EPSON
EXCEED YOUR VISION

S1C17W23
Photoplethysmography (PPG)
Application Notes

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

No part of this material may be reproduced or duplicated in any form or by any means without the written permission of Seiko Epson. Seiko Epson reserves the right to make changes to this material without notice. Seiko Epson does not assume any liability of any kind arising out of any inaccuracies contained in this material or due to its application or use in any product or circuit and, further, there is no representation that this material is applicable to products requiring high level reliability, such as, medical products. Moreover, no license to any intellectual property rights is granted by implication or otherwise, and there is no representation or warranty that anything made in accordance with this material will be free from any patent or copyright infringement of a third party. When exporting the products or technology described in this material, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You are requested not to use, to resell, to export and/or to otherwise dispose of the products (and any technical information furnished, if any) for the development and/or manufacture of weapon of mass destruction or for other military purposes.

All brands or product names mentioned herein are trademarks and/or registered trademarks of their respective companies.

©SEIKO EPSON CORPORATION 2016, All rights reserved.

Summary

This document is intended to provide the reference material for measuring pulse waves by photoplethysmography (PPG) using the S1C17W23, an LED, and an optical sensor (phototransistor or photodiode) to obtain heart rate values.

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 FLS17W23 (file name: fls17w23.elf)
  This file is mandatory for programming the embedded flash memory.
- S1C17W23 PPG Programming Package (this package)
  - Time Variation of Optical Sensor Output Visualization Programming Package
    (s1c17w23_ppg_mon_gnu17vx)
    Excel file with VBA macro included (PpgMon64.xlsm), VBS file (DoPpg64.vbs), and Active-X control file (NonComSck.ocx)
  - Pulse Wave to Heart Rate Conversion and LCD Display Programming Package
    (s1c17w23_ppg_demo_gnu17vx)

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

Table of Contents

1. Specifications

The program described in this application note measures pulse waves by photoplethysmography (PPG) to obtain heart rate values. An AFE is used for amplifying and filtering the optical sensor (phototransistor or photodiode) output signal to obtain the sensed waveform with the voltage level that can be input to the A/D converter (ADC12A) embedded in the S1C17W23.

The sample program s1c17w23_ppg_mon_gnu17vx samples the optical sensor output values in set intervals and sends them to the PC via the UART to write to an Excel sheet in conjunction with Excel including a VBA macro and a VB script (VBS). The values in the array written to the Excel sheet are graphed, this makes it possible to visualize the variations in the optical sensor output values on the time series.

The sample program s1c17w23_ppg_demo_gnu17vx fetches the optical sensor output values periodically in set intervals using the 16-bit timer (T16) and calculates the heart rate from the sensed pulse wave. Then it displays the calculated results on the LCD panel. Also it displays the states of pulse wave and pulsation in real time.

Figure 1-1 shows the evaluation system used in Appendix A.

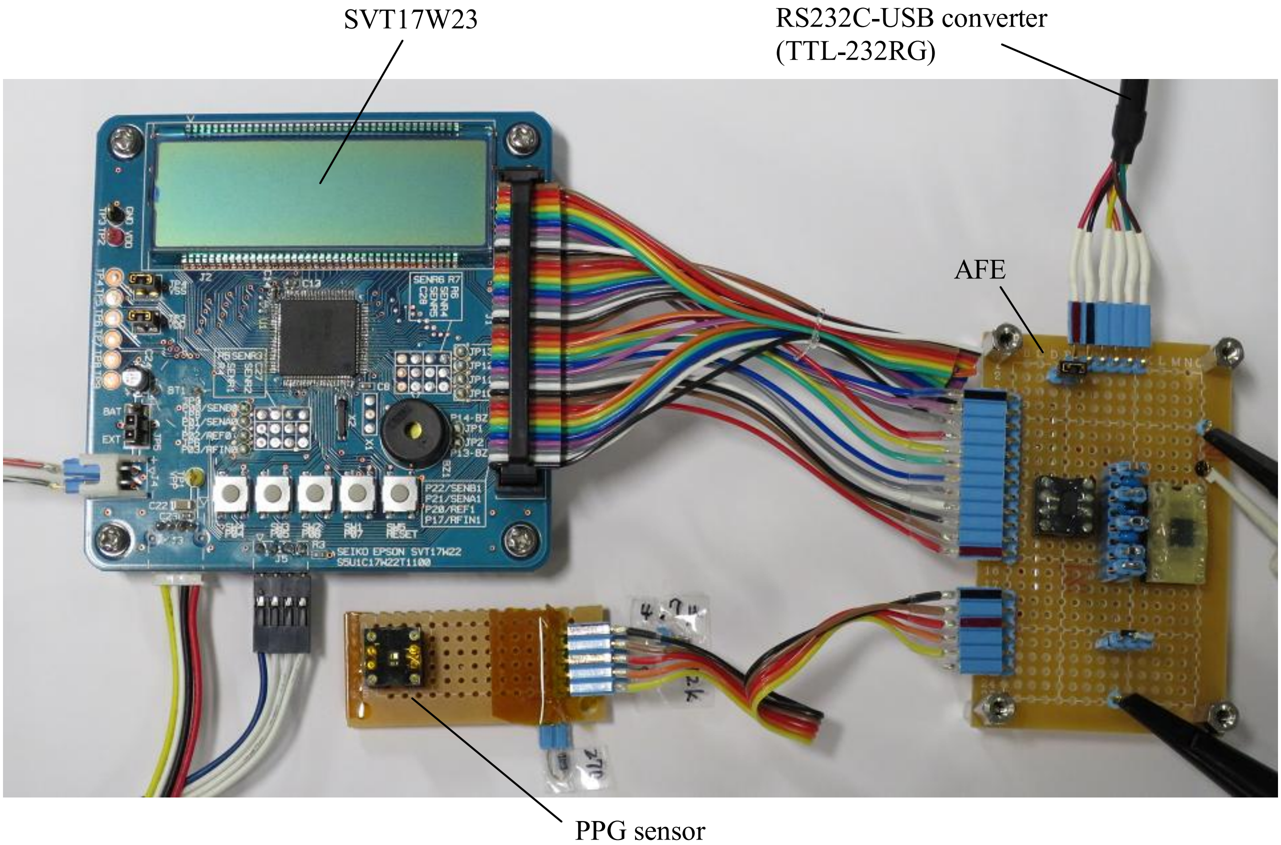


Figure 1-1 Evaluation System Used In Appendix A

2. Descriptions of the Functions Used

PPORT	Five ports, P10, P15, P16, P17, and P20, are connected to FET gates and analog switches and used to drive an LED and to switch analog circuits.
ADC12A Ch.0	Used to convert the optical sensor output values after the amplification and filtering processing by the AFE has completed into the digital values.
T16 Ch.0	Used as an interval timer to run the ADC12A periodically.
T16 Ch.1	Used as a timer to wait for the stabilization time before starting sampling in intermittent drive mode.
T16 Ch.3	Used as a timer to generate the ADC12A operating clock.

The peripheral circuit shown below is used in the s1c17w23_ppg_mon_gnu17vx 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 and VB script executed on the PC.

The peripheral circuit shown below is used in the s1c17w23_ppg_demo_gnu17vx program.

LCD24	Used to display the heart rate determined from the pulse wave on the LCD panel.

System clock
OSC3 (4 MHz internal oscillator) is used as the system clock.
OSC1 (32.768 kHz) is also used in s1c17w23_ppg_demo_gnu17vx.

Interrupts
The following shows the ADC12A Ch.0 vector number and vector address:
    ADC12A Ch.0 vector number: 26 (0x1a)
    ADC12A Ch.0 vector address: 0x8068
The sample program uses the following two interrupts:
    Analog input signal 0 A/D conversion completion interrupt
    Analog input signal 0 A/D conversion result overwrite error interrupt

The following shows the T16 Ch.0 vector number and vector address:
    T16 Ch.0 vector number: 9 (0x09)
    T16 Ch.0 vector address: 0x8024
The sample program uses the following interrupt:
    Underflow interrupt

The following description is applied to s1c17w23_ppg_mon_gnu17vx.

The following shows the UART Ch.0 vector number and vector address:
    UART Ch.0 vector number: 10 (0x0a)
    UART Ch.0 vector address: 0x8028
The sample program uses the following interrupt:
    Receive buffer one byte full interrupt

Figure 2-1 shows the ADC12A configuration.

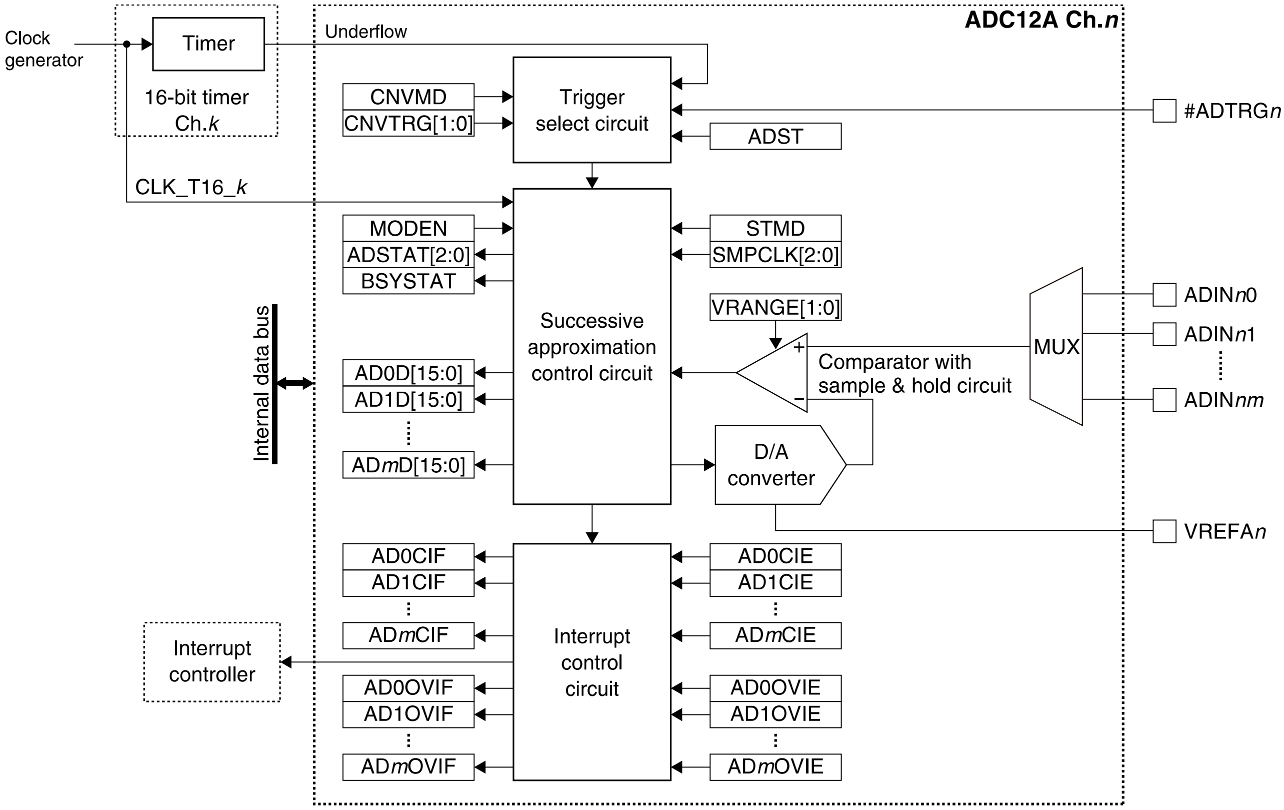


Figure 2-1 ADC12A Configuration

3. Principle of Operation

3.1 Principle of Detection

A pulse wave is a wave motion caused by propagation of a change in pressure within the blood vessel, which is generated by contraction of the heart, from the aorta into peripheral blood vessels. The change in pressure within a blood vessel due to the wave motion is detected as a pressure pulse wave; the change in volume of a blood vessel is detected as plethysmogram.

An optical technique, called photoplethysmography (PPG), is used to detect plethysmogram. There are two photoelectric pulse wave detection methods: transmission type and reflection type.

