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.

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.

SH7145 Group

I/O Voltage Level Conversion and D/A Output

Introduction

Digital values are output from the I/O port of the SH7145, and the voltage levels output from the pins are converted for D/A conversion. In this application, the signals output from the pins are changed during NMI interrupt processing and the state is displayed on a 7-segment LED.

Target Device

SH7145F

Contents

1. Specifications

Specified 10-bit data is output to the PE pins of the port E and is D/A coverted after voltage level conversion. A switch is connected to the NMI pin of the SH7145. On detecting the falling edge of the switch input during a program execution, an NMI interrupt is generated, then specified data is output to the PE pins. The signals outut from the PE pins are subjected to voltage level conversion by hardware and the results are input to a D/A converter. The D/A converter converts the input digital signal (parallel input) into an analog current for output.

The data specified for outupt is changed repeatedly from a maximum value → an intermediate value → a minimum value whenever there is a switch input. According to these changes, the display on the 7-segment LED also changes repeatedly from "1" → "2" → "3".

The clock signal (sampling period) input to the D/A converter is supplied by the multi-function timer pulse unit (MTU) of the SH7145. To adjust the clock signal, the timer control register_0 (TCR_0) in the MTU is set to set up counting on an internal clock (Pϕ/1), and by the setting of the timer I/O control register H_0 (TIORH_0), compare-match of the general register_A (TGRA) and the timer counter_0 (TCNT_0) at a "0" is used to obtain a period of Pϕ/2.

In this sample task a bus state controller (BSC) is used for 7-segment LED display. The BSC controls the display on the 7-segment LED, which is connected as an external memory.

Figure 1 shows a configuration diagram indicating an example of connections of the voltage-level converter, D/A converter, LED display, and NMI switch. Figure 2 shows a circuit diagram for the voltage-level converter and D/A converter. Figure 3 shows a circuit diagram of the 7-segment LED.

