To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

RENESAS

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

Application Note

V850ES/Jx3-L

Sample Program
(16-bit Timer/Event Counters (TMP, TMQ))
Pulse Width Measurement Mode

This document gives an operation overview of the sample program and describes how to use it, as well as how to set and use the pulse width measurement function of the 16-bit timer/event counters (TMP, TMQ). In the sample program, using the pulse width measurement function of the 16-bit timer/event counter (TMP), measurement is performed with the interval during which SW1 is pressed as the pulse width, and LED1 is lit for the length of time that SW1 is pressed.

Target devices
V850ES/JF3-L microcontrollers
V850ES/JG3-L microcontrollers

Document No. U20013EJ1V0AN00
Date Published September 2009 N

CONTENTS

CHAPTER 1 OVERVIEW

In this sample program, an example of using the pulse width measurement function of the 16-bit timer/event counter (TMP) is presented. Measurement is performed with the interval during which SW1 is pressed as the pulse width[Note 1], and LED1 is lit only for the length of time that SW1 is pressed.

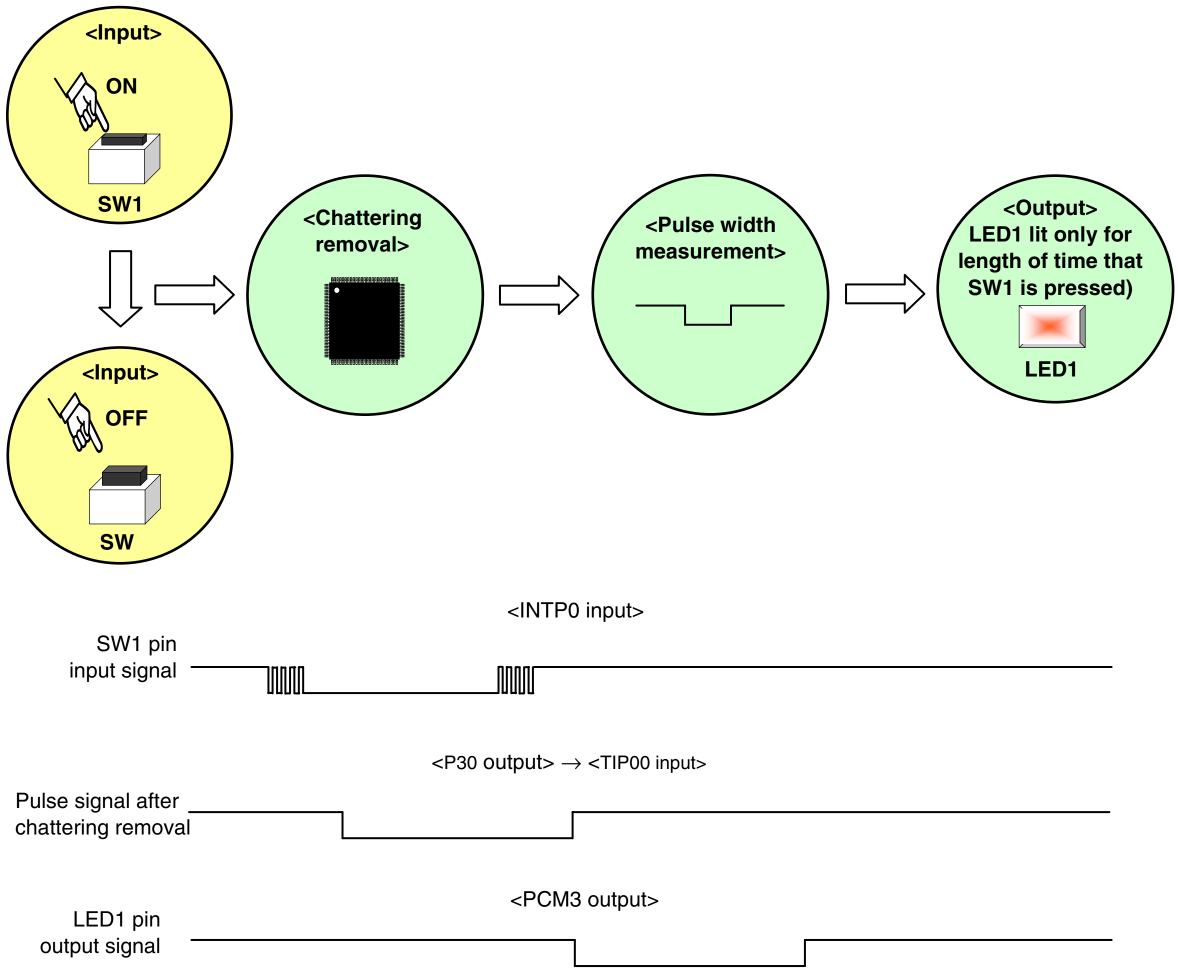
In pulse width measurement, measurement can be performed up to 104.856 ms by counting the set count clock (625 kHz) with a 16-bit counter. In this sample program, measurement for a duration even longer than this is possible by counting the number of times that the 16-bit counter has overflowed with software[Note 2].

Peripherals that stop immediately after a reset and are not used in the sample program are not set up.

Notes 1. In order to eliminate the influence of chattering for SW1 input, a pulse to remove chattering is generated by software, and its pulse width is measured.

2. The switch depression time is 5 seconds maximum, and even if the switch is pressed longer than that, the switch depression time is considered to be 5 seconds. SW1 depression input is not acknowledged while LED1 is lit.

Figure 1-1. Operation Overview



Remark In this sample program, the active level of LED1 and SW1 is the low level.

1.1 Initial Settings

The main initial settings are as follows:

(1) Referencing option byte>

Referencing the oscillation stabilization time immediately after a reset

(2) Setting up on-chip peripherals

- Setting the number of wait cycles to 1 for bus access to on-chip peripheral I/O registers
- Specifying normal operation mode by using the on-chip debug mode register (OCDM)
- Stopping the internal oscillator and watchdog timer 2
- Specifying that the CPU clock frequency not be divided
- Specifying PLL mode and 20 MHz operation (5 MHz × 4)

(3) Pin settings

- Setting up unused pins
- Setting of external interrupt pin through SW1 input (INTP0 function)
- Selecting the LED1 output pin (PCM3 pin)
- Setting of output pin after SW1 input chattering removal (P30 pin)
- Setting of input pin for pulse width measurement (TIP00 function)

(4) External interrupt (INTP0) settings

- Setting of valid edge of INTP0 input signal to both edges
- Setting the priority of the INTP0 interrupt to level 7 and unmasking it

(5) Timer P0 (TMP0) settings

- Specifying the undivided f_{xx}/32 (625 kHz) as the count clock frequency by using the TP0CTL0 register
- Specifying pulse width measurement mode as the timer operation mode by using the TP0CTL1 register
- TP0IOC1: Setting of valid edge of capture trigger input signal to both edges
- Setting the priority of the INTTP0CC0 interrupt to level 7 and unmasking it
- TP0OVMK: Masking of overflow interrupt of 16-bit counter
- TP0CE: Enables TMP0

(6) Variable setting

- Overflow counter initialization

1.2 Servicing External Interrupt Through SW1 Input (INTP0)

Both the rising edge and the falling edge of the INTP0 pin input signal are detected through SW1 input and the corresponding interrupt servicing is executed.

In the case of external interrupt servicing through SW1, approximately 10 ms after the detection of the edge of the INTP0 pin signal, SW1 depression or release is checked for, and the output level of the P30 pin is changed.

If the SW1 input level changes approximately 10 ms after detection of the edge of the INTP0 pin signal, this is judged to be chattering noise and the output level of the P30 pin is not changed.

Remark For the cautions regarding device use, see the user's manual for that device (V850ES/Jx3-L).

1.3 Servicing Capture Interrupts by Using 16-Bit Timer/Event Counter (TMP0)

Both the rising edge and the falling edge of the TIP00 pin input signal are detected through P30 pin output and the corresponding interrupt servicing is executed.

In the case of capture interrupt servicing using the 16-bit timer/event counter (TMP0), the time from when SW1 is pressed until it is released is calculated by measuring the low level width of the pulse from falling edge detection to rising edge detection. LED1 is lit just for the calculated SW1 depression acknowledgment time (5 seconds maximum).

Moreover, external interrupts through detection of the edge of the INTP0 pin input that occur during LED1 lit processing are invalid.

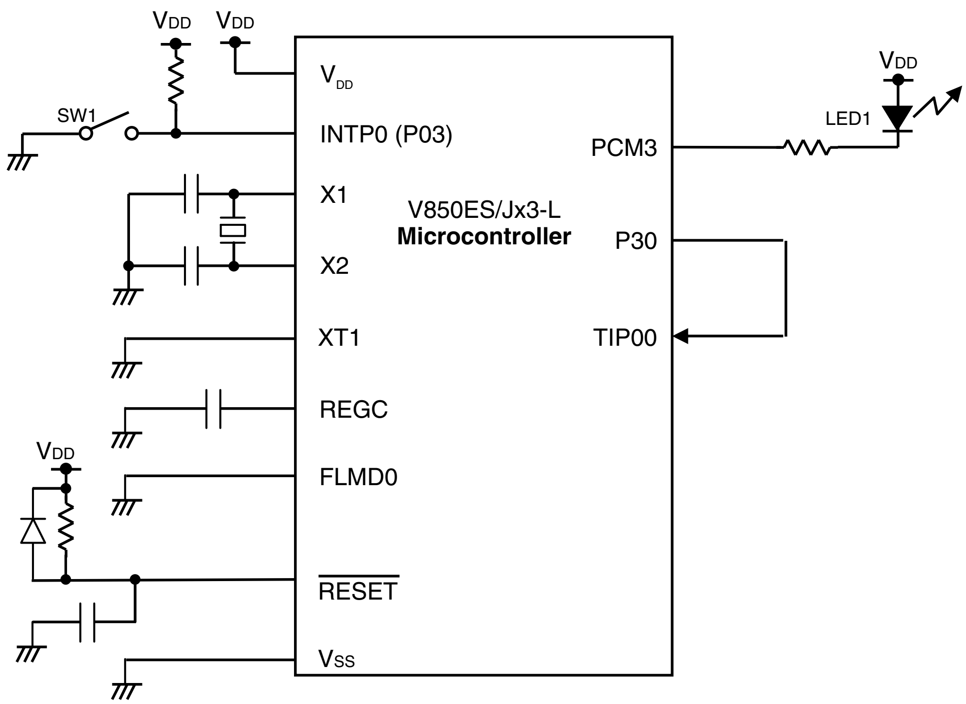
CHAPTER 2 CIRCUIT DIAGRAM

This chapter provides a circuit diagram and describes the peripheral hardware used in the sample program.