The transmission type detector measures pulse waves by holding the measuring part between its light emitting part and light receiving part. Only a fingertip or an earlobe can be used as the measuring part. The reflection type detector can be used by sticking it on the measurement part arbitrarily selected. Hemoglobin in blood has strong absorption spectrums in the light of a certain wavelength band. The intensity of the transmitted or reflected light of a living body irradiated with light having this wavelength band is changed according to the amount of hemoglobin varying with the volume variations of the blood vessel. Pulse waves can be detected by converting this intensity of the transmitted or reflected light into an electrical signal. [1)]

As developed with this technology, the pulse oximeter, which monitors pulse rate and percutaneous arterial blood oxygen saturation (SpO2) by attaching a probe to a fingertip or earlobe without invasion and is used to grasp a patient's condition such as during a surgical operation, is well-known. As shown in Figure 3-1, Absorption of Human Blood (Oxyhemoglobin and Hemoglobin) vs. Wavelength of Light [2)], the pulse oximeter uses a principle that can determine arterial blood oxygen saturation by using light sources with a different wavelength. Specifically, it is common to illuminate skin using two or more LEDs with a different wavelength as the light source and obtain a signal of the transmitted or reflected light intensity using an optical sensor such as a phototransistor or photodiode.

This application note is intended to measure pulses, so it includes a green light source, which is hardly affected by arterial blood oxygen saturation as shown in Figure 3-1 and has a high absorptivity, into the experimental conditions.

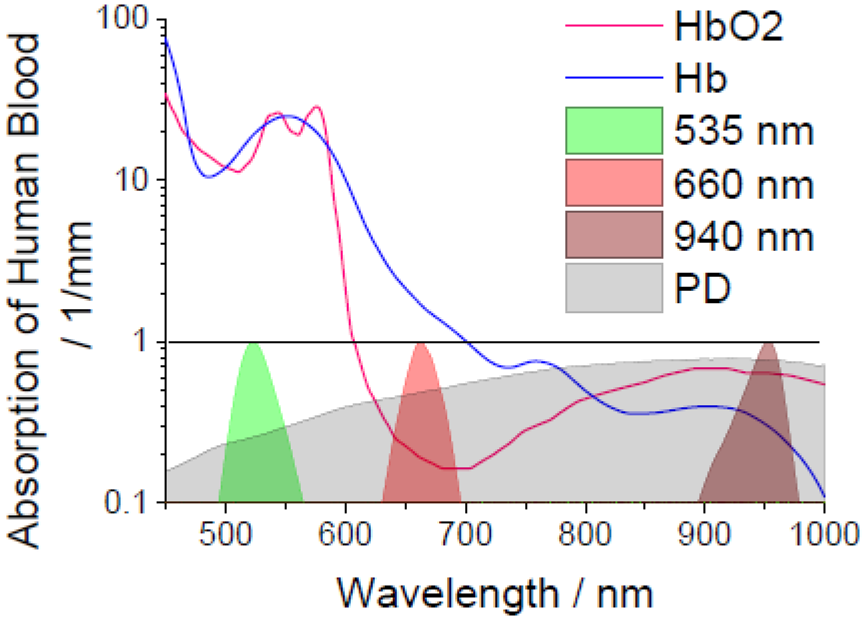


Figure 3-1 Absorption of Human Blood (Oxyhemoglobin and Hemoglobin) vs. Wavelength of Light [2)]

3.2 Method of Detection

This application note uses a reflection type photoelectric pulse wave detector. Figure 3-2 shows the outline circuit block diagram used for reflection type photoelectric pulse wave measurement.

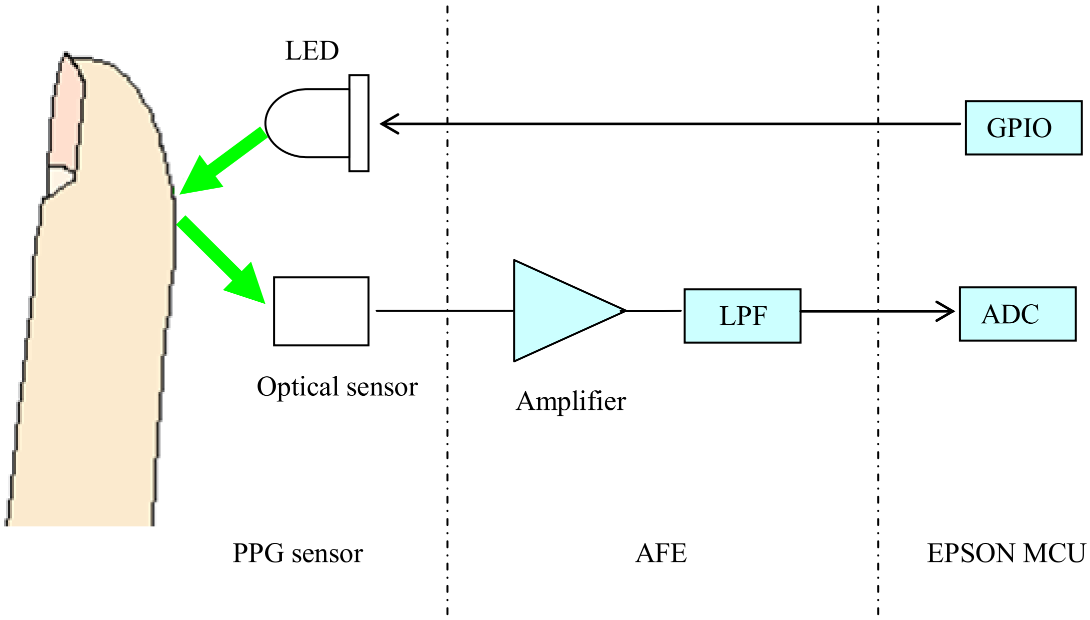
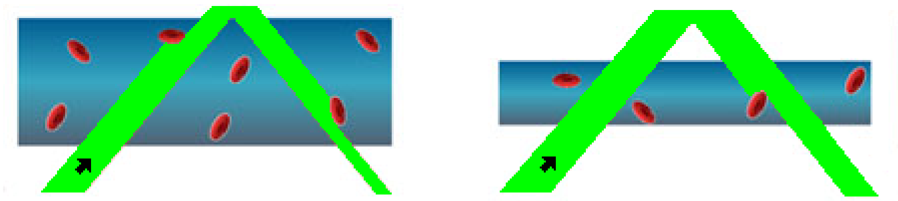


Figure 3-2 Circuit Block Diagram Used For Reflection Type Photoelectric Pulse Wave Measurement

A basic operation is as follows: (1) The GPIO output controls the LED on and off to irradiate light to a fingertip. (2) The optical sensor receives light that was absorbed by hemoglobin in capillary vessels of the finger and then reflected by the bones. (3) The photoelectric current signal that varies with the pulse waves is processed through the amplifier and LPF in AFE. (4) The MCU converts it into digital values using the embedded ADC.

As shown in Figure 3-3, the volume of blood increases at the rising point of the pulse as the blood stream increases (a). This increases red blood cells that absorbs light and the intensity of reflected or transmitted light becomes lower than that in a static condition (b). Thus it can be detected as pulse waves. The detection method in this application note adopts this operation principle.



(a) Blood vessel at rising of pulse (b) Blood vessel in static condition

Figure 3-3 Reason for Light Intensity to Vary with Pulsation [3)]

3.3 Detection Circuit

Pulse wave detector can be broadly divided into two methods: continuous drive method in which the LED is continuously lighted during pulse wave measurement, and intermittent drive method in which the LED lights only while the ADC is converting the optical sensor output value. Figures 3-4 and 3-5 show an example of continuous-drive pulse wave detector AFE circuit [4] and an example of intermittent-drive pulse wave detector AFE circuit, respectively.

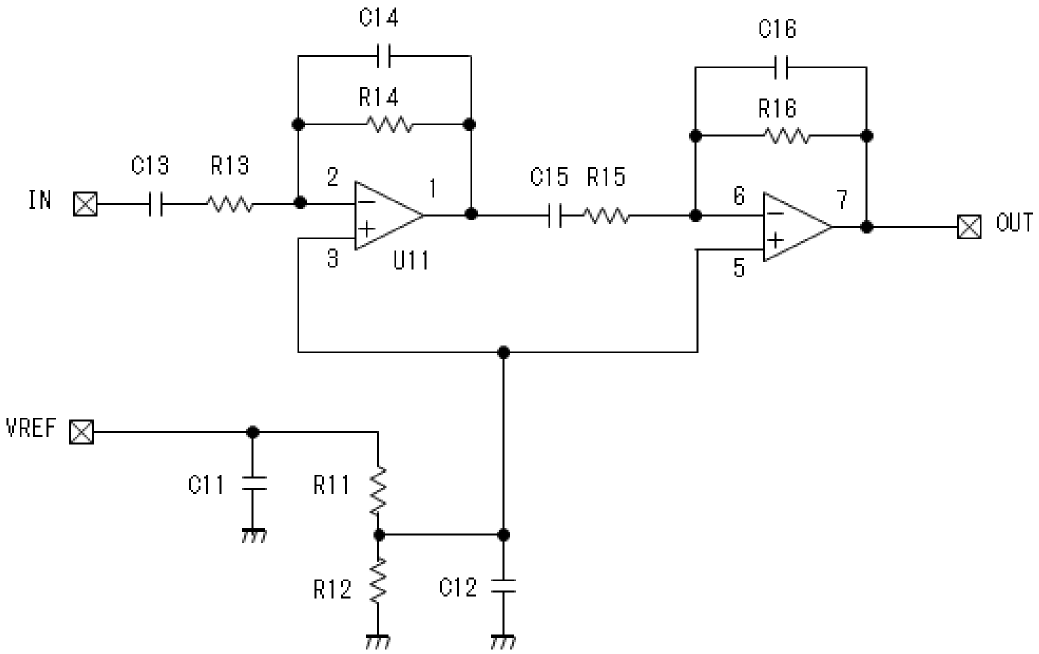


Figure 3-4 Example of Continuous-Drive Pulse Wave Detector AFE Circuit [4]

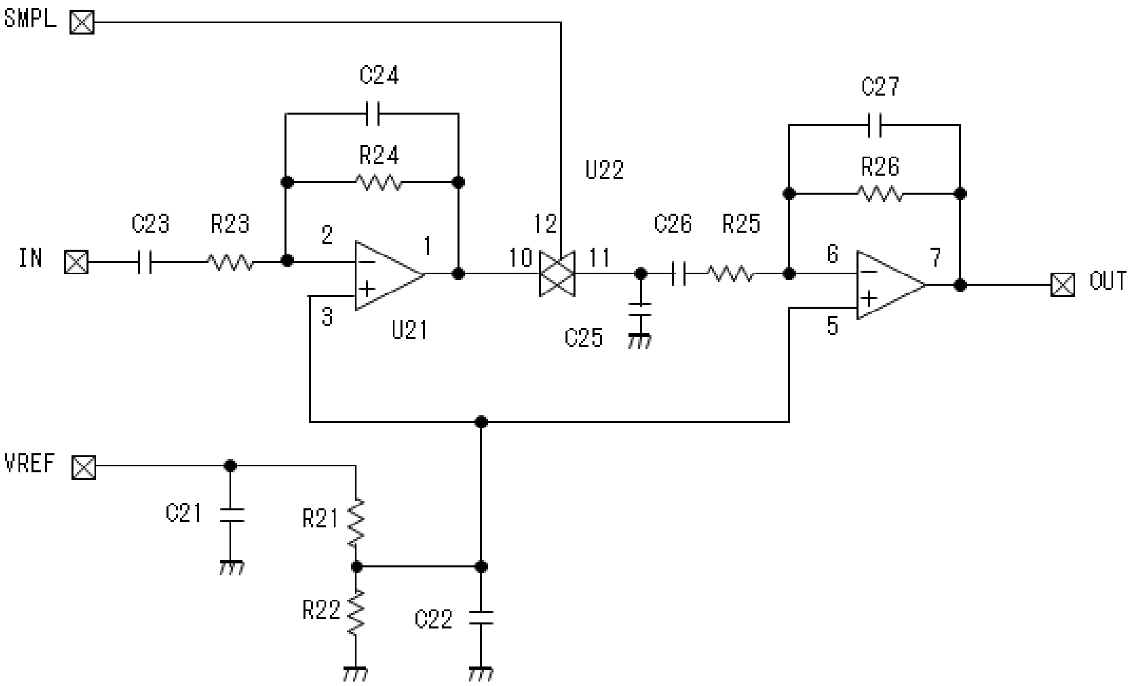


Figure 3-5 Example of Intermittent-Drive Pulse Wave Detector AFE Circuit

Appendix A in this application note uses the circuit shown in Figure 3-6 to check the behavior by electronically switching the two drive modes.