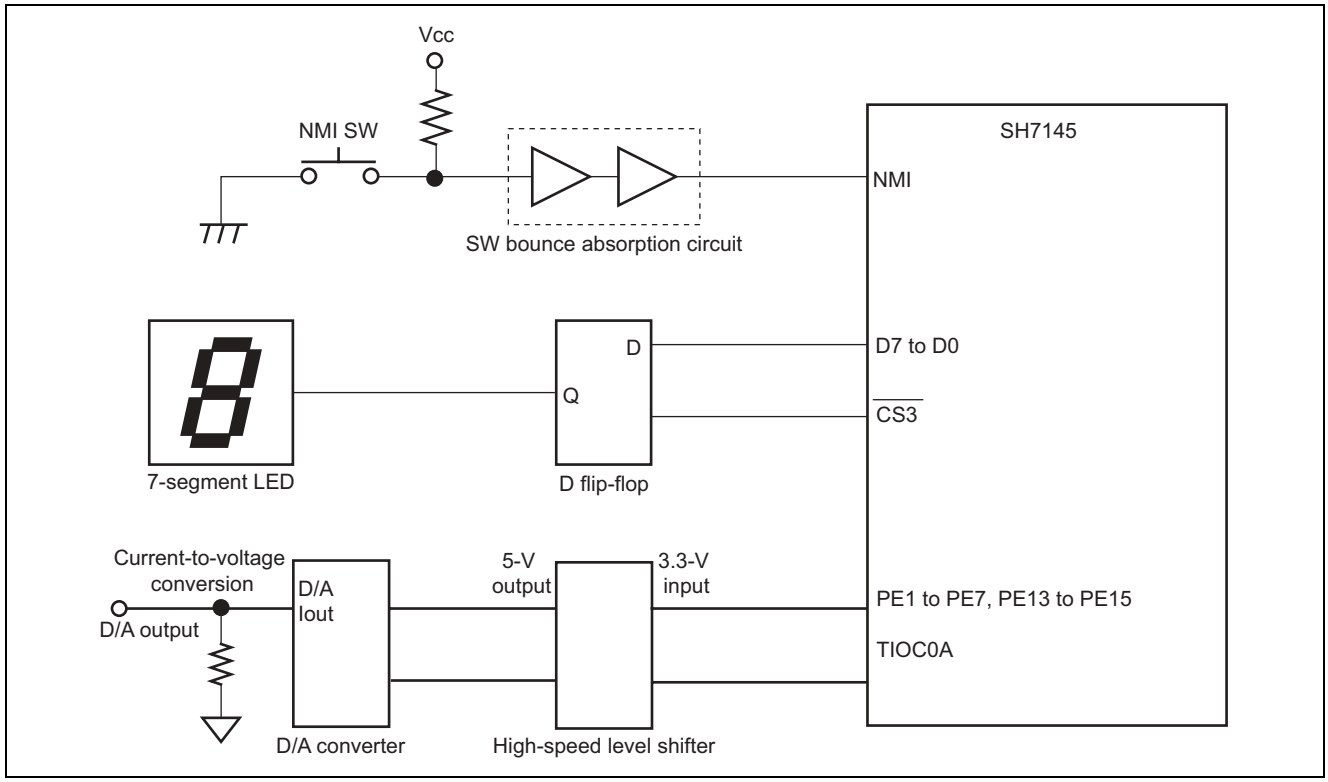


Figure 1 Configuration Diagram

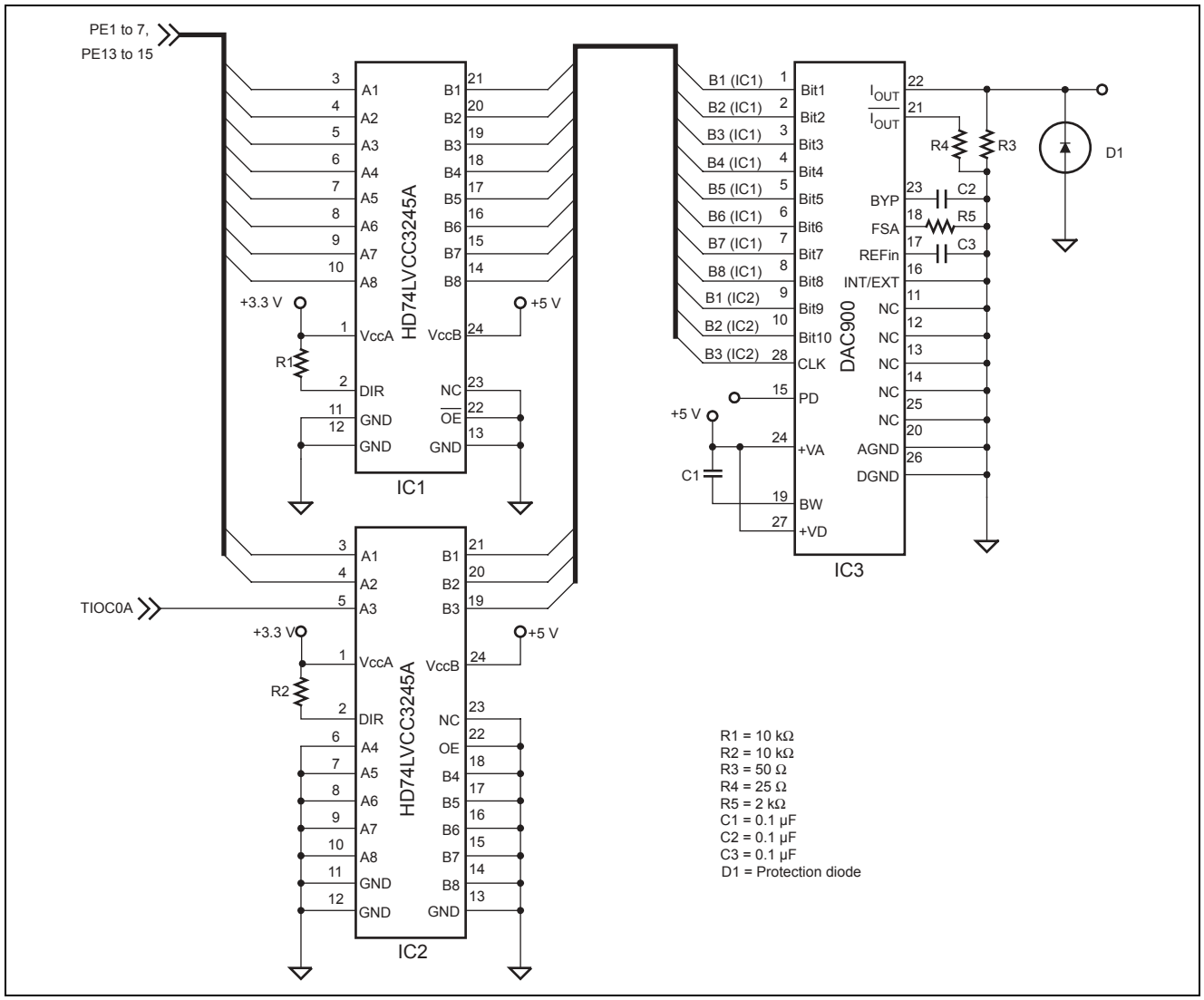


Figure 2 I/O Pin Voltage-Level Converter and D/A Converter

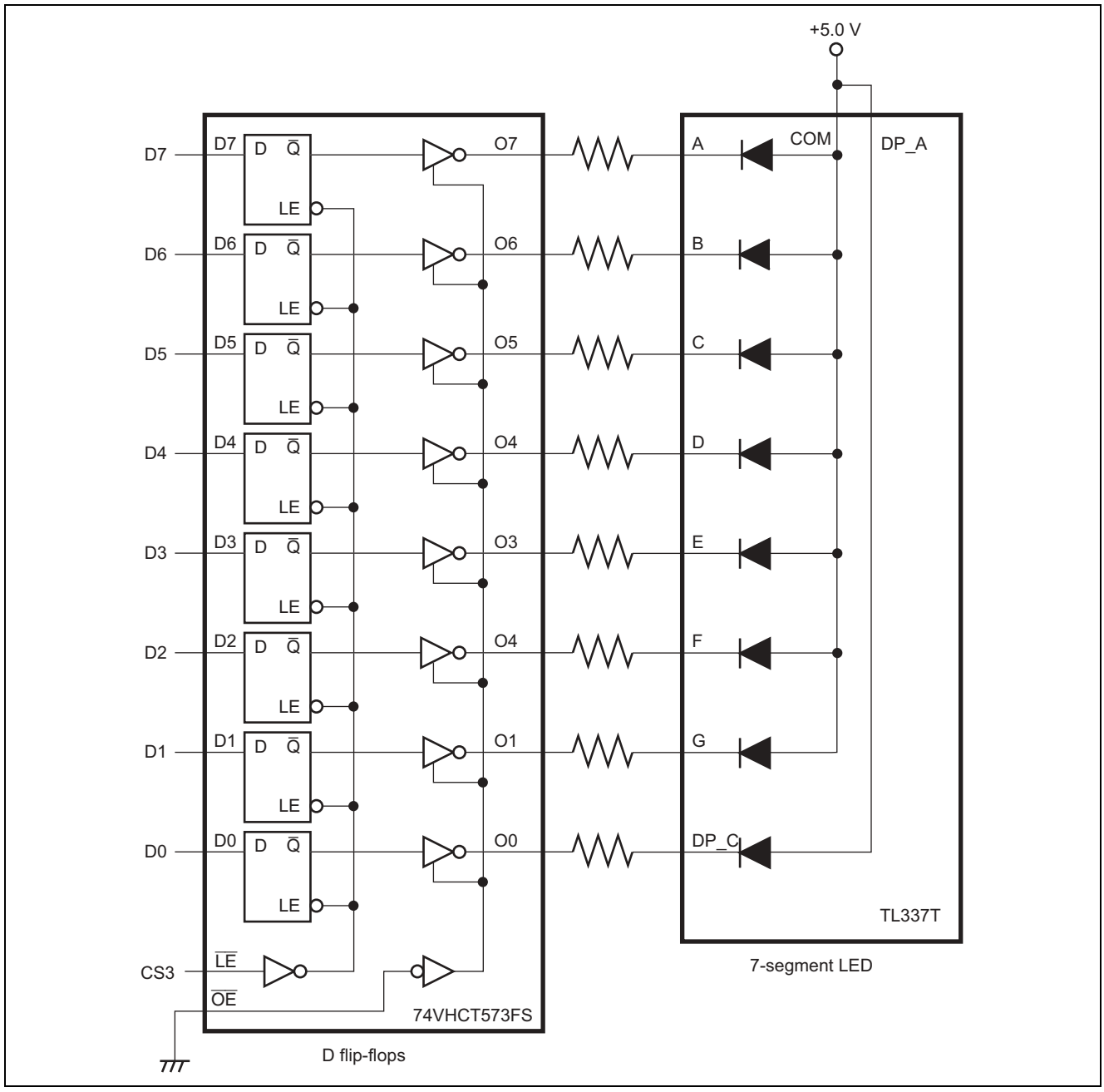


Figure 3 Equivalent Circuit of 7-Segment LED

The analog current output from the D/A converter is subjected to current-to-voltage conversion. Figure 4 shows the current-to-voltage conversion. Table 1 shows the D/A output range.

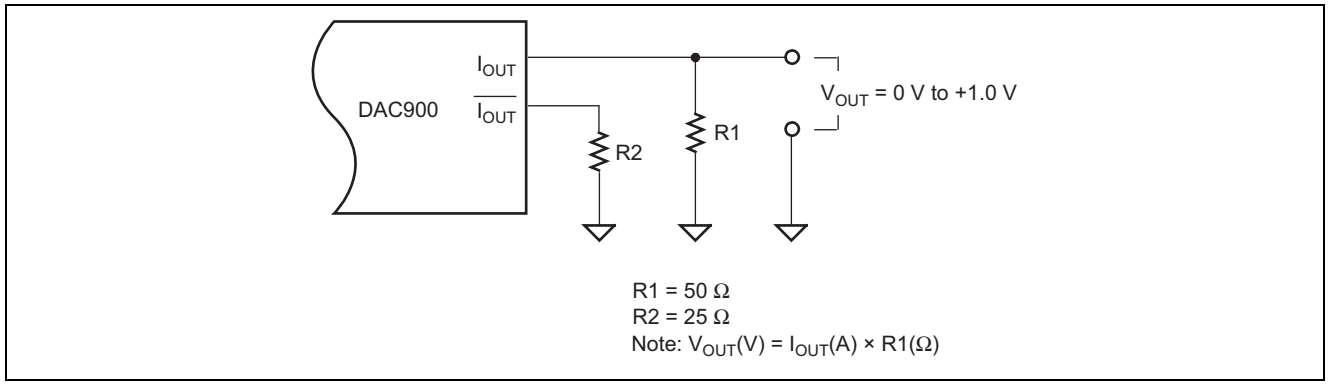


Figure 4 Current-to-Voltage Conversion

Table 1 D/A Output Range

Input Code (D9 to D0)	I_{OUT}	V_{OUT}
11 1111 1111	20 mA	1.0 V
10 0000 0000	10 mA	0.5 V
00 0000 0000	0 mA	0 V

The level shifter used in this sample task is a high-speed level shifter (the HD74LVCC3245A) manufactured by Renesas Technology Corporation. The specifications for this level shifter are summarized below.

Tables 2 and 3 show the specifications of the high-speed level shifter.

Table 2 Function Table

Input		
$\overline{OE}$	DIR	Operation
L	L	B data to A bus
L	H	A data to B bus
H	X	Z

Table 3 Recommended Operating Conditions (Excerpts)

Item	Symbol	Rating	Unit	Conditions
Power supply voltage	V_{CCA}	2.3 to 3.6	V	
	V_{CCB}	3.0 to 5.5		
Input/output voltage	V_I	0 to 5.5	V	DIR, $\overline{OE}$
	$V_{I/O}$	0 to V_{CCA}		A port output "H" or "L"
		0 to 5.5		A port output "Z" or V_{CCA}: OFF
		0 to V_{CCB}		B port output "H" or "L"
		0 to 5.5		B port output "Z" or V_{CCB}: OFF
Input fall/rise time	Δt/ΔV	10	ns/V	

Figure 5 shows a block diagram of the high-speed level shifter.

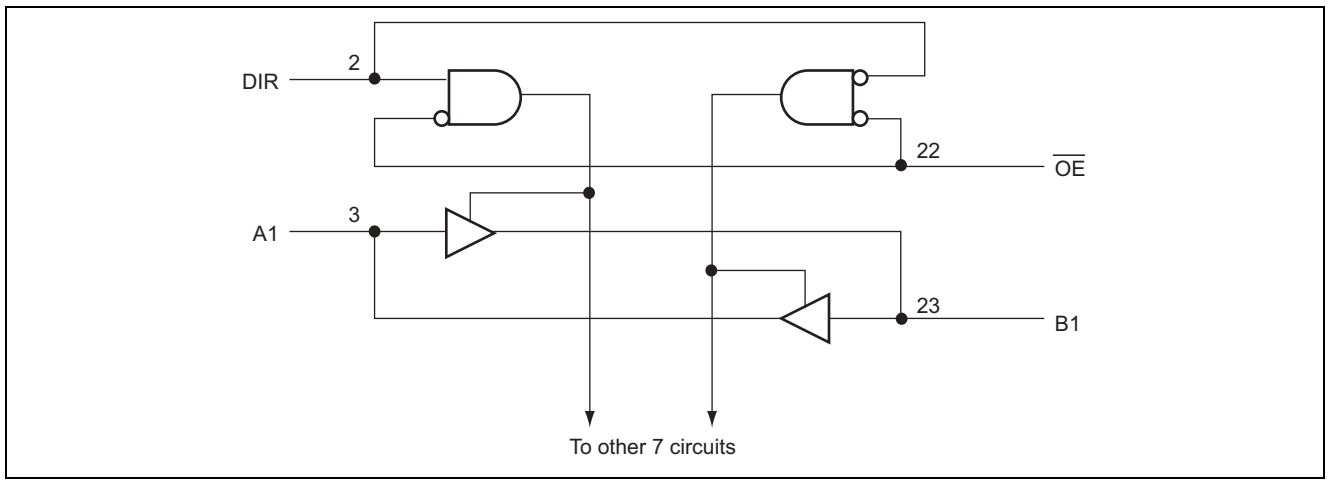


Figure 5 High-Speed Level Shifter

The D/A converter used in this task example is a high-speed 10-bit D/A converter (model: DAC900) from Texas Instruments Corporation. The specifications of this D/A converter are summarized below.

Table 4 shows the D/A converter specifications.

Table 4 Electrical Characteristics (Excerpts)

				Ratings			
	Item	**Symbol**	**Condition**	**Min.**	**Typ.**	**Max.**	**Unit**
Dynamic capability	Output rise time		10% to 90%		2		ns
	Output fall time		10% to 90%		2		ns
Voltage precision	Full-scale output range (FSR)		All bits high, I_{OUT}	2.0		20.0	mA
	Output compliance range			–1.0		+1.25	V
	Output resistance				200		kΩ
Digital input	Logic coding			Straight binary			
	Logic command			Rising edge of clock			
	Logic high voltage	V_{IH}	$+V_D$ = +5V	3.5	+5		V
	Logic low voltage	V_{IL}	$+V_D$ = +5V		0	1.2	V
	Logic high current	I_{IH}	$+V_D$ = +5V		±20		µA
	Logic low current	I_{IL}	$+V_D$ = +5V		±20		µA
Power supply	Supply voltage	$+V_A$		+2.7	+5	+5.5	V
		$+V_D$		+2.7	+5	+5.5	V

Figure 6 shows a block diagram of the high-speed D/A converter.

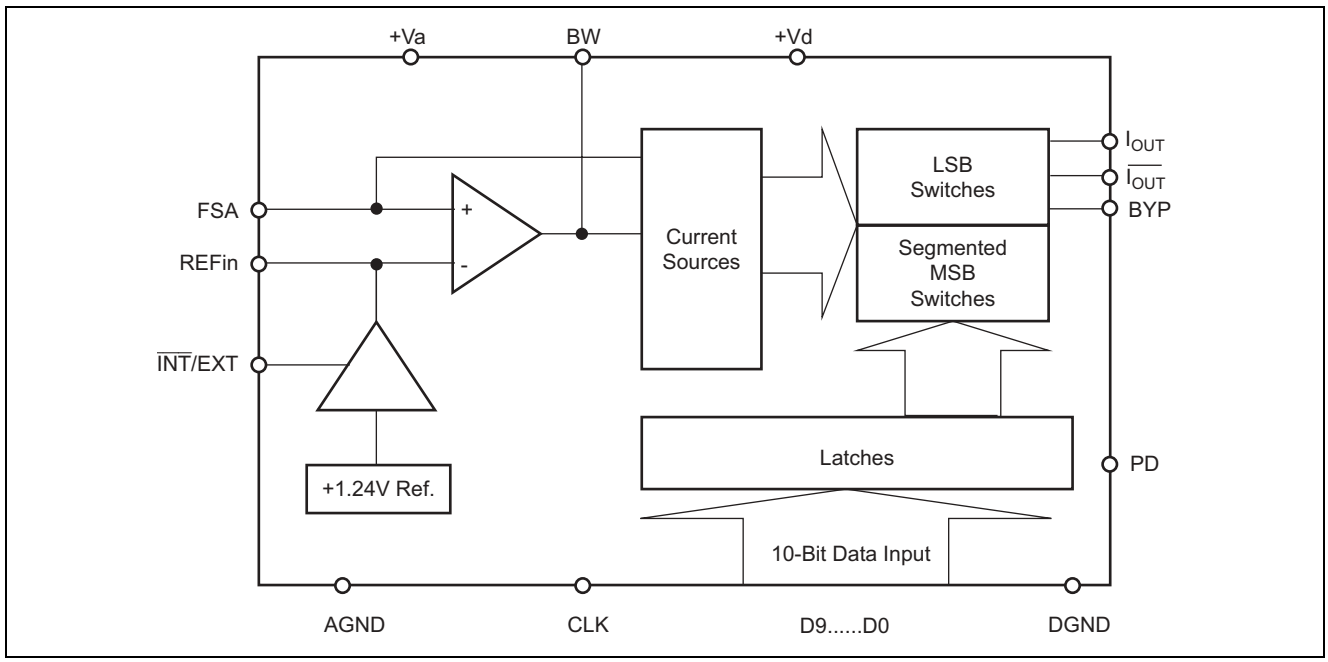


Figure 6 High-Speed D/A Converter Block Diagram

2. Description of Functions

This sample task uses an NMI interrupt to output specified 10-bit data to the PE pins of port E. The functions used in this sample task are described below.

