EPSON
EXCEED YOUR VISION

S1C17 Family Application Note

S1C17700 Series Peripheral Circuit Sample Software 2

SEIKO EPSON CORPORATION

Rev.1.0

NOTICE

Table of Contents

1. Overview

This manual describes how to use the sample software for peripheral devices newly added to the S1C17703 and S1C17706 microcontrollers and its operations. Refer to the following manual for information on peripheral circuits common among the S1C17700 series.

- S1C17700 Series Peripheral Circuit Sample Software

The S1C17700 series peripheral circuit sample software 2 is aimed at presenting examples of how to use peripheral circuits newly added to the S1C17703 and S1C17706 microcontrollers.

The S1C17700 series peripheral circuit sample software 2 is provided for each model to simplify the installation; however the basic operations of each function are the same.

Use this manual along with model information, applicable technical manuals, and the "S5U1C17001C Manual."

1.1 Operating Environment

In order to run the S1C17700 series peripheral circuit sample software, prepare the following items.

- A board on where S1C17703 or S1C17706 is mounted
- S5U1C17001H (hereinafter ICDmini)
- S5U1C17001C (hereinafter GNU17)

Note: This sample software has been confirmed to operate on GNU17v2.0.0.

2. Explanation of Sample Software

This chapter describes the file configuration and execution method of the S1C17700 series peripheral circuit sample software 2.

The S1C17700 series peripheral circuit sample software 2 consists of "sample software" that checks operations of each peripheral circuit, and "sample drivers" that are sample drivers of each peripheral circuit.

2.1 Included Sample Software

The following lists the included sample software items.

Table 2.1 List of included sample software items

Peripheral Circuit	Sample Software
I/O port (P)	○
Clock generator (CLG)	○
16-bit timer (T16)	○
Advanced timer (T16A)	○
Clock timer (CT)	○
Stopwatch timer (SWT)	○
Watchdog timer (WDT)	○
UART that uses OSC3	○
UART that uses IOSC	○
SPI master	○
SPI slave	○
I2C master (I2CM)	○
I2C slave (I2CS)	○
LCD driver (LCD)	○
Power supply voltage detection circuit (SVD)	○
RF converter (RFC)	○
A/D converter (ADC10)	○
Remote controller sending (REMC)	○
Remote controller receiving (REMC)	○
Sleep/Halt	○
Advanced timer (T16A2)	●
LCD driver (LCD32A)	●
Power supply voltage detection circuit (SVD2)	●
Sound generator (SND)	●
Real time clock (RTC2)	●
Electric current measurement	●

●: New or modified, ○: Common among the series

2.2 Included Sample Drivers

The following lists the included sample drivers.

Table 2.2 List of included sample drivers

Peripheral Circuit	Sample Driver
I/O port (P)	○
Clock generator (CLG)	○
16-bit timer (T16)	○
Advanced timer (T16A)	○
Clock timer (CT)	○
Stopwatch timer (SWT)	○
Watchdog timer (WDT)	●
UART	○
SPI	○
I2C master (I2CM)	○
I2C slave (I2CS)	○
LCD driver (LCD)	●
Power supply voltage detection circuit (SVD)	○
RF converter (RFC)	○
A/D converter (ADC10)	○
Remote controller (REMC)	○
Prescaller (PSC)	○
MISC	○
Multiplexer (MUX)	●
Advanced timer (T16A2)	●
LCD driver (LCD32A)	●
Power supply voltage detection circuit (SVD2)	●
Sound generator (SND)	●
Real time clock (RTC2)	●
Power supply control circuit (VD1)	●

●: New or modified, ○: Common among the series

2.3 Directory Structure and File Configuration

The following indicates the directory structure of the S1C17700 series peripheral circuit sample software 2.