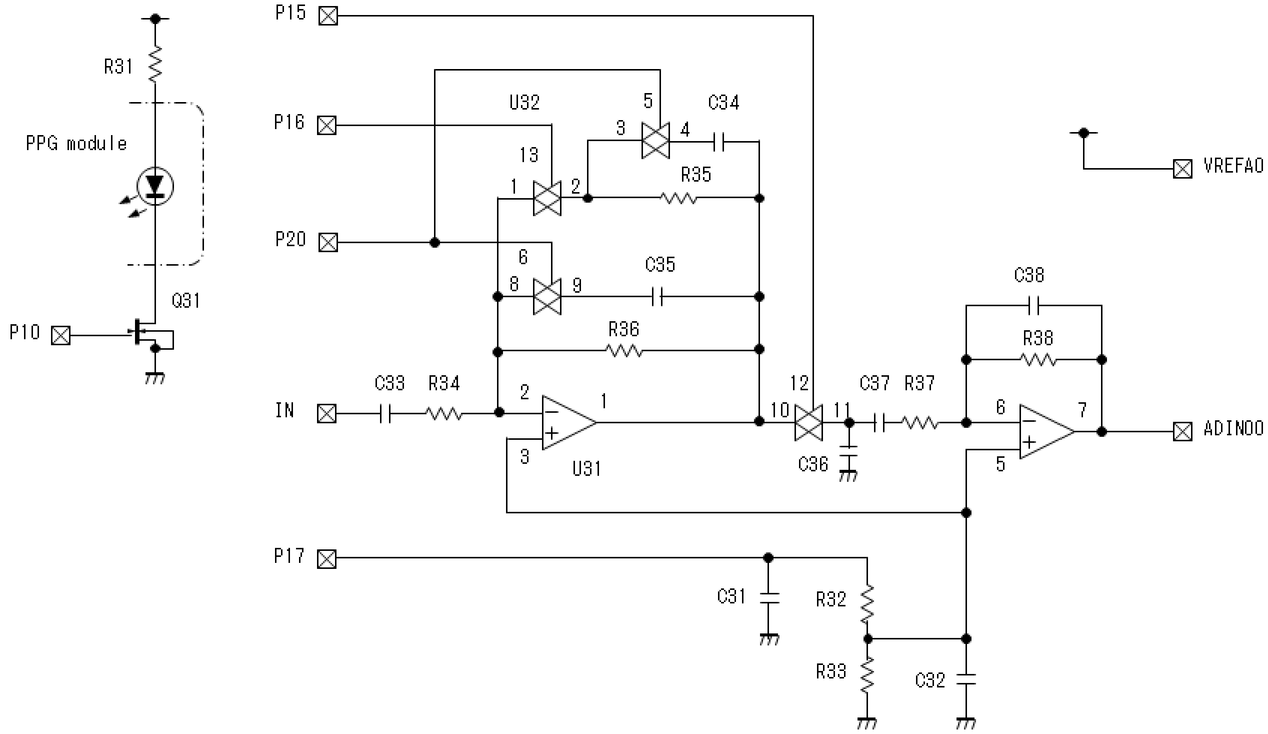


Figure 3-6 Circuit Used in this Application Note

P10, P15, P16, P17, and P20 in the above circuit are connected to the SVT17W23 GPIO ports. Table 3-1 lists their functions.

Table 3-1 P10, P15, P16, P17, and P20 Functions

GPIO	At 'L' level	At 'H' level	Remarks
P10	Turn LED off	Turn LED on	
P15	Hold data	Sample data	Always 'H' in continuous drive mode
P16	Set amplifier gain to high	Set amplifier gain to low	
P17	Disable V_{REF} supply	Enable V_{REF} supply	
P20	Disable preamplifier LPF	Enable preamplifier LPF	

Table 3-2 lists the parts used in the above circuit except for resistor R31 connected in series to the LED of the PPG module.

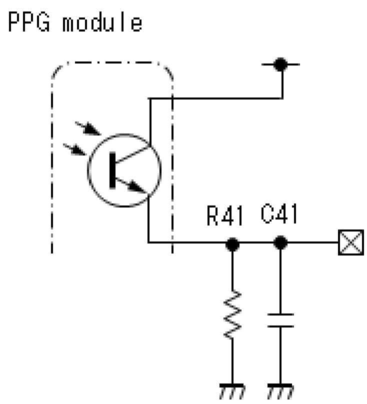
Table 3-2 Parts List for Circuit in Figure 3-6

Name	Symbol	Manufacture	Product number	Specification
Operational amplifier	U31	STMicroelectronics	LMV358LIDT	
Analog switch	U32	Texas Instruments	SN74HC4066PWR	
FET	Q31	NXP Semiconductors	2N7002P, 215	
Resistor	R32	ROHM	MCR01 Series Size = 1005 (0402), Tol = F	47 kΩ
	R33			47 kΩ
	R34			15 kΩ
	R35			560 kΩ
	R36			2.7 MΩ
	R37			15 kΩ
	R38			180 kΩ
Capacitor	C31	muRata	GRM185B10J105KE21	1 µF, 6.3 V, B
	C32		GRM185B10J105KE21	1 µF, 6.3 V, B
	C33		GRM21BF10J106ZE01	10 µF, 6.3 V, F
	C34		GRM155B31H103KA88	0.01 µF, 50 V, B
	C35		GRM155B11H222KA01	2,200 pF, 50 V, B
	C36		GRM155B31C104KA87	0.1 µF, 16 V, B
	C37		GRM21BF10J106ZE01	10 µF, 6.3 V, F
	C38		GRM188B11E333KA01	0.033 µF, 25 V, B

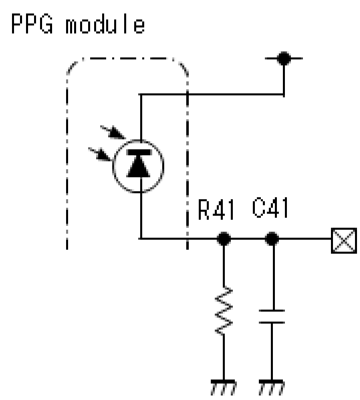
Note that the connection depends on the optical sensor used. Figures 3-7 (a) and (b) show an example of a phototransistor circuit and an example of a photodiode circuit, respectively.

The experiment performed for this application note evaluated four PPG sensors listed in Table 3-3. Table 3-4 lists the component values of the parts used for each module. "@5 mA" and "@10 mA" described with R31 in this table mean the current made to flow through the LED. The experiment described in Appendix A used the component values for a 5 mA condition. To increase the PPG sensor sensitivity easily, the component values for a 10 mA condition may be selected as they satisfy the absolute maximum rating conditions.

If noise does not cause a problem, C41 may be removed to improve the sensor's time responsibility.



(a) Phototransistor Circuit



(b) Photodiode Circuit

Figure 3-7 Examples of Optical Sensor Circuit

Table 3-3 PPG Sensors Used in this Application Note

Manufacture	Product number	LED color	Optical sensor
New Japan Radio	NJL5303R-TE1	Green	Phototransistor
Fairchild Semiconductor	QRE1113GR	Infrared ray	Phototransistor
OSRAM Opto Semiconductors	SFH 7050	Green, Red, Infrared ray	Photodiode
ROHM Semiconductor	RPR-220C1N	Infrared ray	Phototransistor

Table 3-4 Component Values for PPG Sensor Used in this Application Note

Product number	LED color	R31@5mA	(R31@10mA)	R41	C41
NJL5303R-TE1	Green	270 Ω	120 Ω	82 kΩ	4.7 μF
QRE1113GR	Infrared ray	470 Ω	220 Ω	15 kΩ	10 μF
SFH 7050	Green	68 Ω	15 Ω	220 kΩ	0.22 μF
	Red	330 Ω	150 Ω	47 kΩ	
	Infrared ray	470 Ω	220 Ω	56 kΩ	
RPR-220C1N	Infrared ray	470 Ω	220 Ω	82 kΩ	10 μF

Although there is no description in the circuit diagram, it is necessary to connect between SVT17W23 and the PC via UART when executing the Time Variation of Optical Sensor Output Visualization Programming Package (s1c17w23_ppg_mon_gnu17vx, PpgMon64.xls, etc.). For more information, refer to sheet “Note” in the PpgMon64.xls Excel file.

3.4 FIR Filter

This is a Finite Impulse Response filter, a kind of digital filter. Various literature has been provided for presenting the FIR filter design techniques. Refer to them for detailed information.

Figure 3-8 shows an example of FIR filter configuration. The letters in the diagram denote the wiring nodes in the circuit. [5)]

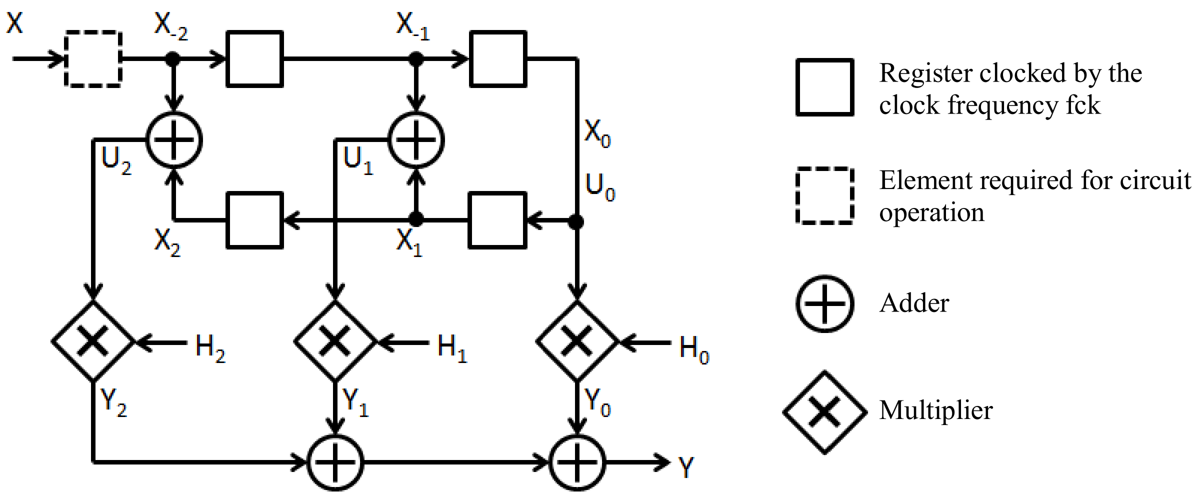


Figure 3-8 FIR Filter Configuration

This application note uses an FIR filter configured as above. The following outlines the design information:

The human pulse rate is a relatively low frequency; in most cases, it is within 40 to 250 beats per minute. Therefore, the pulse wave measurement needs an LPF (Low Pass Filter) that eliminates noise components with a higher frequency than that. The FIR filter used in this time was designed with a 20 ms of fixed sampling cycle and a 4 Hz cutoff frequency.

For the concrete calculation examples, refer to the Excel file (FIRdesign.xls) included in this package that was used to design the FIR window function implemented in the sample program. [6) 7)]

3.5 Square Wave Correlation Filter

The square wave correlation filter is an older technology, but it is an algorithm that can be executed in real time even in embedded MCUs. [8) 9)] Heart rate can be determined by measuring the interval between the bottom positions of the measured pulse wave, as the detected light intensity decreases and the A/D converted value is also reduced in the systole of the heart. This filter is easy to use to suppress noise in the pulse wave data, so it was implemented in the sample program of this application note.

Specifically, the cross-correlation value is obtained using a square pulse window as Figure 3-9 shows for the measured pulse data. The following shows the equation to calculate cross-correlation values from pulse wave data: [10)]

$$
\begin{aligned}
y(t) = (-1) * \{ x(t) + x(t-1) + \ldots + x(t-n+1) \} \\
+ (+1) * \{ x(t-n) + x(t-n-1) + \cdots + x(t-3n+1) \} \\
+ (-1) * \{ x(t-3n) + x(t-3n-1) + \cdots + x(t-4n+1) \}
\end{aligned}
$$

where, y(t) is the cross-correlation value, x(t) is a pulse wave data, and n is the number of frames for 1/4 of the window width.

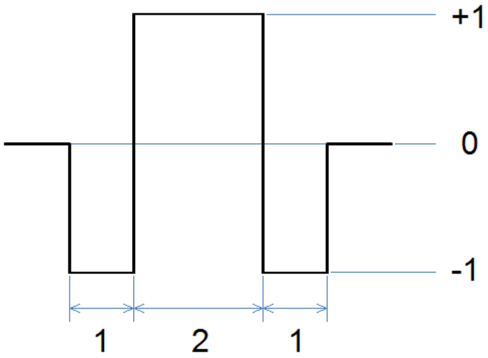


Figure 3-9 Square Pulse Window

4. Software Description

4.1 s1c17w23_ppg_mon_gnu17vx

This s1c17w23_ppg_mon_gnu17vx software, which runs in conjunction with Excel including a VBA macro and a VB script (VBS), samples the PPG sensor output values in certain cycles, and sends them to the PC via the UART to write them to an Excel sheet. The following describes this software.

