega$$

Therefore, 4.15 MΩ calculated as above is sufficiently smaller resistance than that of leakage current and you may judge that this sensing method is practical.

Although the calculated R value is 4.15 MΩ, an E24 series resistance value, such as 4.7 MΩ, 5.1 MΩ, or 5.6 MΩ, is suited to practical use as they can easily be obtained.

However, the actual C value is larger than 5.0 pF since a capacitance resulting from routing of wiring is added even though the board pattern is designed so that the capacitance will be smaller than the value calculated as a parallel plate capacitor (refer to the Board Design chapter for more information). Therefore, the T16B counter value in untouched status will normally be larger than 100.

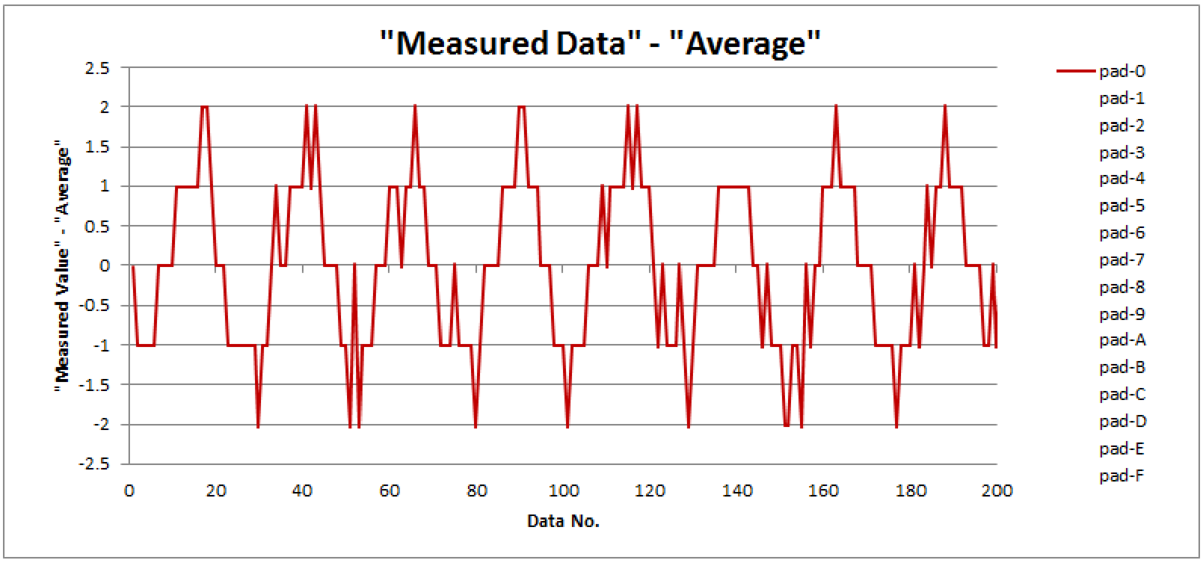
Furthermore, the same results can be obtained if the above calculations are performed by focusing that the touch key voltage changes from Vdd to $V_{T-}$ within the period t- between t3 and t4 as shown in Figure 3-3.

# 4. Suppressing Influence Caused by Noise

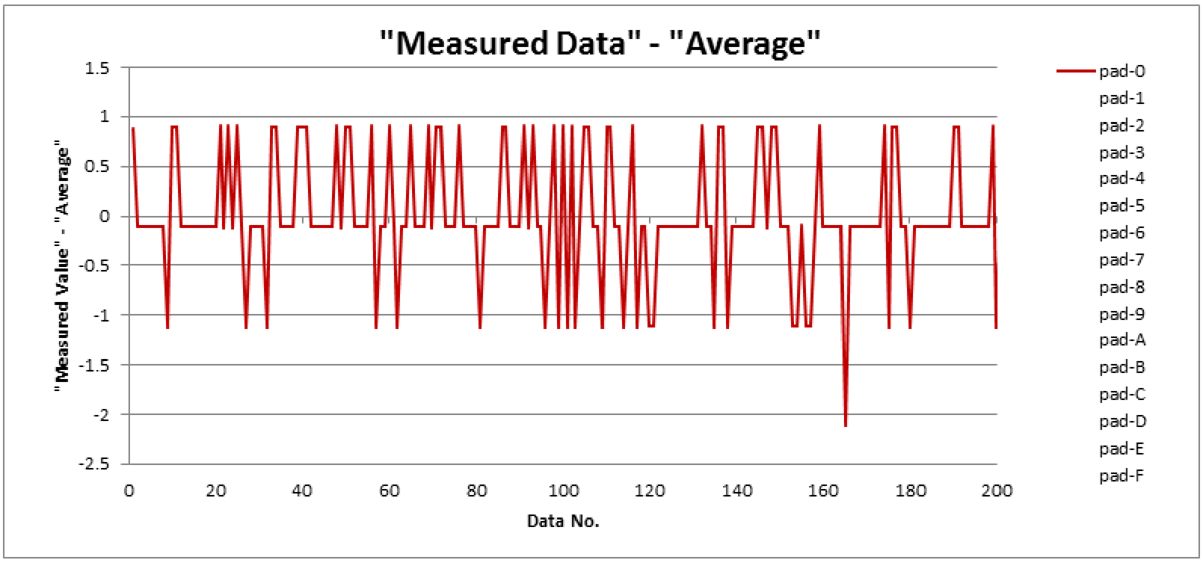
As shown with the detection waveform in Figure 3-3, the touch key capacitance equivalent values can be obtained from one of the two measurement periods t+ and t-. Using both results measured in a very short time suppresses influence caused by noise.

Figure 4-1 (a) shows a plot differences between 200 data sampled in 50 ms intervals and the average of the capacitance equivalent values t+. These data were sampled in a region that uses a 60 Hz commercial AC power line and abnormal waveforms appear about every 25 sampling cycles. This cycle is 1.25 s (= sampling cycle 50 ms x 25 points), therefore, it might be noise by interference with the sampling cycle, not direct influence by 60 Hz AC line.

On the other hand, Figure 4-1 (b) shows a plot differences between 200 data sampled in 50 ms intervals and the average of the sum of the capacitance equivalent values t+ and t-. In this case, about ±2 level of cyclic noise appears in Figure 4-1 (a) are falls within about ±1 level and no waveforms having a specific cycle are observed. In other words, by adding the capacitance equivalent values t+ and t-, influence caused by noise, especially from a commercial AC power line that is the largest noise source, can be suppressed to some extent.