2.1 Circuit Diagram

The circuit diagram is shown below.

Figure 2-1. Circuit Diagram



Cautions 1. Connect the EV_{DD}, AV_{REF0}, and AV_{REF1} pins directly to V_{DD}.

2. Connect the EV_{SS} and AV_{SS} pins directly to GND.

3. Connect the FLMD0 pin to GND in normal operation mode.

4. Connect the REGC pin to GND via a capacitor (recommended capacitance: 4.7 μF).

5. Leave all unused ports open because they will be handled as output ports.

2.2 Peripheral Hardware

The peripheral hardware to be used is shown below.

(1) Switch (SW1)

This switch is used as the interrupt input for pulse output control.

(2) LED (LED1)

This LED is used as the output corresponding to the switch input time.

CHAPTER 3 SOFTWARE

This chapter describes the files included in the compressed files that are downloaded, on-chip peripherals of the microcontroller, and the initial settings, and provides an operational overview of the sample program. Flowcharts are also shown.

3.1 Included Files

The following table shows the files included in the compressed files that are downloaded.

File Name (Tree)	Description	Compressed (*.zip) Files Included	
			
• conf — crtE.s	Startup routine file[Note 1]	–	●
AppNote_Pulse.dir	Link directive file[Note 2]	●	●
AppNote_Pulse.prj	Project file for integrated development environment PM+	–	●
AppNote_PWM.prw	Workspace file for integrated development environment PM+	–	●
• src — main.c	C source file including code for hardware initialization processing and the main microcontroller processing	●	●
minicube2.s	Source file for reserving the area for MINICUBE®2	●	●
opt_b.s	Source file for specifying values for the option byte	●	●

Notes 1. This is the startup file copied when “Copy and Use the Sample file” is selected in the Startup File dialog box in the New WorkSpace wizard. (If the default installation path is used, the startup file will be a copy of C:\Program Files\NEC Electronics Tools\CA850*version*\lib850\r32\crtE.s.)

2. This is the link directive file automatically generated if “Create and Use the Sample file” is selected and “Use Internal memory only” is selected for “Memory Usage” in the LinkDirective File dialog box in the New WorkSpace wizard, and to which a segment for MINICUBE2 is added. (If the default installation path is used, C:\Program Files\NEC Electronics Tools\PM+*version*\bin\w_data\V850_i.dat is used as the reference file.)

Remark  : Only the source file is included.

 : The files used with the integrated development environment PM+ are included.

3.2 On-Chip Peripherals Used

The following on-chip peripherals of the microcontroller are used in this sample program:

- Pulse width measurement function: 16-bit timer/event counter (TMP0)
- External interrupt input (for switch input): INTP0 (SW1)
- Capture trigger input port: TIP00[Note]
- Output port (LED1): PCM3
- Output port (for pulse output to TIP00 pin): P30

Note For the V850ES/JG3-L microcontrollers, this pin also functions as ASCKA0, $\overline{\text{SCKB4}}$, TOP00, or P32.
For the V850ES/JF3-L microcontrollers, this pin also functions as ASCKA0, TOP00, or P32.

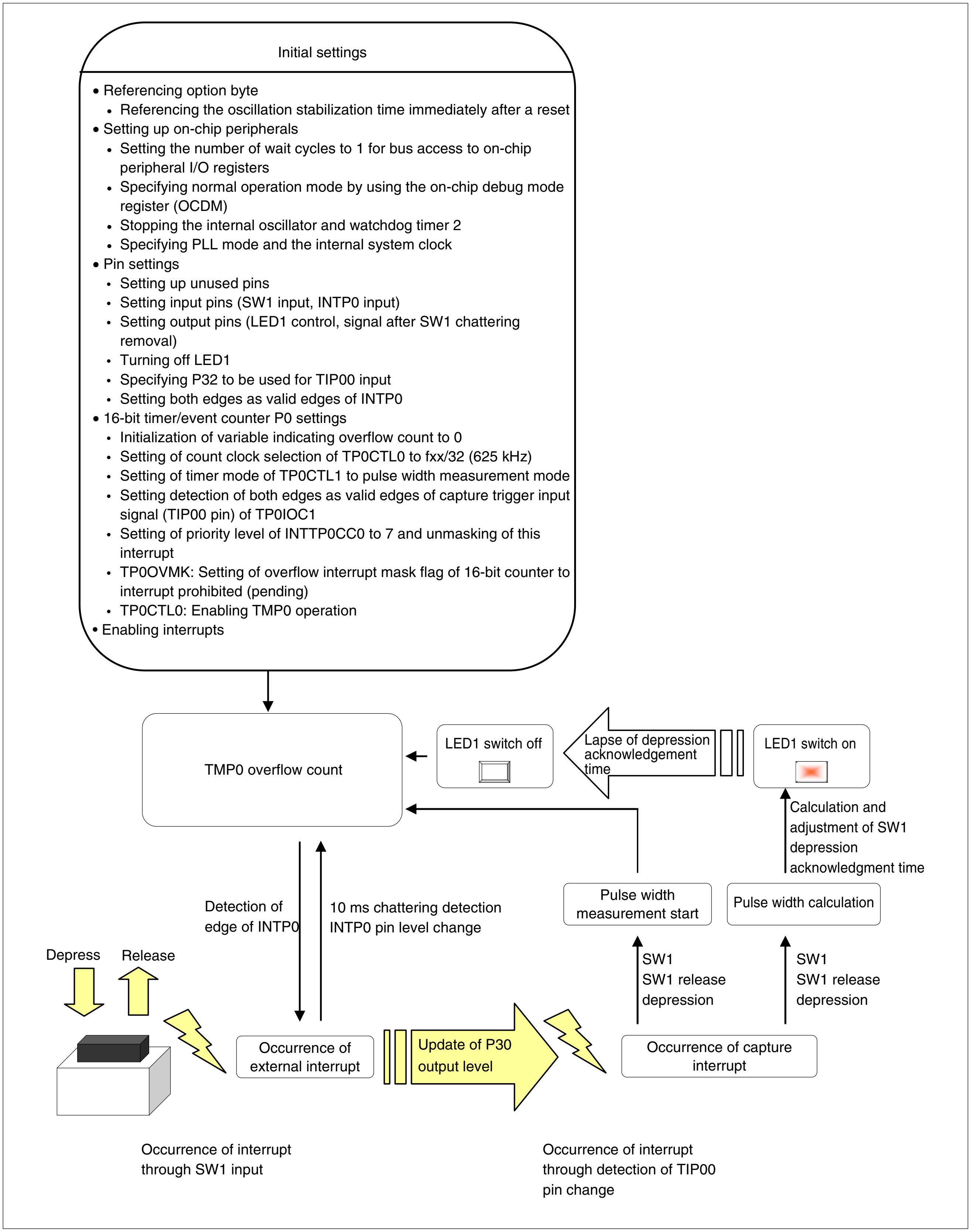
3.3 Initial Settings and Operational Overview

As the initial settings in the sample program, the clock frequency is selected, watchdog timer 2 is stopped, and settings for the I/O ports, pulse width measurement mode from the 16-bit timer/event counter (TMP0), and interrupts are specified.

After the initial settings are specified, the depressed state of SW1 is checked through external interrupt input, and the output value to the P30 pin is switched. Moreover, after the input pulse width from the P30 pin to the TIP00 pin is measured using the pulse width measurement mode of the 16-bit timer/event counter (TMP0) of TIP00, and the time from when SW1 is pressed until it is released is calculated, LED1 is lit. The LED1 lit time corresponds to the SW1 depressed time but is a maximum of 5 seconds, and while LED1 is lit, SW1 depression is not acknowledged.

The details are described in the status transition diagram shown below.