4.1.1 File Configuration (within src folder)

File name	Description
ad12.c	ADC12A driver source file
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
t16_ch1.c	T16 Ch.1 driver source file
t16_ch3.c	T16 Ch.3 driver source file
uart.c	UART driver source file

4.1.2 File Configuration (within inc folder)

File name	Description
reg	S1C17W23 peripheral circuit register definition file folder
c17w23_reg.h	S1C17W23 peripheral circuit header definition file
init.h	Initialization function header definition file
osc.h	OSC driver header definition file
ppg_mon.h	Pulse wave measurement value visualization header definition file
t16_ch0.h	T16 Ch.0 driver header definition file
t16_ch1.h	T16 Ch.1 driver header definition file
t16_ch3.h	T16 Ch.3 driver header definition file
uart.h	UART driver header definition file

4.1.3 Module Description

This section describes the more uncommon functions and variables that need explanation.

File name: main.c

Function name	Description
execFir	If FIR filter processing is selected in the Excel file PpgMon64.xlsm, this function processes the FIR filtering calculation according to the condition written in the FIR sheet of PpgMon64.xlsm and stores the result to pm.firRes.
calcCc	If CC filter processing is selected in PpgMon64.xlsm, this function processes the square wave correlation filtering calculation according to the Num. Frame setting in PpgMon64.xlsm and stores the calculated cross-correlation value to pm.ccRes.
intUartCh0	UART Ch.0 interrupt handler function. This function activates T16 Ch.0 to start pulse wave measurement when the control information is received from the VBA or VBS in PpgMon64.xlsm.
intT16Ch0	T16 Ch.0 interrupt handler function. This function is called repeatedly at the time interval specified with the Scan Interval setting in PpgMon64.xlsm. It obtains the pulse wave A/D converted values according to the control information already received and sends them to the VBS on the PC. When data of the number set in Data Set# of PpgMon64.xls has been sent, this function stops T16 Ch.0 operations.
intAdc12Ch0	ADC12A interrupt handler function. This function stores the measurement result to pm.measuredVal when an A/D conversion has completed and sets pm.stage to 1 (measurement completed).

Structure name: ppg_mon (defined in the ppg_mon.h file)

Variable name	Description
rawData[ ]	Raw data of the pulse wave measurement results. The data range is 0 to 4,095 because it was converted by ADC12A.
firRes[ ]	The value after being processed through the FIR filter. The filtering process makes the shaped width smaller than the value before being converted.
ccRes[ ]	The cross-correlation value calculated using the square pulse window.
firCoef[ ]	The FIR filter coefficient sent from the PC after the value has been multiplied by 65,536.
rcvData[ ]	The control information sent from the PC.
sndData[ ]	The measured values to be sent to the PC. Three 2-byte data are sent in the order of raw data, data after being processed through the FIR filter, and cross-correlation value. Since the cross-correlation value may be negative, it is sent after adding 32,768. If no data is available to be sent, 29,999 is sent.
interm	The control information sent from the PC that specifies either intermittent drive or continuous drive. 0: Continuous drive, 1: Intermittent drive
highGain	The gain control information sent from the PC. 0: Low amplifier gain, 1: High amplifier gain
lpf	The LPF control information sent from the PC. 0: LPF off, 1: LPF on
filter1, filter2	Two different filter control information sent from the PC. 0: None, 1: FIR filter, 2: Square wave correlation filter
skipNum	The number of data information sent from the PC for skipping from the initiation of measurement until the actual start of data measurement.
dataSetNum	The number of measurement data information sent from the PC.
scanInterval	The sampling cycle information sent from the PC.

4. Software Description

Variable name	Description
samplingDelay	The additional stabilization waiting time information sent from the PC for inserting before starting sampling during intermittent drive mode.
firNum	The number of FIR filter coefficients sent from the PC. (N-1)/2 when the tap value = N.
numFrame	The number of frames for 1/4 of the square pulse window width that is sent from the PC.
dataPtr	Pointer to measurement data.
measuredVal	The measured value.
measCount	Counter to count the number of measurements.
skipCount	Counter to count the number of skips at the start of measurement.
availDataNum	Counter to check if enough data are available for calculating the square wave correlation filter.
setNum	4 * numFrame.
step	Status flag. 0: Standby for control data from the PC, 1: Data is being skipped, 2: During measurement, 3: Measurement has completed → The status changes to 0.
stage	ADC measurement status flag. 0: During measurement, 1: Measurement has completed.
p1xBase	Source data for the data to be output from the GPIO P10 to P17 ports. The source data is manipulated with the necessary bits set on or off before being output.

4.1.4 Operation Procedures

The sample software includes two projects for GNU17 Version 2 (hereinafter referred to as GNU17v2) and GNU17 Version 3 (hereinafter referred to as GNU17v3).

Before the sample software for GNU17v2 or GNU17v3 can be used, configure the target model by following the procedure shown below.

(1) Copy the c17w23_reg.h file and the reg folder to the inc subfolder in the s1c17w23_ppg_mon_gnu17vx folder of the sample software.

(2) After launching the GNU17 IDE, select [Import...] from the [File] menu to start the Import Wizard, then select

[General] > [Existing Project to Workspace] (GNU17v2) or

[General] > [Existing Projects into Workspace] (GNU17v3).

(3) Select the project folder that contains the sample program:

s1c17w23_ppg_mon_gnu17v2 folder (GNU17v2) or

s1c17w23_ppg_mon_gnu17v3 folder (GNU17v3).

(4) Select [Copy projects into workspace] and then click the [Finish] button to exit the Import Wizard.

(5) Change the target CPU.

(GNU17v2)

1. Select the imported project in the [C/C++ project] view and select [Properties] from the [Project] menu.
2. Select [GNU17 General] from the property list in the [Properties] dialog box that appears.
3. Select the target CPU from the [Target CPU Device] drop-down list.
4. Click the [Apply] button.

(GNU17v3)

1. Select the imported project in the [Project Explorer] view and select [Properties] from the [Project] menu.
2. Select [GNU17 Setting] from the property list in the [Properties] dialog box that appears.
3. Select the target CPU from the [Target CPU] drop-down list.
4. Click the [OK] button to close the dialog box. Then, go to Step (7).

(6) Set the debugger's startup options. (GNU17v2 only)

1. Select [GNU17 GDB Commands] from the property list.
2. Click the [Create commands from template] button to display the [Create a simple startup command] dialog box.
3. Select "ICD Mini" from the [Debugger:] drop-down list and select [Execute flash ROM writing].
4. Click the [Overwrite] button to close the dialog box. Close the [Properties] dialog box as well.

(7) Build the project.

Build the s1c17w23_ppg_mon_gnu17vx project using IDE.

(8) Perform the preparations shown below for using Excel VBA.

1. Copy the PpgMon64 folder to the desktop. This folder contains the Excel file PpgMon64.xlsm, VB script file DoPpg64.vbs, and 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 PpgMon64 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:\Users*user*\Desktop\PpgMon64”, “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”.

(9) Connect the equipment and turn on the power.

1. Connect an optical sensor and the AFE to the SVT17W23. 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.

(10) Execute Excel VBA.

Double-click the Excel file PpgMon64.xlsm to execute the VBA.

(11) Execute the sample software.

1. Execute the s1c17w23_ppg_mon_gnu17vx project using IDE.
2. Operate Excel VBA to start obtaining the PPG sensor output values.

4.1.5 How to Use PpgMon64.xlsm

Figure 4-1 shows the appearance of PpgMon64.xlsm.

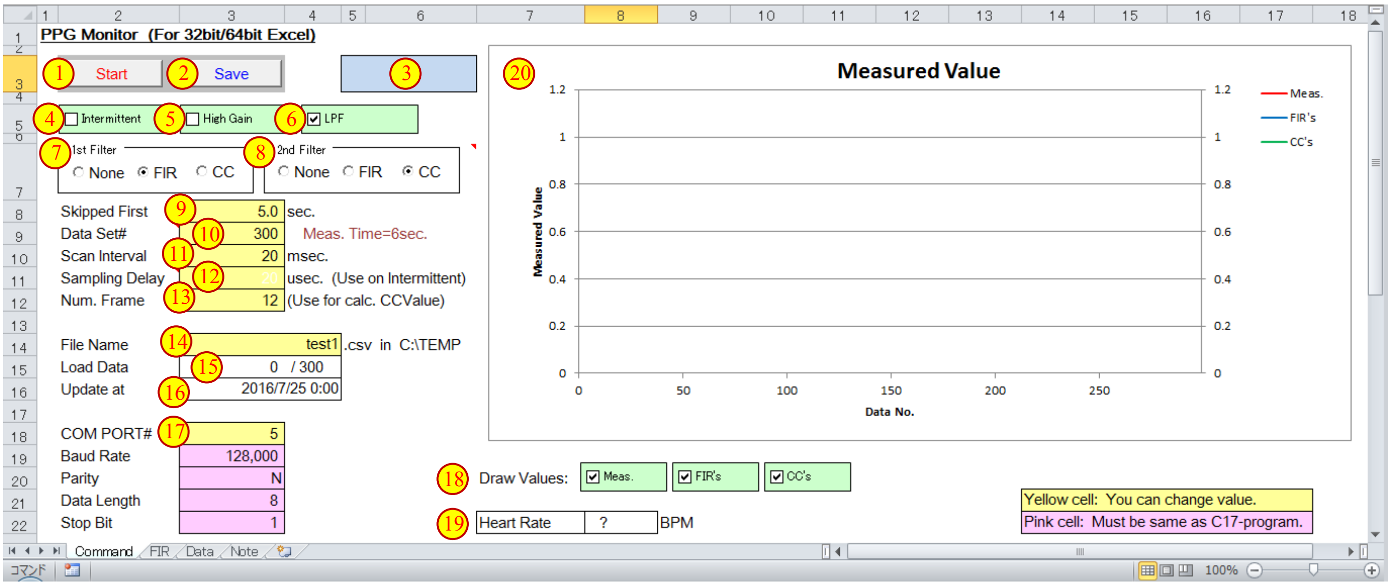


Figure 4-1 Appearance of PpgMon64.xlsm

① Measurement start button	Starts obtaining pulse wave measurement values.
② Data save button	Saves the measured data in CSV format.
③ Status indicator	Indicates the measurement or data saving status.
④ Drive mode select button	☐: Continuous drive ☑: Intermittent drive
⑤ Amplifier gain select button	☐: Low amplifier gain ☑: High amplifier gain
⑥ LPF select button	☐: LPF off ☑: LPF on
⑦ First filter	None: Not used, FIR: FIR filter, CC: Square wave correlation filter Note 1
⑧ Second filter	None: Not used, FIR: FIR filter, CC: Square wave correlation filter
⑨ Skip time	Time for skipping measurement values at the start of measurement.
⑩ Number of data sets	Specify the number of data sets to be loaded.
⑪ Scan interval	Specify the interval to load data.
⑫ Sampling delay time	Additional delay time from turning on the LED to the first sampling in intermittent drive mode.
⑬ Number of frames	The number of frames for 1/4 of the square pulse window width.
⑭ Save file name	Specify the file name to store measurement data.
⑮ Number of loaded data	Displays the number of measured data sets by updating sequentially.
⑯ Data sampling date and time	Displays the last date and time at which data has been sampled.
⑰ Serial communication conditions	COM port number, baud rate, parity, data length, and stop bit length, from the top.
⑱ Display graph selector	Measured raw data, data after FIR filtering, and cross-correlation values after square wave correlation filtering, from the left.

⑲ Pulse rate	The pulse rate that is calculated based on the pulse wave measured.
⑳ Measured data graph	This area plots the specified graphs. The right ordinate indicates cross-correlation value.

Note 1) The filter actually used depends on the combination of first and second filter selections. For detailed information, refer to the comment at the 7th row and 6th column in sheet “Command” of PpgMon64.xlsm.

Note 2) 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. If a such a communication error occurs, try to avoid it by changing the ⑩ and ⑪ setting values.

Note 3) Normally, it is not necessary to change the serial communication parameters at ⑰ except for the COM port number. Note that the related part of s1c17w23_ppg_mon_gnu17vx must be modified if these parameters are changed.

Note 4) The graph ⑳ is drawn after all data has been loaded. While data are being sampled, they are not updated successively and the graph retains the status with data cleared.

4.1.6 Outline of Sample Program Operations

The sample program performs the following processing:

(1) Initializes the following peripherals/functions to be used:
- sets the interrupt levels,
- switches the system clock from IOSC to OSC3 (4 MHz internal oscillator),
- initializes T16 and UART,
- assigns the ADC12A ports to the pins and initializes ADC12A, and
- configures PPORT for output.