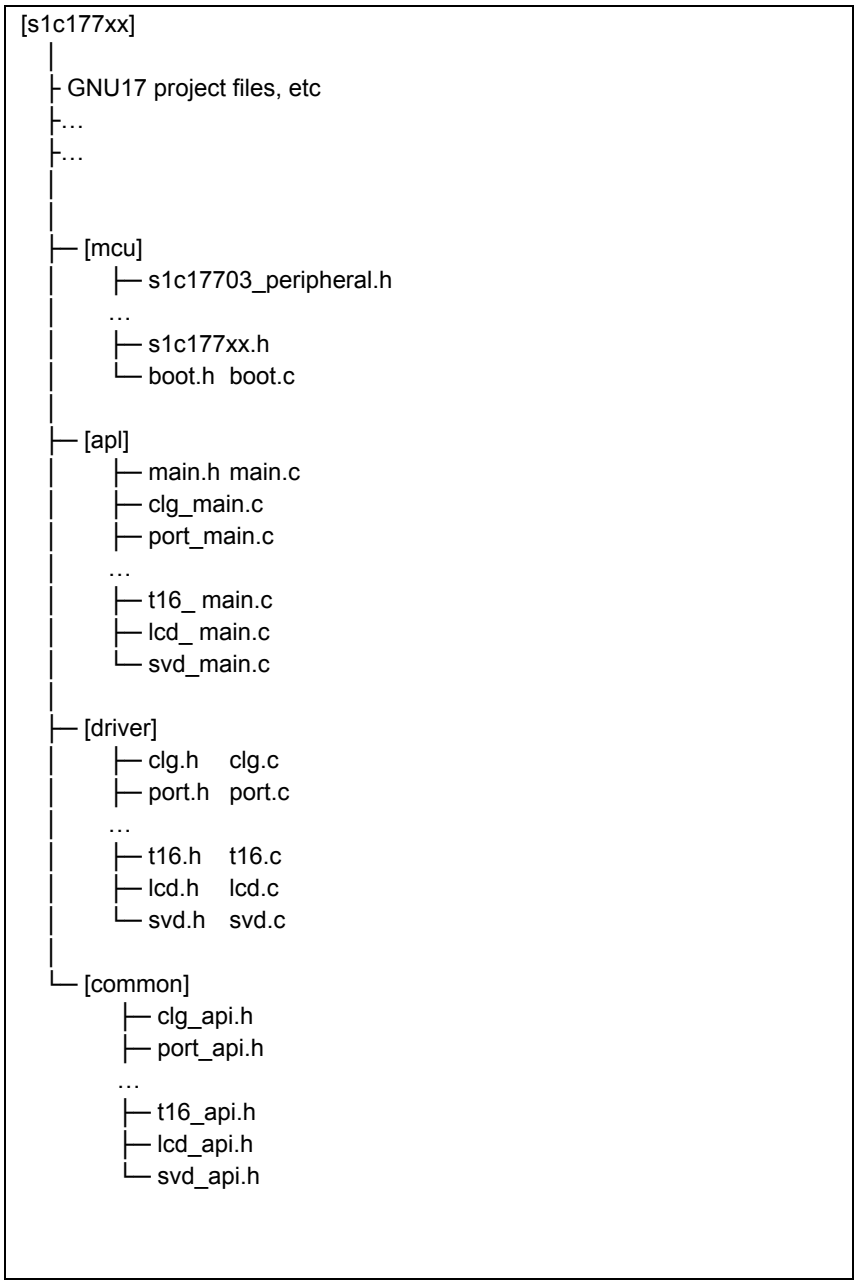

Figure 2.1 Directory structure of the S1C17700 series peripheral circuit sample software 2

(1) "s1c177xx" directory

This directory contains files related to the GNU17 project, and directories where the source code of the sample software is stored.

(2) "mcu" directory

This directory contains files for microcontroller initialization and files which define model-dependent information.

- Header files that define information such as the register address of respective model (s1c17703_peripheral.h, etc)
- Header file common among models (s1c177xx.h)
- Initialization file (boot.c)

(3) "apl" directory

This directory contains sample software for each peripheral circuit and header files which define constants and others used in the sample software.

- Header file for each peripheral circuit (xxx.h)
- Sample software for each peripheral circuit (xxx.c)

(4) "driver" directory

This directory contains sample drivers for each peripheral circuit.

- Header files which define register addresses and bit assignments for each peripheral circuit (xxx.h)
- Program for each peripheral circuit (xxx.c)

(5) "common" directory

This directory contains header files which define prototypes of externally accessible functions provided by sample drivers for each peripheral circuit.

- Header file that defines the constants of arguments and prototypes of functions provided externally by the sample driver of each peripheral circuit (xxx.h)

Software should include the header file in the "common" directory before calling the sample driver function.

2.4 Execution Method

Execute the S1C17700 series peripheral circuit sample software 2 in the following sequence.

(1) Import the project

Start up GNU17 and import a project of the S1C17700 series peripheral circuit sample software 2.

Refer to "3. Software Development Steps" in the "S5U1C17001C Manual" for how to import a project.

(2) Build the project

Build the S1C177xx project with GNU17.

Refer to "5. GNU17 IDE" in the "S5U1C17001C Manual" for how to build a project.

(3) Connect ICDmini

Connect ICDmini to the PC and development board, and turn the power on for the development board.

Refer to the "S5U1C17001H User Manual" for how to use ICDmini.

(4) Load and execute program using debugger

Press the [Debug Configurations] button of GNU17 to start the debugger, and press the [Resume] button in the debug view.

The program is loaded to S1C177xx and starts.

Refer to "10. Debugger" in the "S5U1C17001C Manual" for how to use the debugger.

2.5 Sample Software Menu

When the sample software starts up, a menu screen is displayed on the GNU17 Simulated I/O (hereinafter SimI/O).