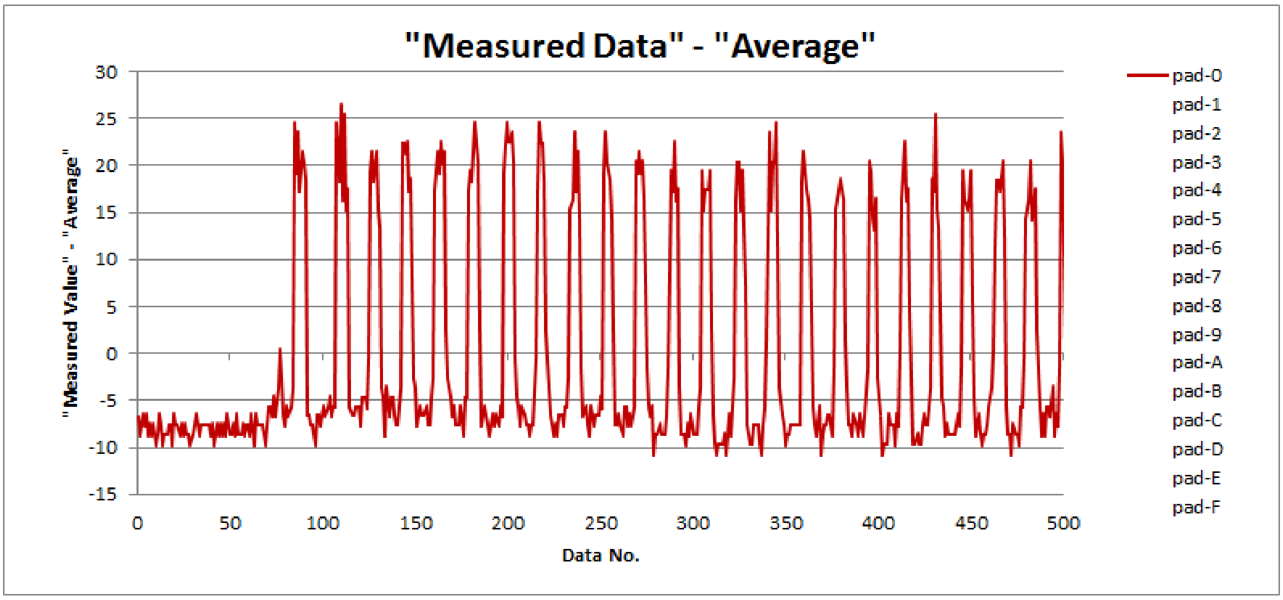

(a) t+



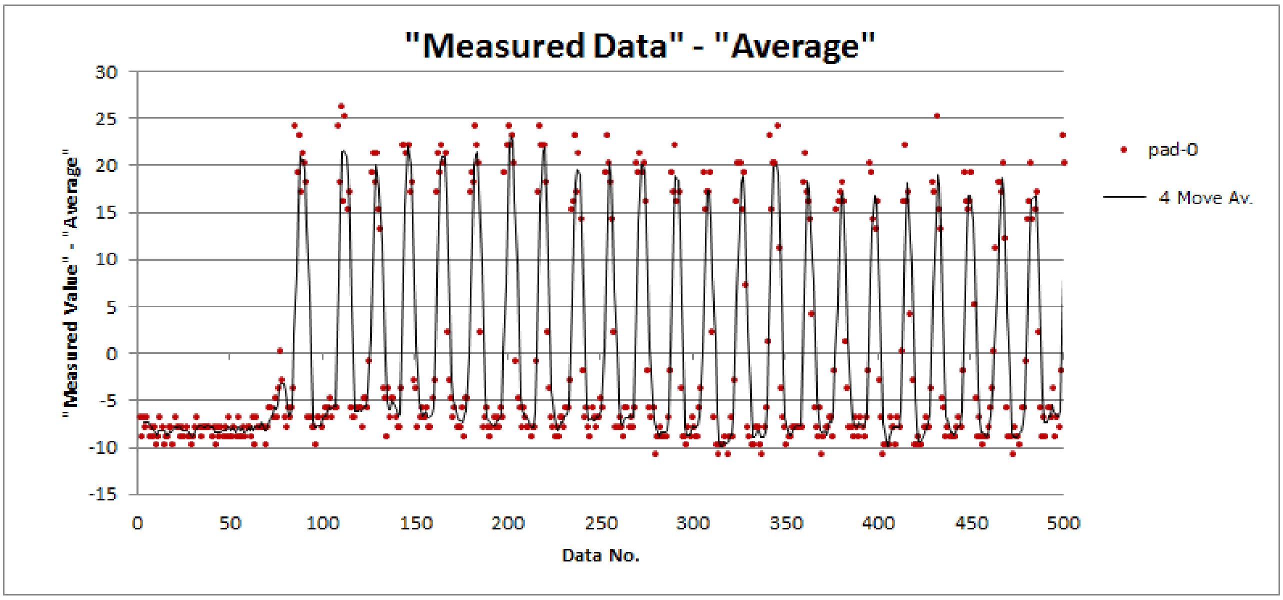

(b) Sum of t+ and t- values

Figure 4-1 Plot Differences Between 200 Data Sampled in 50 ms Intervals and Average of Capacitance Equivalent Values

Figure 4-2 (a) shows a plot differences between 500 data sampled in 25 ms intervals and the average of the capacitance equivalent values when a touch key, which has a 12 mm x 12 mm square shape and an acrylic plate 4 mm thick on the surface, is touched repeatedly with a finger. Although this enables you to determine whether the key is being touched or not, it can be checked more clearly by simply performing moving average processing four times as shown in Figure 4-2 (b).




(a) Raw data




(b) Moving average processing executed four times

Figure 4-2 Plot Differences Between 500 Data Sampled in 25 ms Intervals and Average of Capacitance Equivalent Values

# 5. Board Design

Figure 5-1 shows a touch key print circuit board pattern image.

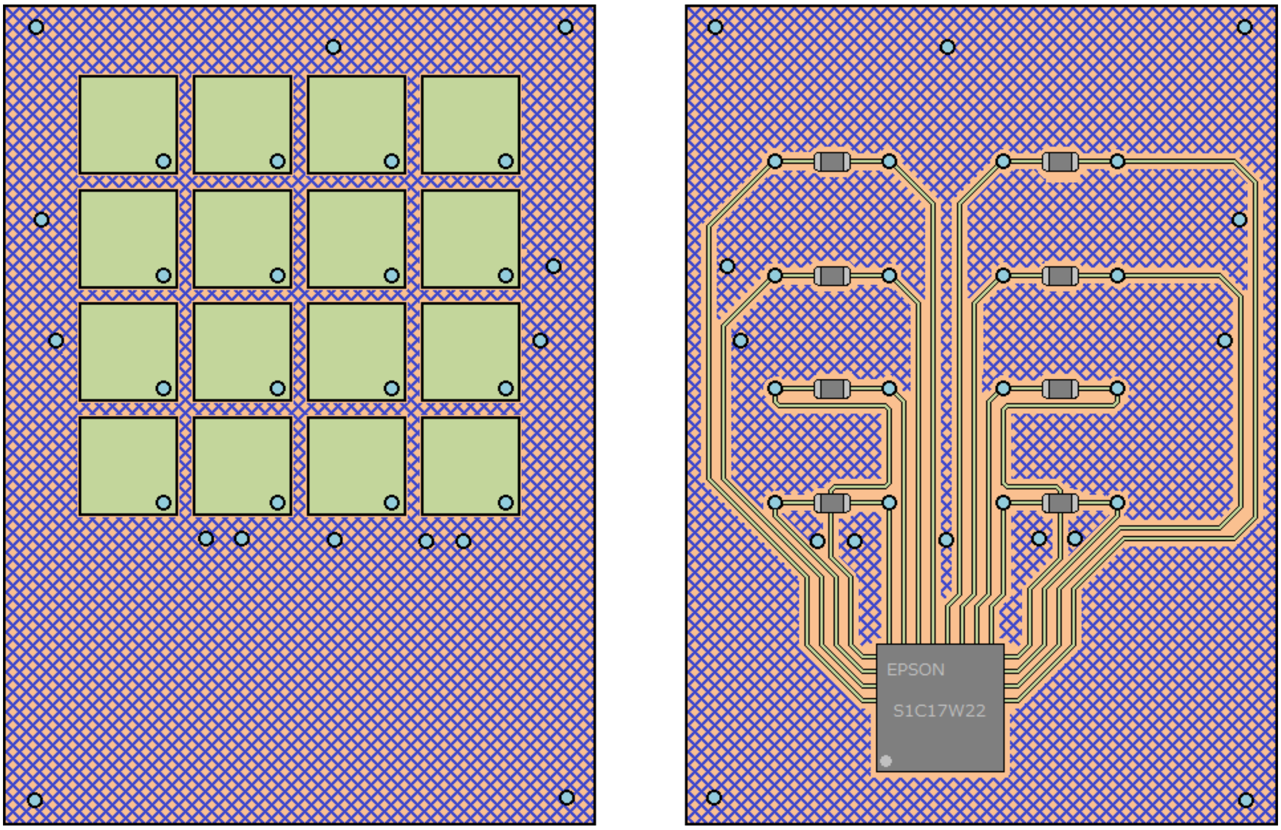



Figure 5-1 Touch Key Print Circuit Board Pattern Image (Front and Back)

The touch key pattern is typically designed between 10 mm x 10 mm and 20 mm x 20 mm taking human finger size into account (any shape can be used, e.g., circle, square, or other). However, since the touch key electrode will be used at very high input impedance, it must be shielded with reliable ground patterns. Furthermore, if a solid ground pattern is printed on the back of the board, horizontal direction of the board aside, the capacitance in a vertical direction (the front of the board from/to the back) increases in untouched status and it causes a lowering of detection sensitivity. To avoid this problem, the ground should be designed as a mesh-shaped pattern with thinner traces as much as possible as shown in Figure 5-1 (drawn in blue). In addition, it is desirable that the traces are located in an oblique 45° direction with a pitch of about 1 mm to reduce interference with other signal traces. To exhibit high shielding effect, use a board with three or more layers and the ground pattern like this placed on the whole backside of the pad.

# 6. Software Description

## 6.1 s1c17w22_w23_touch_oscillo

This s1c17w22_w23_touch_oscillo software runs in conjunction with Excel and a VBA macro. It samples the counter value (capacitance equivalent value) that varies according to the touch key capacitance in certain cycles, and sends the counter values to the PC via the UART to write it to an Excel sheet. The following describes this software.

### 6.1.1 File Configuration (within src folder)

| File name | Description |
|---|---|
| boot.c | Startup module source file |
| init.c | Initialization function source file |
| main.c | Main function source file |
| osc.c | OSC driver source file |
| t16_ch0.c | T16 Ch.0 driver source file |
| t16b_ch0.c | T16B Ch.0 driver source file |
| uart.c | UART driver source file |

### 6.1.2 File Configuration (within inc folder)

| File name | Description |
|---|---|
| reg | S1C17W22 peripheral circuit register definition file folder |
| c17w22_reg.h | S1C17W22 peripheral circuit header definition file |
| init.h | Initialization function header definition file |
| osc.h | OSC driver header definition file |
| t16_ch0.h | T16 Ch.0 driver header definition file |
| t16b_ch0.h | T16B Ch.0 driver header definition file |
| touch_oscillo.h | Touch key capacitance visualization header definition file |
| uart.h | UART driver header definition file |

### 6.1.3 Module Description

This section describes the functions and the array variables, these explanations are especially necessary.

