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.

Application Note

32

SH7046 Group On-Chip I/O Volume

Application Note

Renesas 32-Bit RISC Microcomputer
SuperH™ RISC engine Family/
SH7046 Series

Renesas Electronics
www.renesas.com

Rev.1.00 2003.7

Renesas 32-Bit RISC Microcomputer
SuperH™ RISC engine Family/
SH7046 Series

SH7046 Group
On-Chip I/O Volume

Application Note

RENESAS

REJ05B0076-0100O

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation 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 Corporation 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 Corporation or a third party.
2. Renesas Technology Corporation 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 Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation 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 Corporation 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 Corporation by various means, including the Renesas Technology Corporation 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 Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation 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 Corporation or an authorized Renesas Technology Corporation 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 Corporation 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 Corporation for further details on these materials or the products contained therein.

Preface

The SH7046, SH7048, SH7047, SH7049, and SH7101 are high-performance microcomputers with a 32-bit SH-2 CPU core that uses a RISC (reduced instruction set computer) type instruction set, and comprehensive on-chip peripheral functions.

On-chip peripherals include a CPU, ROM, RAM, a 16-bit multifunction timer pulse unit (MTU), serial communication interface (SCI), port output enable (POE), data transfer controller (DTC) (except for the SH7101), and motor management timer (MMT) (except for the SH7101), enabling these microcomputers to be used for a wide range of applications covering small to large-scale systems.

This Application Note includes sample tasks that use the SH7046 Series' on-chip peripheral functions, which we hope users will find useful as reference material in carrying out software design.

Although the operation of the task programs in this Application Note has been checked, operation should be confirmed again before any of these programs are actually used.

Contents

Section 1 Using the SH7046 Series Application Note

This Application Note illustrates the operation of SH7046 Series' on-chip peripheral functions based on simple sample tasks.

1.1 Organization of On-Chip I/O Volume

In the On-Chip I/O Volume, the layout shown in figure 1.1 is employed to describe the use of the peripheral functions.

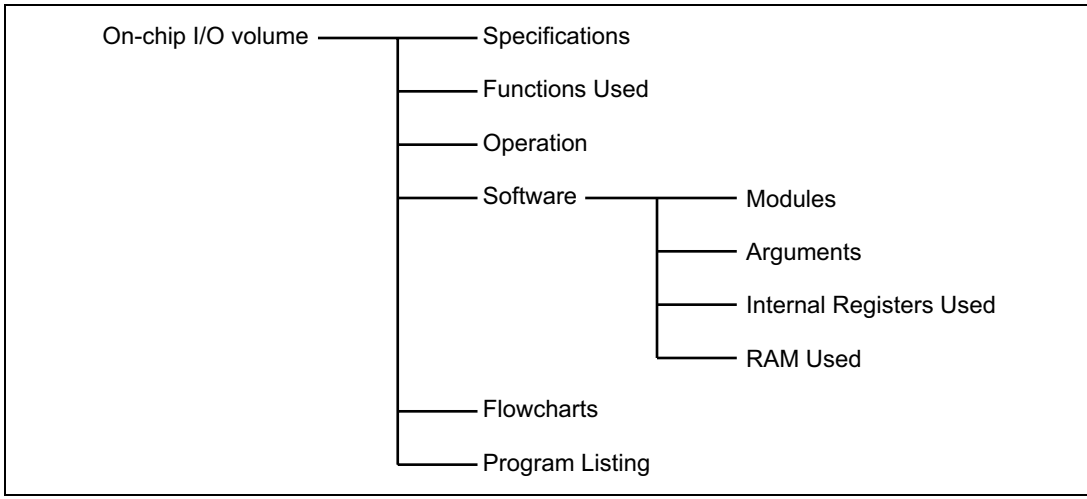


Figure 1.1 On-Chip I/O Volume

(1) Specifications

Describes the system specifications for the sample task.