Refer to "10.4.11 [Simulated I/O] view" in the "S5U1C17001C Manual" for information on Simulated I/O.

Entering the program number and pressing the [Enter] key starts the selected sample software.

See Chapter 3 for the details of each sample software item.

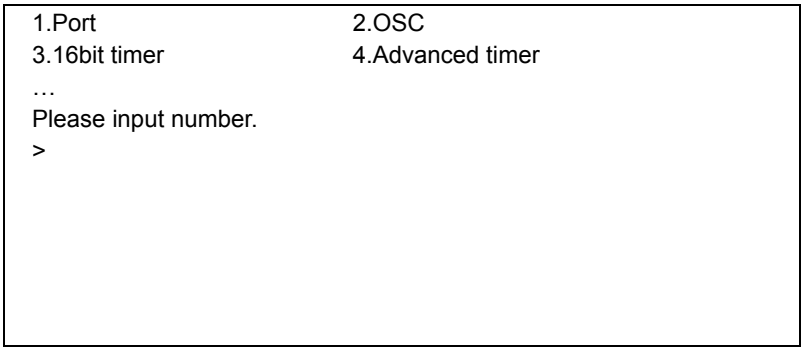

Figure 2.2 Menu screen display example

2.6 Build Method of Specific Modules

The S1C177xx sample software is distributed in the condition where multiple programs will be built.

However, you can build only the sample software for the required peripheral module by modifying the source code of the sample software. The steps are shown below.

(1) File to be modified

Modify the model-specific definition header.

For the case of the S1C17706 sample software, modify the s1c17706_peripheral.h file.

(2) Locations to be modified

Modify the following locations at the bottom of the file.

```
//#undef  PE_PORT
//#undef  PE_CLG
//#undef  PE_T16
//#undef  PE_T16A
//#undef PE_T16A2
//#undef  PE_CT
//#undef  PE_SWT
//#undef PE_RTC2
//#undef  PE_WDT
#undef   PE_UART
#undef   PE_UART_OSC3
#undef   PE_UART_IOSC
#undef   PE_SPI
#undef   PE_SPI_MASTER
#undef   PE_SPI_SLAVE
#undef   PE_I2CM
#undef   PE_I2CS
#undef   PE_LCD
#undef   PE_LCD32A
#undef   PE_SVD
#undef   PE_SVD2
#undef   PE_RFC
#undef   PE_ADC
#undef   PE_REMC
#undef   PE_REMC_TX
#undef   PE_REMC_RX
#undef   PE_SND
#undef   PE_SLEEP_HALT
#undef   PE_VD1
#undef   PE_CURRENT_MEASURE
```

Figure 2.3 Definition modification example for a specific module

For example, if building only the I/O port sample software, disable the "#undef PE_PORT" definition and enable other "#undef PE_XXX" definitions.

If the sample software for the peripheral module you are building uses other peripheral modules, you also need to build sample software for these peripheral modules.

For example, the I2CM sample software uses 16-bit timer; therefore, you need to disable "#undef PE_I2CM" and "#undef PE_T16" definitions when building the I2CM sample software.

3. Sample Software Function Details

This chapter describes the function details of the S1C17700 series peripheral circuit sample software 2.

3.1 Advanced timer (T16A2)

3.1.1 Sample software specifications

This sample software performs the following operations using the advanced timer.

- The software causes advanced timer compare A match interrupts five times in normal mode and acquires counter values of the timer.
- The software causes advanced timer compare B match interrupts five times each in normal and half clock modes and acquires counter values of the timer.
- The software outputs the PWM waveform to the TOUTA0 terminal in normal and half clock modes.
- The software reduces the power consumption while waiting for the interrupt by putting the CPU into HALT mode.

3.1.2 Hardware conditions

This sample software operates in the condition where the crystal oscillator or ceramic oscillator is connected to OSC1 and OSC3.

Refer to the section describing "Clock generator (CLG) oscillation circuit" in the respective technical manual for information on how to connect the oscillator.

Use this sample software by connecting the respective ports of the microcontroller as shown below.

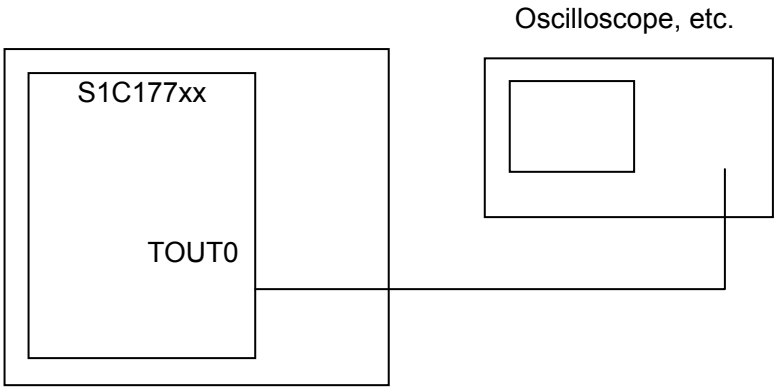

Figure 3.1 Hardware connection diagram for the advanced timer (T16A2) in the sample software