2.1 Multi-Function Timer Pulse Unit (MTU)

The multi-function timer pulse unit (MTU) is comprised of 5 channels of 16-bit timer. This MTU is used to generate the clock signal to be fed to the D/A converter, which uses this clock for the updating of its analog signal. Figure 7 shows a block diagram of the MTU.

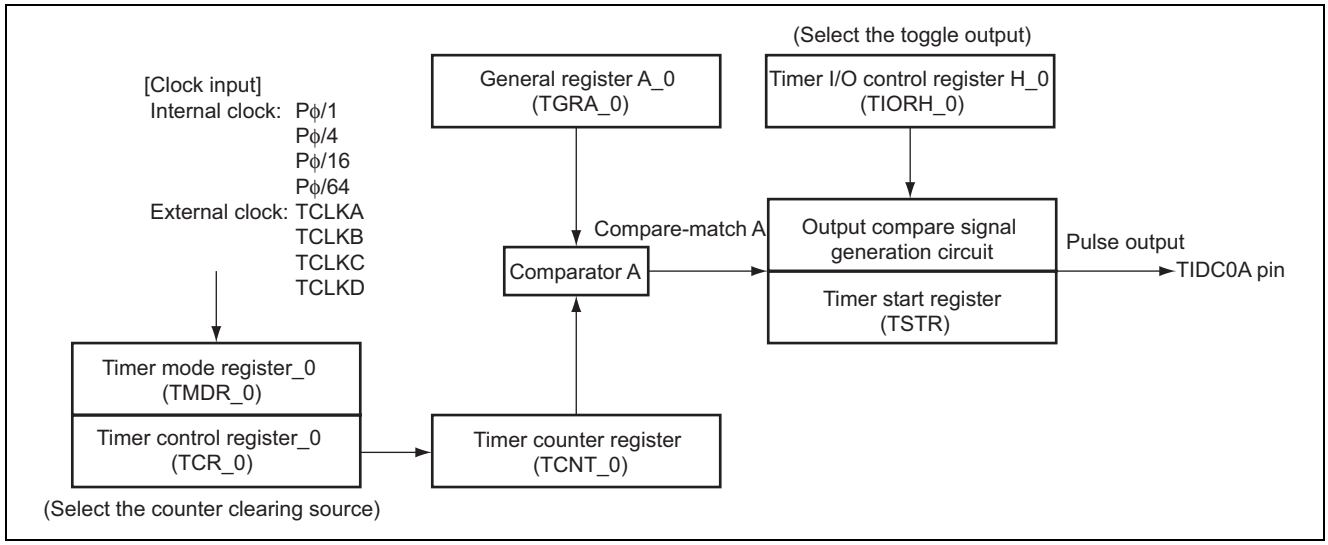


Figure 7 MTU Block Diagram

- Before using the MTU, the module standby mode of the MTU is cancelled using the module standby control register 2 (MSTCR2), which a 16-bit readable/writable register.
- The timer control register (TCR_0), which is an 8-bit readable/writable register, controls TCNT_0 of channel 0. TCR_0 is set to select a mode in which TCNT_0 is cleared on compare-match/input capture of TGRA_0. The TPSC0 to TPSC2 bits of the TCR_0 register are also set so that TCNT_0 counts cycles of the internal clock of Pϕ/1.
- The timer I/O control register (TIORH_0), which is an 8-bit readable/writable register, is set so that the output on the TIOC0A pin toggles on compare-match and its initial output is 0.
- The timer general register A (TGRA_0) is set to 0. TGRA_0 is a 16-bit readable/writable register that can be used for either the output compare or input capture function. In this sample task, TGRA_0 is used for the output compare function.
- The timer counter (TCNT_0), which is a 16-bit readable/writable counter, is set to 0.
- The timer mode register (TMDR_0) is set to select normal operation of TGRA_0. TMDR_0 is an 8-bit read/write enabled register that sets the operating mode of channel 0.
- TCNT_0 is made to operate using the timer start register (TSTR), which is an 8-bit readable/writable register that starts/stops TCNT_0 operation.

2.2 Interrupt Sources

An NMI interrupt is a level-16 interrupt and can always be accepted. The input from the NMI pin is detected by edge, and the falling edge detection is selected by setting the NMI edge select bit (NMIE) in the interrupt control register 1 (ICR1) of the interrupt controller (INTC).

The interrupt control register 1 (ICR1), which is a 16-bit register, selects the detection edge for the NMI external interrupt input pin, and indicates the input level on the NMI pin.

Table 5 shows the interrupt operation in this sample task.

Table 5 NMI Interrupt Operation

Interrupt Source	Interrupt Detection	7-Segment LED Display	Signal to D/A converter	Transition Condition
NMI interrupt	Falling edge of the NMI pin	Displays "1"	10 bits are set to all 1s.	When the NMI pin returns to the high level
NMI interrupt	Falling edge of the NMI pin	Displays "2"	Only the LSB of 10 bits is set to 1.	When the NMI pin returns to the high level
NMI interrupt	Falling edge of the NMI pin	Displays "3"	10 bits are cleared to all 0s.	When the NMI pin returns to the high level

2.3 Bus State Controller (BSC)

The bus state controller (BSC) divedes up the address space and outputs control signals according to the memory type. The use of this controller enables the direct connection of SRAM and ROM to the LSI without requiring additional circuitry. Figure 2.3 shows a BSC block diagram. In this sample task, the BSC accesses the 7-segment LED as an external memory to control the display on the LED.

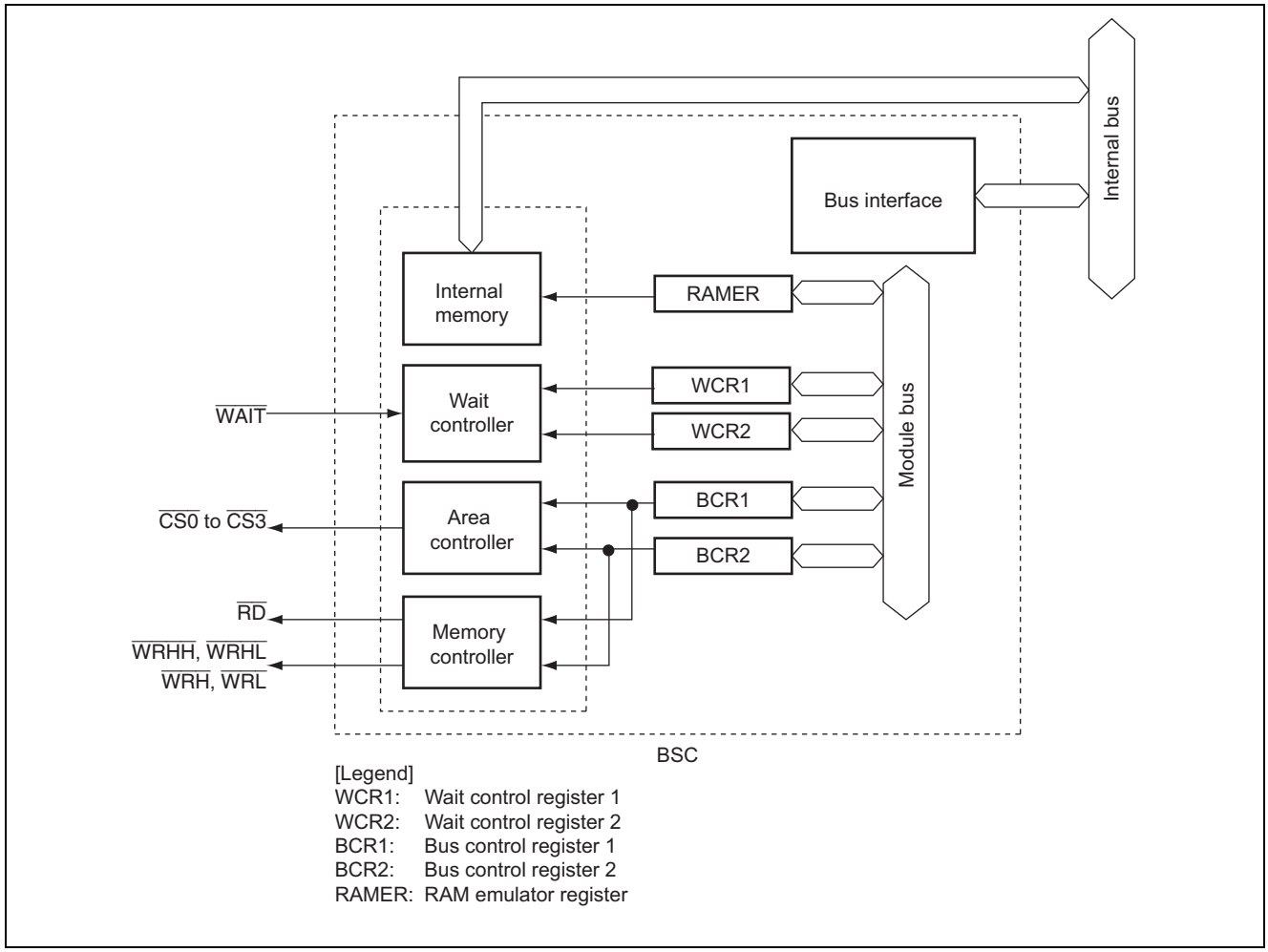


Figure 8 BSC Block Diagram

Bus cycles for external space access are performed in two states. Figure 2.4 shows the basic timing for external space access. For reading, all bits equal to the data bus width of the space (address) to be accessed are latched into the LSI with the timing of the RD signal, irrespective of the size of the operand, and required bytes are internally selected and used. For writing, the byte position at which data is actually written to is specified by the following signals: $\overline{\text{WRHH}}$ (bits 31 to 34), $\overline{\text{WRHL}}$ (bits 23 to 16), $\overline{\text{WRH}}$ (bits 15 to 08), and $\overline{\text{WRL}}$ (bits 7 to 0).