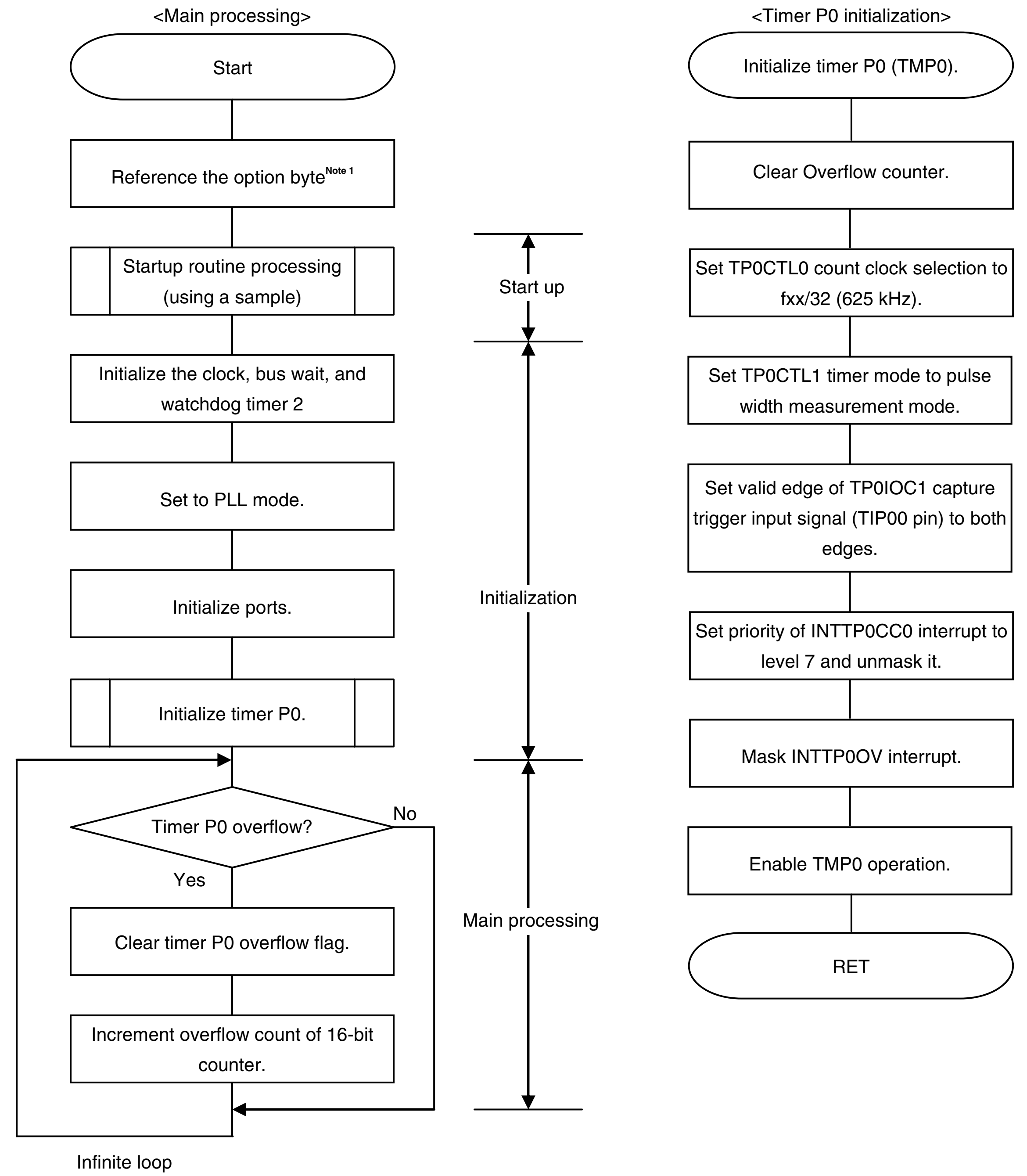
Figure 3-1. Status Transition



3.4 Flowcharts

Flowcharts for the sample program are shown below.

Figure 3-2. Status Transition (1/3)



Note The option byte is automatically referenced by the microcontroller immediately after a reset ends. In this sample program, the oscillation stabilization time immediately after a reset ends is set to 6.554 ms using the option byte.

Figure 3-2. Status Transition (2/3)

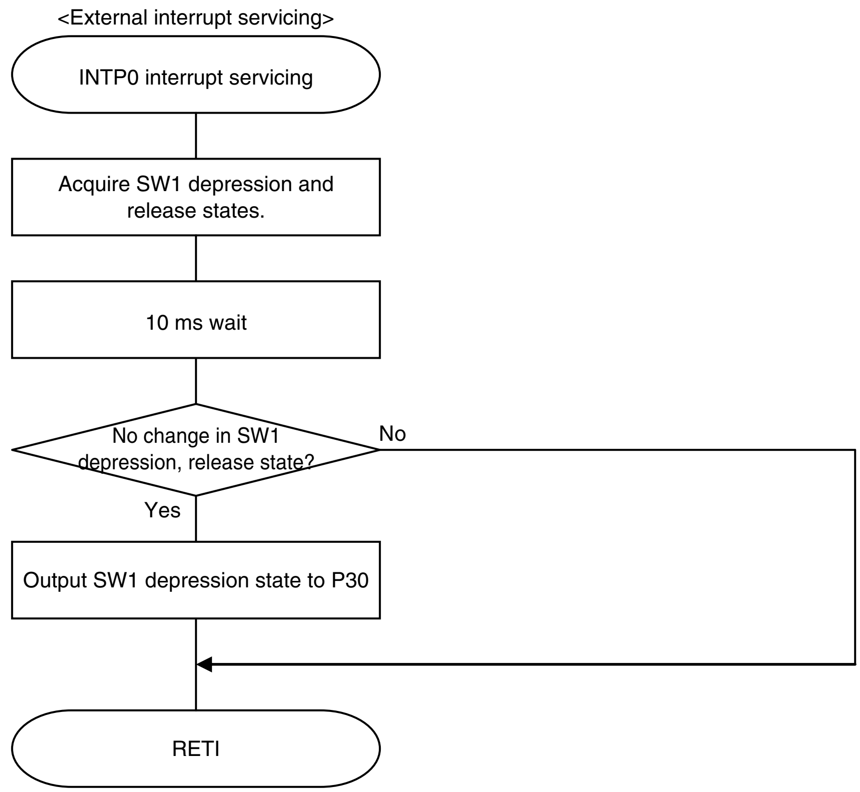
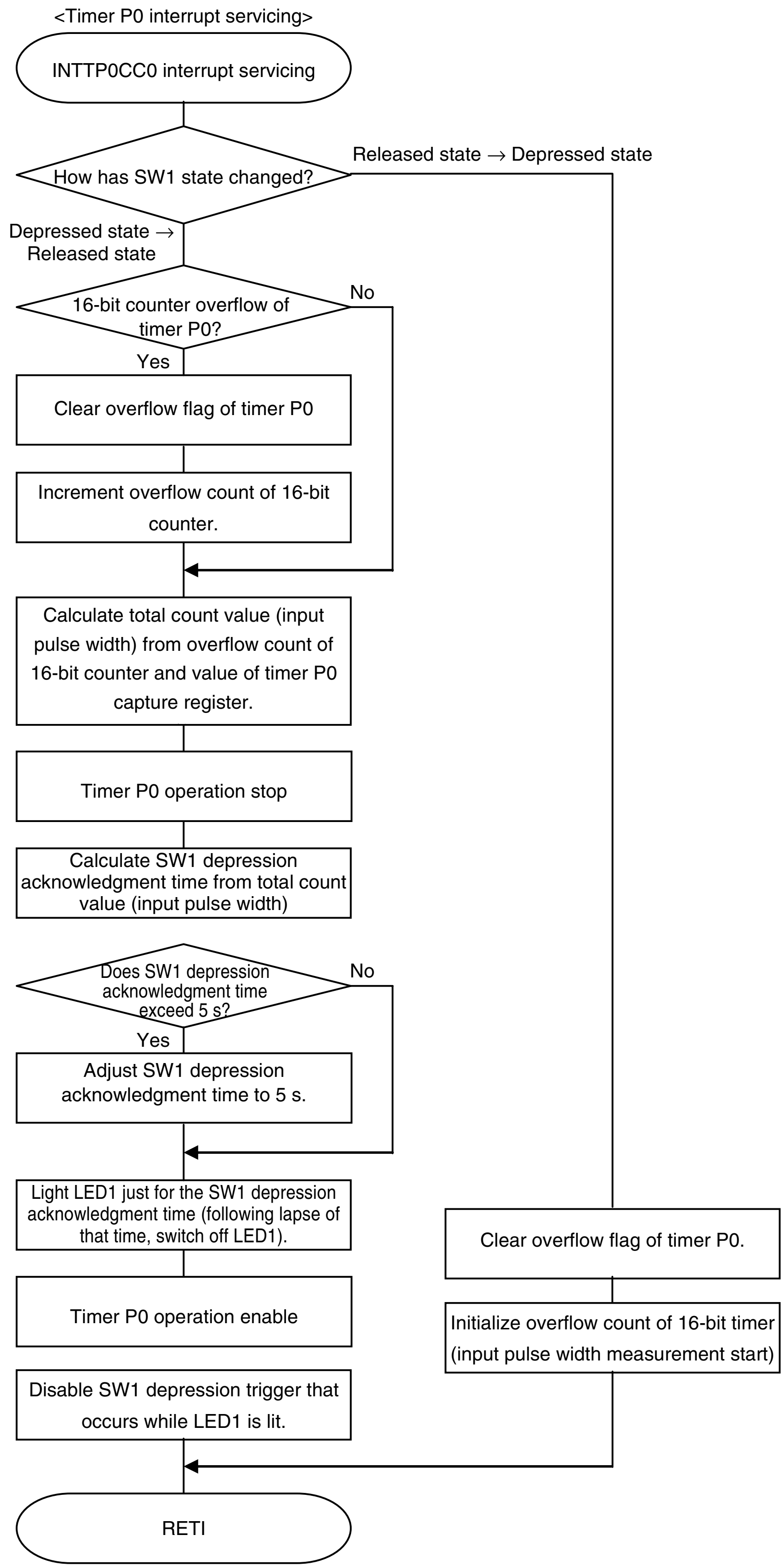


Figure 3-23. Status Transition (3/3)



[Column] Contents of the startup routine

The startup routine is executed before executing the main function immediately after resetting the V850 ends. Basically, the startup routine executes initialization so that the C program can start.
Specifically, the following are performed:

- Allocating the argument space for the main function
- Allocating the stack
- Setting up the reset handler when a reset is input
- Setting up the text pointer (tp)
- Setting up the global pointer (gp)
- Setting up the stack pointer (sp)
- Setting up the element pointer (ep)
- Specifying mask values for the mask registers (r20 and r21)
- Clearing the sbss and bss areas to 0
- Specifying the CTBP value for the prologue/epilogue runtime library
- Specifying r6 and r7 as the arguments for the main function
- Branching to the main function

3.5 Differences Between V850ES/JG3-L and V850ES/JF3-L

The V850ES/JG3-L is the V850ES/JF3-L with its functions, such as I/Os, timer/counters, and serial interfaces, expanded.

In this sample program, the initialization range of P1, P3, P7, P9, and PDH during I/O initialization differs.