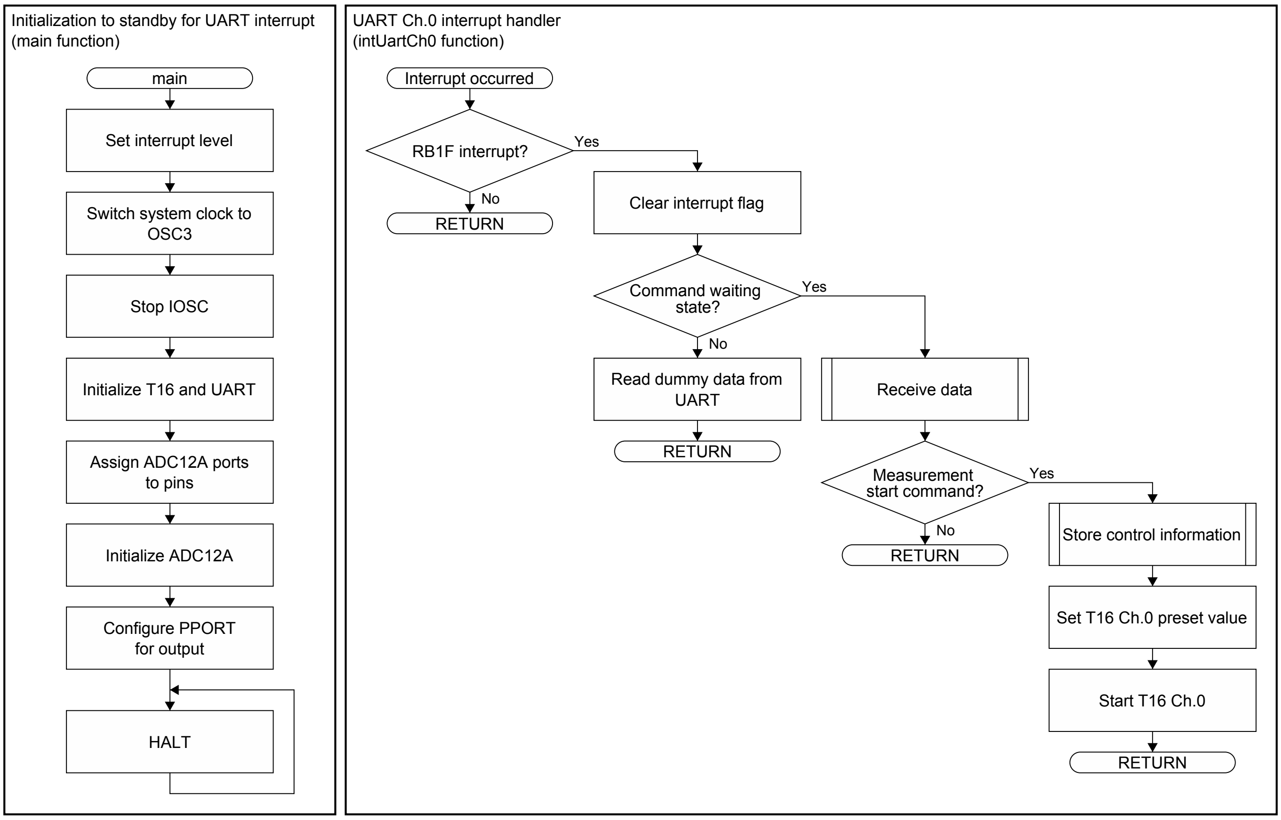
(2) Puts the CPU into HALT mode to wait for a UART interrupt.

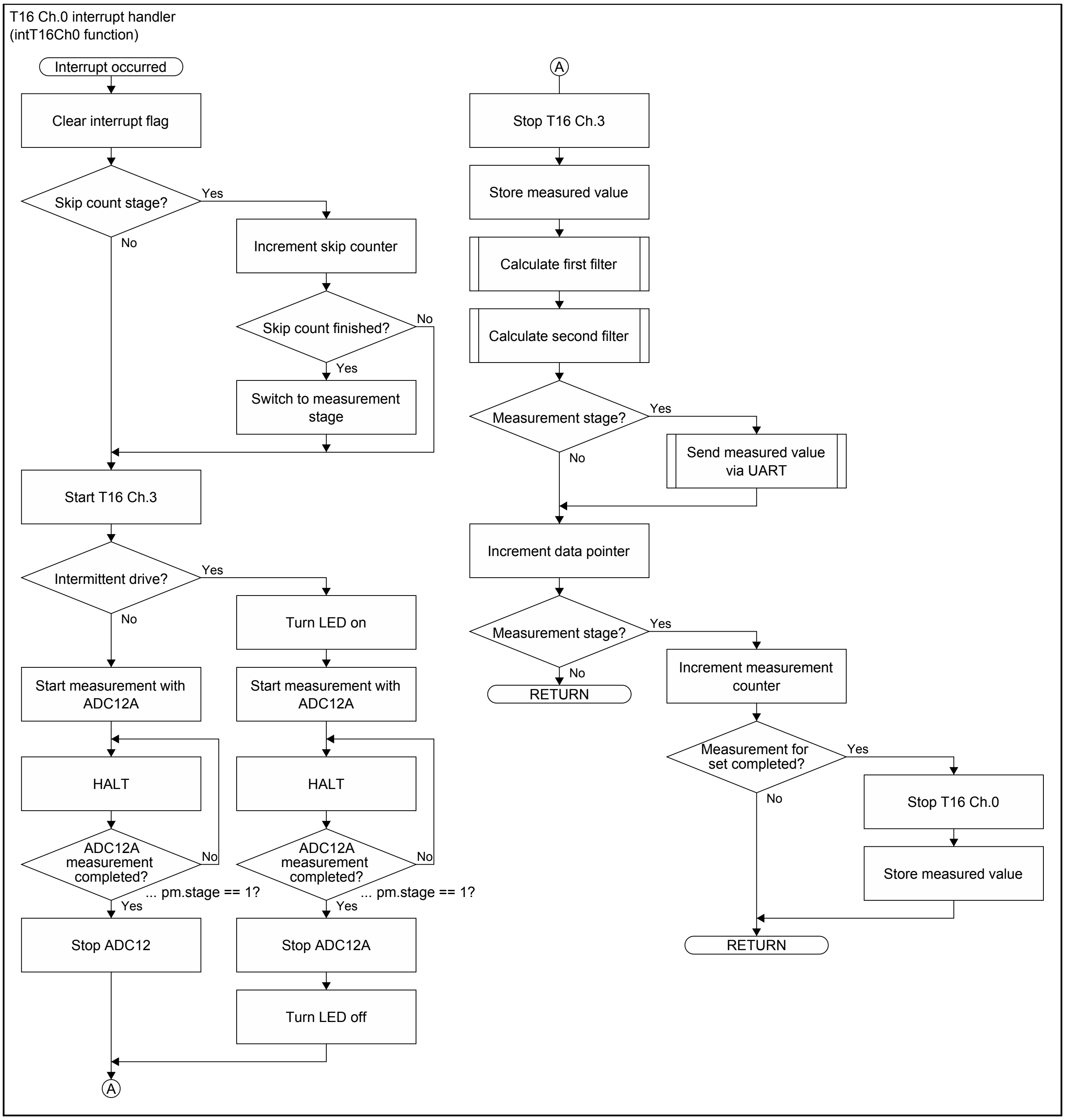
(3) When a UART interrupt occurs, the interrupt handler intUartCh0 performs the following processing:
- receives the control information sent from the PC, and
- starts T16 Ch.0 to initiate the measurement of pulse waves.

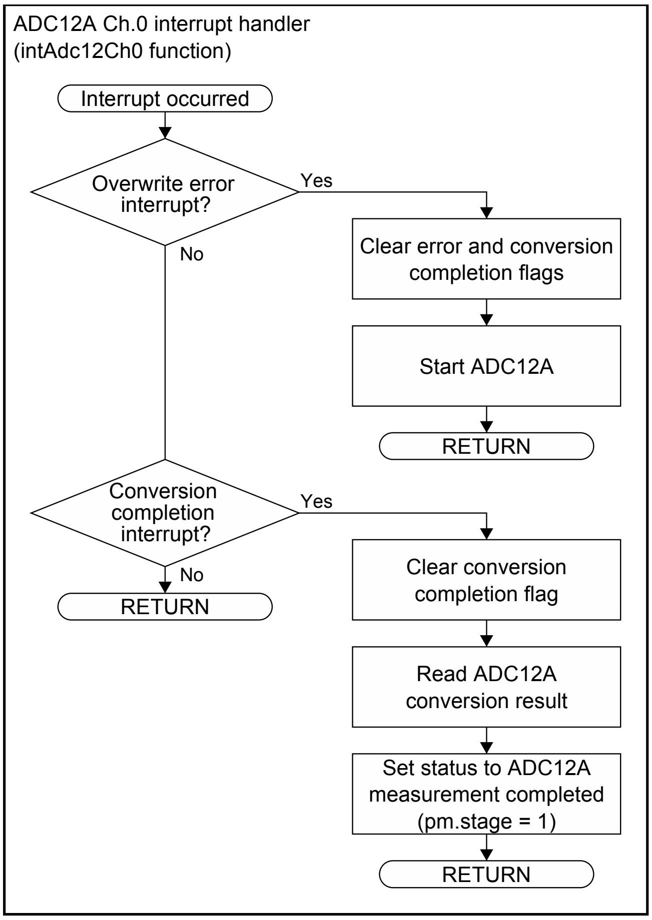
(4) Every time a T16 Ch.0 interrupt occurs, the interrupt handler intT16Ch0 performs the following processing according to the control information that has already been received:
- reads the A/D converted value of the pulse wave,
- performs the filtering calculation, and
- sends the measured data to the PC.

When data of the specified number have been sent, intT16Ch0 stops T16 Ch.0.

The following shows the flowcharts of the above processing:




T16 Ch.0 interrupt handler
(intT16Ch0 function)
Interrupt occurred
Clear interrupt flag
Skip count stage?
Yes
No
Increment skip counter
Skip count finished?
No
Yes
Switch to measurement stage
Start T16 Ch.3
Intermittent drive?
Yes
No
Turn LED on
Start measurement with ADC12A
Start measurement with ADC12A
HALT
HALT
ADC12A measurement completed?
No
... pm.stage == 1?
Yes
ADC12A measurement completed?
No
... pm.stage == 1?
Yes
Stop ADC12
Stop ADC12A
Turn LED off
A
A
Stop T16 Ch.3
Store measured value
Calculate first filter
Calculate second filter
Measurement stage?
Yes
No
Send measured value via UART
Increment data pointer
Measurement stage?
Yes
No
RETURN
Increment measurement counter
Measurement for set completed?
Yes
No
Stop T16 Ch.0
Store measured value
RETURN


ADC12A Ch.0 interrupt handler
(intAdc12Ch0 function)
Interrupt occurred
Overwrite error interrupt?
Yes
No
Clear error and conversion completion flags
Start ADC12A
RETURN
Conversion completion interrupt?
Yes
No
RETURN
Clear conversion completion flag
Read ADC12A conversion result
Set status to ADC12A measurement completed (pm.stage = 1)
RETURN

4.2 s1c17w23_ppg_demo_gnu17vx

This section describes the PPG demonstration software s1c17w23_ppg_demo_gnu17vx that displays pulsation, pulse rate, and pulse wave on the LCD.

4.2.1 File Configuration (within src folder)

File name	Description
ad12.c	ADC12A driver source file
boot.c	Startup module source file
display.c	LCD display function 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
t16_ch1.c	T16 Ch.1 driver source file
t16_ch3.c	T16 Ch.3 driver source file

4.2.2 File Configuration (within inc folder)

File name	Description
reg	S1C17W23 peripheral circuit register definition file folder
ad12.h	ADC12A driver header definition file
c17w23_reg.h	S1C17W23 peripheral circuit header definition file
display.h	LCD display function header definition file
init.h	Initialization function header definition file
lcd_font.h	LCD display font header definition file
osc.h	OSC driver header definition file
ppg_data.h	Pulse wave measurement data header definition file
t16_ch0.h	T16 Ch.0 driver header definition file
t16_ch1.h	T16 Ch.1 driver header definition file
t16_ch3.h	T16 Ch.3 driver header definition file

4.2.3 Module Description

This section describes the more uncommon functions and variables that need explanation.