(2) Functions Used

Describes the features of the peripheral function(s) used in the sample task, and peripheral function assignment.

(3) Operation

Describes the operation of the sample task, using a timing chart.

(4) Software

(a) Modules

Describes the software modules used in the operation of the sample task.

(b) Arguments

Describes the input arguments needed to execute the module, and the output arguments after execution.

(c) Internal Registers Used

Describes the peripheral function internal registers (timer control registers, serial mode registers, etc.) set by the modules.

(d) RAM Used

Describes the labels and functions of RAM used by the modules.

(5) Flowcharts

Describes the software that executes the sample task, using general flowcharts.

(6) Program Listing

Shows a program listing of the software that executes the sample task.

2.1 Pulse High and Low Width Measurement

Pulse High and Low Width Measurement	MCU: SH7046/47	Functions Used: MTU (Input Capture)

Specifications

(1) Pulse high width and low width times are measured and the results are stored in RAM as shown in figure 2.1.

(2) When operating with on-chip peripheral clock Pϕ = 20.0 MHz, the pulse high width and low width can be measured in a range of 50.0 ns to 3.27 ms in 50.0 ns units.

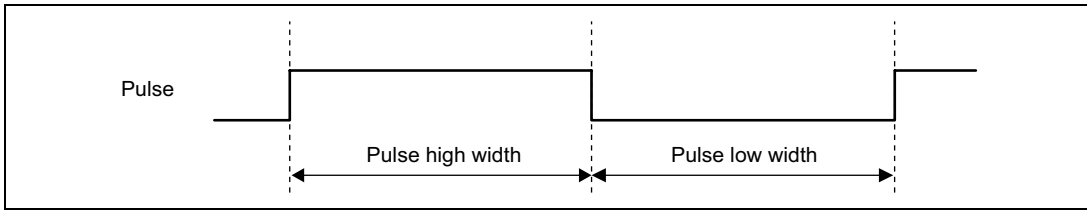


Figure 2.1 Pulse Width Measurement Timing

Functions Used

(1) In this sample task, the high width and low width of a pulse are measured using channel 0 (ch0).

(a) Figure 2.2 shows a block diagram of ch0. This task uses the following functions.

- A function that performs pulse rising edge and falling edge detection, and sets the timer value at that time in an internal register (input capture)
- A function that clears the timer counter when input capture occurs (counter clearing)
- A function that initiates interrupt handling when a pulse rising edge or falling edge is detected

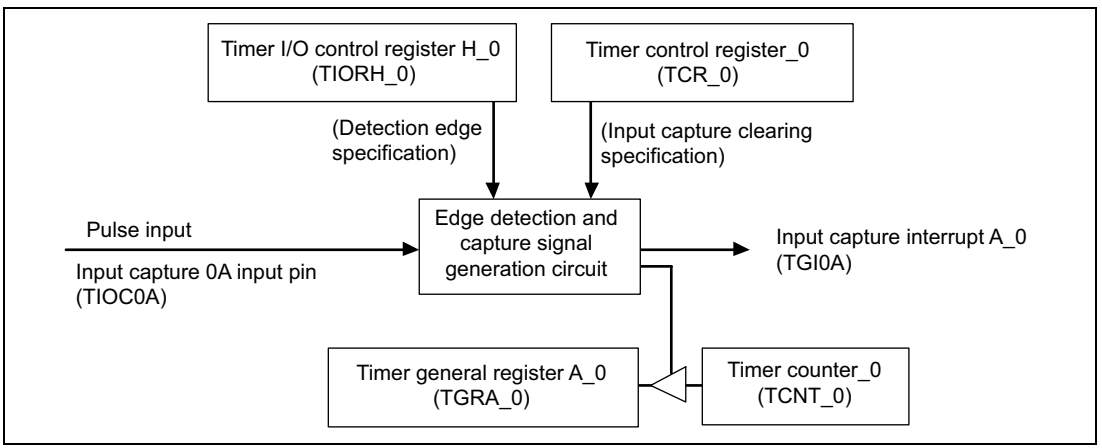


Figure 2.2 Block Diagram of MTU/ch0

(2) Table 2.1 shows the function assignments used in this sample task. The high width and low width of a pulse are measured by assigning MTU functions as shown in the table.