See **APPENDIX A PROGRAM LIST** for details about the sample program.

3.6 Difference Between TMP and TMQ

16-bit timer/event counter P (TMP) and 16-bit timer/event counter Q (TMQ) differ in the number of capture trigger pins, timer output pins, and capture/compare registers.

In the sample program, 16-bit timer/event counter P (TMP) is used. When using 16-bit timer/event counter Q (TMQ), see **CHAPTER 4 SETTING UP REGISTERS** and **APPENDIX A PROGRAM LIST** for the settings.

3.7 Security ID

The flash memory can be protected from unauthorized reading by using a 10-byte ID code for authentication when executing on-chip debugging using an on-chip debug emulator.

For details about the security ID, see the **V850ES/Jx3-L Sample Program (Interrupt) External Interrupt Generated by Switch Input Application Note**.

CHAPTER 4 SETTING UP REGISTERS

This chapter describes the settings of 16-bit timer/event counter P (TMP) and 16-bit timer/event counter Q (TMQ).
Peripherals that are stopped immediately after a reset and are not used in this sample program are not set up.
For details about how to set up registers, see the user's manual for the device used.

- V850ES/JG3-L 32-bit Single-Chip Microcontroller Hardware User's Manual
- V850ES/JF3-L 32-bit Single-Chip Microcontroller Hardware User's Manual

For details about extended C code, see the **CA850 C Compiler Package C Language User's Manual**.

4.1 Setting Up 16-bit Timer/Event Counter P (TMP)

The following nine registers are used to set up 16-bit timer/event counter P (TMP):

- TMPn control register 0 (TPnCTL0)
- TMPn control register 1 (TPnCTL1)
- TMPn I/O control register 0 (TPnIOC0)
- TMPn I/O control register 1 (TPnIOC1)
- TMPn I/O control register 2 (TPnIOC2)
- TMPn option register 0 (TPnOPT0)
- TMPn capture/compare register 0 (TPnCCR0)
- TMPn capture/compare register 1 (TPnCCR1)
- TMPn counter read buffer register (TPnCNT)

Remarks 1. n = 0 to 5 (V850ES/JG3-L), n = 0 to 2, or 5 (V850ES/JF3-L)

2. n = 0 in the sample program

3. The eleven registers below are used to set up 16-bit timer/event counter Q (TMQ).
The description on the following pages is of TMP. Therefore, when using TMQ, read the above registers as those below, and read TP in the bit names as TQ.

- TMQ0 control register 0 (TQ0CTL0)
- TMQ0 control register 1 (TQ0CTL1)
- TMQ0 I/O control register 0 (TQ0IOC0)
- TMQ0 I/O control register 1 (TQ0IOC1)
- TMQ0 I/O control register 2 (TQ0IOC2)
- TMQ0 option register 0 (TQ0OPT0)
- TMQ0 capture/compare register 0 (TQ0CCR0)
- TMQ0 capture/compare register 1 (TQ0CCR1)
- TMQ0 capture/compare register 2 (TQ0CCR2)
- TMQ0 capture/compare register 3 (TQ0CCR3)
- TMQ0 counter read buffer register (TQ0CNT)

4.1.1 Setting up 16-bit timer/event counter P (TMP) operation clock

TMPn control register 0 (TPnCTL0) selects the count clock for 16-bit timer/event counter P (TMP) and controls the counter.

Values must be specified for the TPnCKS2 to TPnCKS0 bits when the TPnCE bit is 0.

In this sample program, fxx/32 (625 kHz) is selected by clearing the TPnCKS2 to TPnCKS0 bits at initialization. After specifying the settings for the 16-bit timer/event counter P (TMP) registers, set the TPnCE bit to 1.

Figure 4-1. TPnCTL0 Register Format

TMPn control register 0 (TPnCTL0)
Address: TP0CTL0 0xFFFFF590, TP1CTL0 0xFFFFF5A0,
TP2CTL0 0xFFFFF5B0, TP3CTL0 0xFFFFF5C0,
TP4CTL0 0xFFFFF5D0, TP5CTL0 0xFFFFF5E0

7	6	5	4	3	2	1	0
TPnCE	0	0	0	0	TPnCKS2	TPnCKS1	TPnCKS0

TPnCE	TMPn operation control
0	TMPn disabled (TMPn reset asynchronously).
1	TMPn enabled. TMPn starts.

TPnCKS2	TPnCKS1	TPnCKS0	Internal count clock selection	
			n = 0, 2, 4	n = 1, 3, 5
0	0	0	fxx	
0	0	1	fxx/2	
0	1	0	fxx/4	
0	1	1	fxx/8	
1	0	0	fxx/16	
1	**0**	**1**	fxx/32	
1	1	0	fxx/64	fxx/256
1	1	1	fxx/128	fxx/512

Remark The red values are specified in this sample program.

4.1.2 Setting up 16-bit timer/event counter P (TMP) operation mode

TMPn control register 1 (TPnCTL1) specifies the operation mode of 16-bit timer/event counter P (TMP).

In this sample program, the pulse width measurement is specified by specifying 110b for the TPnMD2 to TPnMD0 bits.

Figure 4-2. TPnCTL1 Register Format

TMPn control register 1 (TPnCTL1)

Address: TP0CTL1 0xFFFFF591, TP1CTL1 0xFFFFF5A1,
TP2CTL1 0xFFFFF5B1, TP3CTL1 0xFFFFF5C1,
TP4CTL1 0xFFFFF5D1, TP5CTL1 0xFFFFF5E1

7	6	5	4	3	2	1	0
0	TPnEST	TPnEEE	0	0	TPnMD2	TPnMD1	TPnMD0

TPnEST	Software trigger control
0	–
1	Generate a valid signal for external trigger input.

TPnEEE	Count clock selection
0	Disable operation with external event count input[Note].
1	Enable operation with external event count input.

TPnMD2	TPnMD1	TPnMD0	Timer mode selection
0	0	0	Interval timer mode
0	0	1	External event count mode
0	1	0	External trigger pulse output mode
0	1	1	One-shot pulse output mode
1	0	0	PWM output mode
1	0	1	Free-running timer mode
1	**1**	**0**	Pulse width measurement mode
1	1	1	Setting prohibited

Note Clock selected using the TPnCK0 to TPnCK2 bits of the TPnCTL0 register

Remark The red values are specified in this sample program.

4.1.3 Controlling timer output

TMPn I/O control register 0 (TPnIOC0) controls timer output.

In the case of operation in the pulse width measurement mode, control of TMPn I/O control register 0 is not required, so this register is not controlled in this sample program.

Figure 4-3. TPnIOC0 Register Format

TMPn I/O control register 0 (TPnIOC0)

Address: TP0IOC0 0xFFFFF592, TP1IOC0 0xFFFFF5A2,
TP2IOC0 0xFFFFF5B2, TP3IOC0 0xFFFFF5C2,
TP4IOC0 0xFFFFF5D2, TP5IOC0 0xFFFFF5E2

7	6	5	4	3	2	1	0
0	0	0	0	TPnOL1	TPnOE1	TPnOL0	TPnOE0

TPnOL1	TOPn1 pin output level setting
0	TOPn1 pin starts output at high level
1	TOPn1 pin starts output at low level

TPnOE1	TOPn1 pin output setting
0	Timer output disabled
1	Timer output enabled

TPnOL0	TOPn0 pin output level setting
0	TOPn0 pin starts output at high level
1	TOPn0 pin starts output at low level

TPnOE0	TOPn0 pin output setting
0	Timer output disabled
1	Timer output enabled

Remarks 1. The red values are specified in this sample program.

2. The 16-bit timer/event counter Q (TMQ) differs in that, in addition to TQ being substituted for TP, the TQ0OL3, TQ0OE3, TQ0OL2, and TQ0OE2 bits are assigned to bits 7 to 4 of the TQ0IOC0 register.

4.1.4 Controlling valid edge of capture trigger input signal

TMPn I/O control register 1 (TPnIOC1) controls the valid edge of the capture trigger input signal (from the TIPn0 and TIPn1 pins).

In this sample program, detection of both edges is set for the capture trigger input signal (TIPn0 pin) by specifying 11b for the TPnIS1 and TPnIS0 bits.

Figure 4-4. TPnIOC1 Register Format

TMPn I/O control register 1 (TPnIOC1)

Address: TP0IOC1 0xFFFFF593, TP1IOC1 0xFFFFF5A3,
TP2IOC1 0xFFFFF5B3, TP3IOC1 0xFFFFF5C3,
TP4IOC1 0xFFFFF5D3, TP5IOC1 0xFFFFF5E3

7	6	5	4	3	2	1	0
0	0	0	0	TPnIS3	TPnIS2	TPnIS1	TPnIS0

TPnIS3	TPnIS2	Capture trigger input signal (TIPn1 pin) valid edge setting
0	0	No edge detection
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TPnIS1	TPnIS0	Capture trigger input signal (TIPn0 pin) valid edge setting
0	0	No edge detection
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Remarks 1. The TPnIOC1 register is not used in this sample program.

2. The 16-bit timer/event counter Q (TMQ) differs in that, in addition to TQ being substituted for TP, the TQ0IS7, TQ0IS6, TQ0IS5, and TQ0IS4 bits are assigned to bits 7 to 4 of the TQ0IOC1 register.

4.1.5 Controlling external input signals

TMPn I/O control register 2 (TPnIOC2) controls the valid edge of the external event count input signal (from the TIPn0 pin) and external trigger input signal (from the TIPn0 pin).

In the case of operation in the pulse width measurement mode, control of TMPn I/O control register 0 is not required, so this register is not controlled in this sample program.