File name: main.c

| Function name | Description |
|---|---|
| measCap | This function measures the capacitance equivalent value of the sensor pad specified by to.scanPort, and returns to the caller after the measurement has completed. |
| intUartCh0 | UART Ch.0 interrupt handler function. This function activates T16 Ch.0 to start measurement of the sensor pad capacitance equivalent values when the control information is received from the VBA on the PC. |
| intT16Ch0 | T16 Ch.0 interrupt handler function. This function sequentially measures the capacitance equivalent values of the sensor pads specified by the control information already received and sends them to the VBA on the PC. When the specified number of data have been sent, this function stops T16 Ch.0 operations. |
| intT16bCh0 | T16B Ch.0 interrupt handler function. When a capture 0 interrupt occurs, this function reads capture 0 data as the capacitance equivalent value at the t+ period and then reconfigures the PPORT and UPMUX so that the capacitance equivalent value at the t- period will be measured. When a capture 1 interrupt occurs, this function reads capture 1 data as the capacitance equivalent value at the t- period and then reconfigures the PPORT and UPMUX to finish the measurement of the current sensor pad. |

The s1c17w22_w23_touch_oscillo software is coded assuming that the touch keys are connected to the 16 PPORT pins shown below. The same background color indicates the paired ports. A resistor of about 5 MΩ is connected between these ports (touch key).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| P00 | P01 | P02 | P03 | P04 | P05 | P06 | P07 |
| P10 | P11 | P12 | | | P15 | P16 | P17 |
| P20 | P21 | | | | | | |

When confirming the array variable values in the file, refer to the table below taking the conditions above into account.

File name: touch_oscillo.h

| Array variable name | Description |
| --- | --- |
| touchPort | Port group number of PPORT. 0: P0x, 1: P1x, 2: P2x |
| touchOen | PPORT output enable. This variable is shared in a port pair, as they have the same value. 0: disable, 1: enable |
| touchDat | Output value of the other PPORT in the port pair to be sensed.<br>0: L output, 1: H output |
| touchDatHH | Output value for both the paired ports. This variable is shared in a port pair, as they have the same value. 0: L output, 1: H output |
| touchIOEN | Output enable value of the other PPORT in the port pair to be sensed.<br>0: disable, 1: enable |
| touchUpmux1 | PORT to be assigned to UPMUX for t+ measurement. This array has two elements for even and odd PPORT numbers. |
| touchUpmux2 | PORT to be assigned to UPMUX for t- measurement. This array has two elements for even and odd PPORT numbers. |
| touchUpmuxNo | P[0–2]UPMUXx register number (= x) for UPMUX configuration. |
| modSel | P[0–2]MODSEL setting value to use a peripheral circuit. |
| fncSel | P[0–2]FNCSEL setting value to use Func#1. |

### 6.1.4 Operation Procedures

Importing project

(1) Launch IDE and import the s1c17w22_w23_touch_oscillo project.

* For the import method, refer to Chapter 3, "Software Development Procedures," in the S5U1C17001C Manual.

Building

(1) Build the s1c17w22_w23_touch_oscillo project using IDE.

Preparation for using Excel VBA

(1) Copy the MeasTouch folder to the desktop. This folder contains the Excel file MeasTouch.xlsm and the Active-X control file NonComSck.ocx. Note that the Active-X control file MSCOMM32.OCX provided by Microsoft Corporation is not included in this package. Please get it from a reliable download site and copy to the MeasTouch folder.

(2) Click [Start] - [All Programs] - [Accessories]. Right-click [Command Prompt] and select [Run as administrator] to open the [Administrator: Command Prompt] window.

(3) Execute "cd C:\Desktop\MeasTouch," "regsvr32.exe MSCOMM32.OCX," and "regsvr32.exe NonComSck.ocx" to enable the Active-X control files.

(4) If something goes wrong with the operation above, copy the Active-X control files to C:\Windows\System32, and then execute "cd C:\Windows\System32," "regsvr32.exe MSCOMM32.OCX," and "regsvr32.exe NonComSck.ocx."

Connection and power up

(1) Connect the SVT17W23 to the ICDmini and then it to the PC with a USB cable.

(2) Connect between UART Ch.0 of the SVT17W23 and the serial connector on the PC in which Excel VBA is executed (in general, USB I/F is used for serial communication. See sheet "Note" in the Excel VBA file for commercial cables that can be used). When using a PC with sufficient processing ability, IDE and Excel can be run simultaneously on that PC, otherwise two PCs should be used.

(3) Reset the SVT17W23 and ICDmini.

Excel VBA execution

(1) Double-click the Excel file MeasTouch.xlsm to execute the VBA.

Execution

(1) Edit the path to fls17w22.elf described in the command file s1c17w22_w23_touch_oscillo_gnu17IDE.cmd as necessary. By default, it is set as C:/EPSON/GNU17/mcu_model/17W22/fls/fls17w22.elf.

* For how to edit a command file, refer to the S5U1C17001C Manual.

(2) Execute the s1c17w22_w23_touch_oscillo project using IDE.

(3) Operate Excel VBA to start touch key scan.

### 6.1.5 How to Use MeasTouch.xlsm

Figure 6-1 shows the appearance of MeasTouch.xlsm.

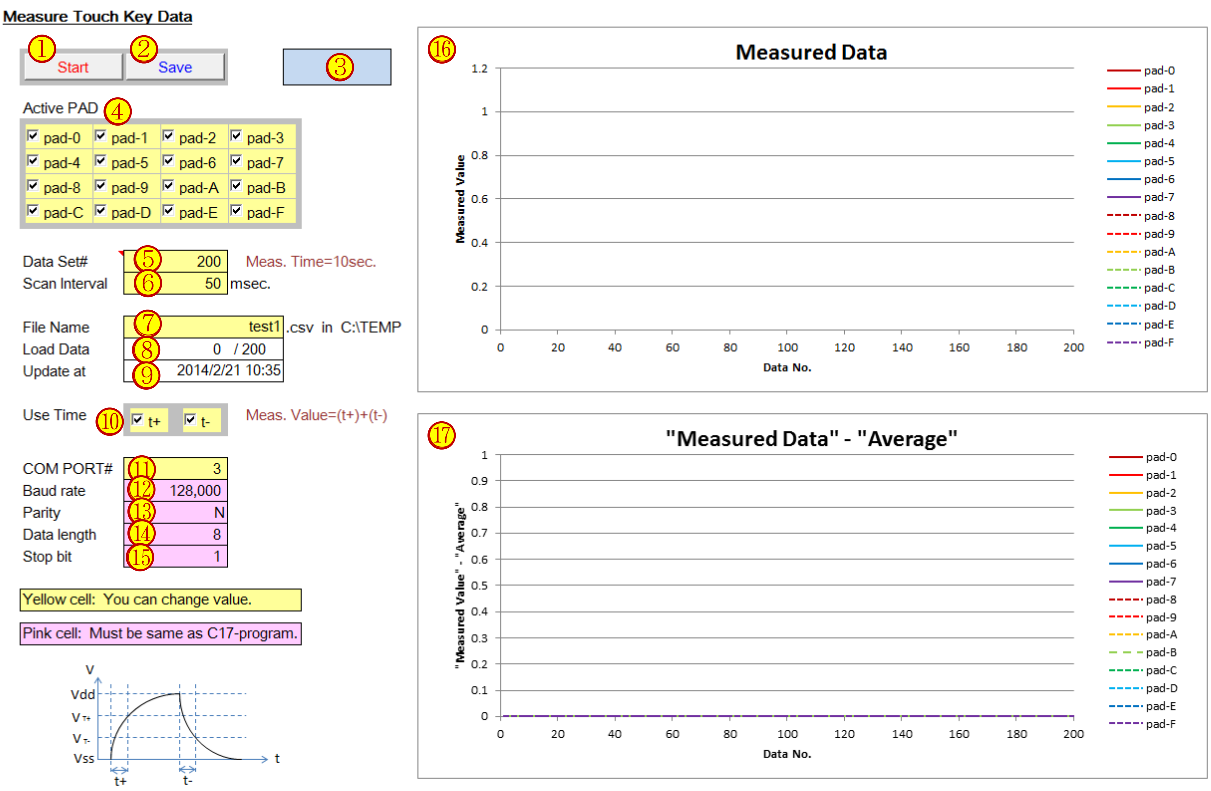



Figure 6-1 Appearance of MeasTouch.xlsm

| | |
|---|---|
| ① Measurement start button | Starts measurement of touch key capacitance equivalent values. |
| ② Data save button | Saves the measured data in CSV format. |
| ③ Status indicator | Indicates the measurement or data saving status. |
| ④ Measuring touch key selector | Selects the touch keys to be measured. |
| ⑤ Number of sampling data | Sets the number of data to be sampled. |
| ⑥ Scan interval | Sets the interval between data samplings. |
| ⑦ Save file name | Specifies the name of the file for storing measured data. |
| ⑧ Number of data sampled | Displays the successively updated number of data sampled. |
| ⑨ Data sampling date and time | Displays the date and time at which the latest data is sampled. |
| ⑩ t+/t- selector | Selects the period to be converted into capacitance equivalent values from t+, t-, or both. |
| ⑪ COM port number | Specifies the COM port number used for serial communication. |
| ⑫ Baud rate | Specifies the baud rate for serial communication. |
| ⑬ Parity | Enables serial data parity addition/check with even or odd specified, or disables it. |
| ⑭ Data length | Specifies the serial data length. |
| ⑮ Stop bit | Specifies the number of stop bits for serial communication. |
| ⑯ Measured data graph | Draws a graph representing measured data on the vertical axis. |
| ⑰ Measured data - average graph | Draws a graph representing difference between measured data and the average value on the vertical axis. |

Note 1) This sample program assumes two-wire serial communication using the UART. On the other hand, the SVT17W23 is designed to send data one-sidedly. Therefore, communication data will be lost if the receive buffer on the PC cannot follow. The program detects this error by checking if the high- or low-order data is out of the range. If a such an error message is displayed, try to avoid it by changing the ④, ⑤, and ⑥ setting values.