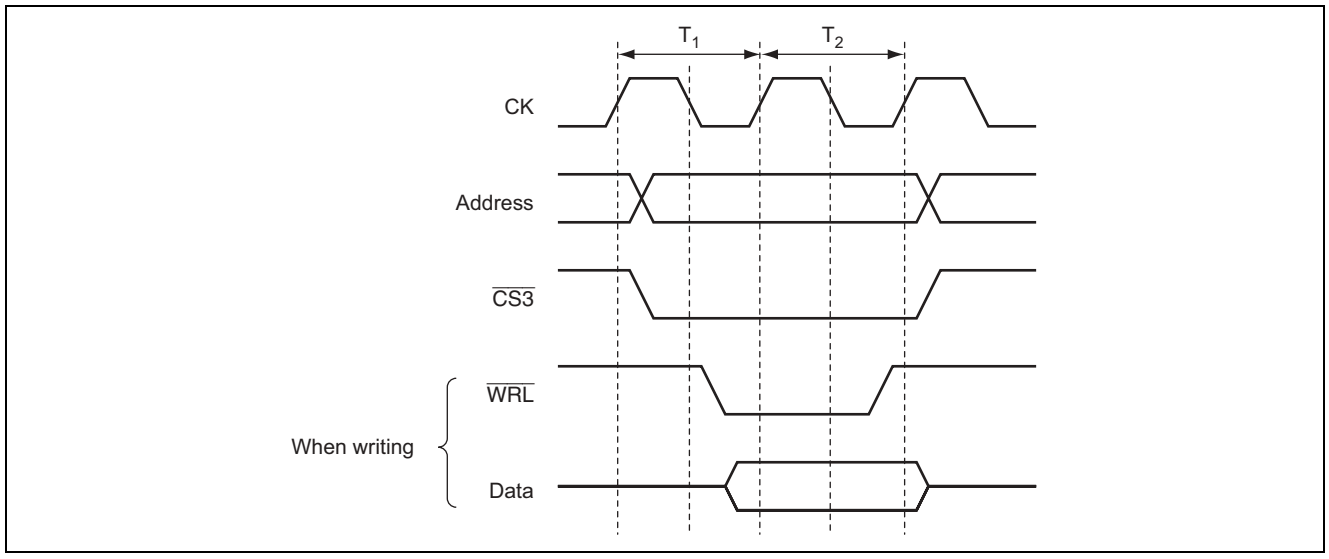


Figure 9 Basic Timing for External Space Access

Table 6 shows memory allocation for the 7-segment LED.

Table 6 Memory Allocation

Area	Address of Allocation	Device Type	Bus Size	Access Wait
CS3	From H'00C00000	7-segment LED	8 bits	None

Figure 10 identifies each segment of the 7-segment LED display. Table 7 is a segment correspondence table. Each segment is lit by negative logic.

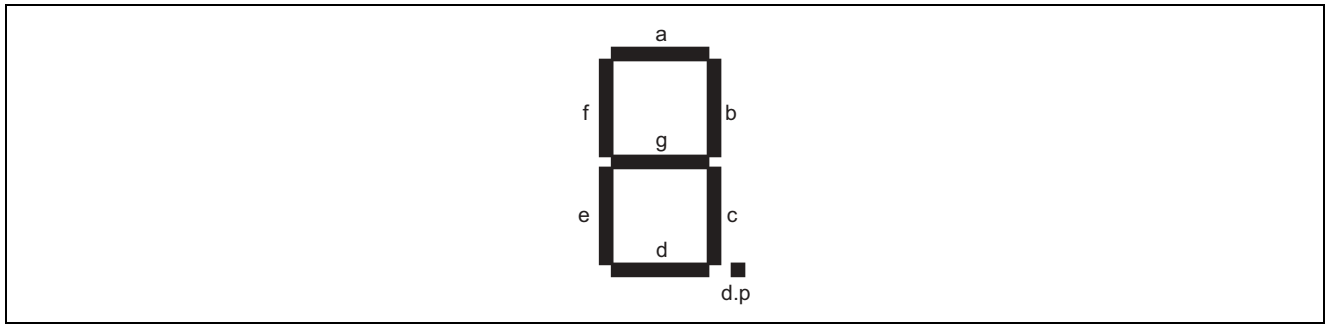


Figure 10 Correspondence of 7-Segment LED Display

Table 7 Segment Correspondence

CS3 Data Bus Name	D7	D6	D5	D4	D3	D2	D1	D0
Segment	dp	g	f	e	d	c	b	a
Character "A" display	1	0	0	0	1	0	0	0
Character "S" display	1	0	0	1	0	0	1	0
Character "." display	0	1	1	1	1	1	1	1

Note: Negative logic lighting (0 = on, 1 = off)

The bus control register 1 (BCR1), which is a 16-bit readable/writable register, is used to enable writing to the MTU control register and to specify a bus size for each CS space. Bits 7 to 0 of the BCR1 should be written to during initialization after a power-on reset, and must not be modified after that. In on-chip ROM enabled mode, the CS spaces should not be accessed until the initialization of the register is complete. In on-chip ROM disabled mode, CS spaces other than space CS0 should not be accessed until the initialization of the register is complete.

Note: In this sample task, the system operates in on-chip ROM enabled mode.

The wait control register 1 (WCR1), which is a 16-bit readable/writable register, specifies the number of wait cycles (0 to15) for each CS space.

2.4 Pin Function Controller (PFC)

The pin function controller (PFC) is comprised of registers for selecting functions of multiplexed pins and their I/O directions.

The port D control registers L1 and L2 (PDCRL1and PDCRL2), which are 16-bit readable/writable registers, select the functions of multiplexed pins of Port D. This sample task selects data bus functions (D0 to D7) to control display on the 7-segment LED.

The port E I/O register L (PEIORL) is a 16-bit readable/writable register that is used to select I/O directions of the port E pins. In this sample task, this register selects the direction as output in order to transmit digital data to the D/A converter.

The port E control registers L1 and L2 (PECRL1, L2) are 16-bit readable/writable registers. These registers are used to select the functions of the multiplexed pins in port E. In this sample task, these registers select port E functions (PE1 to PE7, PE13 to PE15) to output digital data to the D/A converter.

2.5 Function Assignment

Table 8 shows the assignment of functions in this sample task.

Table 8 Assignment of Functions

Register	Description
MSTCR2	Controls the module standby mode.
TCR_0	Controls TCNT_0.
TIORH_0	Controls TGRA_0
TGRA_0	Used for either the output compare or input capture function.
TCNT_0	Timer counter for channel 0
TMDR_0	Selects normal operation of TGRA_0.
TSTR	Controls the TCNT.
ICR1	Sets the input signal detection mode for the external interrupt input pin (NMI).
BCR1	Specifies the bus size for the CS space.
WCR1	Specifies the number of wait cycles for the CS space.
PDCRL1	Selects the functions of multiplexed pins (port D or data bus) of port D.
PEIORL	Selects the input/output directions of port E pins.
PECRL1	Selects functions of the multiplexed pins (port E or signals of on-chip peripheral functions) of port E.
PECRL2	Selects functions of the multiplexed pins (port E or signals of on-chip peripheral functions) of port E.

3. Description of Operation

Through the hardware and software processing illustrated in figure 11, I/O voltage level conversion and D/A output are implemented. Figure 12 shows the operation that generates the clock for input to the D/A converter.