Figure 4-5. TPnIOC2 Register Format

TMPn I/O control register 2 (TPnIOC2)

Address: TP0IOC2 0xFFFFF594, TP1IOC2 0xFFFFF5A4,
TP2IOC2 0xFFFFF5B4, TP3IOC2 0xFFFFF5C4,
TP4IOC2 0xFFFFF5D4, TP5IOC2 0xFFFFF5E4

7	6	5	4	3	2	1	0
0	0	0	0	TPnEES1	TPnEES0	TPnETS1	TPnETS0

TPnEES1	TPnESS0	External event count input signal (TIPn0 pin) valid edge setting
0	0	No edge detection
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

TPnETS1	TPnETS0	External trigger input signal (TIPn0 pin) valid edge setting
0	0	No edge detection
0	1	Detection of rising edge
1	0	Detection of falling edge
1	1	Detection of both edges

Remark The TPnIOC2 register is not used in this sample program.

4.1.6 Controlling capture/compare operation

TMPn option register 0 (TPnOPT0) controls the capture/compare operation setting and overflow detection.

In the PWM output mode, the TPnOPT0 register does not have to be controlled. Therefore, the register is not controlled in this sample program.

Figure 4-6. TPnOPT0 Register Format

TMPn option register 0 (TPnOPT0)

Address: TP0OPT0 0xFFFFF595, TP1OPT0 0xFFFFF5A5,
TP2OPT0 0xFFFFF5B5, TP3OPT0 0xFFFFF5C5,
TP4OPT0 0xFFFFF5D5, TP5OPT0 0xFFFFF5E5

7	6	5	4	3	2	1	0
0	0	TPnCCS1	TPnCCS0	0	0	0	TPnOVF

TPnCCS1	TPnCCR1 register capture/compare selection
0	Compare register selected
1	Capture register selected

TPnCCS0	TPnCCR0 register capture/compare selection
0	Compare register selected
1	Capture register selected

TPnOVF	TMPn overflow detection flag
Set (1)	Overflow occurred
Reset (0)	TPnOVF bit 0 written or TPnCTL0.TPnCE bit = 0

Remarks 1. The red parts in the table above are the values referenced by the sample program, and if they have been set, they are reset by software.

2. The 16-bit timer/event counter Q (TMQ) differs in that, in addition to TQ being substituted for TP, the TQ0CCS3, TQ0CCS2, TQ0CCS1, and TQ0CCS0 bits are assigned to bits 7 to 4 of the TQ0OPT0 register.

4.1.7 Referencing count value during capturing

The capture function and compare function of the TMPn capture/compare register 0 (TPnCCR0) and TMPn capture/compare register 1 (TPnCCR1) can be switched only in the free-running timer mode.

In the case of operation in the pulse width measurement mode, upon detection of the valid edge of the capture trigger input (TIPn0 pin and TIPn1 pin), the count value of the 16-bit counter is saved to TMPn capture/compare register 0 (TPnCCR0) and TMPn capture/compare register 1 (TPnCCR1), and the 16-bit counter is cleared.

Remark In the sample program, TIP00 is used as the input pin for pulse width measurement, so that the count value is reflected to the TP0CCR0 register.

Figure 4-7. TPnCCR0 Register Format

TMPn capture/compare register 0 (TPnCCR0)

Address: TP0CCR0 0xFFFFF596, TP1CCR0 0xFFFFF5A6,
TP2CCR0 0xFFFFF5B6, TP3CCR0 0xFFFFF5C6,
TP4CCR0 0xFFFFF5D6, TP5CCR0 0xFFFFF5E6

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Figure 4-8. TPnCCR1 Register Format

TMPn capture/compare register 1 (TPnCCR1)

Address: TP0CCR1 0xFFFFF598, TP1CCR1 0xFFFFF5A8,
TP2CCR1 0xFFFFF5B8, TP3CCR1 0xFFFFF5C8,
TP4CCR1 0xFFFFF5D8, TP5CCR1 0xFFFFF5E8

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Remark The TPnCCR1 register is not used in the sample program.

Four capture/compare registers (TQ0CCR0 to TQ0CCR3) are provided for 16-bit timer/event counter Q (TMQ). These registers are used in the same way as the TPnCCR0 and TPnCCR1 registers.

4.1.8 Referencing timer count value

The TPnCNT register is a read buffer register from which a 16-bit counter value can be read.

The current counter value can be read by reading this register while the timer is running (TPnCTL0.TPnCE bit = 1).

If this register is read while the timer is stopped (TPnCTL0.TPnCE bit = 0), 0000H is returned.

In the sample program, this register is not used because counter values do not have to be referenced.

Figure 4-9. TPnCNT Register Format

TMPn counter read buffer register (TPnCNT)

Address: TP0CNT 0xFFFFF59A, TP1CNT 0xFFFFF5AA,
TP2CNT 0xFFFFF5BA, TP3CNT 0xFFFFF5CA,
TP4CNT 0xFFFFF5DA, TP5CNT 0xFFFFF5EA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Remark The TPnCNT register is not used in the sample program.

4.1.9 Example of timer settings

[Example 1] When starting the timer with 16-bit timer/event counter (TMP) set to the pulse width measurement mode and the detection edge for count clock and capture trigger input signal specified (same as the sample program)

Setup procedure:
<1> Specify fxx/32 (625 kHz) as the count clock frequency.
<2> Set the pulse width measurement mode.
<3> Set the capture trigger input signal to both edges.
<4> Unmask the relevant interrupt.
<5> Start the timer operation.

Program example (same as the sample program)

```
/* Timer P0 settings */
TP0CTL0 = 0x05;          /* Specifies fxx/32 as the count clock frequency.              */ }<1>
                         /* Specifies Failsafe (Clears TP0CE to 0) to stop TMP0.        */
TP0CTL1 = 0x06;          /* Specifies the pulse width measurement mode.                 */ }<2>
TP0IOC1 = 0x03;          /* Detects both edges of capture trigger input signal.         */ }<3>

/* Interrupt control register setup */
TP0CCIC0 = 0x07;         /* Sets the priority of INTTP0CC0 to level 7 and unmasks it.   */ }<4>

TP0CE = 1;               /* Starts TMP0.                                                */ }<5>
```

[Example 2] When starting the timer with 16-bit timer/event counter Q (TMQ) set to the pulse width measurement mode and the detection edge of count clock and capture trigger input signal specified

Setup procedure:

<1> Specify fxx/32 (625 kHz) as the count clock frequency.
<2> Set the pulse width measurement mode.
<3> Set the capture trigger input signal to both edges.
<4> Unmask the relevant interrupt.
<5> Start the timer operation.

Program example

```
/* Timer Q0 settings */
TQ0CTL0 = 0x05;         /* Specifies fxx/32 as the count clock frequency.            */} <1>
                        /* Specifies Failsafe (Clears TQ0CE to 0) to stop TMP0.      */
TQ0CTL1 = 0x06;         /* Specifies the pulse width measurement mode.               */} <2>
TQ0IOC1 = 0x03;         /* Detects both edges of capture trigger input signal.       */} <3>

/* Interrupt control register setup */
TQ0CCIC0 = 0x07;        /* Sets the priority of INTTQ0CC0 to level 7 and unmasks it. */} <4>

TO0CE = 1               /* Starts TMQ0.                                              */} <5>
```

4.2 Setting SW1 Input Interrupt Pins

In this sample program, the P03 pin is used as the external interrupt pin for SW1 input, and interrupt output is set for detection of both the rising and falling edges.

4.2.1 Setting up port 0 mode register (PM0)

The PM0 register enables control of the I/O mode of the P02 to P06 pins.

In this sample program, the pin state is read through chattering verification, to set the I/O mode of the P03 pin to the "input mode".

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to FFH.

Figure 4-10. PM0 Register Format

Port 0 mode register (PM0)
Address: 0xFFFFF420

7	6	5	4	3	2	1	0
1	PM06	PM05[Note]	PM04	PM03	PM02	1	1

PM03	Control of I/O mode of P03 pin
0	Output mode
1	Input mode

Remarks 1. The red parts in the table above are the setting values in the sample program.

2. If PM03 is set to the output mode, the value of the output latch during P03 read is read out, so caution is required.

4.2.2 Setting up port 0 mode control register (PMC0)

The PMC0 register can specify the operation mode of pins P02 to P06.

In this sample program, the operation mode of the P03 pin is set to INTP0 (external interrupt) input.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Figure 4-11. PMC0 Register Format

Port 0 mode control register (PMC0)
Address: 0xFFFFF440

7	6	5	4	3	2	1	0
0	PMC06	PMC05[Note]	PMC04	PMC03	PMC02	0	0

PM03	Specification of operation mode of P03 pin
0	I/O port
1	INTP0 input/ADTRG input

Note The P05/INTP2/$\overline{\text{DRST}}$ pin is used as the $\overline{\text{DRST}}$ pin regardless of the value of the PMC05 bit when OCDM.OCDM0 bit 0 = 1.

Remarks 1. The red parts in the table above are the setting values in the sample program.

2. Pins P02 to P06 have hysteresis characteristics during alternate function input, but in the port mode, they do not have hysteresis characteristics.

4.2.3 Setting up external interrupt falling edge, rising edge specification register 0 (INTF0, INTR0)

This is an 8-bit register that specifies falling edge and rising edge detection for the NMI pin with bit 2 and for the external interrupt pins (INTP0 to INTP3) with bits 3 to 6.

Detection of both edges is set in this sample program.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Figure 4-12. INTF0/INTR0 Register Format

External interrupt falling edge specification register 0 (INTF0)
Address: 0xFFFFFC00

7	6	5	4	3	2	1	0
0	INTF06	INTF05	INTF04	INTF03	INTF02	0	0
	INTP3	INTP2	INTP1	INTP0	NMI		

External interrupt rising edge specification register 0 (INTR0)
Address: 0xFFFFFC20

7	6	5	4	3	2	1	0
0	INTR06	INTR05	INTR04	INTR03	INTR02	0	0
	INTP3	INTP2	INTP1	INTP0	NMI		

Specification of Valid Edge

INTF03	INTR03	Valid edge specification
0	0	No edge detection
0	1	Detection of rising edge
1	0	Detection of falling edge
1	**1**	Both edges

Remark The red parts in the table above are the setting values in the sample program.