3.1.3 Operation overview

(1) Overview of sample software operations

- The sample software enables compare A match interrupts and compare B match interrupts, and then starts the advanced timer.
- The sample software acquires the counter value of the up counter when a compare A match interrupt or compare B match interrupt occurs.
- The sample software stops the advanced timer after the 5th compare B match interrupts, displays the interrupt type and counter value on SimI/O, and terminates the sample program.

```
<<< PWM timer(T16A2) demonstration start >>>
Normal clock mode start
*** PWM compare A interrupt :633 ***
*** PWM compare B interrupt :   0 ***
*** PWM compare A interrupt :633 ***
…
*** PWM Interrupt B interrupt:     0 ***

Half clock mode start
*** PWM Interrupt B interrupt:     0 ***
...
<<< PWM timer demonstration finish >>>
```

Figure 3.2 Display example of the advanced timer (T16A2) on the sample software screen

(2) Stopping the sample software

When all the operations described in the above "Overview of sample software operations" are completed, the sample software is terminated and the display returns to the menu screen.

3.2 LCD Driver (LCD32A)

3.2.1 Sample software specifications

This sample software performs the following operations using the LCD driver (LCD32A).

- The sample software turns all LCD pixels on and turns all LCD pixels off.
- The sample software turns on and off the specified COM/SEG.
- The sample software displays the LCD in reverse video.
- The sample software modifies the contrast and displays the LCD in 4-step gradation.
- The sample software displays the LCD in 4-step gradation using the grayscale function.

3.2.2 Hardware conditions

This sample software operates in the condition where the crystal oscillator or ceramic oscillator is connected to OSC1 and OSC3.

Refer to the section describing "Clock generator (CLG) oscillation circuit" in the respective technical manual for information on how to connect the oscillator.

Use this sample software by connecting the respective ports of the microcontroller as shown below.

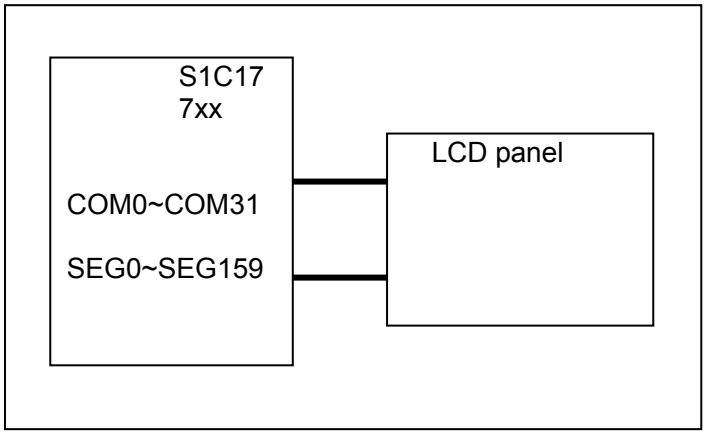

Figure 3.3 Hardware connection diagram of the LCD driver (LCD32A) in the sample software

3.2.3 Operations overview

(1) Overview of sample software operations

- When you enter "1" and press the Enter key from the menu, the sample software turns all pixels on on the LCD.
- When you enter "2" and press the Enter key from the menu, the sample software turns all pixels off on the LCD.
- When you enter "3" and press the Enter key from the menu, the sample software waits for the SEG/COM number, and when you enter "(SEG number), (COM number)" and press the Enter key, it turns the specified segment on.
- When you enter "4" and press the Enter key from the menu, the sample software waits for the SEG/COM number, and when you enter "(SEG number), (COM number)" and press the Enter key, it turns the specified segment off.
- When you enter "5" and press the Enter key from the menu, the sample software displays a checkered pattern followed by the same pattern in reserve video and repeats this sequence once more at a certain interval.

- When you enter "6" and press the Enter key from the menu, the sample software displays the LCD in 4-step gradation using the LCD32A interrupt. (The 1st and 5th lines are displayed in black, 2nd and 6th are in 75% gray, 3rd and 7th are in 50% gray, and 4th and 8th are in 25% gray.)
- When you enter "7" and press the Enter key from the menu, the sample software displays the LCD in 4-step gradation using the grayscale function. (The 1st is displayed in grayscale = 0, 2nd line is in grayscale = 1, 3rd line is in grayscale = 2, and 4th line is in grayscale = 3.)
- When you enter "8" and press the Enter key from the menu, the sample software displays Kanji characters.
- When you enter "9" and press the Enter key from the menu, the sample software terminates itself.

```
<<< LCD driver demonstration start >>>
1.All on                    2.All off
3.Turn dot on               4.Turn dot off
5.Reverse                   6.Grayscale1
7.Grayscale2                8.Kanji
9.exit

<<< LCD driver demonstration finish >>>
```

Figure 3.4 Display example of the LCD driver (LCD32A) on the sample software screen

(2) Stopping the sample software

Entering "9" and pressing the Enter key from the menu terminates the sample software and the display returns to the menu screen.