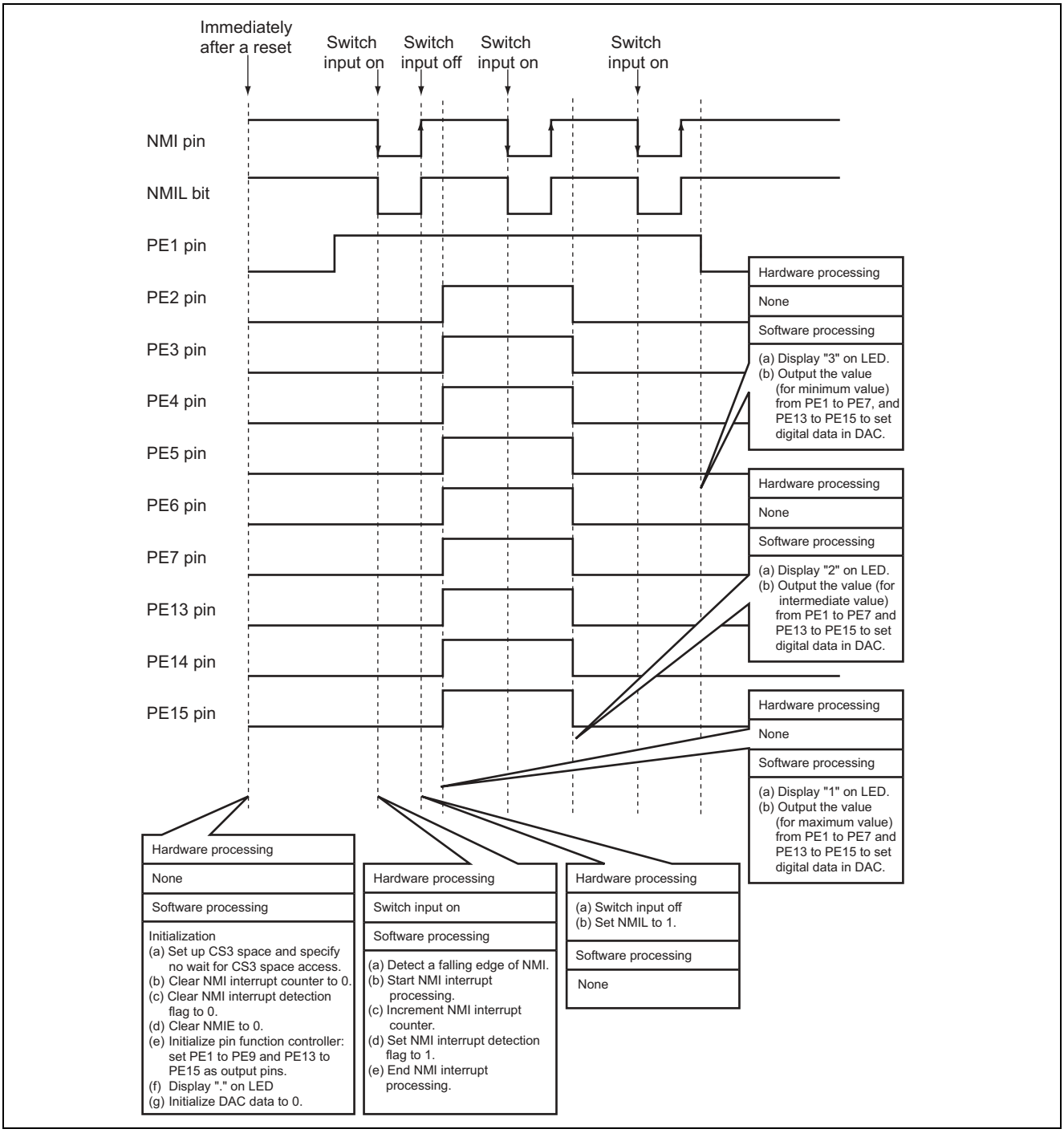


Figure 11 I/O Pin Voltage Level Conversion and D/A Output

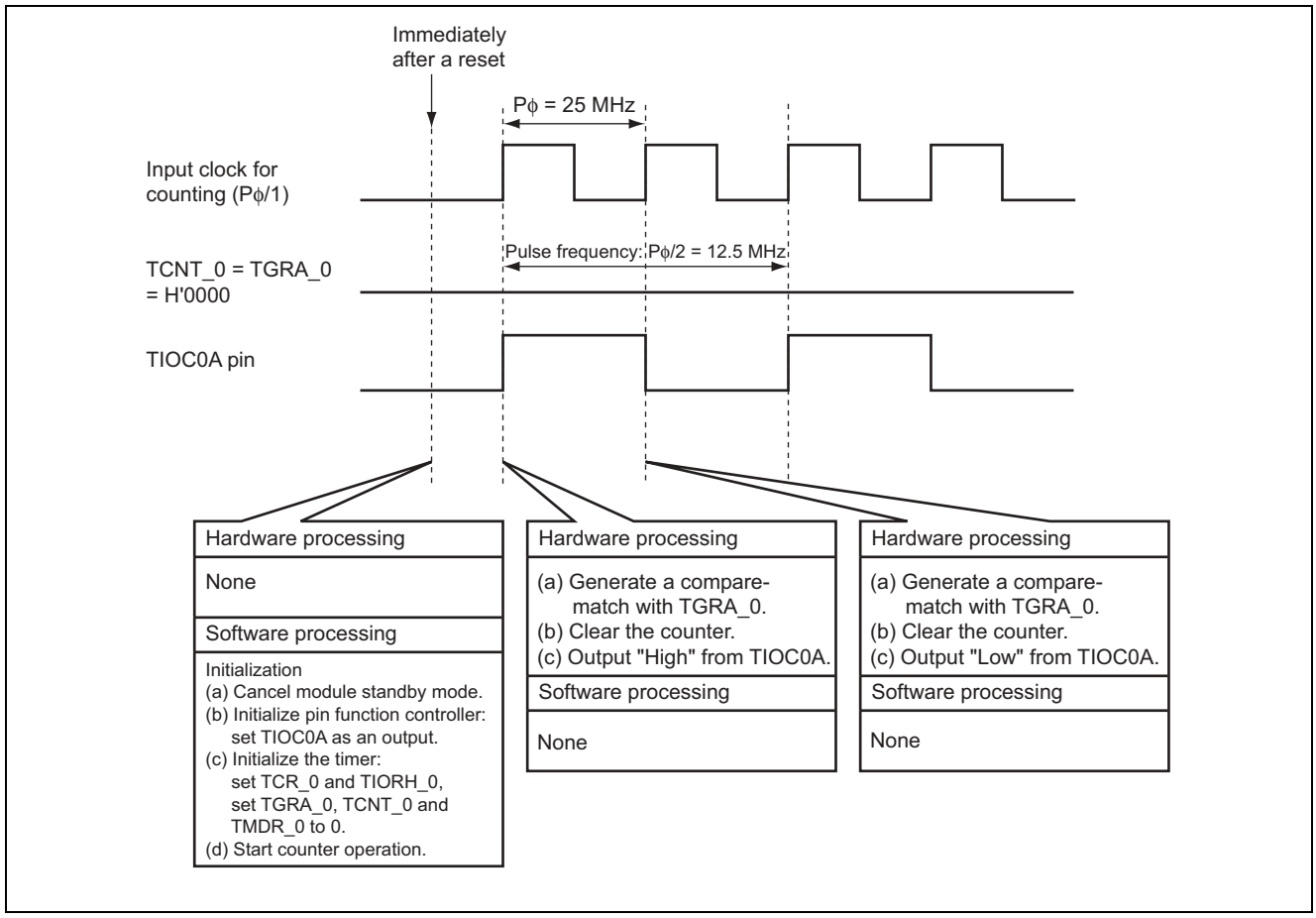


Figure 12 Generation of Clock for Input to the D/A Converter

4. Description of Software

4.1 Modules

Table 9 describes the modules used in this sample task.

Table 9 Description of Modules

Module Name	Label Name	Function
Main routine	main	Calls initialization routine and pulse(), displays on 7-segment LED to indicate the initial state, and initializes the output to the D/A converter. Calls value() upon every NMI interrupt, and iteratively sets the 10-bit data (scaled value) of maximum, intermediate, and minimum values in sequence; and at the same time, also calls led7(), to iteratively display "1", "2", then "3" on the 7-segment LED.
Initialization	init	Sets an access width and wait cycles for the BSC, makes data bus pin function settings with the PFC, initializes the MNI interrupt counter, initializes the NMI interrupt detection flag, selects the detection edge for NMI interrupt, and specifies the functions and I/O directions of port E pins with the PFC.
7-segment LED display processing	led7	Displays on the 7-segment LED.
Pulse output processing	pulse	Uses channel 0 of the MTU to output a clock from the TIOC0A pin to the D/A converter.
10-bit data output processing	value	Outputs a 10-bit data value from the PE1 to PE7 and PE13 to PE15 port pins to the D/A converter.
Wait processing	wait	Performs wait operation.
NMI interrupt processing	nmisub	Counts NMI interrupts and sets the NMI interrupt detection flag.

4.2 Arguments

Table 10 shows the arguments used in this sample task.

Table 10 Description of Arguments

Argument Name	Function	Used in	Data Size	Input/ Output
data	Index for the array dsp_data that stores 7-segment LED display data	7-segment LED display processing	1 byte	Input
dot	Flag data for outputting a dot	7-segment LED display processing	1 byte	Input
dadata	Index for the array PEDATA that stores data for output from the PE1 to PE7 and PE13 to PE15 port pins	10-bit data output processing	1 byte	Input

4.3 Internal Registers

Table 11 describes the internal registers used in this sample task.

Table 11 Description of Internal Registers

Register Name		Function	Address	Setting
ICR1	NMIL	Interrupt control register 1 (NMI input level) NMIL = 0 indicates that a low level is input to the NMI pin. NMIL = 1 indicates that a high level is input to the NMI pin.	H'FFF8358 Bit 15	—
	NMIE	Interrupt control register 1 (NMI edge select) When NMIE = 0, NMI interrupt request is detected on the falling edge of the NMI input.	H'FFF8358 Bit 8	0
BCR1	A3LG	Bus control register 1 (CS3 space long size specification) When A3LG = "0", the bus size is according to the value of the A3SZ bit of this register.	H'FFF8620 Bit 7	0
	A3SZ	Bus control register 1 (CS3 space size specification) When A3SZ = "0", byte size (8 bits) is selected.	H'FFF8620 Bit 3	0
WCR1	W33 W32 W31 W30	Wait control register 1 (CS3 and CS7 space wait specification) When W33 = "0", W32 = "0", W31 = "0", and W30 = "0", no wait cycle is inserted (external wait input disabled).	H'FFF8624 Bit 15 Bit 14 Bit 13 Bit 12	W33 = 0 W32 = 0 W31 = 0 W30 = 0
PDCRL1	PD7MD	Port D control register L1 (PD7 mode bit) When PD7MD = "1", D7 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 7	PD7MD = 1
PDCRL1	PD6MD	Port D control register L1 (PD6 mode bit) When PD6MD = "1", D6 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 6	PD6MD = 1
PDCRL1	PD5MD	Port D control register L1 (PD5 mode bit) When PD5MD = "1", D5 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 5	PD5MD = 1
PDCRL1	PD4MD	Port D control register L1 (PD4 mode bit) When PD4MD = "1", D4 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 4	PD4MD = 1
PDCRL1	PD3MD	Port D control register L1 (PD3 mode bit) When PD3MD = "1", D3 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 3	PD3MD = 1

Register Name		Function	Address	Setting
PDCRL1	PD2MD	Port D control register L1 (PD2 mode bit) When PD2MD = "1", D2 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 2	PD2MD = 1
PDCRL1	PD1MD	Port D control register L1 (PD1 mode bit) When PD1MD = "1", D1 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 1	PD1MD = 1
PDCRL1	PD0MD	Port D control register L1 (PD0 mode bit) When PD0MD = "1", D0 I/O pin function (BSC) is selected.	H'FFFF83AD Bit 0	PD0MD = 1
MSTCR2	MSTP13	Module standby control register 2 When MSTP13 = 0, cancels MTU module standby mode. When MSTP13 = 1, sets MTU module standby mode.	H'FFFF861E Bit 13	MSTP13 = 0
PEIORL	PE15IOR to PE01OR	Port E I/O register L When set to 1, the corresponding pin functions as an output pin. When cleared to 0, the corresponding pin functions as an input pin. In this sample task, TIOC0A, PE1 to PE7 and PE13 to PE15 pins are specified for output.	H'FFFF83B4 Bits 0 to 7, Bits 13 to 15	H'E0FE
PECRL1	PE15MD1 PE15MD0	Port E control register L1 (PE15 mode bits) When PE15MD1 = 0 and PE15MD0 = 0, PE15 I/O (port) pin function is selected.	H'FFFF83B8 Bits 15 and 14	PE15MD1 = 0 PE15MD0 = 0
	PE14MD1 PE14MD0	Port E control register L1 (PE14 mode bits) When PE14MD1 = 0 and PE14MD0 = 0, PE14 I/O (port) pin function is selected.	H'FFFF83B8 Bits 13 and 12	PE14MD1 = 0 PE14MD0 = 0
	PE13MD1 PE13MD0	Port E control register L1 (PE13 mode bits) When PE13MD1 = 0 and PE13MD0 = 0, PE13 I/O (port) pin function is selected.	H'FFFF83B8 Bits 11 and 10	PE13MD1 = 0 PE12MD0 = 0
PECRL2	PE7MD1 PE7MD0	Port E control register L2 (PE7 mode bits) When PE7MD1 = 0 and PE7MD0 = 0, PE7 I/O (port) pin function is selected.	H'FFFF83BA Bits 15 and 14	PE7MD1 = 0 PE7MD0 = 0
	PE6MD1 PE6MD0	Port E control register L2 (PE6 mode bits) When PE6MD1 = 0 and PE6MD0 = 0, PE6 I/O (port) pin function is selected.	H'FFFF83BA Bits 13 and 12	PE6MD1 = 0 PE6MD0 = 0
	PE5MD1 PE5MD0	Port E control register L2 (PE5 mode bits) When PE5MD1 = 0 and PE5MD0 = 0, PE5 I/O (port) pin function is selected.	H'FFFF83BA Bits 11 and 10	PE5MD1 = 0 PE5MD0 = 0