4.2.4 Setting up interrupt control register (PIC0)

This register is allocated for each interrupt request signal (maskable interrupt) and sets the control conditions for each interrupt.

In this sample program, INTP0 can be used at the lowermost priority level.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 47H.

Caution To read out the PIC0.PIF0 bit, do so in the interrupt disabled (DI) status. If the PIF0 bit is read out in the interrupt enabled (EI) status and the interrupt acknowledgment and bit readout timings conflict, the correct value may not be read.

Figure 4-13. PIC0 Register Format

INTP0 interrupt control register (PIC0)
Address: 0xFFFFF112

7	6	5	4	3	2	1	0
PIF0	PMK0	0	0	0	PPR02	PPR01	PPR00

PIF0	Interrupt request flag
0	No interrupt request signal
1	Interrupt request signal

PMK0	Interrupt mask flag
0	Interrupt servicing enabled
1	Interrupt servicing disabled (held pending)

PPR02	PPR01	PPR00	Interrupt priority specification
0	0	0	Level 0 (highest)
1	0	1	Level 1
0	1	0	Level 2
0	1	1	Level 3
1	0	0	Level 4
1	0	1	Level 5
1	1	0	Level 6
1	1	1	Level 7 (lowest)

Remark The red parts in the table above are the setting values in the sample program.

Interrupt request flag PIF0
The PIF0 interrupt request flag of the PIC0 register is set to "1" when an interrupt source occurs, and upon acknowledgment of an interrupt request signal, it is automatically reset by hardware.

4.3. Setting Up Capture Interrupt Pin for Pulse Width Measurement Mode

In this sample program, settings are done so that the P30 pin is used as the pulse output pin for SW1 input confirmation, and the P32 pin is used as the capture interrupt pin for the pulse width measurement mode.

4.3.1 Setting up port 3 mode register (PM3)

The PM3 register can control the I/O mode of pins P30 to P39.

In this sample program, the mode is set to the output mode in order to use the P30 pin as the pulse output pin for SW1 input confirmation.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to FFFFH.

Figure 4-14. PM3 Register Format

Port 3 mode register (PM3)
Address: 0xFFFFF426

15	14	13	12	11	10	9	8
1	1	1	1	1	1	PM39	PM38

7	6	5	4	3	2	1	0
PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30

PM30	Control of I/O mode of P30 pin
0	Output mode
1	Input mode

Remark The red parts in the table above are the setting values in the sample program.

4.3.2 Setting up port 3 mode control register (PMC3)

The PMC3 register can set the operation mode of the P30 to P35 pins, the P38 pin, and the P39 pin.

In this sample program, the operation mode of the P32 pin is set to TIP00 input.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 0000H.

Figure 4-15. PMC3 Register Format

Port 3 mode control register (PMC3)
Address: 0xFFFFF446

15	14	13	12	11	10	9	8
0	0	0	0	0	0	PM39	PM38

7	6	5	4	3	2	1	0
0	0	PM35	PM34	PM33	PM32	PM31	PM30

PM32	Control of operation mode of P32 pin
0	I/O port
1	ASCKA0 input/$\overline{\text{SCKB4}}$ I/O/TIP00 input

Remark The red parts in the table above are the setting values in the sample program.

4.3.3 Setting up port 3 function control register (PFC3) and port 3 function control expansion register (PFCE3L)

The PFC3 register and the PFCE3L register can specify the alternate functions of the P30 to P35 pins, the P38 pin, and the P39 pin.

In this sample program, TIP00 input is set in order to use the P32 pin as the capture interrupt pin for the pulse width measurement mode.

This register can be read or written in 16-bit, 8-bit, or 1-bit units.

Reset sets this register to 0000H.

Figure 4-16. PFC3 Register Format

Port 3 function control register (PFC3)
Address: 0xFFFFF466

15	14	13	12	11	10	9	8
0	0	0	0	0	0	PFC39	PFC38

7	6	5	4	3	2	1	0
0	0	PFC35	PFC34	PFC33	PFC32	PFC31	PFC30

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Figure 4-17. PFCE3L Register Format

Port 3 function control register (PFCE3L)
Address: 0xFFFFF706

7	6	5	4	3	2	1	0
0	0	0	0	0	PFCE32	0	0

PM32	PFC32	Specification of alternate function of P32 pin
0	0	ASCKA0 input
0	1	$\overline{\text{SCKB4}}$ I/O
1	**0**	TIP00 input
1	1	TOP00 output

Remark The red parts in the table above are the setting values in the sample program.

4.4 Setting Up Pin for LED Output

In this sample program, after the LED1 output is set to output that does not cause the LED to light up in the initial state, settings are done to make the PCM3 pin operate as an output port.

4.4.1 Setting up port CM register (PCM)

The PCM register can control the output data of the PCM0 to PCM3 pins in 1-bit units.

In this sample program, "Output 1" is set in order to output the PCM3 pin as LED switched off in the initial state.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to 00H.

Figure 4-18. PCM Register Format

Port CM register (PCM)
Address: 0xFFFFF00C

7	6	5	4	3	2	1	0
0	0	0	0	PCM3	PCM2	PCM1	PCM0

PCM3	Control of output data of PCM3 pin (in output mode)
0	Output 0
1	Output 1

Remark The red parts in the table above are the setting values in the sample program.

4.4.2 Setting up port CM mode register (PMCM)

The PMCM register can specify the operation mode of the PCM0 to PCM3 pins.

In this sample program, the operation mode of the PCM3 pin is set to output.

This register can be read or written in 8-bit or 1-bit units.

Reset sets this register to FFH.

Figure 4-19. PMCM Register Format

Port 3 mode control register (PMCM)

Address: 0xFFFFF02C

7	6	5	4	3	2	1	0
1	1	1	1	PMCM3	PMCM2	PMCM1	PMCM0

PCM3	Control of I/O mode of PCM3 pin
0	Output
1	Input

Remark The red parts in the table above are the setting values in the sample program.

CHAPTER 5 RELATED DOCUMENTS

Document Name	Document Number
V850ES/JF3-L Hardware User's Manual Hardware	U18952E
V850ES/JG3-L Hardware User's Manual Hardware	U18953E
PM+ Ver. 6.30 User's Manual	U18416E
CA850 Ver. 3.20 C Compiler Package Operation User's Manual	U18512E
CA850 Ver. 3.20 C Compiler Package C Language User's Manual	U18513E
CA850 Ver. 3.20 C Compiler Package Assembly Language User's Manual	U18514E
CA850 Ver. 3.20 C Compiler Package Link Directives User's Manual	U18515E
V850ES Architecture User's Manual	U15943E
QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual	U18371E
ID850QB Ver. 3.40 Integrated Debugger Operation User's Manual	U18604E

URL of document search website: http://www.necel.com/micro/en/documentation.html

APPENDIX A PROGRAM LIST

The V850ES/JG3-L microcontroller source code is shown below.

● opt_b.s

```
#-------------------------------------------------------------------------------
#
#    NEC Electronics     V850ES/Jx3-L microcontroller
#
#-------------------------------------------------------------------------------
#    V850ES/JG3-L JF3-L  sample program
#-------------------------------------------------------------------------------
#    Pulse width measurement mode
#-------------------------------------------------------------------------------
#[History]
#    2009.8.--   Released
#-------------------------------------------------------------------------------
#[Overview]
#    This sample program specifies a value for the option byte.
#-------------------------------------------------------------------------------

    .section "OPTION_BYTES"
    .byte 0b00000101 -- 0x7a (5MHz: Sets the oscillation stabilization time to 6.554 ms)
    .byte 0b00000000 -- 0x7b        ↑
    .byte 0b00000000 -- 0x7c        ↑
    .byte 0b00000000 -- 0x7d 0x00 must be specified for addresses 0x7b to 0x7f.
    .byte 0b00000000 -- 0x7e        ↓
    .byte 0b00000000 -- 0x7f        ↓
```

● minicube2.s

```
#-----------------------------------------------------------------------------
#
#    NEC Electronics     V850ES/Jx3-L microcontroller
#
#-----------------------------------------------------------------------------
#    V850ES/JG3-L JF3-L  sample program
#-----------------------------------------------------------------------------
#    Pulse width measurement mode
#-----------------------------------------------------------------------------
#[History]
#    2009.8.--   Released
#-----------------------------------------------------------------------------
#[Overview]
#    This sample program allocates the resources required when using MINICUBE2.
#       (Example of using MINICUBE2 via CSIB0)
#-----------------------------------------------------------------------------

    -- Allocation of a 2 KB space as the monitor ROM section
    .section "MonitorROM", const
    .space 0x800, 0xff

    -- Allocation of an interrupt vector for debugging
    .section "DBG0"
    .space 4, 0xff

    -- Allocation of a reception interrupt vector for serial communication
    .section "INTCB0R"
    .space 4, 0xff

    -- Allocation of a 16-byte space as the monitor RAM section
    .section "MonitorRAM", bss
    .lcomm monitorramsym, 16, 4
```

● AppNote_Pulse.dir

```
#   Sample link directive file (not use RTOS/use internal memory only)
#
#   Copyright (C) NEC Electronics Corporation 2002
#   All rights reserved by NEC Electronics Corporation.
#
#   This is a sample file.
#   NEC Electronics assumes no responsibility for any losses incurred by customers or
#   third parties arising from the use of this file.
#
#   Generated        : PM+ V6.31  [ 9 Jul 2007]
#   Sample Version   : E1.00b [12 Jun 2002]
#   Device           : uPD70F3738 (C:\Program Files\NEC Electronics Tools\DEV\DF3738.800)
#   Internal RAM     : 0x3ffb000 - 0x3ffefff
#
#   NOTICE:
#        Allocation of SCONST, CONST and TEXT depends on the user program.
#
#        If interrupt handler(s) are specified in the user program then
#        the interrupt handler(s) are allocated from address 0 and
#        SCONST, CONST and TEXT are allocated after the interrupt handler(s).

SCONST  : !LOAD ?R {
       .sconst           = $PROGBITS    ?A .sconst;
};

CONST   : !LOAD ?R {
       .const            = $PROGBITS    ?A .const;
};

TEXT    : !LOAD ?RX {
       .pro_epi_runtime = $PROGBITS    ?AX .pro_epi_runtime;
       .text             = $PROGBITS    ?AX .text;
};

### For MINICUBE2 ###
MROMSEG : !LOAD ?R V0x03F800{
      MonitorROM = $PROGBITS ?A MonitorROM;
};
```