Table 2.1 Function Assignments

Pin or Register Name	Function	Function Assignment
TCR_0	Register	Counter clearing source selection
TIORH_0	Register	Selects input edge of input capture signal
TGRA_0	Register	Stores counter value at pulse rising edge or falling edge
TGIA_0	Register bit	Initiates pulse high and low width measurement at pulse rising edge or falling edge
TIOC0A	Pin	Inputs pulse to be measured

Operation

(1) Figure 2.3 illustrates the principles of operation of this sample task. Pulse high width and low width measurement is performed by SH7046 hardware and software processing as shown in the figure.

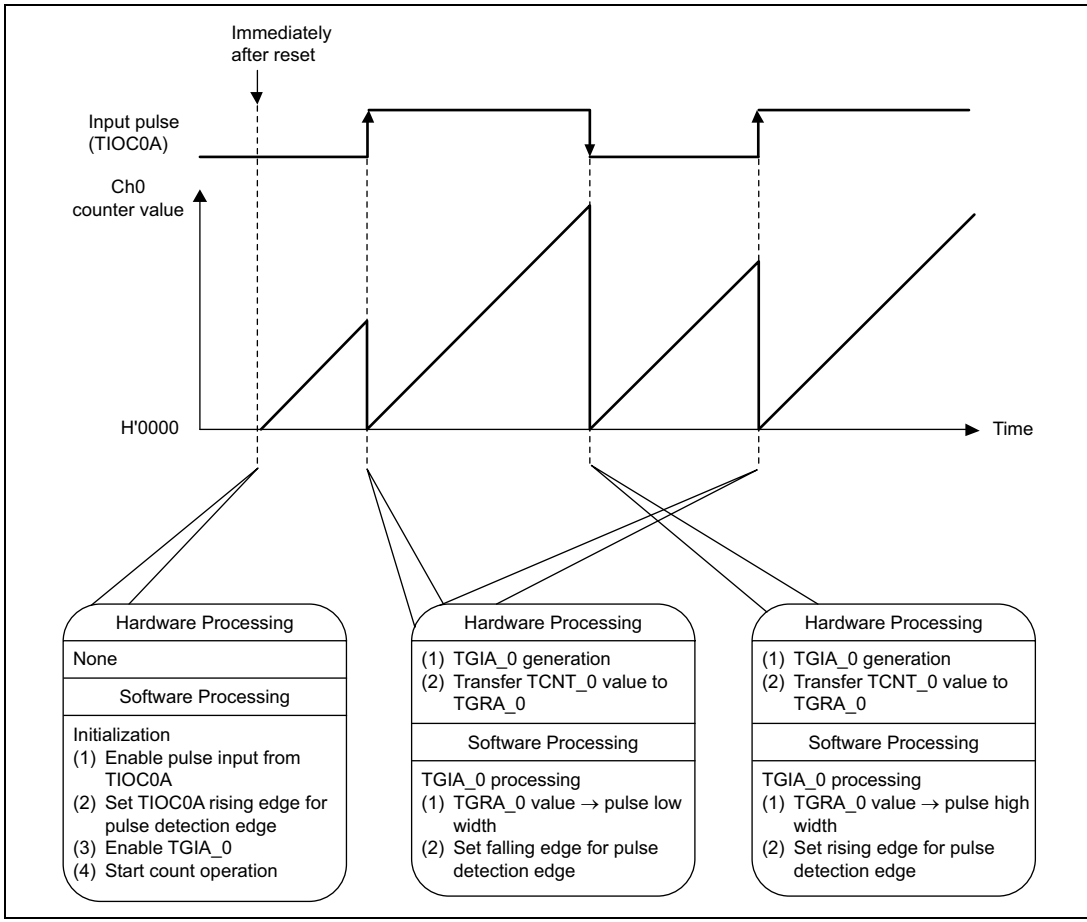


Figure 2.3 Principles of Operation of Pulse Width Measurement

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	pwhlmn	MTU initialization
Pulse high width and low width measurement	pwhl1	Initiated by TGIA_0. Measures pulse high width and low width based on TGRA_0 value, and stores results in RAM