Register Name		Function	Address	Setting
PECRL2	PE4MD1 PE4MD0	Port E control register L2 (PE4 mode bits) When PE4MD1 = 0 and PE4MD0 = 0, PE4 I/O (port) pin function is selected.	H'FFFF83BA Bits 9 and 8	PE4MD1 = 0 PE4MD0 = 0
	PE3MD1 PE3MD0	Port E control register L2 (PE3 mode bits) When PE3MD1 = 0 and PE3MD0 = 0, PE3 I/O (port) pin function is selected.	H'FFFF83BA Bits 7 and 6	PE3MD1 = 0 PE3MD0 = 0
	PE2MD1 PE2MD0	Port E control register L2 (PE2 mode bits) When PE2MD1 = 0 and PE2MD0 = 0, PE2 I/O (port) pin function is selected.	H'FFFF83BA Bits 5 and 4	PE2MD1 = 0 PE2MD0 = 0
	PE1MD1 PE1MD0	Port E control register L2 (PE1 mode bits) When PE1MD1 = 0 and PE1MD0 = 0, PE1 I/O (port) pin function is selected.	H'FFFF83BA Bits 3 and 2	PE1MD1 = 0 PE1MD0 = 0
	PE0MD1 PE0MD0	Port E control register L2 (PE0 mode bits) When PE0MD1 = 0 and PE0MD0 = 1, TIOC0A I/O (MTU) pin function is selected.	H'FFFF83BA Bits 1 and 0	PE0MD1 = 0 PE0MD0 = 1
PEDRL	PE15DR to PE0DR	Port E data register L When the pins are specified for general output, data written to PEDRL is directly output to the pins. When the pins are specified for general input, the pin states are directly read by reading PEDRL.	H'FFFF83B0	H'0000 or H'0002 or H'E0FE
TCR_0	CCLR2 CCLR1 CCLR0	Timer control register_0 When CCLR2 = 0, CCLR1 = 0, and CCLR0 = 1, TCNT_0 is cleared upon compare-match/input capture of TGRA.	H'FFFF8280 Bits 5 to 7	CCLR2 = 0 CCLR1 = 0 CCLR0 = 1
	CKEG1 CKEG0	When CKEG1 = 0 and CKEG0 = 0, TCNT_0 counts on the rising edge.	H'FFFF8280 Bits 4 and 3	CKEG1 = 0 CKEG0 = 0
	TPSC2 TPSC1 TPSC0	When TPSC2 = 0, TPSC1 = 0, and TPSC0 = 0, TCNT_0 counts Pϕ/1 (internal clock) cycles.	H'FFFF8280 Bits 0 to 2	TPSC2 = 0 TPSC1 = 0 TPSC0 = 0
TIORH_0	IOA3 IOA2 IOA1 IOA0	Timer I/O control register H_0 When IOA3 = 0, IOA2 = 0, IOA1 = 1, and IOA0 = 1, sets initial output to 0 and toggles output when a compare-match occurs.	H'FFFF8262 Bits 0 to 3	IOA3 = 0 IOA2 = 0 IOA1 = 1 IOA0 = 1
TGRA_0		Timer general register A_0	H'FFFF8268	H'0000
TCNT_0		Timer counter_0	H'FFFF8266	H'0000
TMDR_0	BFB BFA MD3 MD2 MD1 MD0	Timer mode register_0 When BFB = 0, TGRB and TGRD operate in normal mode. When BFA = 0, TGRA and TGRC operate in normal mode. When MD3 = 0, MD2 = 0, MD1 = 0, and MD0 = 0, normal mode operation is selected.	H'FFFF8261 Bits 0 to 5	BFB = 0 BFA = 0 MD3 = 0 MD2 = 0 MD1 = 0 MD0 = 0
TSTR	CST0	Timer start register When CST0 = 1, TCNT_0 performs counting operation.	H'FFFF8240 Bit 0	CST0 = 1

4.4 RAM Usage

Table 12 describes the RAM usage in this sample task.

Table 12 Description of RAM

Label Name	Function	Address	Used in
nmicount	Stores the NMI interrupt count data.	H'00407000	Main routine, initialization routine, NMI interrupt processing
nmi_sw_down	Stores the NMI interrupt detection flag data.	H'00407001	Main routine, initialization routine, NMI interrupt processing

4.5 Data Tables

This sample task stores 7-segment LED display data and the 10-bit data that is output to the D/A converter (using the PE1 to PE7 and PE13 to PE15 port pins) in ROM in the form of one-dimensional array data tables. Table 13 shows the 7-segment LED display data table (dsp_data[]). Table 4.6 shows the table of data that is output to the D/A converter (PEDATA[]).

Table 13 7-Segment LED Display Data Table

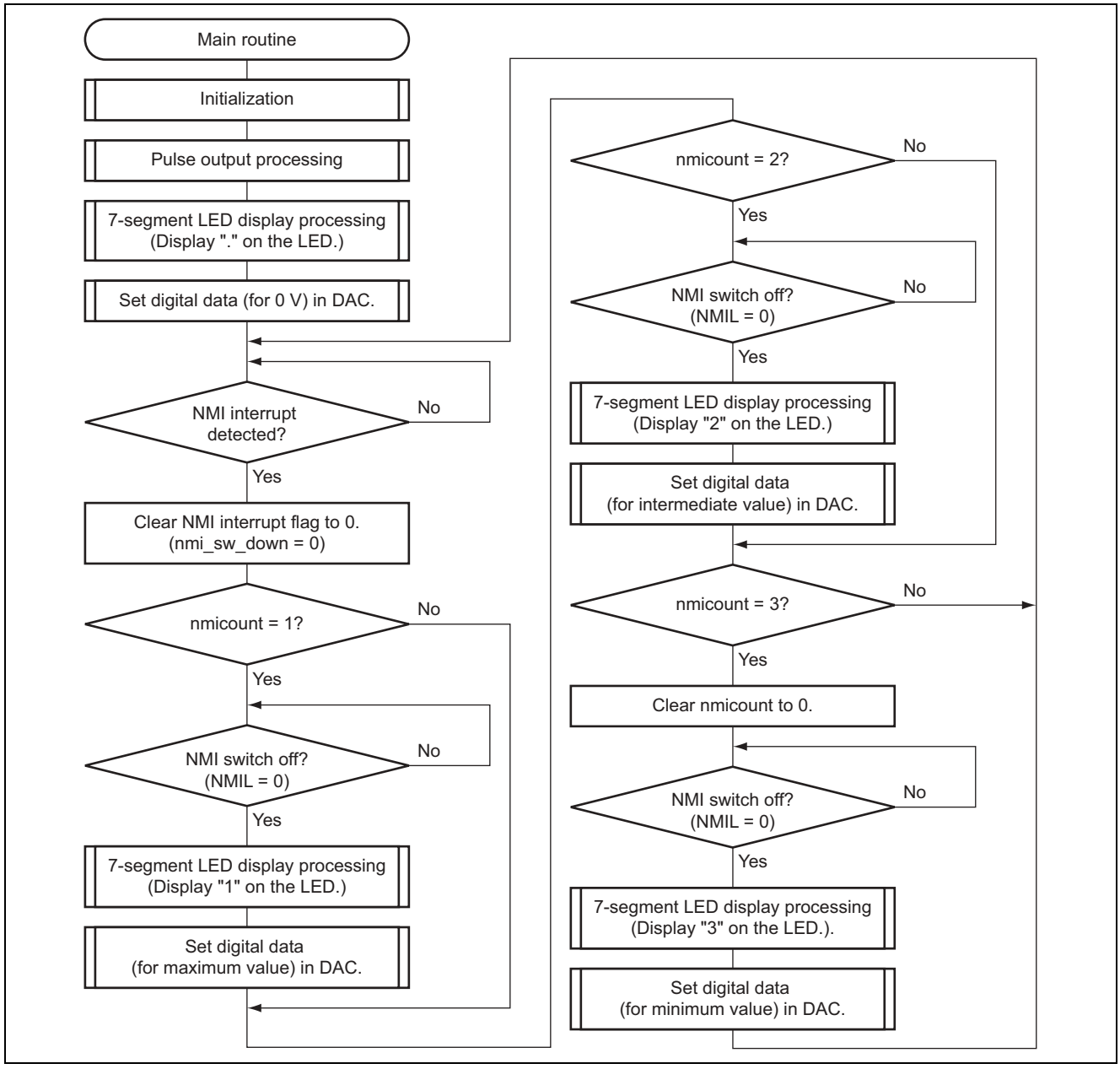
Array Name	Data	Description	Size	Address
dsp_data[0]	H'06	Data for displaying "1" on the LED	1 byte	H'22E8
dsp_data[1]	H'5B	Data for displaying "2" on the LED	1 byte	H'22E9
dsp_data[2]	H'4F	Data for displaying "3" on the LED	1 byte	H'22EA
dsp_data[3]	H'00	Data for displaying "." on the LED	1 byte	H'22EB

Table 14 Table of Data for Output to the D/A Converter

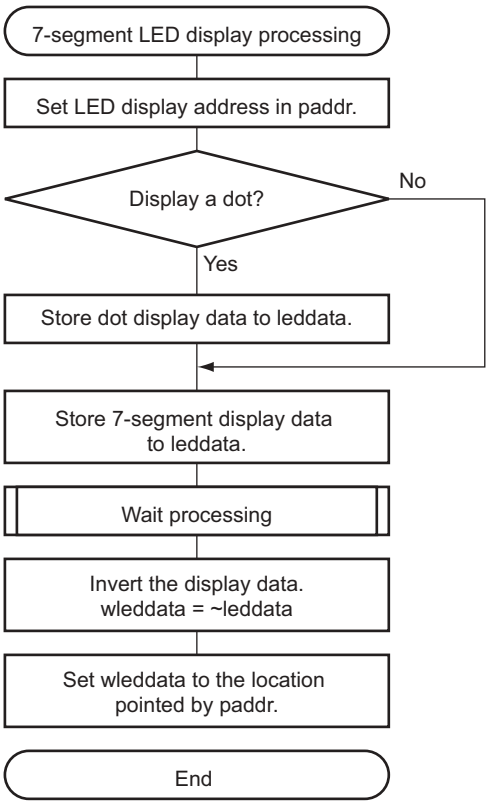
Array Name	Data	Description	Size	Address
PEDATA[0]	H'E0FE	Data for outputting the maximum value of D/A converter (PE1 to PE7 and PE13 to PE15 are all 1s)	2 bytes	H'22EC
PEDATA[1]	H'0002	Data for outputting the intermediate value of D/A converter (PE1 is 1, and PE2 to PE7 and PE13 to PE15 are 0s)	2 bytes	H'22EE
PEDATA[2]	H'0000	Data for outputting the minimum value of D/A converter (PE to PE7 and PE13 to PE15 are all 0s)	2 bytes	H'22F0