0x01F800 if the internal ROM size is 128 KB

Difference from the default link directive file (additional code)

An area reserved for MINICUBE2 is allocated.

```
SIDATA   : !LOAD ?RW V0x3ffb000 {
        .tidata.byte    = $PROGBITS     ?AW .tidata.byte;
        .tibss.byte     = $NOBITS       ?AW .tibss.byte;
        .tidata.word    = $PROGBITS     ?AW .tidata.word;
        .tibss.word     = $NOBITS       ?AW .tibss.word;
        .tidata         = $PROGBITS     ?AW .tidata;
        .tibss          = $NOBITS       ?AW .tibss;
        .sidata         = $PROGBITS     ?AW .sidata;
        .sibss          = $NOBITS       ?AW .sibss;
};

DATA     : !LOAD ?RW V0x3ffb100 {
        .data          = $PROGBITS     ?AW   .data;
        .sdata         = $PROGBITS     ?AWG .sdata;
        .sbss          = $NOBITS       ?AWG .sbss;
        .bss           = $NOBITS       ?AW   .bss;
};

### For MINICUBE2 ###
MRAMSEG : !LOAD ?RW V0x03FFEFF0{
MonitorRAM = $NOBITS ?AW MonitorRAM;
};

__tp_TEXT @ %TP_SYMBOL;
__gp_DATA @ %GP_SYMBOL &__tp_TEXT{DATA};
__ep_DATA @ %EP_SYMBOL;
```

Difference from the default link directive file (additional code)

An area reserved for MINICUBE2 is allocated.

● main.c

```
/*------------------------------------------------------------------------------*/
/*
/*    NEC Electronics     V850ES/Jx3-L microcontroller
/*
/*------------------------------------------------------------------------------*/
/*    V850ES/JG3-L sample program
/*------------------------------------------------------------------------------*/
/*    Pulse width measurement mode
/*------------------------------------------------------------------------------*/
/*[History]
/*    2009.8.--   Released
/*------------------------------------------------------------------------------*/
/*[Overview]
/*    This sample program shows an example of using the pulse width measurement mode of the
/*    16-bit timer/event counter (TMP0).
/*
/*    The falling edge/rising edge of the switch input is detected and an interrupt is output,
/*    and following chattering removal, the signal is output from the P30 port and input to the
/*    TIP00 port.
/*
/*    The pulse width of the TIP00 port is measured in the pulse width measurement mode of the
/*    16-bit timer/event counter (TMP0), and LED1 is lit just for the switch depression time,
/*    with an upper limit of 5 s.
/*
/*    LED1 lights immediately after the switch is released, and switch depressions while LED1 is
/*    lit are invalid.
/*
/*    Regarding the peripheral functions in the operation stopped state following reset release,
/*    in this sample program, peripheral functions that are not used are not set.
/*
/*
/*
/*    <Main settings>
/*    • Using pragma directives to enable setting up the interrupt handler and specifying
/*      peripheral I/O register names
/*    • Defining the adjustment value of 10 ms wait for chattering
/*    • Defining the adjustment value to make the maximum acknowledgment time of switch input 5 s
/*    • Defining the count value during overflow of the 16-bit counter of timer P0 (TMP0)
/*    • Declaring prototypes
/*    • Setting up a bus wait for on-chip peripheral I/O registers, stopping watchdog timer 2,
/*      and setting up the clock
/*    • Initializing unused ports
/*    • Initializing external interrupt port (INTP0)
/*    • Initializing P30 port, TIP00 port, and PCM3 port
/*    • Initializing timer P0 (TMP0)
/*
/*    <Timer P0 settings>
```

```
/*     • Setting fxx/32 (625 kHz) as the count clock to TP0CTL0
/*     • Setting the pulse width measurement mode as the operation mode to TP0CTL1
/*     • Setting detection of both edges as the capture trigger input signal to TP0IOC1
/*     • Setting the priority of the INTTP0CC0 interrupt to level 7 and unmasking it
/*     • Masking interrupt INTTP0OV
/*
/*     <Main processing>
/*     • Counting the number of times the 16-bit counter of TPM0 has overflowed and clearing the
/*       overflow flag
/*
/*     <External interrupt (INTP0) interrupt servicing>
/*     • Chattering removal of switch input
/*     • Outputting the signal after chattering removal from the P30 port and inputting it to the
/*       TIP00 port
/*
/*     <Timer P0 (INTTP0CC0) interrupt servicing>
/*     • Detecting the edge of the capture trigger and lighting LED1
/*
/*[Port I/O settings]
/*
/* Input: P03, P32(TIP00)
/* Output: P30, PCM3
/* Unused ports: P02, P04 to P06, P10 to P11, P31, P33 to P39, P50 to P55, P70 to P711,
/*               P90 to P915,PCM0 to PCM2, PCT0, PCT1, PCT4, PCT6, PDH0 to PDH5, PDL0 to PDL15
/*               *Preset all unused ports as output ports (low-level output).
/*
/*---------------------------------------------------------------------------*/

/*---------------------------*/
/*     pragma directives     */
/*---------------------------*/
  #pragma ioreg                           /* Enables the specification of peripheral I/O
                                             registers.                                  */
#pragma interrupt INTP0     f_int_intp0       /* External interrupt (INTP0) interrupt handler
                                             specification                               */
#pragma interrupt INTTP0CC0 f_int_inttp0cc0   /* Specifies the timer P0 interrupt (INTTP0CC0) handler.
                                                                                         */
```

When using TMQ, specify INTTQ0CC0.

```
/*---------------------------*/
/*   Constant definitions    */
/*---------------------------*/
#define LIMIT_10ms_WAIT     ( 0x6F9B )                /* Definition of constant for 10 ms wait
                                                         adjustment                          */
#define LIMIT_5sec_VALUE    ( 20 * 1000 * 1000 * 5 )  /* Definition of 5 s count clock constant
                                                                                             */
#define OVER_FLOW_VALUE     ( 65536 )                 /* 16-bit counter overflow value       */
#define TMP0_COUNT_CLK_DIV  ( 32 )                    /* Count clock division ratio of 16-bit
```

```
                                                         counter                              */
#define PORT_HIGH            ( 1 )                    /* Port status (HIGH)                   */
#define PORT_LOW             ( 0 )                    /* Port status (LOW)                    */

/*---------------------------------*/
/*   Define the global variables   */
/*---------------------------------*/
static unsigned long ulOVFCnt;                /* Overflow counter */



/*---------------------------*/
/*   Prototype definitions   */
/*---------------------------*/
static  void f_init( void );                  /* Initialization function                          */
static  void f_init_clk_bus_wdt2( void );     /* Clock bus/WDT2 initialization function           */
static  void f_init_port_func( void );        /* Port/alternate-function initialization function  */
static  void f_init_int_tmp0( void );         /* TMP0 initialization function                     */

/*****************************/
/*      Main module          */
/*****************************/
void main( void )
{
    f_init();                    /* Executes initialization.                                  */

     __EI();                     /* Enables interrupts.                                       */

    while( 1 )                   /* Main loop (infinite loop)                                 */
    {
        __DI();                  /* Interrupt prohibited                                      */
                                  To use TMQ, check TQ0OVF graph
        if ( TP0OVF == 1 )       /* 16-bit counter of TMP0 has overflowed                     */
        {
            TP0OVF = 0;          /* Clear overflow flag of TMP0                               */
            ulOVFCnt++;          /* Increment the overflow count of the 16-bit counter        */
        }                To use TMQ, clear the TQ0OVF flag
        __EI();                  /* Interrupt enable                                          */
    }
}
```

```
/*---------------------------------*/
/*      Initialization module      */
/*---------------------------------*/
static void f_init( void )
{
    f_init_clk_bus_wdt2();          /* Specifies a bus wait for on-chip peripheral I/O registers,
                                       stops WDT2, and sets up the clock. */

    f_init_port_func();             /* Sets up ports and alternate functions.                */

    f_init_int_tmp();               /* Sets up the TMP0 timer.                               */

    return;
}

/*------------------------------------*/
/* Initializing clock, bus wait, WDT2 */
/*------------------------------------*/
static void f_init_clk_bus_wdt2( void )
{
    VSWC  = 0x01;                    /* Sets a bus wait for on-chip peripheral I/O registers. */

                                     /* Specifies normal operation mode for OCDM. */
#pragma asm
    st.b    r0, PRCMD
    st.b    r0, OCDM
#pragma endasm
```

Caution is required because access to special registers must be specified in assembly language.

```
    RSTOP  = 1;                     /* Stops the internal oscillator. */
    WDTM2  = 0x00;                  /* Stops watchdog timer 2. */

                                    /* Specifies that the clock not be divided */
#pragma asm
    push    r10
    mov     0x80, r10
    st.b    r10, PRCMD
    st.b    r10, PCC
    pop    r10
#pragma endasm
```