3.3 Power Supply Voltage Detection Circuit (SVD2)

3.3.1 Sample software specifications

This sample software performs the following operations using the power supply voltage detection circuit (hereinafter SVD2 circuit).

- The sample software detects the power supply voltage using the SVD2 circuit.

3.3.2 Hardware conditions

This sample software operates in the condition where the crystal oscillator or ceramic oscillator is connected to OSC1 and OSC3.

Use this sample software by specifying any power supply voltage.

3.3.3 Operations overview

(1) Overview of sample software operations

- The sample software detects the power supply voltage (VDD) using the SVD2 circuit and displays the current VDD voltage on SimI/O. The comparison voltage is in the range from 1.8V to 3.2V.
- The sample software displays "SVD interrupt did not occur" on SimI/O if the power supply voltage is less than 1.8V, or 3.2V or more.

```
<<< SVD demonstration start >>>
Vdd=2.5V
<<< SVD demonstration finish >>>
```

Figure 3.5 Display example of the power supply voltage detection circuit (SVD2) on the sample software screen

Note: The detected voltage may change depending on models. Check the source code of each model.

(2) Stopping the sample software

When all the operations described in the above "Overview of sample software operations" are completed, the sample software is terminated and the display returns to the menu screen.

3.4 Sound Generator (SND)

3.4.1 Sample software specifications

This sample software performs the following operations using the sound generator.

- The sample software outputs the waveform while raising the frequency in SND normal mode to the BZOUT terminal.

3.4.2 Hardware conditions

This sample software operates in the condition where the crystal oscillator or ceramic oscillator is connected to OSC1 and OSC3.

Refer to the section describing "Clock generator (CLG) oscillation circuit" in the respective technical manual for information on how to connect the oscillator.

Use this sample software by connecting the respective ports of the microcontroller as shown below.

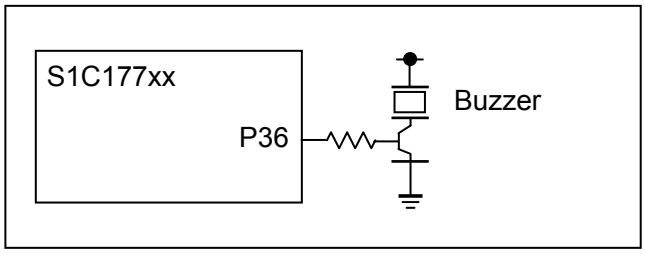

Figure 3.6 Hardware connection diagram for the sound generator (SND) in the sample software

3.4.3 Operations overview

(1) Overview of sample software operations

- The sample software starts the clock timer interrupt.
- The sample software emits a beep at the frequency of 1170.8Hz.
- The sample software puts the CPU in HALT mode.
- The sample software takes the CPU out of HALT mode and stops the beep when a clock timer interrupt occurs.
- Then the sample software emits a beep at a higher frequency for one second and returns to HALT mode. The above sequence is repeated until the frequency reaches 4096.0Hz. (The frequency is displayed on SimI/O.)
- The sample software terminates the clock timer and itself when it has completed the sequence for 4096.0Hz.

```
<<<Sound Generator demonstration start >>>
***Buzzer Frequency: 1170.8Hz***
***Buzzer Frequency: 1365.3Hz***
…
***Buzzer Frequency: 4096.0Hz***
<<< Sound Generator demonstration finish >>>
```

Figure 3.7 Display example of the sound generator (SND) on the sample software screen

(2) Stopping the sample software

When all the operations described in the above "Overview of sample software operations" are completed, the sample software is terminated and the display returns to the menu screen.

3.5 Real Time Clock (RTC2)

3.5.1 Sample software specifications

This sample software performs the following operations using the real time clock.

- The sample software displays the sample program menu for the real time clock.
- The sample software acquires the time from the real time clock.
- The sample software sets the time on the real time clock.
- The sample software displays the number of real time clock interrupts.

3.5.2 Hardware conditions

This sample software operates in the condition where the crystal oscillator or ceramic oscillator is connected to OSC1 and OSC3.

Refer to the section describing "Clock generator (CLG) oscillation circuit" in the respective technical manual for information on how to connect the oscillator.