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
pwh_hdata	Used to set timer value for pulse high width Pulse high width is calculated using following equation: Pulse high width (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz operation)	1 word	Pulse high width and low width measurement	Output
pwh_ldata	Used to set timer value for pulse low width Pulse low width is calculated using following equation: Pulse low width (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz operation)	1 word		

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_PORTE.PECRL2	Enables pulse input from TIOC0A input pin	H'FFFF83BA	H'0001
P_MTU0.TCR_0	TCNT counter clock selection, and setting of input capture A as counter clearing source	H'FFFF8260	H'20
P_MTU0.TIORH_0	Sets transfer from TCNT_0 to TGRA_0 on detection of pulse rise or fall	H'FFFF8262	H'08
P_MTU0.TIER_0	Enables interrupt by TGIA_0	H'FFFF8264	H'41
P_MTU0.TGRA_0	TCNT_0 values at time of pulse rising edge and falling edge are stored, and pulse period is calculated from these values	H'FFFF8268	pwh_ldata pwh_hdata
P_INTC.IPRD	Sets 15 as TGIA_0 interrupt priority level	H'FFFF834E	H'f000
P_STBY.MSTCR2	MTU module standby mode clearing	H'FFFF861E	H'd2fd

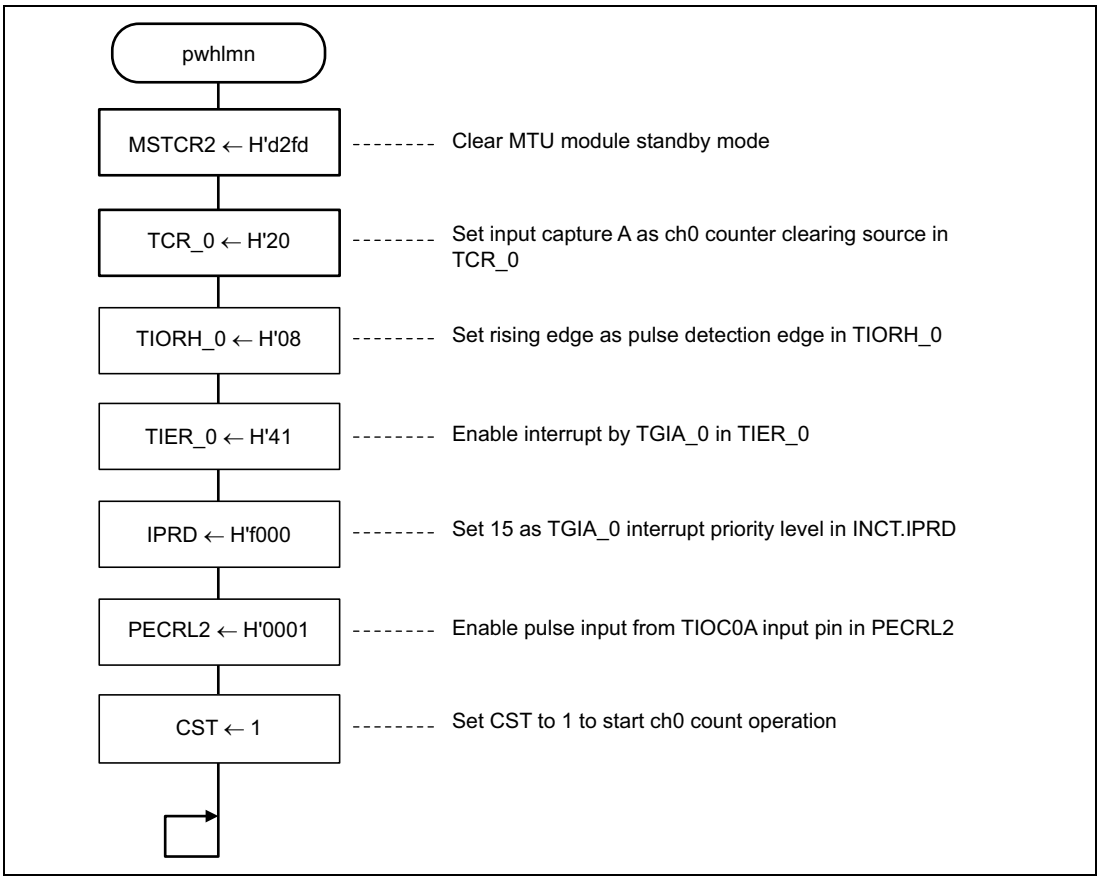
(4) RAM Used

This sample task does not use any RAM apart from the arguments.

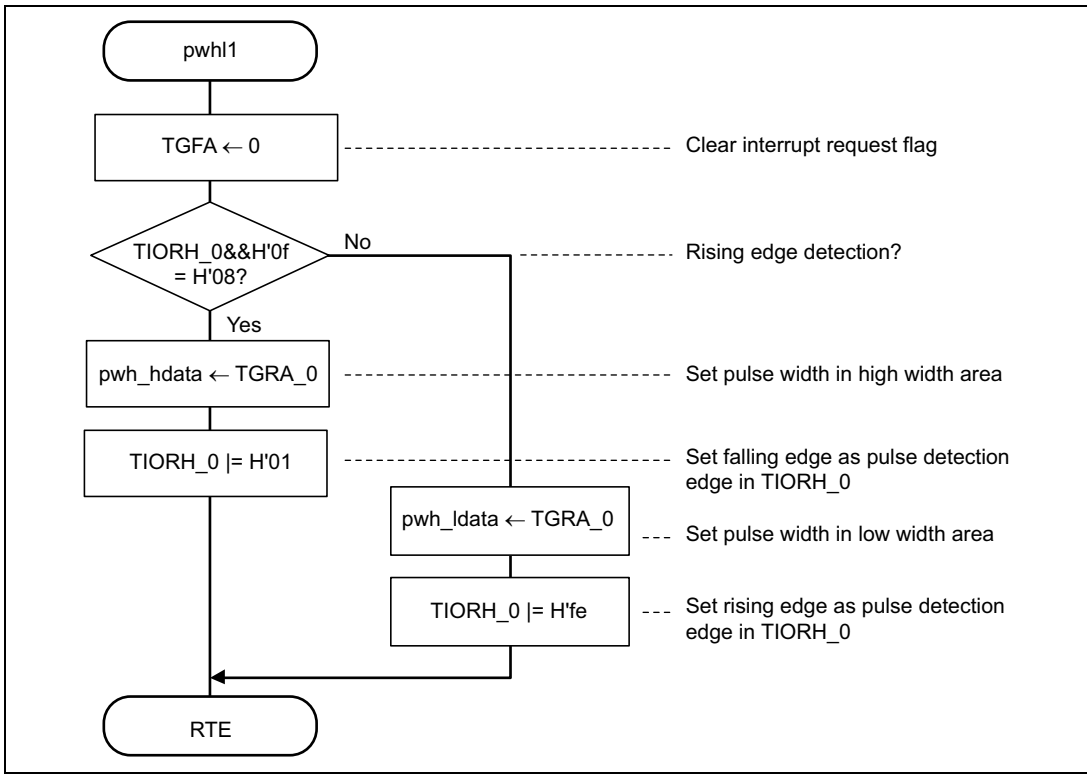
Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