Caution is required because access to special registers must be specified in assembly language.

```
    PLLCTL = 0x03;                  /* Sets PLL mode. */

    return;
}
```

```
/*-------------------------------------------*/
/* Setting up ports and alternate functions */
/*-------------------------------------------*/
static void f_init_port_func( void )
{
    P0    = 0x00;                        /* Sets P02 to P06 to output a low level signal. */
    PM0   = 0x83;
    PMC0  = 0x00;

    P1    = 0x00;                        /* Sets P10 and P11 to output a low level signal. */
    PM1   = 0xFC;

/* Set P32 to be used as TOP00 input */
      P3    = 0x0001;                    /* Sets P30 to HIGH, P31 to P39 to a low level.   */
      PM3   = 0xFC04;                    /* Sets P30, P31, and P33 to P39 to output mode.  */
      PFC3  = 0x0000;                    /* Sets P32 to be used as TIP00 input.            */
      PMC3  = 0x04;
      PMC3  = 0x0004;

#if(0) /* Because P4 is used via CSIB0 when using MINICUBE2,          */
       /* P4 is not initialized as an unused pin (QB-V850ESJG3L-TB) */
    P4    = 0x00;                        /* Sets P40 to P42 to output a low level signal.  */
    PM4   = 0xF8;
    PMC4  = 0x00;
#endif

    P5    = 0x00;                        /* Sets P50 to P55 to output a low level signal.  */
    PM5   = 0xC0;
    PMC5  = 0x00;

    P7H   = 0x00;                        /* Sets P70 to P711 to output a low level signal. */
    P7L   = 0x00;
    PM7H  = 0xF0;
    PM7L  = 0x00;

    P9    = 0x0000;                      /* Sets P90 to P915 to output a low level signal. */
    PM9   = 0x0000;
    PMC9  = 0x0000;

    PCM   = 0x08;                        /* Sets PCM0 to PCM2 to output a low level signal */
    PMCM  = 0xF0;                        /* and specifies the turn-off pattern for PCM3.   */
    PMCCM = 0x00;

    PCT   = 0x00;                        /* Sets PCT0, PCT1, PCT4, and PCT6 to output a   */
    PMCT  = 0xAC;                        /* low level signal.                              */
    PMCCT = 0x00;
```

PM1 = 0xFC: For the V850ES/JF3-L, the setting is 0xFE.

P10 and P11: For the V850ES/JF3-L, only P10 is set.

P31 to P39 (P3): For the V850ES/JF3-L, P31 to P35, P38, and P39 are set.

P33 to P39 (PM3): For the V850ES/JF3-L, P33 to P35, P38, and P39 are set.

PFC3 = 0x0000: For the V850ES/JF3-L, the setting is 0xFCC4.

P7H, PM7H: For the V850ES/JF3-L, these are not set up because the registers do not exist.

P70 to P711: For the V850ES/JF3-L, P70 to P77 are set.

PM9 = 0x0000: For the V850ES/JF3-L, the setting is 0x1C3C.

P90 to P915: For the V850ES/JF3-L, P90, P91, P96 to P99, and P913 to P915 are set.

```
    PDH   = 0x00;                         /* Sets PDH0 to PDH5 to output a low level signal. */
    PMDH  = 0xC0;
    PMCDH = 0x00;
```

For the V850ES/JF3-L, the setting is 0xFC.

For the V850ES/JF3-L, PDH0 and PDH1 are set.

```
    PDL   = 0x0000;                       /* Sets PDL0 to PDL15 to output a low level signal. */
    PMDL  = 0x0000;
    PMCDL = 0x0000;

    /* Specifies both edges of INTP0 */
    INTF0 = 0x08;                   /* Specifies the falling edge of INTP0                  */
    INTR0 = 0x08;                   /* Specifies the rising edge of INTP0                   */

    PIC0  = 0x07;                   /* Sets the priority of INTP0 to level 7 and unmasks it */

    return;
}

/*---------------------------------------------------------------------------*/
/*     Timer P0 (TMP0) settings (specification of pulse width measurement)  */
/*---------------------------------------------------------------------------*/
static void f_init_int_tmp0( void )
{
/* Initialization of variables */
ulOVFCnt = 0;                        /* Clears overflow counter                               */

    /* Timer P0 function settings */
    TP0CTL0 = 0x05;                  /* Sets count clock to fxx/32                            */
                                     /* Specifies FailSafe (TP0CE = 0) to stop TMP0           */
    TP0CTL1 = 0x06;                  /* Specifies the pulse width measurement mode            */
    TP0IOC1 = 0x03;                  /* Detects both edges of the capture trigger input signal */
```

When using TMQ, specify values for the TQ0xxx registers.

```
/* Caution 1: The following registers are not set up in the pulse width measurement mode. */
/*                         <Registers that are not set up> */
/*                         - TP0IOC0 register              */
/*                         - TP0IOC2 register              */
/*                         - TP0OPT0 register              */
/*                         - TP0CCR0 register              */
/*                         - TP0CCR1 register              */
/*                         - TP0CNT register             */
```

```
    /* Interrupt control register settings */
    TP0CCIC0 = 0x07            /* Sets the priority of INTTP0CC0 to level 7 and unmasks it     */
    TP0OVMK  = 1;              /* Masks the INTTP0OV overflow interrupt of the 16-bit counter */
    TP0CE    = 1;              /* Starts TMP0                                                   */

    return;
}
```

When using TMQ, specify values for the TQ0xxx registers.

```
/*******************************************/
/*  Interrupt module (external interrupt)  */
/*******************************************/
__interrupt
void f_int_intp0( void )
{
    unsigned long loop_wait;
    unsigned char start_sw;
    unsigned char end_sw;

    /* Acquires the switch input value before the 10 ms wait */
    start_sw = ( unsigned char ) P0.3;

    /* 10 ms wait for chattering countermeasure */
    for( loop_wait = 0 ; loop_wait < LIMIT_10ms_WAIT ; loop_wait++ )
    {
        __nop();
    }

    /* Acquires the switch input value after the 10 ms wait */
    end_sw = ( unsigned char ) P0.3;

    if ( start_sw == end_sw )              /* The switch state remained unchanged before and
                                              after the 10 ms wait                          */
    {
        P3L.0 = start_sw;                  /* Updates the P30 port output value             */
    }

    return;
```

```
}


/***********************************/
/*  Interrupt module (TMP0)        */
/***********************************/
__interrupt
void f_int_inttp0cc0( void )
{

    unsigned long ulTMP0CLKCnt;
    unsigned long ulCPUCLKCnt;
    unsigned long loop_wait;
```

When using TMQ, check the TQ0OVF flag.

```
    if ( P3L.0 == PORT_HIGH )
    {                                 /* If switch changes from ON to OFF (P30 port is HIGH)   */
        if ( TP0OVF == 1 )            /* 16-bit counter of TMP0 has overflowed                 */
        {
            TP0OVF = 0;               /* Clears the overflow flag of TMP0                      */
            ulOVFCnt++;               /* Increments the overflow count of the 16-bit counter   */
        }
```

When using TMQ, acquire the count value of the TQ0CCR0 register

```
        ulTMP0CLKCnt = OVER_FLOW_VALUE * ulOVFCnt + TP0CCR0; /* Acquires the timer count value
                                                                 of TMP0                       */

        TP0CE = 0;                         /* Stops TMP0                                       */
```

When using TMQ, stop the timer with the TQ0CE flag

```
    ulCPUCLKCnt = ulTMP0CLKCnt * TMP0_COUNT_CLK_DIV; /* Converts the timer count value of TMP0 to
                                                        the CPU clock count                    */

        if ( ulCPUCLKCnt >= LIMIT_5sec_VALUE )  /* If the switch input time exceeds 5 s         */
        {
            ulCPUCLKCnt = LIMIT_5sec_VALUE;   /* Sets the switch input time to 5 s              */
        }

        ulCPUCLKCnt /= 6;                  /* Converts the CPU clock value to loop count
                                              (1 loop = 6 clocks)                              */
```

```
    PCM.3 = PORT_LOW;                     /* LED1 is on                                    */

    /* Wait corresponding to switch input time */
    for ( loop_wait = 0 ; loop_wait < ulCPUCLKCnt ; loop_wait++ )
    {
        __nop();
    }

    PCM.3 = PORT_HIGH;                    /* LED1 is off                                   */

    TP0CE = 1;                            /* Enables TMP0                                  */
    PIF0  = 0;                            /* Ignores external interrupts (INTTP0) that occur
                                             while the LED is on                           */
}
else
{
    TP0OVF = 0;                           /* Clears the overflow flag of TMP0              */
    ulOVFCnt = 0;                         /* Initializes the overflow count of the 16-bit
                                             counter                                       */
}

return;
```

When using TMQ, start the timer with the TQ0CE flag

For further information, please contact:

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668,
Japan
Tel: 044-435-5111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554, U.S.A.
Tel: 408-588-6000
800-366-9782
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211-65030
http://www.eu.necel.com/

Hanover Office
Podbielskistrasse 166 B
30177 Hannover
Tel: 0 511 33 40 2-0

Munich Office
Werner-Eckert-Strasse 9
81829 München
Tel: 0 89 92 10 03-0

Stuttgart Office
Industriestrasse 3
70565 Stuttgart
Tel: 0 711 99 01 0-0

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908-691-133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01-3067-5800

Sucursal en España
Juan Esplandiu, 15
28007 Madrid, Spain
Tel: 091-504-2787

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 638 72 00

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02-667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven
The Netherlands
Tel: 040 265 40 10

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27 ZhiChunLu Haidian
District, Beijing 100083, P.R.China
Tel: 010-8235-1155
http://www.cn.necel.com/

Shanghai Branch
Room 2509-2510, Bank of China Tower,
200 Yincheng Road Central,
Pudong New Area, Shanghai, P.R.China P.C:200120
Tel:021-5888-5400
http://www.cn.necel.com/

Shenzhen Branch
Unit 01, 39/F, Excellence Times Square Building,
No. 4068 Yi Tian Road, Futian District, Shenzhen,
P.R.China P.C:518048
Tel:0755-8282-9800
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
Unit 1601-1613, 16/F., Tower 2, Grand Century Place,
193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: 2886-9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R. O. C.
Tel: 02-8175-9600
http://www.tw.necel.com/

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253-8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku,
Seoul, 135-080, Korea
Tel: 02-558-3737
http://www.kr.necel.com/

G0706