Note 2) Normally, it is not necessary to change the serial communication parameters ⑪, ⑫, ⑬, ⑭, and ⑮. Note that the related part of s1c17w22_w23_touch_oscillo must be modified if these parameters are changed.

Note 3) The graphs ⑯ and ⑰ are drawn after all data has been loaded. While data are being sampled, they are not updated successively and the graphs retain the previous plot.

### 6.1.6 Outline of Sample Program Operations

① Switches the system clock from IOSC to OSC3 (4 MHz internal oscillator) after setting interrupt levels and initializing variables.

② Initializes the UART, T16, and T16B.

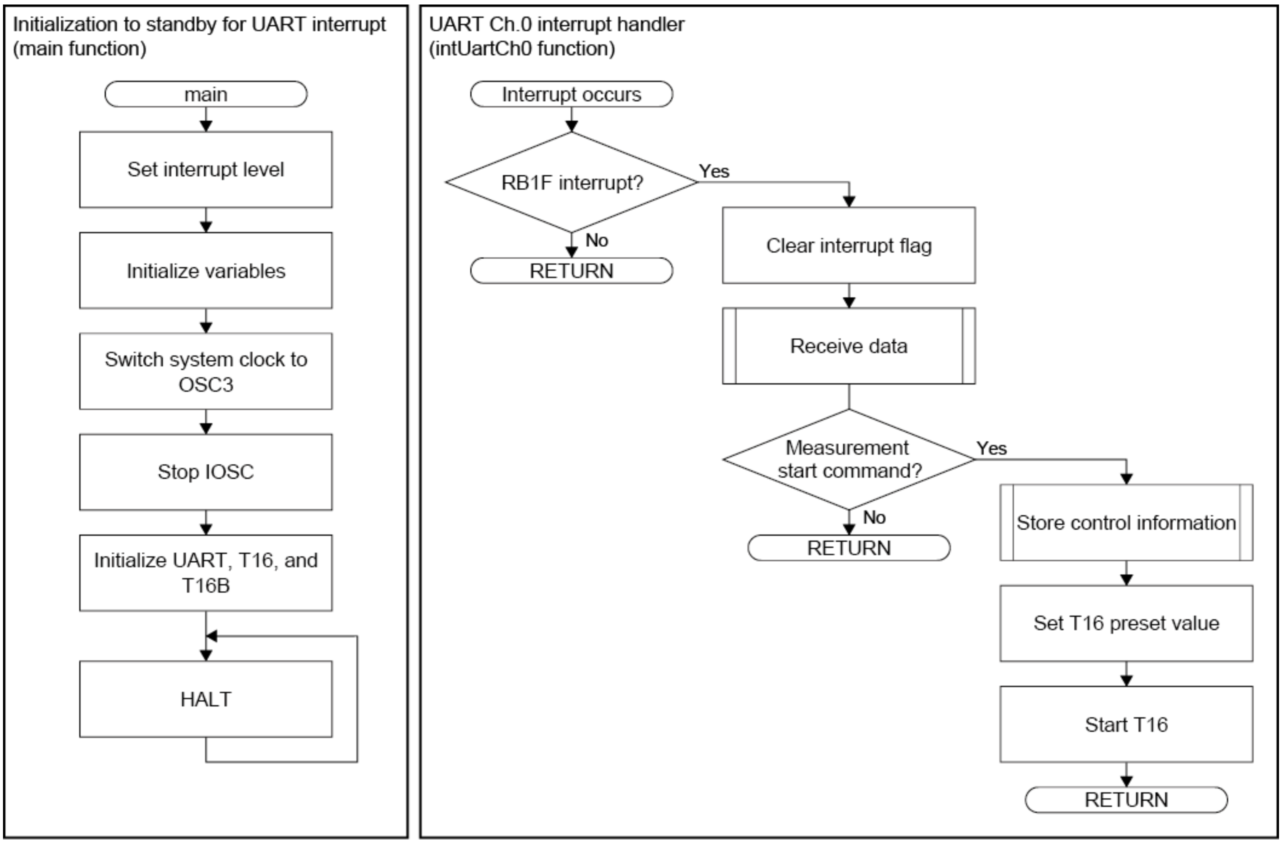
③ Places the CPU into HALT status while waiting for a UART interrupt.

④ When a UART interrupt occurs, the program receives the control information sent from the PC and starts T16 Ch.0 operations to start measurement of sensor pad capacitance equivalent values.

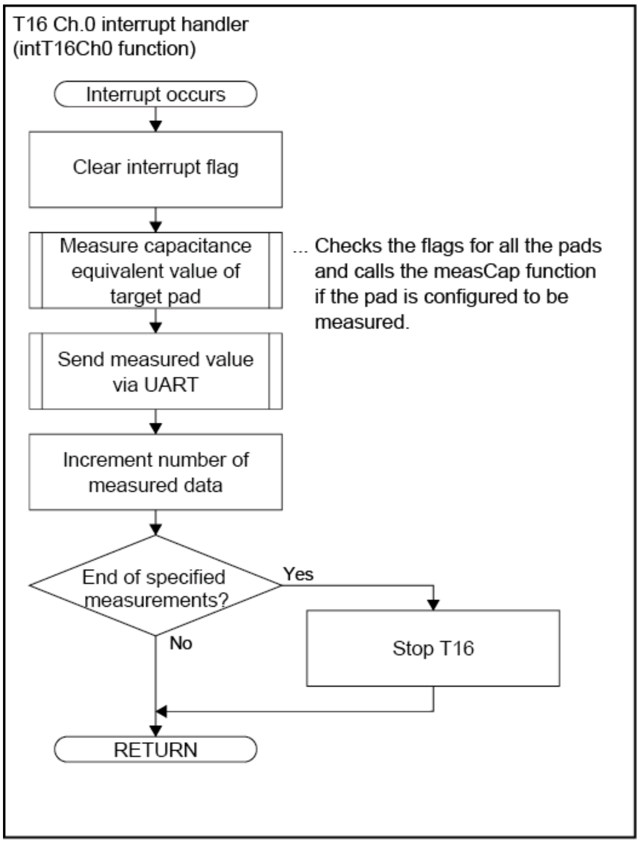
⑤ Every time a T16 Ch.0 interrupt occurs, the program sequentially measures the capacitance equivalent values of the sensor pads specified by the control information already received and sends them to the PC. When the specified number of data have been sent, the program stops T16 Ch.0 operations.

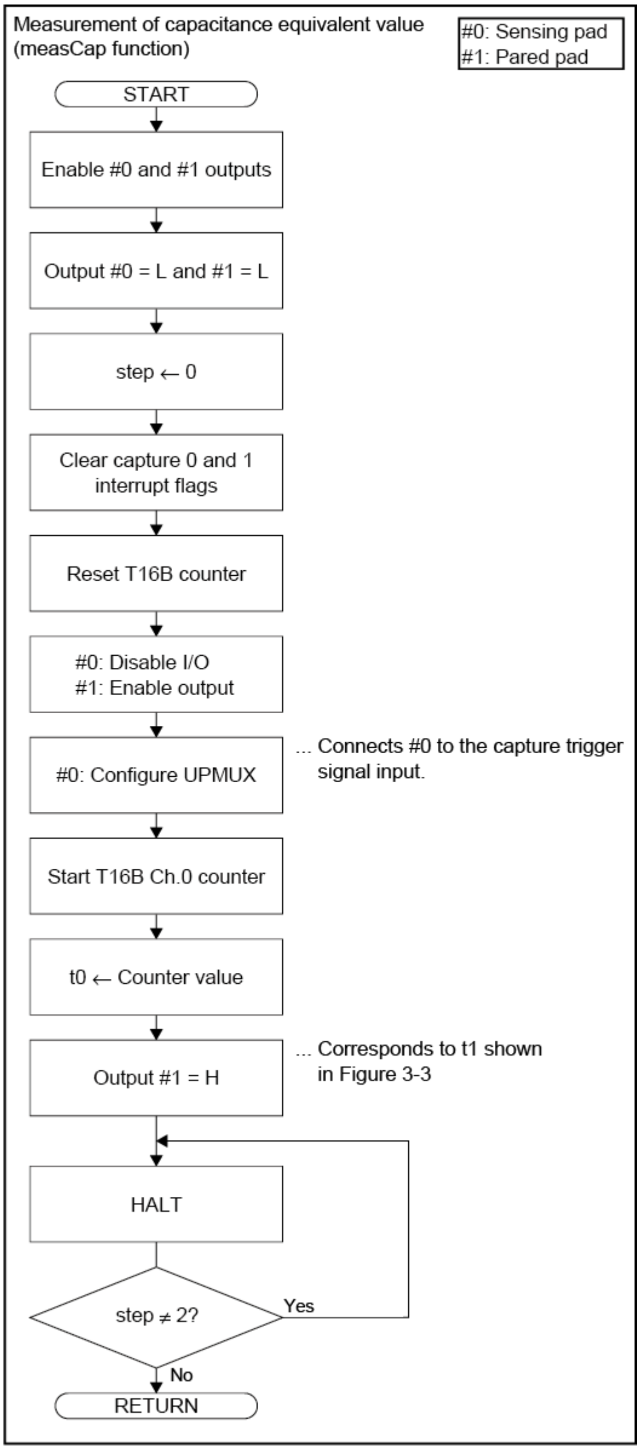
⑥ When a capture 0 interrupt occurs from T16B Ch.0, the program reads capture 0 data as the capacitance equivalent value at the t+ period and then reconfigures the PPORT and UPMUX so that the capacitance equivalent value at the t- period will be measured. When a capture 1 interrupt occurs, this function reads capture 1 data as the capacitance equivalent value at the t- period and then reconfigures the PPORT and UPMUX to finish the measurement of the current sensor pad.