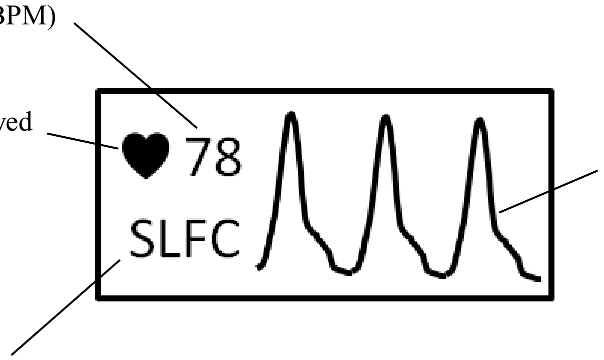
File name: main.c

Function name	Description
calcHeartRate	Calculates the pulse rate from the interval of the minimum peaks (points becoming dark by the pulsation) of cross-correlation values. The number of frames for 1/4 of the pulse window width varies within a certain range by following the pulse rate.
getMinMax	Obtains the maximum and minimum values of the raw data that are used to detect pulsation.
checkHeartBeatTiming	Returns 1 if the latest measured raw data is smaller than the threshold value to determine that it denotes systole, otherwise this function returns 0.
execFir	Processes the raw data through the FIR filter and stores the result to pd.firRes.
calcCc	Process pd.firRes through the square wave correlation filter and stores the obtained cross-correlation value to pd.ccRes.
val2Str4	Converts a positive integer val into a numeric character string of “digit” digits (up to four digits) and sets it to the return value dst. If zeroSup = 1, this function suppresses zeros in the number, otherwise it does not zero-suppress.
dispHeartMark	If mark = 0, this function displays a small heart mark, which represents systole, at the predetermined position on the LCD, otherwise it displays a large hart mark.
dispHeartRate	If the heart rate has been calculated, this function displays the value at the predetermined position on the LCD, otherwise it displays a ‘?’ instead of a value.
dispStat	Displays the currently selected measurement conditions on the LCD. For more information, see Figure 4-2.
setP1Base	Sets the value to be the base of the data that is output to the P1 ports to pd.p1xBase according to the selected conditions.
clearGraph	Clears the graph display area on the LCD.
intPport	This is an interrupt handler called when a switch among SW1 to SW4 is pressed.
intT16Ch0	This function is called every 20 ms to obtain the A/D conversion value of the pulse wave according to the control information and store it to pd.rawData. After that, it performs the FIR filtering and square wave correlation filtering calculations. This function updates the graph display. Also it updates the pulse rate and heart mark displays repeatedly at fixed intervals.
intAdc12Ch0	This is the ADC12A Ch.0 interrupt handler. When an A/D conversion has been completed, this function stores the measurement result to pm.measuredVal and sets pm.stage to 1 (measurement completed).

Structure name: ppg_data (defined in the ppg_data.h file)

Variable name	Description
rawData[ ]	Raw data of the pulse wave measurement results. The data range is 0 to 4,095 because it was converted by ADC12A.
firRes[ ]	The value after being processed through the FIR filter. The filtering process makes the shaped width smaller than the value before being converted.
ccRes[ ]	The cross-correlation value calculated using the square pulse window.
dataPtr	Pointer to measurement data.
availDataNum	Counter to check if enough data is available for calculating the square wave correlation filter.
measuredVal	The measured value.
calcHRIter	Iteration counter to generate timings for heart rate calculation.
numFrame	The number of frames for 1/4 of the square pulse window width.

Variable name	Description
heartRate	Heart rate in BPM (Beats Per Minute).
rawMinVal	The minimum value of the raw data.
rawMaxVal	The maximum value of the raw data.
rawRange	Raw data range (= maximum value - minimum value).
rawSystoleLevel	Threshold value to determine systole. If raw data is smaller than this value, it is determined as systole.
currHeartMark	The currently displayed heart mark used to minimize the LCD update frequency.
systoleCycleCounter	Systole cycle counter.
hRFromSystole	The heart rate calculated from the systole cycle. This is used to automatically adjust numFrame.
running	Measurement status: 0 = idle, 1 = during measurement
interm	Drive mode control information: 0 = continuous drive, 1 = intermittent drive
highGain	Gain control information: 0 = low amplifier gain, 1 = high amplifier gain
lpf	LPF control information: 0 = LPF off, 1 = LPF on
samplingDelay	The additional stabilization waiting time for inserting before starting sampling during intermittent drive mode.
stage	ADC12A measurement status flag: 0 = during measurement, 1 = measurement completed
p1xBase	Source data for the data to be output from the GPIO P10 to P17 ports. The source data are manipulated with the necessary bits set on or off before being output.
overRunTimes	T16 Ch.0 overrun counter. Up to the overrun count value until the measurement is started by pressing SW4 is allowed. If the count exceeds this allowed value, the intT16Ch0 function cannot be processed in time. In this case, the process in the intT16Ch0 function must be moved to the main function to do event processing.



SW	Symbol and meaning			
SW4	S	Measurement halted	R	During measurement
SW3	L	Low amplifier gain	H	High amplifier gain
SW2	F	LPF on	–	LPF off
SW1	C	Continuous drive	I	Intermittent drive

Figure 4-2 Information Displayed on LCD

4.2.4 Operation Procedures

The sample software includes two projects for GNU17 Version 2 (hereinafter referred to as GNU17v2) and GNU17 Version 3 (hereinafter referred to as GNU17v3).

Before the sample software for GNU17v2 or GNU17v3 can be used, configure the target model by following the procedure shown below.

(1) Copy the c17w23_reg.h file and the reg folder to the inc subfolder in the s1c17w23_ppg_demo_gnu17vx folder of the sample software.

(2) After launching the GNU17 IDE, select [Import...] from the [File] menu to start the Import Wizard, then select

[General] > [Existing Project to Workspace] (GNU17v2) or

[General] > [Existing Projects into Workspace] (GNU17v3).

(3) Select the project folder that contains the sample program:

s1c17w23_ppg_demo_gnu17v2 folder (GNU17v2) or

s1c17w23_ppg_demo_gnu17v3 folder (GNU17v3).

(4) Select [Copy projects into workspace] and then click the [Finish] button to exit the Import Wizard.

(5) Change the target CPU.

(GNU17v2)

1. Select the imported project in the [C/C++ project] view and select [Properties] from the [Project] menu.
2. Select [GNU17 General] from the property list in the [Properties] dialog box that appears.
3. Select the target CPU from the [Target CPU Device] drop-down list.
4. Click the [Apply] button.

(GNU17v3)

1. Select the imported project in the [Project Explorer] view and select [Properties] from the [Project] menu.
2. Select [GNU17 Setting] from the property list in the [Properties] dialog box that appears.
3. Select the target CPU from the [Target CPU] drop-down list.
4. Click the [OK] button to close the dialog box. Then, go to Step (7).

(6) Set the debugger's startup options. (GNU17v2 only)

1. Select [GNU17 GDB Commands] from the property list.
2. Click the [Create commands from template] button to display the [Create a simple startup command] dialog box.
3. Select "ICD Mini" from the [Debugger:] drop-down list and select [Execute flash ROM writing].
4. Click the [Overwrite] button to close the dialog box. Close the [Properties] dialog box as well.

(7) Build the project.

Build the s1c17w23_ppg_demo_gnu17vx project using IDE.

(8) Connect the equipment and turn on the power.

1. Connect an optical sensor and the AFE to the SVT17W23. Connect the SVT17W23 to the ICDmini and then it to the PC with a USB cable. If the same PC environment as s1c17w23_ppg_mon_gnu17vx previously described is used, disconnect the cable for serial communication via UART, as it may cause interference in the LCD drive pins.
2. Reset the SVT17W23 and ICDmini.

(9) Execute the sample software.

1. Execute the s1c17w23_ppg_demo_gnu17vx project using IDE.
2. Pressing a switch among SW1 to SW4 starts a demonstration of PPG operations.

4.2.5 Outline of Sample Program Operations

The sample program performs the following processing:

(1) Initializes the following peripherals/functions to be used:
- sets the interrupt levels,
- initializes the variables,
- enables for OSC1 (32.768 kHz) to start oscillating,
- initializes the LCD registers to clear the display.
- initializes PPORT, T16 Ch.0, T16 Ch.1, and T16 Ch.3,
- assigns the ADC12 ports to the pins and initializes ADC12, and
- initializes PPORT used for changing the AFE circuit configuration.

(2) Updates the LCD display to the initial screen

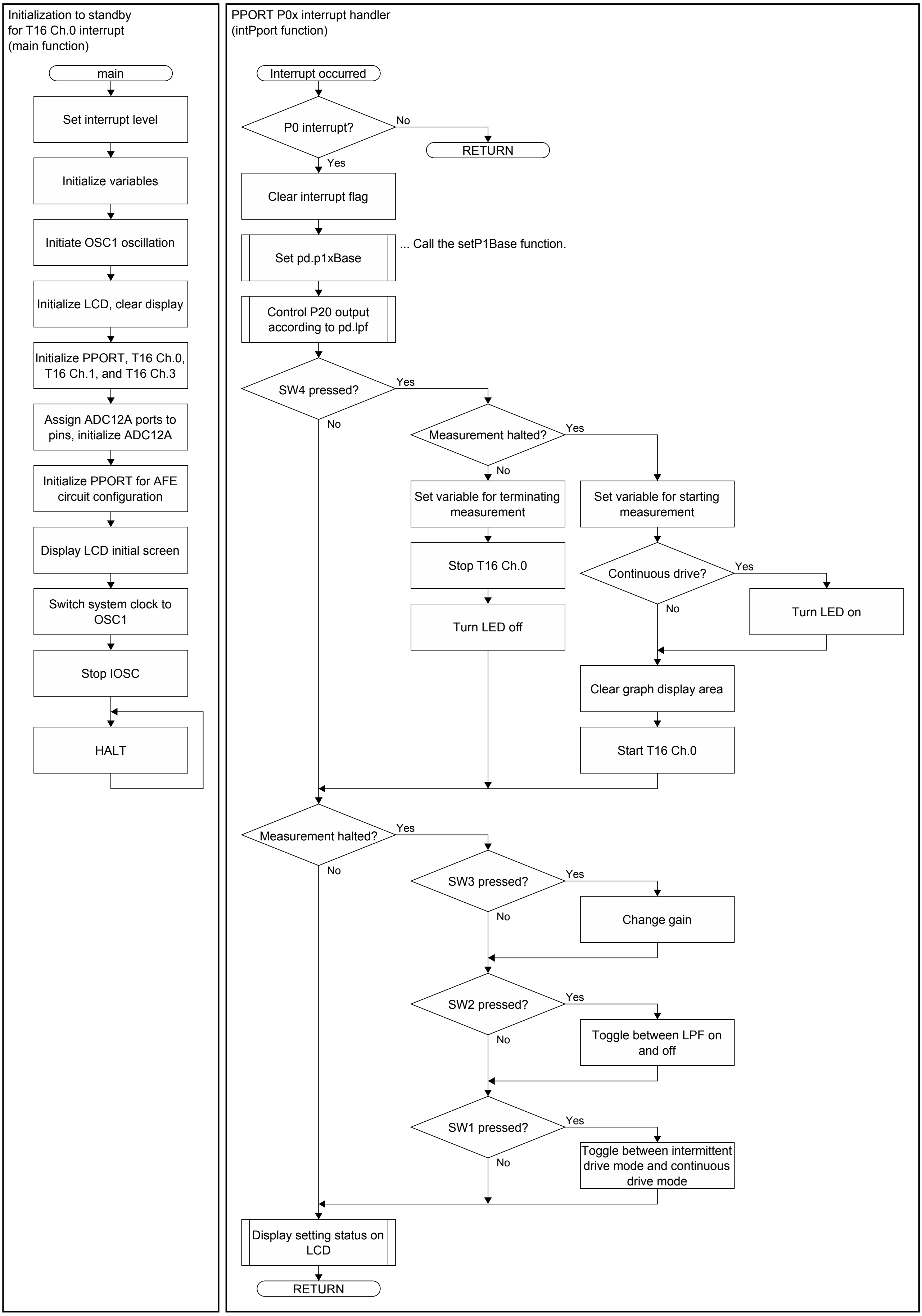
(3) Switches the system clock from IOSC to OSC1 (32.768 kHz).