3.5.3 Operations overview

(1) Overview of sample software operations

- When the sample program starts the RTC2 sample program menu is displayed.
- When you enter "1" and press the Enter key from the menu, the sample software obtains the time from RTC2 and displays in the 24-hour notation.
- When you enter "2" and press the Enter key from the menu, the sample software sets the time to RTC2.
- When you enter "3" and press the Enter key from the menu, the sample software displays the number of RTC2 interrupts.
- When you enter "4" and press the Enter key from the menu, the sample software terminates the RTC2 sample program.

```
<<< Real Time Clock demonstration start >>>
1.get RTC                          2.set RTC
3.indicate the count of interrupt  4.exit
Please input number.
> 1
10/03/01(Mon) 10:00:00

1.get RTC                          2.set RTC
3.indicate the count of interrupt  4.exit
...
> 4
<<< Real Time Clock demonstration finish >>>
```

Figure 3.8 Display example of the real time clock (RTC) on the sample software screen

(2) Stopping the sample software

Entering "4" and pressing the Enter key from the menu terminates the sample software and the display returns to the menu screen.

3.6 Electric Current Measurement

3.6.1 Sample software specifications

This sample software drives the CPU in the following status in order to measure the electric current.

- The sample software puts the CPU into SLEEP mode.
- The sample software puts the CPU into HALT mode while oscillating only OSC1.
- The sample software puts the CPU into HALT mode while oscillating OSC1 and OSC3.
- The sample software puts the CPU into HALT mode while oscillating OSC1, OSC3, and IOSC.

3.6.2 Hardware conditions

This sample software operates in the condition where the crystal oscillator or ceramic oscillator is connected to OSC3.

Refer to the section describing "Clock generator (CLG)" in the "S1C177xx Series Technical Manual" for information on how to connect the oscillator.

3.6.3 Execution method

Build this sample software as described below to make it executable.

(1) Program to be modified

Modify the model-specific definition header.

For the case of the S1C17706 sample software, modify the s1c17706_peripheral.h file.

(2) Locations to be modified

Modify the following locations at the bottom of the file.

```
//#undef   PE_SND
//#undef   PE_SLEEP_HALT
//#undef   PE_VD1
//#undef   PE_CURRENT_MEASURE

/*************************************************************************/
/* Current measurement mode selecting                                    */
/*************************************************************************/
// Comment out "#undef        PE_CURRENT_MEASURE", when using current measure program.
#ifdef PE_CURRENT_MEASURE
#define CURRENT_MEASURE_SLEEP                          (0x00)
#define CURRENT_MEASURE_HALT_OSC1                      (0x01)
#define CURRENT_MEASURE_HALT_OSC1_OSC3                 (0x02)
#define CURRENT_MEASURE_HALT_OSC1_OSC3_IOSC            (0x03)
// Remove the comment on the selected measurement mode.
//#define CURRENT_MEASURE_MODE CURRENT_MEASURE_SLEEP
//#define CURRENT_MEASURE_MODE CURRENT_MEASURE_HALT_OSC1
#define CURRENT_MEASURE_MODE CURRENT_MEASURE_HALT_OSC1_OSC3
//#define CURRENT_MEASURE_MODE CURRENT_MEASURE_HALT_OSC1_OSC3_IOSC
#endif
```

Figure 3.9 Modification example for electric current measurement sample software definitions

For example, when putting the CPU into HALT mode while oscillating OSC1 and OSC3, disable the "#undef PE_CURRENT_MEASURE" definition and enable the "#define CURRENT_MEASURE_MODE CURRENT_MEASURE_HALT_OSC1_OSC3" definition before building.

(3) Loading and executing program by debugger

Start the debugger and load the sample software to S1C177xx.

(4) Executing the program

Remove ICDmini from the development board and reset S1C177xx in order to eliminate the influence from the electric current from ICDmini.

(5) Operation overview

- The electric current measurement sample software is executed after the reset.

Carry out step (2) to (4) if you want to measure the electric current in a different condition.

(6) Stopping the sample software

There is no procedure for stopping. Simply stop the power supply to the development board.

4. List of Sample Driver Functions

This chapter lists the sample drivers for each peripheral circuit.

4.1 Advanced timer (T16A2)

Table 4.1 shows the list of functions of this sample driver. Refer to source code t16a2.c for details of the function.

Table 4.1 List of functions of the advanced timer (T16A2) sample driver

Function Name	Description Name
T16A2_setInputClock	Input clock setting
T16A2_controlInputClock	Input clock supply allow/disallow setting
T16A2_init	Advanced time initialization
T16A2_setTimerMode	Advanced timer mode setting
T16A2_setComparatorCapture	Comparator/capture setting
T16A2_getCounterData	Counter data acquisition
T16A2_setCompareData	Compare data setting
T16A2_getCaptureData	Capture data acquisition
T16A2_resetTimer	Advanced timer reset
T16A2_setTimerRun	Advanced timer start/stop setting
T16A2_initInt	Advanced timer interrupt initialization
T16A2_controlInt	Advanced timer interrupt allow/disallow setting
T16A2_resetIntFlag	Advanced timer interrupt factor flag reset
T16A2_checkIntFlag	Advanced timer interrupt factor flag check

You can find descriptions of this sample driver in 16a2.c, t16a2.h, and t16a2_api.h.