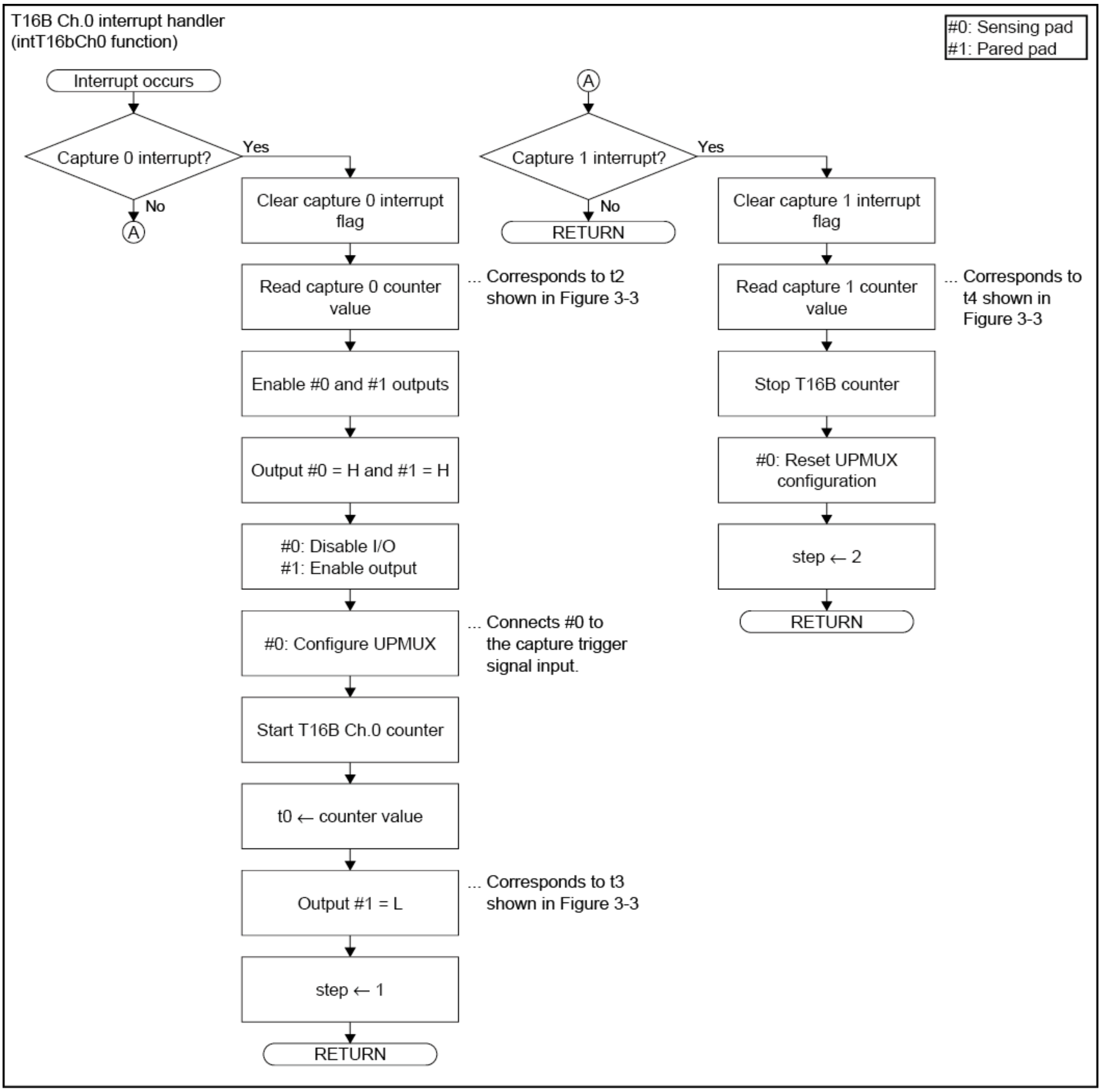
The following shows the flowcharts of the operations above.



# 6. Software Description




T16B Ch.0 interrupt handler
(intT16bCh0 function)
#0: Sensing pad
#1: Pared pad
Interrupt occurs
Capture 0 interrupt?
Yes
No
A
Clear capture 0 interrupt flag
Read capture 0 counter value
... Corresponds to t2 shown in Figure 3-3
Enable #0 and #1 outputs
Output #0 = H and #1 = H
#0: Disable I/O
#1: Enable output
#0: Configure UPMUX
... Connects #0 to the capture trigger signal input.
Start T16B Ch.0 counter
t0 ← counter value
Output #1 = L
... Corresponds to t3 shown in Figure 3-3
step ← 1
RETURN
A
Capture 1 interrupt?
Yes
No
RETURN
Clear capture 1 interrupt flag
Read capture 1 counter value
... Corresponds to t4 shown in Figure 3-3
Stop T16B counter
#0: Reset UPMUX configuration
step ← 2
RETURN

## 6.2 s1c17w22_w23_touch_demo

This section describes the s1c17w22_w23_touch_demo software that displays the touch key status and capacitance equivalent values as a touch key demonstration.

### 6.2.1 File Configuration (within src folder)

| File name | Description |
|---|---|
| boot.c | Startup module source file |
| display.c | LCD display function source file |
| init.c | Initialize function source file |
| main.c | Main function source file |
| osc.c | OSC driver source file |
| snda.c | SNDA driver source file |
| t16_ch0.c | T16 Ch.0 driver source file |
| t16b_ch0.c | T16B Ch.0 driver source file |

### 6.2.2 File Configuration (within inc folder)

| File name | Description |
|---|---|
| reg | S1C17W22 peripheral circuit register definition file folder |
| c17w22_reg.h | S1C17W22 peripheral circuit header definition file |
| display.h | LCD driver header definition file |
| init.h | Initialization function header definition file |
| lcd_font.h | LCD font definition file |
| osc.h | OSC driver header definition file |
| snda.h | SNDA driver header definition file |
| t16_ch0.h | T16 Ch.0 driver header definition file |
| t16b_ch0.h | T16B Ch.0 driver header definition file |
| touch_sw.h | Touch key control header definition file |

### 6.2.3 Module Description

This section describes the functions and the array variables, these explanations are especially necessary.

File name: main.c

| Function name | Description |
|---|---|
| measCap | This function measures the capacitance equivalent value of the sensor pad specified by ts.scanPort, and returns to the caller after the measurement has completed. |
| val2Str3 | This function replaces the character range specified by the start position (argument pos) and the number of digits (argument digit) within the character string (argument str) with the character string converted from the specified value (argument val). |
| dispStat | This function displays the character string “NoTouchKey,” “(Button number) = (Detection value)/(Background level of the target button),” or “Chattering” on the first line when the argument (touched) is 0, 1, or 2, respectively. It also displays whether a key has been touched or not (*/ (blank)), the touch judgment level, and the touch status of all keys in a four-digit hexadecimal number on the second line. |
| intT16Ch0 | T16 Ch.0 interrupt handler function. This function switches the system clock from OSC1 (32.768 kHz) for standby status to OSC3 (4 MHz internal oscillator). It sequentially measures the capacitance equivalent values of the 16 sensor pads and processes data for moving average calculations. The processing in this function depends on the ts.stage value as follows:<br>When ts.stage = 0: When the data collected reaches the necessary number for moving average processing, the sequence proceeds to the next stage. Until that time, data collection continues.<br>When ts.stage = 1: When the data collected reaches the necessary number to determine the background level, the sequence proceeds to the next stage. Until that time, data collection for background level calculation continues.<br>When ts.stage = 2: This stage is a waiting period making preparations for the next stage (touching the specified pad with a finger).<br>When ts.stage = 3: In this stage, the capacitance equivalent value of the sensor pad connected to P00 is measured to obtain a capacitance equivalent value variation when the pad is touched.<br>When ts.stage = 4: This stage is a normal operation period. This function checks the capacitance equivalent value variations of all the sensor pads to determine whether the pad is in On status or not, and displays the information of the sensor pad that has the largest variation on the LCD panel. Furthermore, it updates the background level in the fixed intervals to follow environmental changes. When On status has successively occurred exceeding the predefined number, this function regards it as a new background level to recover from abnormal operations that may occur, for example, if the equipment is put in a pocket. If Off status is continued, this function extends the scan interval and displays that change. When On status of a sensor pad is detected, it immediately shortens the scan interval. With these operations, both high response and low power consumption are achieved. A click sound is generated when a sensor pad changes Off status to On status.<br>When the processing of any one of the stages above has completed, this function switches the system clock from OSC3 (4 MHz internal oscillator) to OSC1 (32.768 kHz) for standby status. |
| intT16bCh0 | T16B Ch.0 interrupt handler function. When a capture 0 interrupt occurs, this function reads capture 0 data as the capacitance equivalent value at the t+ period and then reconfigures the PPORT and UPMUX so that the capacitance equivalent value at the t- period will be measured. When a capture 1 interrupt occurs, this function reads capture 1 data as the capacitance equivalent value at the t- period and then reconfigures the PPORT and UPMUX to finish the measurement of the current sensor pad. |

## 6. Software Description

The s1c17w22_w23_touch_demo software is coded assuming that the touch keys are connected to the 16 PPORT pins shown below the same as s1c17w22_w23_touch_oscillo. The same background color indicates the paired ports. A resistor of about 5 MΩ is connected between these ports (touch key).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| P00 | P01 | P02 | P03 | P04 | P05 | P06 | P07 |
| P10 | P11 | P12 | | | P15 | P16 | P17 |
| P20 | P21 | | | | | | |

When confirming the array variable values in the file, refer to the table below taking the conditions above into account.