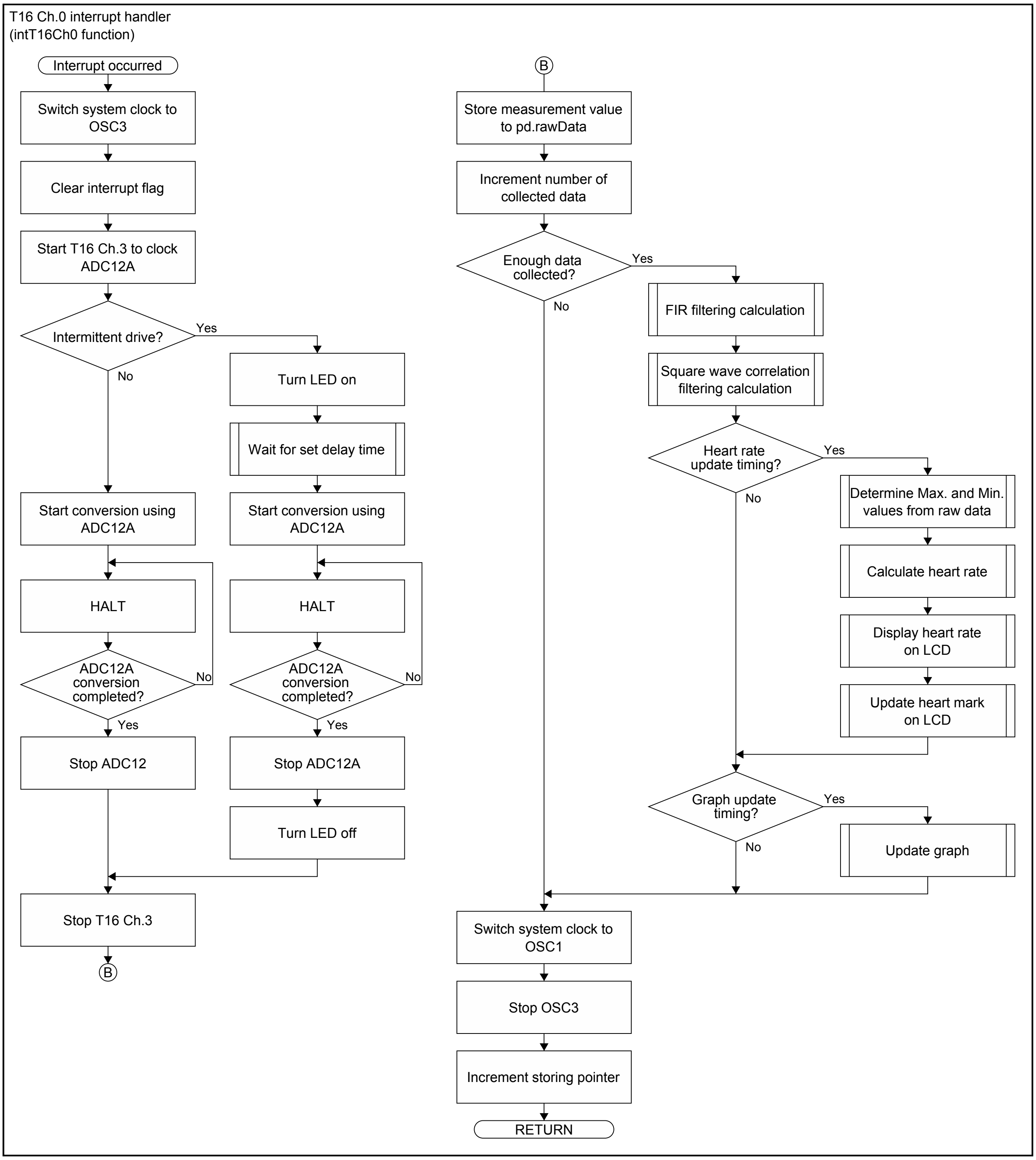
(4) Puts the CPU into HALT mode to wait for a PPORT interrupt.

(5) When a PPORT interrupt has occurred by pressing a switch among SW1 to SW4, the sample program performs the following processing:
- if measurement is in halt state, pressing SW4 starts T16 Ch.0 to initiate measurement. Pressing a switch among SW1 to SW3 changes the AFE circuit configuration and updates the LCD display according to the change.
- if the measurement is underway, pressing SW4 stops T16 Ch.0 to terminate the measurement. SW1 to SW3 are ineffective in this case and the intT16Ch0 function will be called every 20 ms by a T16 Ch.0 interrupt.

(6) In intermittent drive mode, the intT16Ch0 function turns the LED on and triggers ADC12A to initiate an A/D conversion after the predefined delay time has elapsed. It obtains the A/D conversion result and then turns the LED off.
In continuous drive mode, the intT16Ch0 function triggers ADC12 and obtains the A/D conversion result without a delay time inserted.
After that, the intT16Ch0 function performs the FIR filtering and square wave correlation filtering calculations, and the regular pulse rate calculation. If systole is determined, it updates the heart mark and graph on the LCD.

The figures below shows the flowcharts of the above processing, except for the ADC12A Ch.0 interrupt handler (intAdc12Ch0 function).
The intAdc12Ch0 function has exactly the same processing flow as s1c17w23_ppg_mon_gnu17vx\src\main.c previously described, except that the structure variable name is changed from “pm” to “pd”.


Initialization to standby for T16 Ch.0 interrupt (main function)
main
Set interrupt level
Initialize variables
Initiate OSC1 oscillation
Initialize LCD, clear display
Initialize PPORT, T16 Ch.0, T16 Ch.1, and T16 Ch.3
Assign ADC12A ports to pins, initialize ADC12A
Initialize PPORT for AFE circuit configuration
Display LCD initial screen
Switch system clock to OSC1
Stop IOSC
HALT
PPORT P0x interrupt handler (intPport function)
Interrupt occurred
P0 interrupt?
No
RETURN
Yes
Clear interrupt flag
Set pd.p1xBase
... Call the setP1Base function.
Control P20 output according to pd.lpf
SW4 pressed?
Yes
No
Measurement halted?
Yes
No
Set variable for terminating measurement
Set variable for starting measurement
Stop T16 Ch.0
Continuous drive?
Yes
No
Turn LED on
Turn LED off
Clear graph display area
Start T16 Ch.0
Measurement halted?
Yes
No
SW3 pressed?
Yes
No
Change gain
SW2 pressed?
Yes
No
Toggle between LPF on and off
SW1 pressed?
Yes
No
Toggle between intermittent drive mode and continuous drive mode
Display setting status on LCD
RETURN


T16 Ch.0 interrupt handler
(intT16Ch0 function)
Interrupt occurred
Switch system clock to OSC3
Clear interrupt flag
Start T16 Ch.3 to clock ADC12A
Intermittent drive?
Yes
No
Turn LED on
Wait for set delay time
Start conversion using ADC12A
Start conversion using ADC12A
HALT
HALT
ADC12A conversion completed?
No
Yes
ADC12A conversion completed?
No
Yes
Stop ADC12
Stop ADC12A
Turn LED off
Stop T16 Ch.3
B
B
Store measurement value to pd.rawData
Increment number of collected data
Enough data collected?
Yes
No
FIR filtering calculation
Square wave correlation filtering calculation
Heart rate update timing?
Yes
No
Determine Max. and Min. values from raw data
Calculate heart rate
Display heart rate on LCD
Update heart mark on LCD
Graph update timing?
Yes
No
Update graph
Switch system clock to OSC1
Stop OSC3
Increment storing pointer
RETURN

Appendix A. Example of Pulse Wave Detection Experiment

This chapter shows the results obtained from the experiment that actually measured pulse waves with different conditions as an example.

In this experiment, NJL5303R-TE1, which combines a green LED and a phototransistor, listed at the top of Table 3-3 was used as the PPG sensor. Nearly the same results were obtained using some other PPG sensors. The component values for the PPG sensor circuit were selected as follows: R31@5 mA = 270 Ω, R41 = 82 kΩ, and C41 = 4.7 µF, as listed in Table 3-4.

(1) Time required for stability of measurement waveform

With an optical sensor (phototransistor or photodiode), it is difficult to observe pulse waves, as the variations of its output absolute values are very small. In order to solve this problem, the circuit was designed so that it amplifies only an AC component of the output to detect the practical level variation. However, it causes the side effect of taking a time until the waveform stabilizes. So we examined how much time is required for the waveforms to stabilize in different conditions. Table A-1 lists the conditions used for the examination and the results. (The list does not include all combinations of the conditions, as some of them cause a problem. For example, the measurement at a fingertip in continuous drive mode with high amplifier gain causes the waveform to be saturated.) Figure A-1 shows the actually measured waveforms.

In the measurement part, fingertip points to the pad of a forefinger and wrist points to the position above the radial artery. “Scan Interval” was fixed at 20 ms.

Table A-1 Conditions Used for Examination and Results

No.	Measurement part	Drive mode	Amplifier gain	LPF	Waveform stabilization time
1	Fingertip	Continuous drive	Low	Not used	3 seconds
2	Fingertip	Continuous drive	Low	Used	5 seconds
3	Fingertip	Intermittent drive	Low	Not used	20 seconds
4	Fingertip	Intermittent drive	Low	Used	20 seconds
5	Fingertip	Intermittent drive	High	Not used	36 seconds
6	Fingertip	Intermittent drive	High	Used	36 seconds
7	Wrist	Continuous drive	Low	Not used	6 seconds
8	Wrist	Continuous drive	Low	Used	8 seconds
9	Wrist	Continuous drive	High	Not used	11 seconds
10	Wrist	Continuous drive	High	Used	8 seconds

The results above show that intermittent drive mode requires a long waveform stabilization time, so it is thought that intermittent drive mode was not compatible with the circuit. Using LPF did not improve the waveform stabilization time that much. Hence in the next experiment to examine the effects of the FIR filter and square wave correlation filter, intermittent mode and LPF used conditions were excluded from the evaluation. (Only condition No.2, 8, and 10 in Table A-1, which are highlighted in a light yellow, were evaluated.)

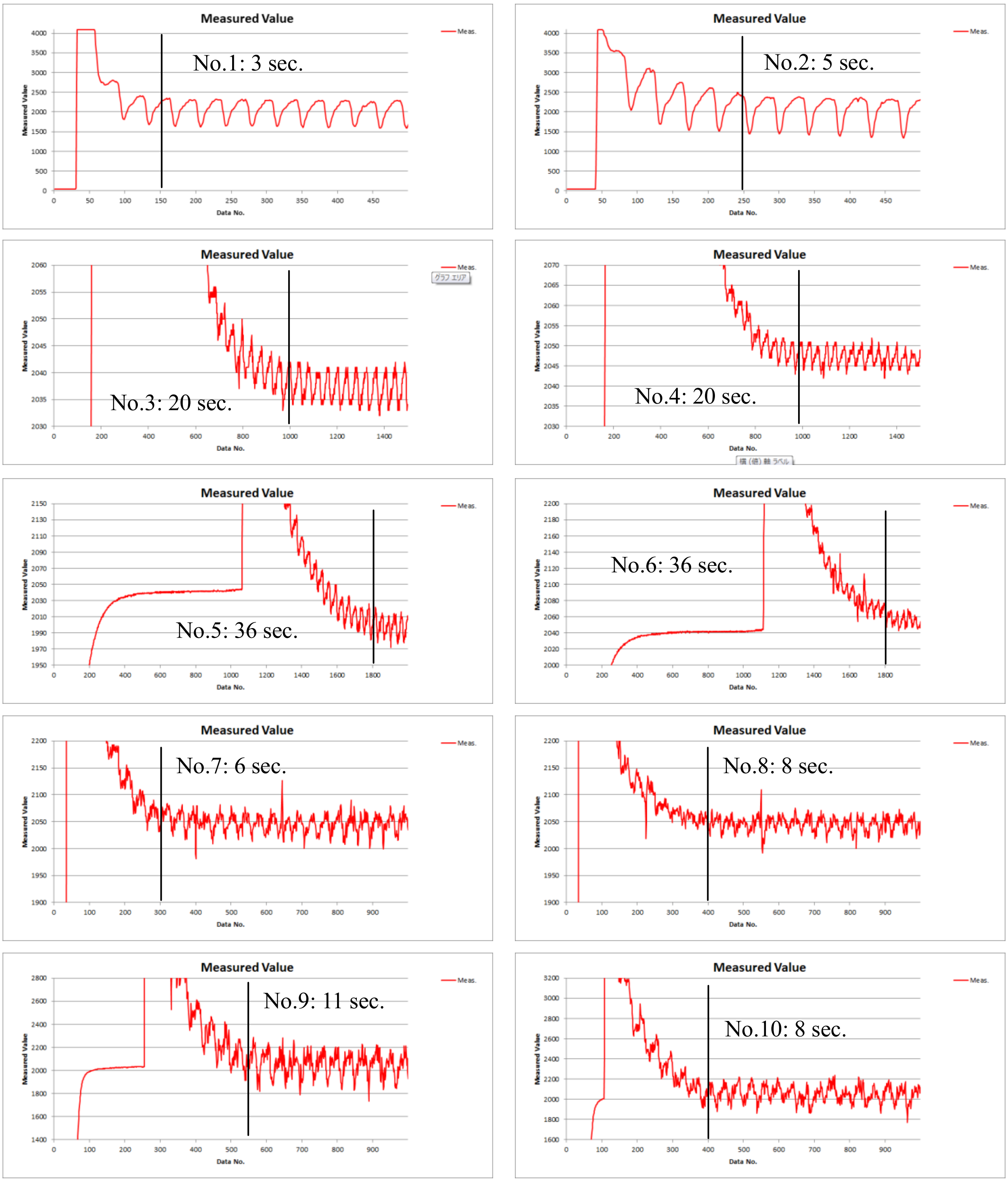


Figure A-1 Actually Measured Waveforms

(2) Confirming effects of FIR filter and square wave correlation filter

The effects of the FIR filter and square wave correlation filter were confirmed under the three conditions, No.2, No.8, and No.10, with "Skipped First" set to 10 seconds.