5. Flowchart

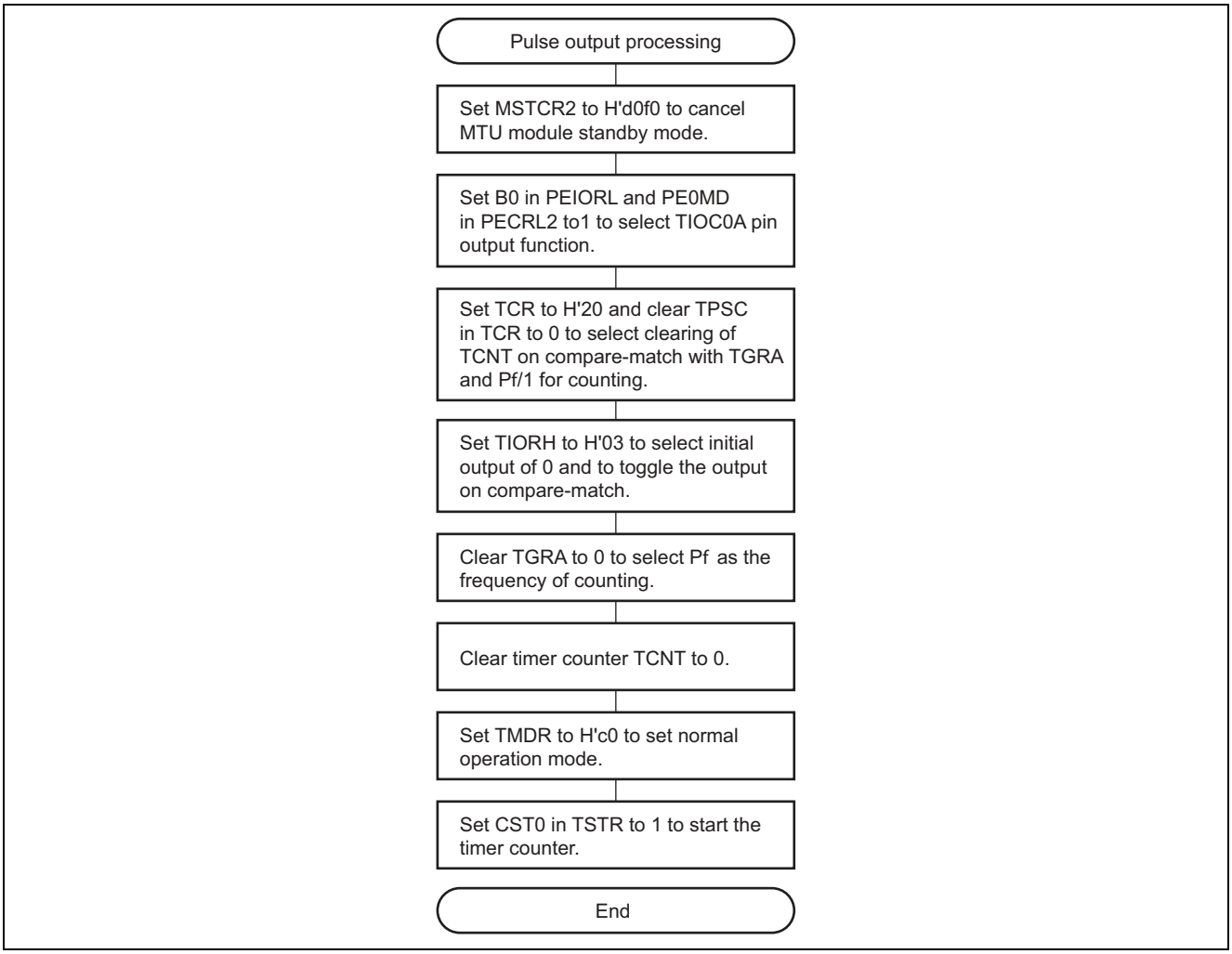
5.1 Main Routine



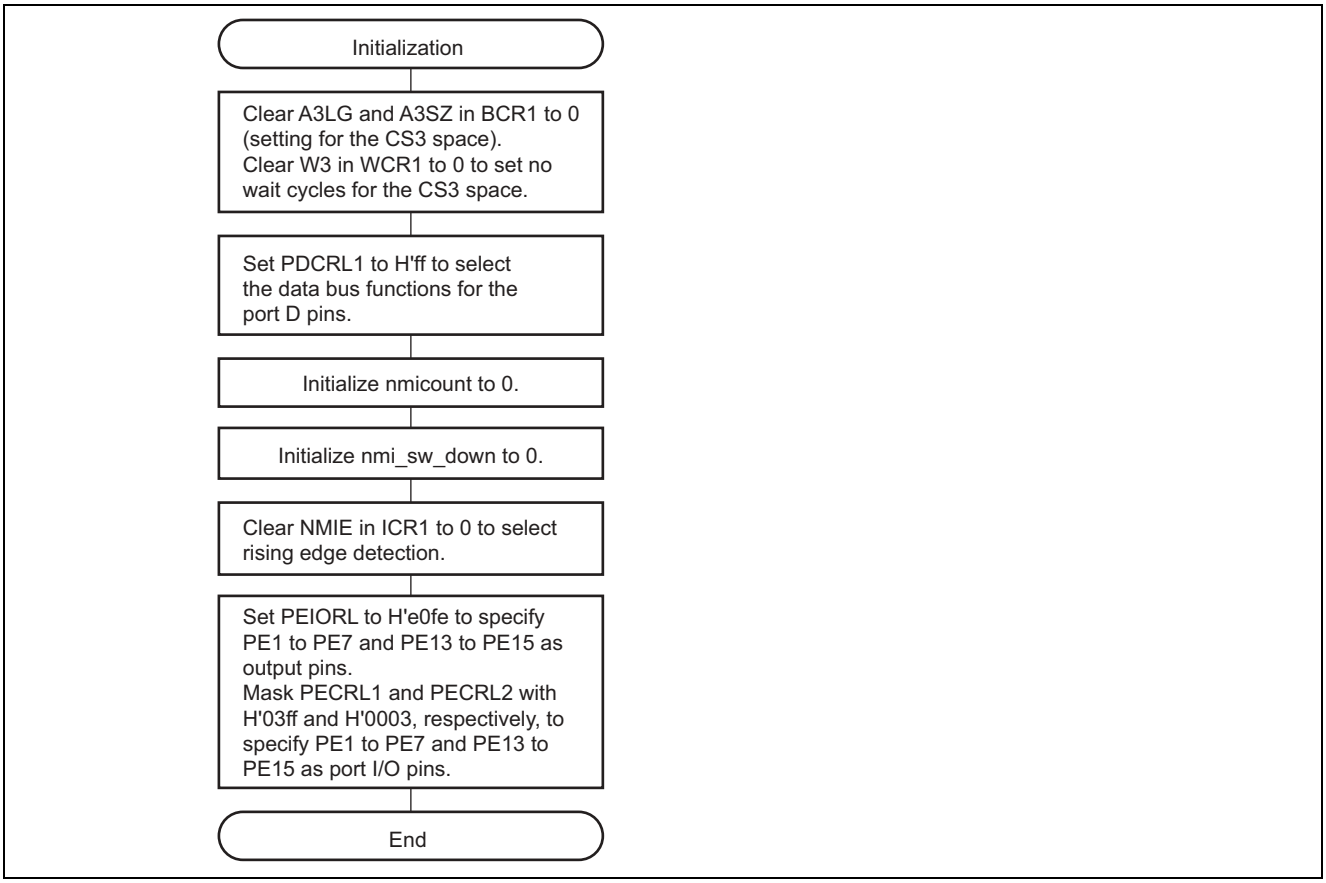
5.2 7-Segment LED Display Processing



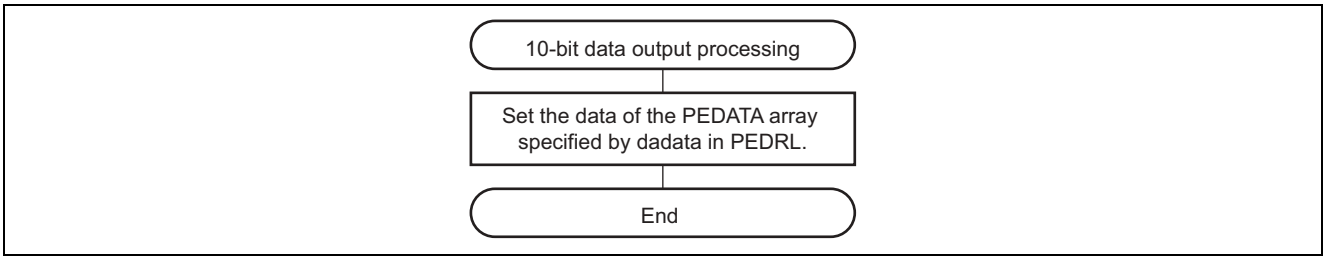
5.3 Pulse Output Processing



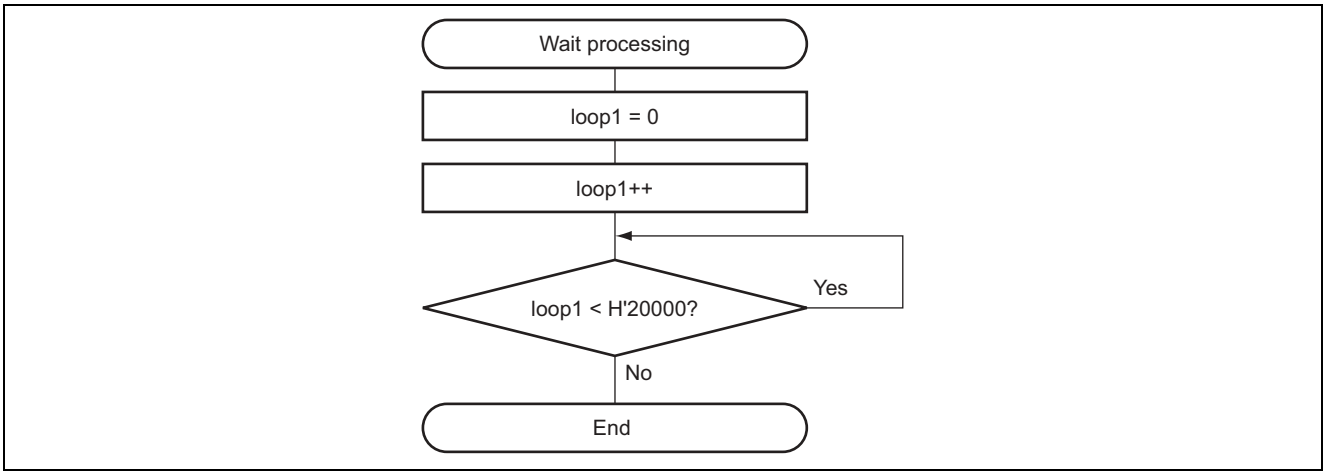
5.4 Initialization Processing



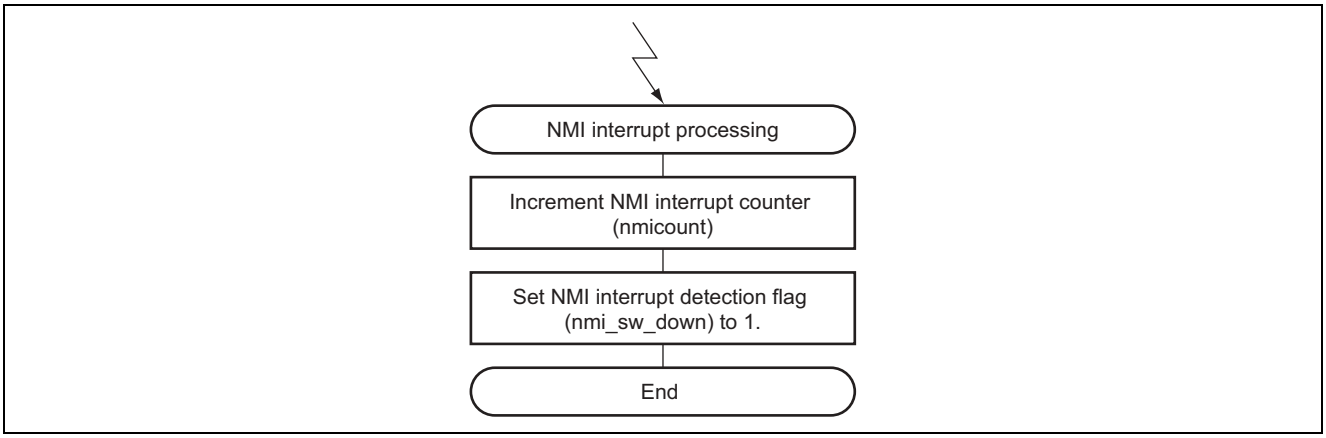
5.5 Bit-Data Output Processing



5.6 Wait Processing



5.7 NMI Interrupt Processing



6. Program Listing

INIT.C (program listing)

```
extern    _INITSCT(void) ;
extern    void main(void) ;

void _INIT()
{
    _INITSCT() ;
    main() ;
    for(;;) ;
}
```

VEC_TBL.SRC (program listing)

```
.SECTION    VECT,DATA,LOCATE=H'0000

.IMPORT __INIT
.IMPORT __STACK
.IMPORT _nmisub

.ORG    H'00000000
.DATA.L __INIT
.DATA.L __STACK
.DATA.L __INIT
.DATA.L __STACK

.ORG    H'0000002c
.DATA.L _nmisub
.END
```

IO.c (program listing)

```
/************************************************************************************************/
/* SH7145 Series -SH7145- Application note                                                      */
/* Application Note                                                                             */
/* Adjust Output Voltage And Digital-To-Analog Converter                                        */
/*                                                                                              */
/* Function : I/O Level Conversion(from 3.3V to 5.0V) From CPU                                  */
/*              And Digital-To-Analog Conversion(Scale is 0-1V)                                 */
/* Peripheral Clock       : 25MHz                                                               */
/* Internal Clock         : 50MHz                                                               */
/************************************************************************************************/

/************************************************************************************************/
/*  Include File                                                                                */
/************************************************************************************************/
#include <machine.h>
#include "IODEFINE.H"

/************************************************************************************************/
/*  I/O Definition                                                                              */
/************************************************************************************************/
#define LED    0x00C00000                          /* LED Port                                  */
```

```
/*****************************************************************************************************/
/*  Function Prototype                                                                               */
/*****************************************************************************************************/
void init(void);                                    /* Initialize Function                          */
void led7(unsigned char data,unsigned char dot);    /* 7 Segment LED Display Function               */
void wait(void);                                    /* Wait Function                                */
void pulse(void);                                   /* Output Pulse Function                        */
void value(unsigned char dadata);                   /* Digital Date Function                        */
void main(void);                                    /* Main Function                                */
#pragma interrupt(nmisub)                           /* NMI Interrupt Handler Function               */

/*****************************************************************************************************/
/*  RAM/ROM Allocation                                                                               */
/*****************************************************************************************************/

/* LED Display Data Table */
const unsigned char dsp_data[4] =
{
    0x06,                                           /* LED display data = "1"                       */
    0x5b,                                           /* LED display data = "2"                       */
    0x4f,                                           /* LED display data = "3"                       */
    0x00                                            /* LED display data = "."                       */
};

/* DAC Digital Data */
const unsigned short PEDATA[3] =
{
    0xe0fe,                                         /* D/A Output Maximum                           */
    0x0002,                                         /* D/A Output Mid                               */
    0x0000                                          /* D/A Output Minimum                           */
};

unsigned char nmicount;                             /* NMI Interrupt Counter                        */
unsigned char nmi_sw_down;                          /* NMI Switch                                   */

/*****************************************************************************************************/
/*  Main Function                                                                                    */
/*****************************************************************************************************/
void main(void)
{
    init();                                         /* Initialize Function                          */
    pulse();                                        /* Output Pulse Function                        */
    led7(3,1);                                      /* LED Display "."                              */
    value(2);                                       /* Initial Write Minimum Digital Data           */

    while(1){
        if(nmi_sw_down ==1){                        /* Check NMI Switch Down?                       */
            nmi_sw_down = 0;                        /* Clear NMI Switch                             */
            if(nmicount ==1){                       /* NMI Counter = 1?                             */
                while(INTC.ICR1.BIT.NMIL == 0x0);   /* NMI Switch Release?                          */
                    led7(0,0);                      /* LED Display "1"                              */
                    value(0);                       /* Write Maximum Digital Data                   */
            }else if(nmicount == 2){                /* NMI Counter = 2?                             */