File name: touch_sw.h

| Array variable name | Description |
|---|---|
| touchPort | Port group number of PPORT. 0: P0x, 1: P1x, 2: P2x |
| touchOen | PPORT output enable. This variable is shared in a port pair, as they have the same value. 0: disable, 1: enable |
| touchDat | Output value of the other PPORT in the port pair to be sensed.<br>0: L output, 1: H output |
| touchDatHH | Output value for both the paired ports. This variable is shared in a port pair, as they have the same value. 0: L output, 1: H output |
| touchIOEN | Output enable value of the other PPORT in the port pair to be sensed.<br>0: disable, 1: enable |
| touchUpmux1 | PORT to be assigned to UPMUX for t+ measurement. This array has two elements for even and odd PPORT numbers. |
| touchUpmux2 | PORT to be assigned to UPMUX for t- measurement. This array has two elements for even and odd PPORT numbers. |
| touchUpmuxNo | P[0–2]UPMUXx register number (= x) for UPMUX configuration. |
| modSel | P[0–2]MODSEL setting value to use a peripheral circuit. |
| fncSel | P[0–2]FNCSEL setting value to use Func#1. |

The contents described in the above file are almost the same as the touch_oscillo.h file of the s1c17w22_w23_touch_oscillo software except for the descriptions of the ports in which the LCD and buzzer signals are assigned.

### 6.2.4 Operation Procedures

Importing project

(1) Launch IDE and import the s1c17w22_w23_touch_demo project.

* For the import method, refer to Chapter 3, “Software Development Procedures,” in the S5U1C17001C Manual.

Building

(1) Build the s1c17w22_w23_touch_demo project using IDE.

Connection and power up

(1) Connect the SVT17W23 to the ICDmini and then it to the PC with a USB cable. If the same system as s1c17w22_w23_oscillo is used, disconnect the serial communication cable for interfacing with the UART.

(2) Reset the SVT17W23 and ICDmini.

Execution

(1) Edit the path to fls17w22.elf described in the command file s1c17w22_w23_touch_demo_gnu17IDE.cmd as necessary. By default, it is set as C:/EPSON/GNU17/mcu_model/17W22/fls/fls17w22.elf.

* For how to edit a command file, refer to the S5U1C17001C Manual.

(2) Execute the s1c17w22_w23_touch_demo project using IDE.

(3) Step (2) starts a demonstration of touch key operation.

### 6.2.5 Outline of Sample Program Operations

① Enables the OSC1 (32.768 kHz) to start oscillating.

② Initializes the LCD driver and clears the display contents.

③ Initializes and activates SNDA for the buzzer.

④ Initializes T16 Ch.0 and T16B Ch.0.

⑤ Starts T16 Ch.0.

⑥ Displays “Init.-1/3” and “Chk BgLvl” on the LCD panel to indicate that Step 1 of the three initialization steps is being executed, or the background data of the sensor pads is being collected.

⑦ Switches the system clock from IOSC to OSC1 (32.768 kHz).

⑧ Places the CPU into HALT status while waiting for a T16 Ch.0 interrupt.

⑨ Every time a T16 Ch.0 interrupt occurs, the program measures the capacitance equivalent values of the 16 sensor pads and sequentially starts initialization according to the measured value. Do not bring the hand or fingers close to the sensor pads while “Init.-1/3” and “Chk BgLvl” are displayed on the LCD panel.

⑩ The LCD display is changed to “Init.-2/3” and “Waiting...” to indicate that Step 2 of the three initialization steps is being executed, or in the stage to wait for the next step. While this message is displayed, touch the sensor pad connected to P00 to get ready for the next step.

⑪ The LCD display is changed to “Init.-3/3” and “Touch Btn0.” Keep touching the sensor pad connected to P00 while this message is displayed. When this stage has completed, a key click sound is generated and the sequence proceeds to the normal operation stage.

⑫ The normal operation stage checks the capacitance equivalent value variations of all the sensor pads to determine whether a pad is in On status or not, and displays the information of the sensor pad that has the largest variation on the LCD panel. The display contents are shown below.

Line 1: "(Sensor pad number) = (Measured value)/(Background level)" or
"NoTouchKey" when no touch status is detected, or
"Chattering" when a chattering is detected

Line 2: "(Symbol)_(Touch judgment threshold value)_(Touch status of all keys)"
(Symbol: "*" when touch status is detected or "(blank)" when no touch status is detected)

For example, when the 12th sensor pad (No.11), which is connected to P15, is detected as the pad having the largest capacitance equivalent value variation, the information shown below is displayed.

Line 1: B=020/0625

Line 2: *_016_0800

The information displayed on Line 2 is used to check if multiple sensor pads are touched simultaneously. When all the sensor pads are regarded as being touched, the information shown below is displayed.