Figure A-2 shows the results when the raw data are processed through the square wave correlation filter only. Figure A-3 shows the results when the raw data are processed through the FIR filter and the square wave correlation filter in sequence.

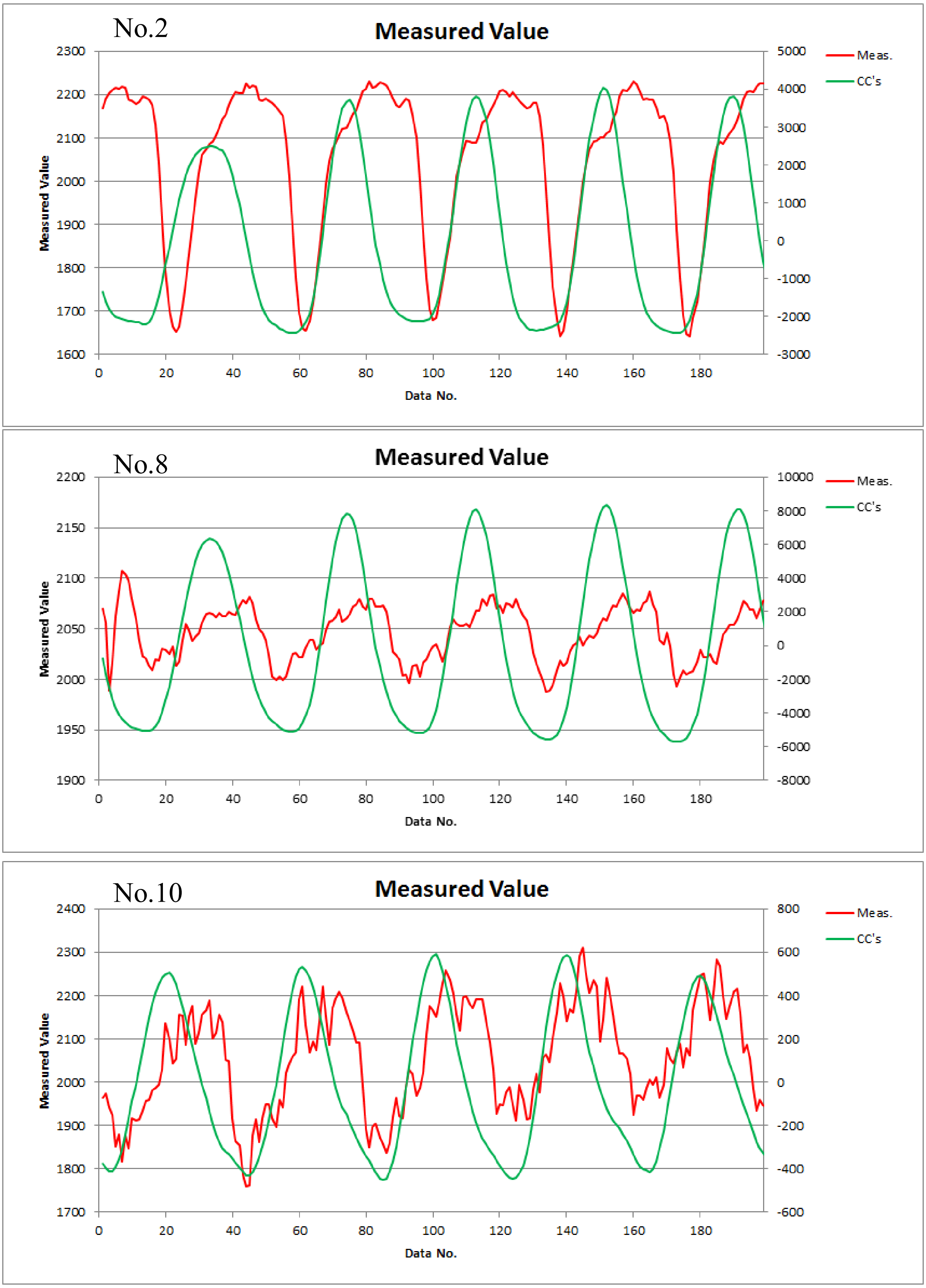


Figure A-2 Results with Square Wave Correlation Filtering Only

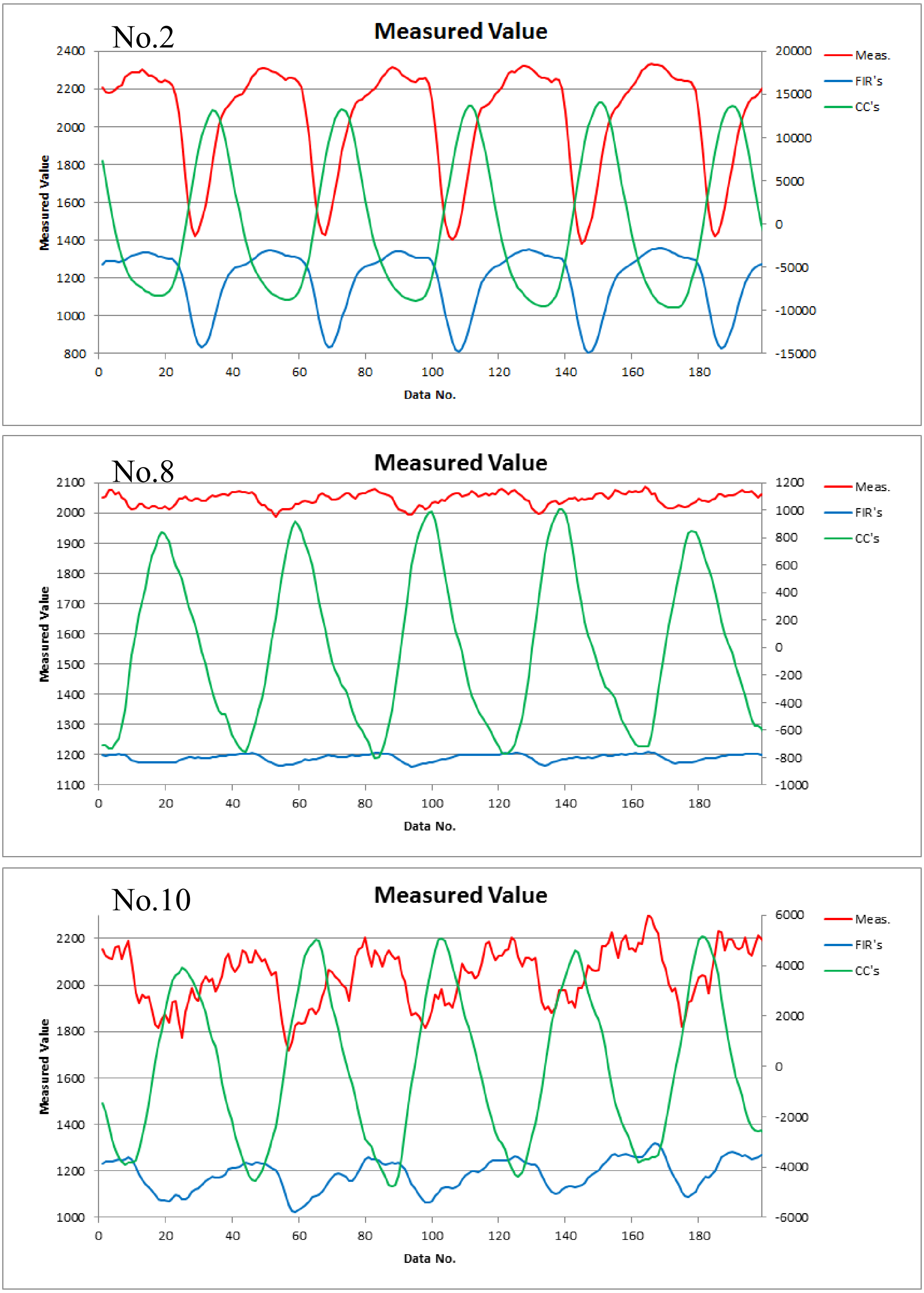


Figure A-3 Results with FIR and Square Wave Correlation Filtering

Figures A-2 and A-3 show that there is no apparent advantage between the FIR filtering results and the FIR and square wave correlation filtering results. As for shaping pulse waves, the square wave correlation filter is more effective than the FIR filter. Therefore, it may be concluded that the pulse rate can be calculated by counting the appearance frequency of the minimum peaks of cross-correlation value per unit time. Of course, it was affected by the hardware LPF processing.

The sample program s1c17w23_ppg_demo_gnu17vx calculates the heart rate by processing data through the FIR and square wave correlation filters in sequence in order to reduce the influence of noise.

Note that the pulse wave measurement on a wrist is subject to noise, as the signal variation is very small. Especially when obtaining the pulse rate during exercise, it should be calculated in conjunction with another sensor, such as a motion sensor. This application note does not use a special optical system and parts difficult to obtain to facilitate confirming reproducibility of the results. Needless to say, to enhance practicality, it is necessary to examine adaption of an optical system having excellent light condensing performance using a lens, a large-area light receiving element having high sensitivity, and a high-luminance LED.

List of references

1) Japan Patent Office, *Iryo Yo Sindankigu 4.2.2 Myakuhakei (Koden-Shiki)*
(Medical diagnostic Instrument 4.2.2 Pulse Wave Meter (Photoelectric)) (1997)

https://web.archive.org/web/20130331080635/https:/www.jpo.go.jp/shiryou/s_sonota/map/ippan04/4/4-2.htm

2) SFH7050 – Photoplethysmography Sensor Application Note. OSRAM. Fig.2. p.2 (2014).

http://www.osram-os.com/media/resource/HIRES/615710/989832/sfh-7050---photoplethysmography-sensor.pdf

3) *Keitai Yogo No Kiso Chishiki, Part 377: Myakuhaku Sensa Toha*
(Basic Knowledge of Mobile Phone Terms, Part 377: What is Pulse Rate Sensor?), Impress Watch Corporation (2008)

http://k-tai.watch.impress.co.jp/cda/article/keyword/40664.html

4) *Denno Densetsu Vintagechips, Myakuhakei Ga Kansei*
(Denno Densetsu Vintage chips, Pulse Wave Meter Has Been Completed.) (2012)

https://vintagechips.wordpress.com/2012/08/13/%E8%84%88%E6%8B%8D%E8%A8%88%E3%81%8C%E5%AE%8C%E6%88%90/

5) *Digital Filter Sekkei No Ie (Jisso Tantousya Muke Digital Shingo Shori Nyumon)*
(Digital Filter Design House (Introduction to Digital Signal Processing for Person in Charge of Implementation))

http://www7b.biglobe.ne.jp/~dfd_house/

6) *Ishikawa Kosen Yamada Yoji Kenkyushitsu Home Page*, *Digital Filter Design Services*
(National Institute of Technology, Ishikawa College, Yoji Yamada Laboratory Home Page, Digital Filter Design Services) (2006)

http://dsp.jpn.org/dfdesign/

7) *Shizukuryoan Nichiroku, Excel De Shingo Shori*
(Shizukuryoan Nichiroku, Signal Processing using Excel) (2009)

http://d.hatena.ne.jp/licheng/20091127/p1

8) Examined patent application publication H07-067440 (1990)

9) [*Ore Senshingu*] "*PC No Kamera De Hi-sesshoku Baitaru Senshingu Ga Dekiru*"
([My Sensing] "Contactless Vital Sensing Using PC Camera") (2013)

http://www.neo-tech-lab.co.uk/WebCam/

10) *Google OS Jikkenshitsu – Moonlight Asuka – Myakuhaku Senshingu Ni Charenji (3)*
(Google OS laboratory – Moonlight Asuka – Challenge to Pulse Wave Sensing (3)) (2015)

http://google-os.blog.jp/archives/50818067.html

Revision History

Attachment-1

Rev. No.	Date	Page	Category	Contents
Rev 1.0	2016/08/09	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.
4th Floor, Tower 1 of China Central Place, 81 Jianguo Road, Chaoyang District, Beijing 100025 China
Phone: +86-10-8522-1199 FAX: +86-10-8522-1120

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
Room 804-805, 8 Floor, Tower 2, Ali Center,No.3331
Keyuan South RD(Shenzhen bay), Nanshan District, Shenzhen
518054, CHINA
Phone: +86-10-3299-0588 FAX: +86-10-3299-0560

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
19F, 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 Department
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5816 FAX: +81-42-587-5117

Document Code: 413346700
Issue October 2016 in JAPAN Ⓛ