                while(INTC.ICR1.BIT.NMIL == 0x0);   /* NMI Switch Release?                          */
```

```
                    led7(1,0);                      /* LED Display "2"                          */
                    value(1);                       /* Write Mid Digital Data                   */
                }else if(nmicount == 3){            /* NMI Counter = 3?                         */
                    nmicount = 0;                   /* Clear NMI Counter                        */
                    while(INTC.ICR1.BIT.NMIL == 0x0);  /* NMI Switch Release?                   */
                    led7(2,0);                      /* LED Display "3"                          */
                    value(2);                       /* Write Minimum Digital Data               */
                }
            }
        }
    }
/*****************************************************************************************************/
/*  Initialize Function                                                                              */
/*****************************************************************************************************/
void init(void)
{
    BSC.BCR1.BIT.A3LG = 0x0;                        /* Clear A3LG Bit for A3SZ Bit              */
    BSC.BCR1.BIT.A3SZ = 0x0;                        /* Byte Size CS3 Memory Area                */
    BSC.WCR1.BIT.W3   = 0x0;                        /* No Wait CS3 Memory Area                  */

    PFC.PDCRL1.BYTE.L = 0xff;                       /* Use From D0-D7 Input And Output(BSC) Port*/
    PFC.PDCRL2.BYTE.L = 0x0;                        /* Use From D0-D7 Input And Output(BSC) Port*/

    nmicount = 0;                                   /* Initialize NMI Interrupt Counter         */
    nmi_sw_down = 0;                                /* Initialize NMI Switch                    */
    INTC.ICR1.BIT.NMIE = 0x0;                       /* Set NMI Falling Edge Interrupt           */
    PFC.PEIORL.WORD = 0xe0fe;                       /* Output PE1 to PE7 and PE13-PE15          */
    PFC.PECRL1.WORD = PFC.PECRL1.WORD & 0x03ff;     /* Input and Output PE13-PE15               */
    PFC.PECRL2.WORD = PFC.PECRL2.WORD & 0x0003;     /* Input and Output PE1-PE7                 */
}
/*****************************************************************************************************/
/*  7 Segment LED Display                                                                            */
/*****************************************************************************************************/
void led7(unsigned char data,unsigned char dot)
{
    unsigned char leddata;                          /* LED Display Data                         */
    unsigned long wleddata;                         /* LED Display Output Value                 */
    unsigned long *paddr;                           /* LED Port Address                         */

    paddr = (unsigned long *)LED;                   /* Set Paddr As LED Port                    */

    if( dot!=0 ){                                   /* Output "." On 7 Segment LED Display?     */
        leddata=(dsp_data[data]|0x80);              /* Output "."                               */
    } else {
        leddata=(dsp_data[data]&0x7f);              /* 7 Segment LED Display Output Value       */
    }

    wait();                                         /* Wait                                     */
    wleddata =~leddata;                             /* Save 7 Segment LED Display Output Value  */
        *paddr = wleddata;                          /* LED Display                              */
}
```

```
/*****************************************************************************************************/
/*  Output Pulse Function                                                                            */
/*****************************************************************************************************/
void pulse(void)
{
    MST.CR2.WORD = 0xd0f0;                          /* MTU Module Standby Mode clear            */
    PFC.PEIORL.BIT.B0 = 1;                          /* TIOC0A Output                            */
    PFC.PECRL2.BIT.PE0MD = 1;                       /* FE0 Function Is TIOC0A                   */
    MTU0.TCR.BYTE = 0x20;                           /* Timer Counter Clear by TGRA              */
    MTU0.TCR.BIT.TPSC = 0;                          /* Count By Pphi/1                          */
    MTU0.TIOR.BYTE.H = 0x03;                        /* Initial Output 0 and toggle              */
    MTU0.TGRA = 0;                                  /* Counter Period Is Pphi                   */
    MTU0.TCNT = 0x0000;                             /* Clears Timer Counter                     */
    MTU0.TMDR.BYTE = 0xc0;                          /* Set Mode                                 */
    MTU.TSTR.BIT.CST0 = 0x1;                        /* Start Timer Counter                      */
}

/*****************************************************************************************************/
/*  Digital Date Function                                                                            */
/*****************************************************************************************************/
void value(unsigned char dadata)
{
    PE.DRL.WORD = PEDATA[dadata];                   /* Write PE Data                            */
}

/*****************************************************************************************************/
/*  Wait Function                                                                                    */
/*****************************************************************************************************/
void wait(void)
{
    unsigned long loop1;
    for(loop1=0;loop1<0x20000;loop1++){             /* wait                                     */
    }
}

/*****************************************************************************************************/
/*  NMI Interrupt Function                                                                           */
/*****************************************************************************************************/
void nmisub(void)
{
    nmicount++;                                     /* Increment NMI Interrupt Counter          */
    nmi_sw_down = 1;                                /* Set NMI Switch Down                      */
}
```

Revision Record

		Description	
Rev.	Date	Page	Summary
1.00	Sep.16.04	—	First edition issued

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.