Line 1: 3=034/0548

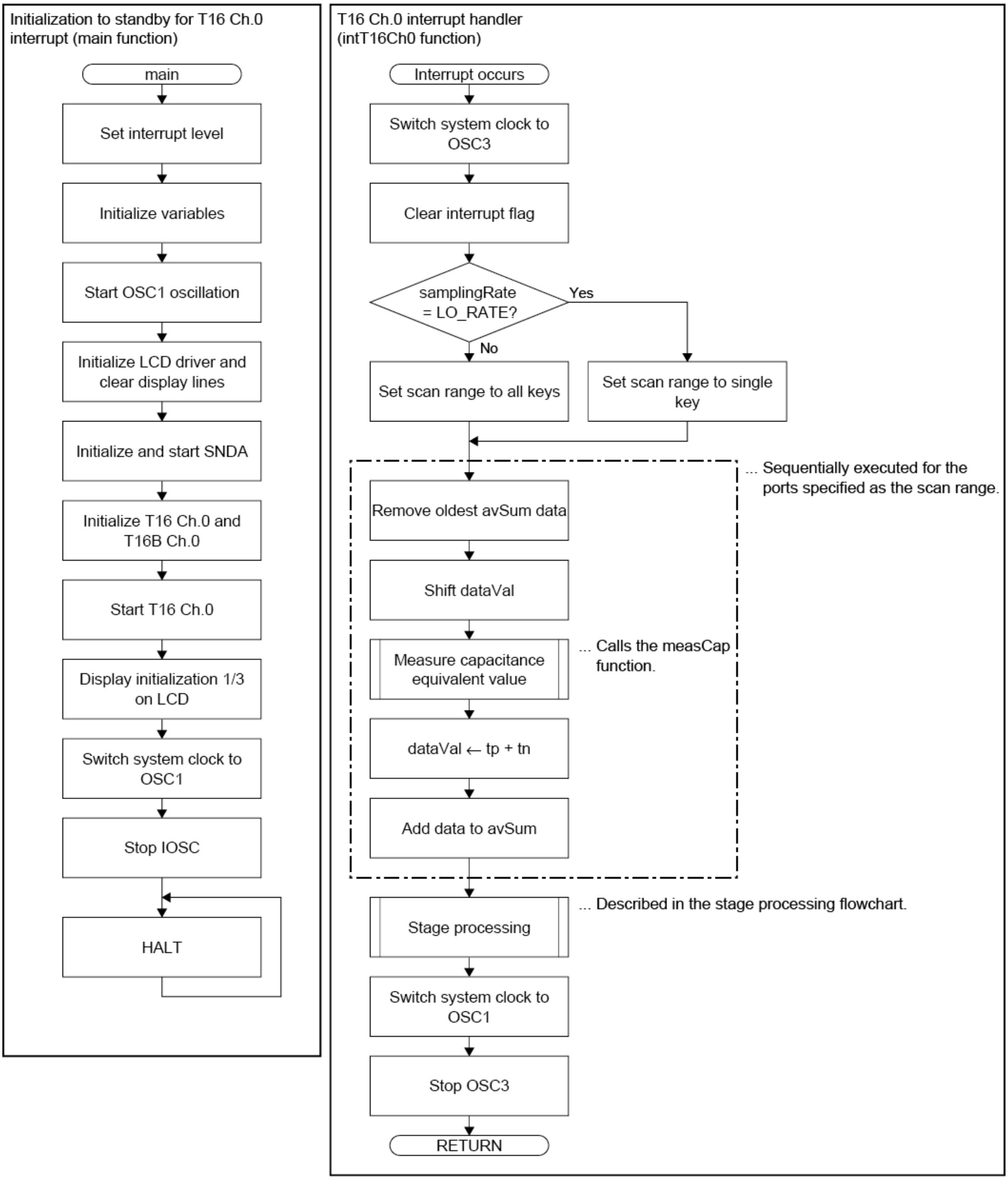
Line 2: *_016_FFFF

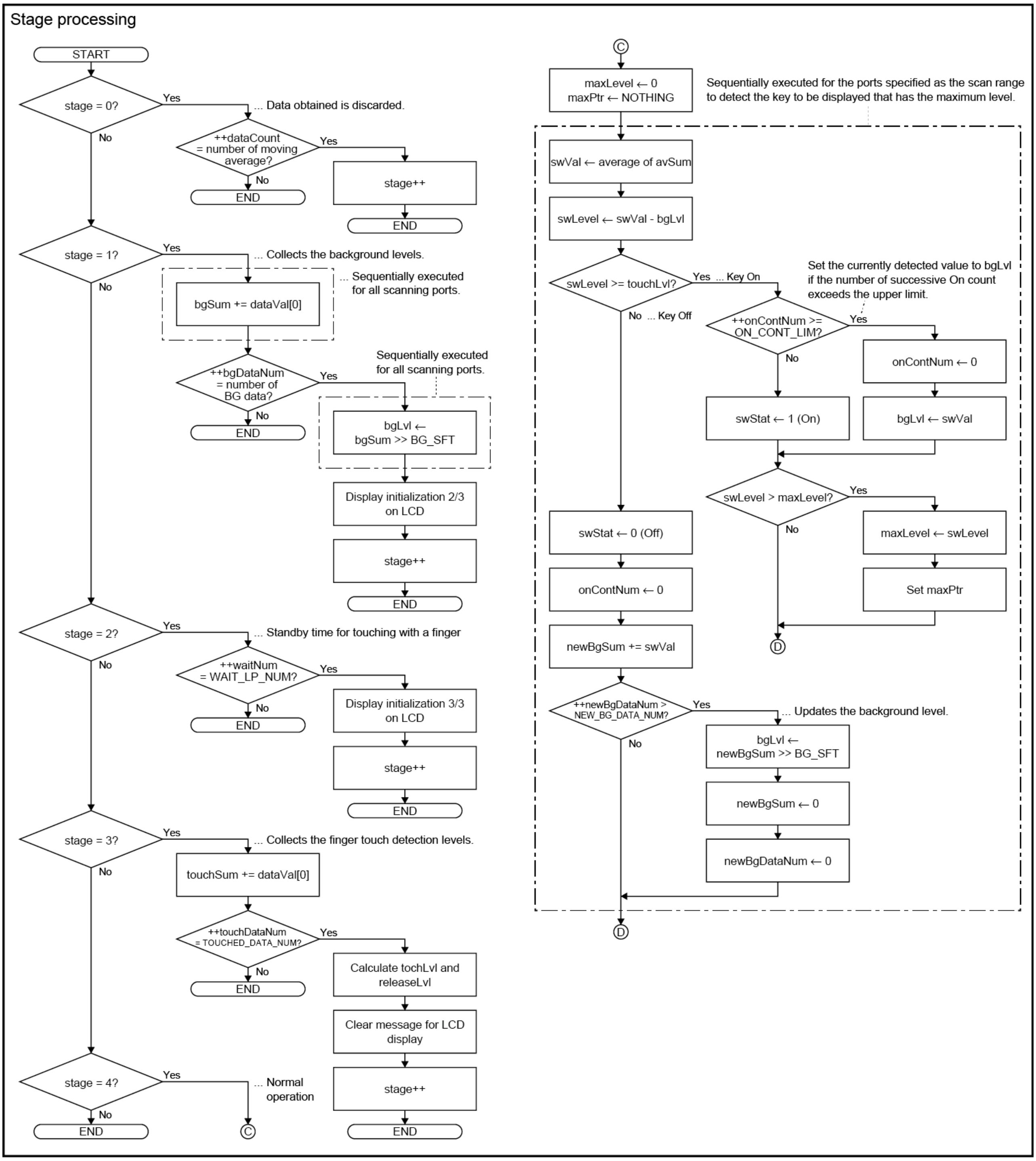
If no key touch is detected within the defined time, the target sensor pad to measure capacitance equivalent values is forcibly set to one that is predefined (to which the power on function is assigned) and the measurement cycle is extended for power saving. In this case, "Sleeping..." is displayed on the LCD panel. This status returns to normal operation when the enabled sensor pad is touched.

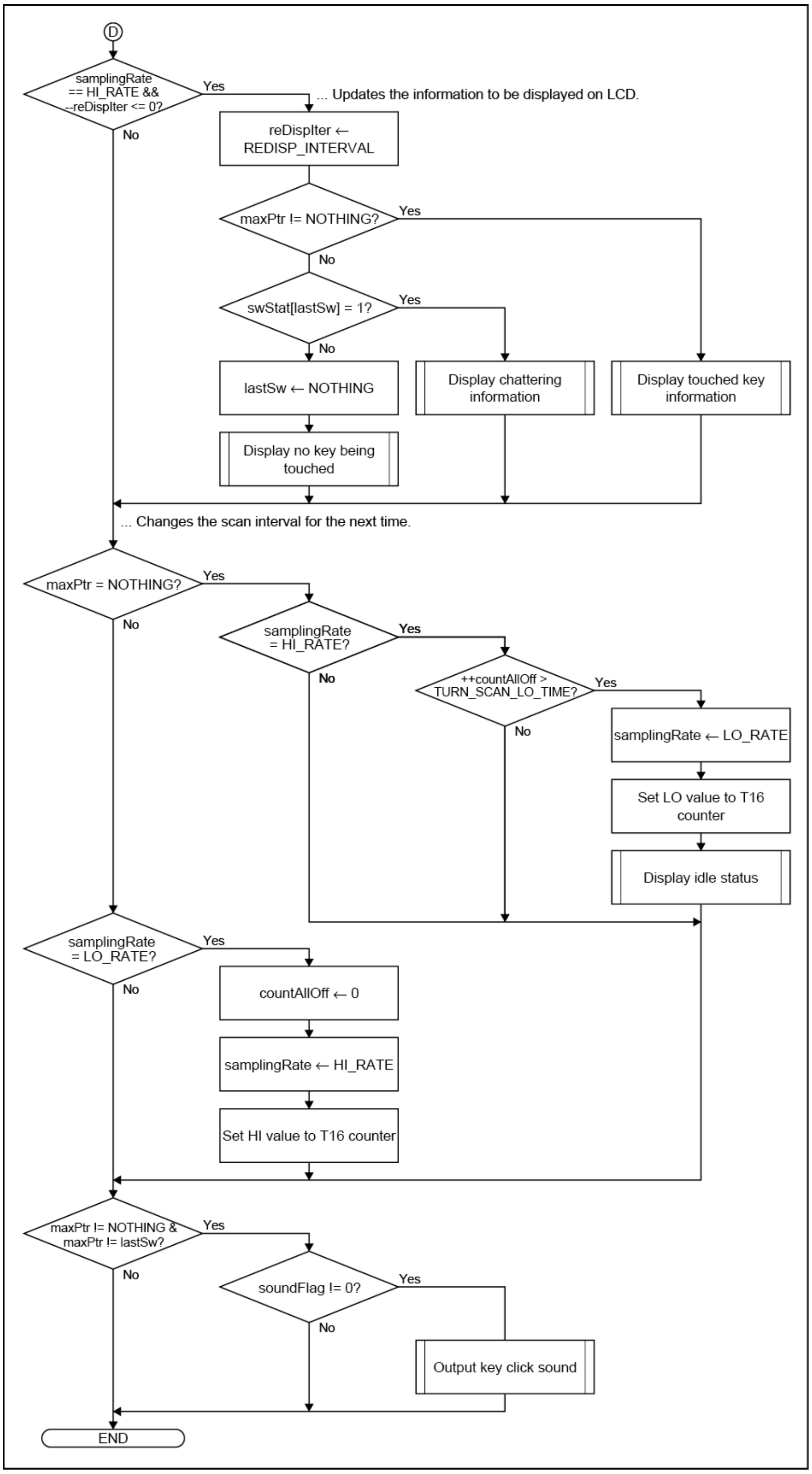
The following two functions have the same processing flow as those of s1c17w22_w23_touch_oscillo/src/main.c except that the structure variable name is changed from "to" to "ts." Therefore no flowcharts are provided in this section.

- T16B Ch.0 interrupt handler (intT16bCh0 function)
- Capacitance equivalent value measurement (measCap function)

The following shows the flowcharts of other processing.