Include t16a2_api.h in programs which use this sample driver.

4.2 LCD Driver (LCD)

Table 4.2 shows the list of functions of this sample driver. Refer to source code lcd.c for details of the function.

Table 4.2 List of functions of the LCD driver (LCD) sample driver

Function Name	Description Name
LCD_initPower	LCD power supply initialization
LCD_controlBooster	Power supply voltage booster circuit setting
LCD_setRegulatorSource	Power supply voltage selection for the LCD system voltage regulator circuit
LCD_init	LCD initialization
LCD_setSEGAssignment	SEG terminal memory assignment setting
LCD_setCOMAssignment	COM terminal memory assignment setting
LCD_setDisplayArea	LCD display area setting
LCD_setDisplayReverse	LCD reverse video display setting
LCD_controlDisplay	LCD display control
LCD_setContrast	LCD contrast setting
LCD_display1Seg	1 segment display
LCD_initInt	LCD interrupt initialization
LCD_controlInt	LCD interrupt allow/disallow setting
LCD_resetIntFlag	LCD interrupt factor flag reset
LCD_checkIntFlag	LCD interrupt factor flag check
LCD_setCOMPinAssignment	COM terminal pin assignment setting
LCD_displayDraw	Rectangle display
LCD_setLCDClock	LCD clock setting
LCD_controlLCDClock	LCD clock supply allow/disallow setting
LCD_allClear	LCD display area all clear

You can find descriptions of this sample driver in lcd.c, lcd.h, and lcd_api.h.

Include lcd_api.h in programs which use this sample driver.

4.3 LCD Driver (LCD32A)

Table 4.3 shows the list of functions of this sample driver. Refer to source code lcd32a.c for details of the function.

Table 4.3 List of functions of the LCD driver (LCD32A) sample driver

Function Name	Description Name
LCD32A_initPower	LCD power supply initialization
LCD32A_controlBooster	Power supply voltage booster circuit setting
LCD32A_setRegulatorSource	Power supply voltage selection for the LCD system voltage regulator circuit
LCD32A_init	LCD32A initialization
LCD32A_setSEGAssignment	SEG terminal memory assignment setting
LCD32A_setCOMAssignment	COM terminal memory assignment setting
LCD32A_setDisplayArea	LCD32A display area setting
LCD32A_setDisplayReverse	LCD32A reverse video display setting
LCD32A_controlDisplay	LCD32A display control
LCD32A_setContrast	LCD32A contrast setting
LCD32A_controlGrayscale	Grayscale function setting
LCD32A_display1Seg	1 segment display
LCD32A_initInt	LCD32A interrupt initialization
LCD32A_controlInt	LCD32A interrupt allow/disallow setting
LCD32A_resetIntFlag	LCD32A interrupt factor flag reset
LCD32A_checkIntFlag	LCD32A interrupt factor flag check
LCD32A_setCOMPinAssignment	COM terminal pin assignment setting
LCD32A_displayDraw	Rectangle display
LCD32A_controlComDrive	COM terminal drive allow/disallow setting
LCD32A_controlSegDrive	SEG terminal drive allow/disallow setting
LCD32A_setLCDClock	LCD32A clock setting
LCD32A_controlLCDClock	LCD32A clock supply allow/disallow setting
LCD32A_allClear	LCD32A display area all clear

You can find descriptions of this sample driver in lcd32a.c, lcd32a.h, and lcd32a_api.h.

Include lcd32a_api.h in programs which use this sample driver.

4.4 Power Supply Voltage Detection Circuit (SVD2)

Table 4.4 shows the list of functions of this sample driver. Refer to source code svd2.c for details of the function.

Table 4.4 List of functions of the power supply voltage detection circuit (SVD2) sample driver

Function Name	Description Name
SVD2_setCompareVoltage	SVD2 comparison voltage setting
SVD2_setCompareMode	SVD2 comparison mode and times setting
SVD2_setResetMode	SVD2 reset mode setting
SVD2_controlDetection	SVD2 detection start/stop setting
SVD2_setDetectionResult	SVD2 detection result acquisition
SVD2_initInt	SVD2 interrupt initialization
SVD2_controlInt	SVD2 interrupt allow/disallow setting
SVD2_resetIntFlag	SVD2 interrupt factor flag reset
SVD2_checkIntFlag	SVD2 interrupt factor flag check
SVD2_setSVD2Clock	SVD2 clock setting
SVD2_controlSVD2Clock	SVD2 clock supply allow/disallow setting

You can find descriptions of this sample driver in svd2.c, svd2.h, and svd2_api.h.