Initialization to standby for T16 Ch.0 interrupt (main function)
main
Set interrupt level
Initialize variables
Start OSC1 oscillation
Initialize LCD driver and clear display lines
Initialize and start SNDA
Initialize T16 Ch.0 and T16B Ch.0
Start T16 Ch.0
Display initialization 1/3 on LCD
Switch system clock to OSC1
Stop IOSC
HALT
T16 Ch.0 interrupt handler (intT16Ch0 function)
Interrupt occurs
Switch system clock to OSC3
Clear interrupt flag
samplingRate = LO_RATE?
Yes
No
Set scan range to all keys
Set scan range to single key
... Sequentially executed for the ports specified as the scan range.
Remove oldest avSum data
Shift dataVal
Measure capacitance equivalent value
... Calls the measCap function.
dataVal ← tp + tn
Add data to avSum
Stage processing
... Described in the stage processing flowchart.
Switch system clock to OSC1
Stop OSC3
RETURN


Stage processing
START
stage = 0?
Yes
... Data obtained is discarded.
++dataCount = number of moving average?
Yes
No
END
stage++
END
stage = 1?
Yes
... Collects the background levels.
bgSum += dataVal[0]
... Sequentially executed for all scanning ports.
++bgDataNum = number of BG data?
Yes
No
END
Sequentially executed for all scanning ports.
bgLvl ← bgSum >> BG_SFT
Display initialization 2/3 on LCD
stage++
END
stage = 2?
Yes
... Standby time for touching with a finger
++waitNum = WAIT_LP_NUM?
Yes
No
END
Display initialization 3/3 on LCD
stage++
END
stage = 3?
Yes
... Collects the finger touch detection levels.
touchSum += dataVal[0]
++touchDataNum = TOUCHED_DATA_NUM?
Yes
No
END
Calculate tochLvl and releaseLvl
Clear message for LCD display
stage++
END
stage = 4?
Yes
... Normal operation
No
END
C
maxLevel ← 0
maxPtr ← NOTHING
Sequentially executed for the ports specified as the scan range to detect the key to be displayed that has the maximum level.
swVal ← average of avSum
swLevel ← swVal - bgLvl
swLevel >= touchLvl?
Yes ... Key On
No ... Key Off
Set the currently detected value to bgLvl if the number of successive On count exceeds the upper limit.
++onContNum >= ON_CONT_LIM?
Yes
No
onContNum ← 0
swStat ← 1 (On)
bgLvl ← swVal
swLevel > maxLevel?
Yes
No
swStat ← 0 (Off)
maxLevel ← swLevel
onContNum ← 0
Set maxPtr
newBgSum += swVal
D
++newBgDataNum > NEW_BG_DATA_NUM?
Yes
... Updates the background level.
No
bgLvl ← newBgSum >> BG_SFT
newBgSum ← 0
newBgDataNum ← 0
D

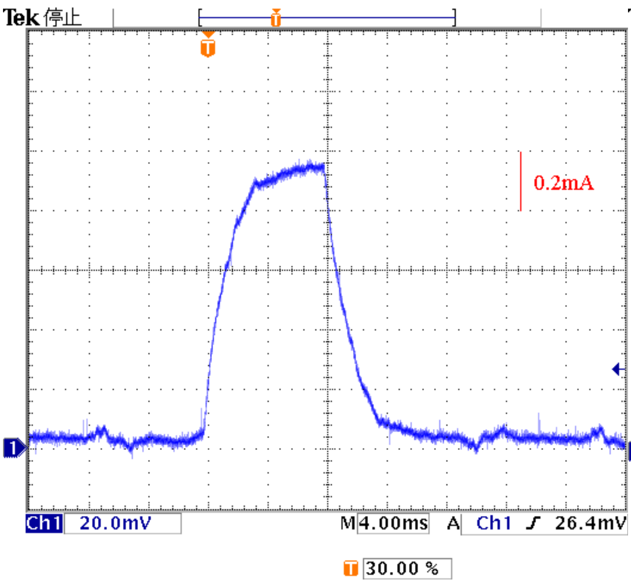

D
samplingRate == HI_RATE && --reDispIter <= 0?
Yes
No
... Updates the information to be displayed on LCD.
reDispIter ← REDISP_INTERVAL
maxPtr != NOTHING?
swStat[lastSw] = 1?
lastSw ← NOTHING
Display chattering information
Display touched key information
Display no key being touched
... Changes the scan interval for the next time.
maxPtr = NOTHING?
samplingRate = HI_RATE?
++countAllOff > TURN_SCAN_LO_TIME?
samplingRate ← LO_RATE
Set LO value to T16 counter
Display idle status
samplingRate = LO_RATE?
countAllOff ← 0
samplingRate ← HI_RATE
Set HI value to T16 counter
maxPtr != NOTHING & maxPtr != lastSw?
soundFlag != 0?
Output key click sound
END

### 6.2.6 Estimate of Battery Life

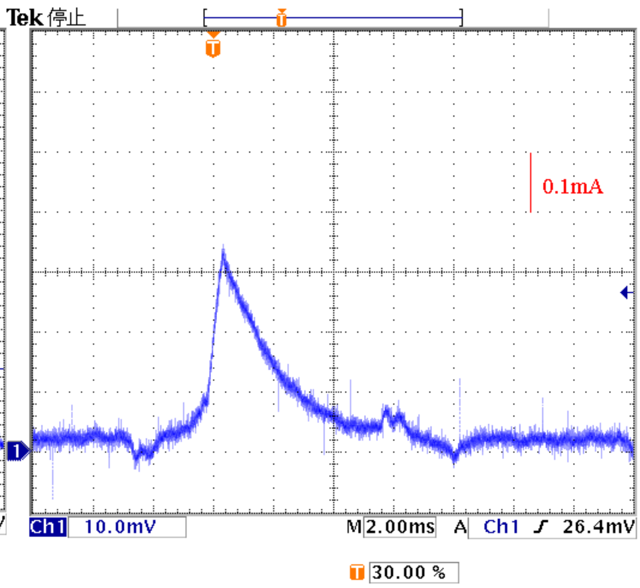
A battery life when touch keys are driven with a coin cell battery is estimated as follows based on the actual consumption current value measured by executing the sample program s1c17w22_w23_touch_demo with the SVT17W23. The calculation conditions are shown below.

- Coin cell battery: CR2023 (nominal capacity = 220 mAh) One
- Touch key read time: 30 minutes per day

Figure 6-2 (a) shows the current waveform monitoring result when all the 16 keys are scanned 30 times per second for reading touch key status. Figure 6-2 (b) shows the current waveform monitoring result in idle status in which the specific key only is scanned five times per second.




(a) When touch key is read




(b) In idle status

Figure 6-2 Current Waveform Monitoring Results

(1) When touch key is read

0.94 mA x 0.009 s x 1000 / 33.3 = 0.25 mA

(2) In idle status

0.35 mA x 0.006 s x 1000 / 200 = 0.0105 mA and 0.5 µA ($I_{HALT2}$: HALT with OSC1)

From these results,

220 mAh / (0.25 mA x 0.5 h + (0.0105 mA + 0.0005 mA) x 24 h) = 566 days = about 1.55 years

# Revision History

Attachment-1

| Rev. No. | Date | Page | Category | Contents |
|---|---|---|---|---|
| Rev 1.0 | 2014/06/12 | All | new | New establishment |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

EPSON

# International Sales Operations


## AMERICA

**EPSON ELECTRONICS AMERICA, INC.**
214 Devcon Drive,
San Jose, CA 95112, USA
Phone: +1-800-228-3964 FAX: +1-408-922-0238

## EUROPE

**EPSON EUROPE ELECTRONICS GmbH**
Riesstrasse 15, 80992 Munich,
GERMANY
Phone: +49-89-14005-0 FAX: +49-89-14005-110

## ASIA

**EPSON (CHINA) CO., LTD.**
7F, Jinbao Bldg., No.89 Jinbao St.,
Dongcheng District,
Beijing 100005, CHINA
Phone: +86-10-8522-1199 FAX: +86-10-8522-1125

**SHANGHAI BRANCH**
7F, Block B, Hi-Tech Bldg., 900 Yishan Road,
Shanghai 200233, CHINA
Phone: +86-21-5423-5577 FAX: +86-21-5423-4677

**SHENZHEN BRANCH**
12F, Dawning Mansion, Keji South 12th Road,
Hi-Tech Park, Shenzhen 518057, CHINA
Phone: +86-755-2699-3828 FAX: +86-755-2699-3838

**EPSON HONG KONG LTD.**
Unit 715-723, 7/F Trade Square, 681 Cheung Sha Wan Road,
Kowloon, Hong Kong.
Phone: +852-2585-4600 FAX: +852-2827-4346

**EPSON TAIWAN TECHNOLOGY & TRADING LTD.**
14F, No. 7, Song Ren Road,
Taipei 110, TAIWAN
Phone: +886-2-8786-6688 FAX: +886-2-8786-6660

**EPSON SINGAPORE PTE., LTD.**
1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 FAX: +65-6271-3182

**SEIKO EPSON CORP.**
**KOREA OFFICE**
5F, KLI 63 Bldg., 60 Yoido-dong,
Youngdeungpo-Ku, Seoul 150-763, KOREA
Phone: +82-2-784-6027 FAX: +82-2-767-3677

**SEIKO EPSON CORP.**
**MICRODEVICES OPERATIONS DIVISION**

**IC Sales & Marketing Department**
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 FAX: +81-42-587-5117



Document Code: 412786300
Issue June 2014 in JAPAN