Include svd2_api.h in programs which use this sample driver.

4.5 Sound Generator (SND)

Table 4.5 shows the list of functions of this sample driver. Refer to source code snd.c for details of the function.

Table 4.5 List of functions of the sound generator (SND) sample driver

Function Name	Description Name
SND_init	SND initialization
SND_controlSND	SND beep start/stop setting
SND_setSNDMode	SND mode setting
SND_setSNDQuality	SND frequency setting
SND_setSNDDuty	SND volume setting
SND_setSNDClock	SND clock setting
SND_controlSNDClock	SND clock supply allow/disallow setting

You can find descriptions of this sample driver in snd.c, snd.h, and snd_api.h.

Include snd_api.h in programs which use this sample driver.

4.6 Real Time Clock (RTC2)

Table 4.6 shows the list of functions of this sample driver. Refer to source code rtc2.c for details of the function.

Table 4.6 List of functions of the real time clock (RTC2) sample driver

Function Name	Description Name
RTC2_init	RTC initialization
RTC2_setTimerRun	RTC start/stop setting
RTC2_setTimerMode	RTC24/12H mode setting
RTC2_setAdj	RTC 30-second adjustment setting
RTC2_setTime	RTC time setting
RTC2_getTime	RTC time acquisition
RTC2_checkBusy	RTC busy check
RTC2_initInt	RTC interrupt initialization
RTC2_controlInt	RTC interrupt allow/disallow setting
RTC2_resetIntFlag	RTC2 interrupt factor flag reset
RTC2_checkIntFlag	RTC2 interrupt factor flag check
RTC2_get1HzCounter	RTC2 1Hz counter acquisition

You can find descriptions of this sample driver in rtc2.c, rtc2.h, and rtc2_api.h.

Include rtc2_api.h in programs which use this sample driver.

4.7 Power Supply Control Circuit (VD1)

Table 4.7 shows the list of functions of this sample driver. Refer to source code vd1.c for details of the function.

Table 4.7 List of functions of the power supply control circuit (VD1) sample driver

Function Name	Description Name
VD1_setMode	Heavy load mode setting

You can find descriptions of this sample driver in vd1.c, vd1.h, and vd1_api.h.

Include vd1_api.h in programs which use this sample driver.

4.8 Multiplexer (MUX)

Table 4.8 shows the list of functions of this sample driver. Refer to source code mux.c for details of the function.

Table 4.8 List of functions of the multiplexer (MUX) sample driver

Function Name	Description Name
MUX_init	MUX initialization
MUX_setREMCport	REMC port setting
MUX_setADCport	ADC port setting
MUX_setSPIport	SPI port setting
MUX_setUARTport	UART port setting
MUX_setRFCport	RFC port setting
MUX_setI2CMport	I2C master port setting
MUX_setI2CSport	I2C slave port setting
MUX_setLCDport	LCD port setting
MUX_setDBGport	Debug port setting
MUX_setCLGport	CLG port setting
MUX_setT16Aport	Advanced timer port setting
MUX_setT16A2port	Advanced timer (T16A2) port setting
MUX_setSNDport	Sound generator (SND) port setting

You can find descriptions of this sample driver in mux.c, mux.h, and mux_api.h.

Include mux_api.h in programs which use this sample driver.

4.9 Watchdog Timer (WDT)

Table 4.9 shows the list of functions of this sample driver. Refer to source code wdt.c for details of the function.

Table 4.9 List of functions of the watchdog timer (WDT) sample driver

Function Name	Description Name
WDT_resetTimer	Watchdog timer reset
WDT_setTimerRun	Watchdog timer start/stop setting
WDT_setTimerMode	Watchdog timer mode setting
WDT_checkNMI	Watchdog timer NMI occurrence check
WDT_setCountClock	Watchdog timer clock setting
WDT_controlCountClock	Watchdog timer clock supply allow/disallow setting

You can find descriptions of this sample driver in wdt.c, wdt.h, and wdt_api.h.

Include wdt_api.h in programs which use this sample driver.

Revision History

Attachment-1

Rev. No.	Date	Page	Category	Contents
Rev 1.0	2010/10/xx	All	new	

EPSON

International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.
2580 Orchard Parkway,
San Jose, CA 95131, 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.
20/F, Harbour Centre, 25 Harbour Road,
Wanchai, Hong Kong
Phone: +852-2585-4600 FAX: +852-2827-4346
Telex: 65542 EPSCO HX

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

Device Sales & Marketing Dept.
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 FAX: +81-42-587-5117

Document Code: 412017400
First Issue October 2010 in JAPAN Ⓥ