(2) Pulse high and low width measurement



Program Listing

```
/******************************************************************************/
/*                                INCLUDE FILE                                */
/******************************************************************************/
#include <machine.h>
#include "iodefine_7046.h"
/******************************************************************************/
/*                                PROTOTYPE                                   */
/******************************************************************************/
void  pwhlmn(void);
#pragma interrupt(pwhl1)
/******************************************************************************/
/*                              RAM ALLOCATION                                */
/******************************************************************************/
#define pwh_hdata  (*(unsigned short  *)0xffffd000)
#define pwh_ldata  (*(unsigned short  *)0xffffd002)
/******************************************************************************/
/*                                MAIN PROGRAM                                */
/******************************************************************************/
void pwhlmn(void)
 {
      set_imask(0xf);
      P_STBY.MSTCR2.WORD = 0xd2fd;
      P_MTU0.TCR_0.BYTE = 0x20;       /* timer clear input capture with TGRA_0 */
                                      /* counter clock = ϕ/1 */
      P_MTU0.TIORH_0.BYTE = 0x08;     /* input capture by TIOC0A rising edge */
      P_MTU0.TIER_0.BYTE = 0x41;      /* enable TGIA interrupt */
      P_INTC.IPRD.WORD = 0xf000;      /* set initialize level = 15 */
      P_PORTE.PECRL2.WORD = 0x0001;
      P_MTU34.TSTR.BIT.CST = 1;       /* start TCNT_0 */
      set_imask(0x0);
      while(1);
 }

void pwhl1()
 {
     P_MTU0.TSR_0.BIT.TGFA = 0;       /* clear interrupt flag */
     if((P_MTU0.TIORH_0.BYTE & 0x0f) == 0x08)
      {
        pwh_hdata = P_MTU0.TGRA_0.BYTE;
        P_MTU0.TIORH_0 |= 0x01;
      }
     else
      {
        pwh_ldata = P_MTU0.TGRA_0.BYTE;
         P_MTU0.TIORH_0 |= 0xfe;
      }
  }
```

2.2 Pulse Output

Pulse Output	MCU: SH7046/47	Functions Used: MTU (Output Compare)

Specifications

(1) Using MTU ch0, a 50% duty pulse with a period set in RAM is output as shown in figure 2.4.

(2) When operating with on-chip peripheral clock Pϕ = 20.0 MHz, the output pulse width can be set arbitrarily in the range 100.0 ns to 3.27 ms.

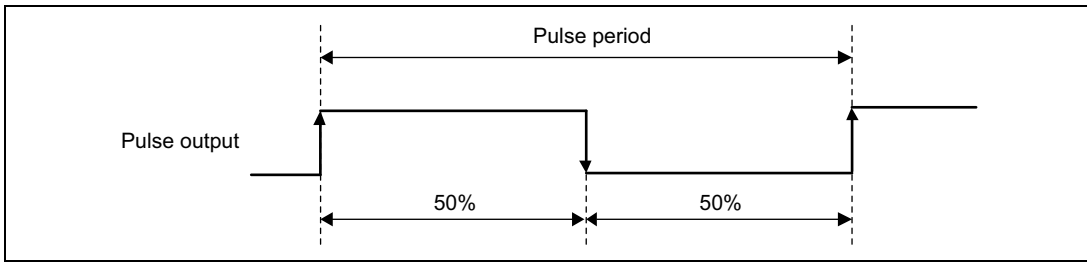


Figure 2.4 Pulse Output

Functions Used

(1) In this sample task, a pulse with a 50% duty is output using MTU channel 0 (ch0).

(a) Figure 2.5 shows a block diagram of MTU/ch0 as used in this task.

The following ch0 functions are used.

- A function that outputs pulses automatically by hardware without software intervention (output compare)
- A function that clears a counter when a compare match occurs (counter clearing)
- A function that reverses output each time a compare match occurs (toggle output)

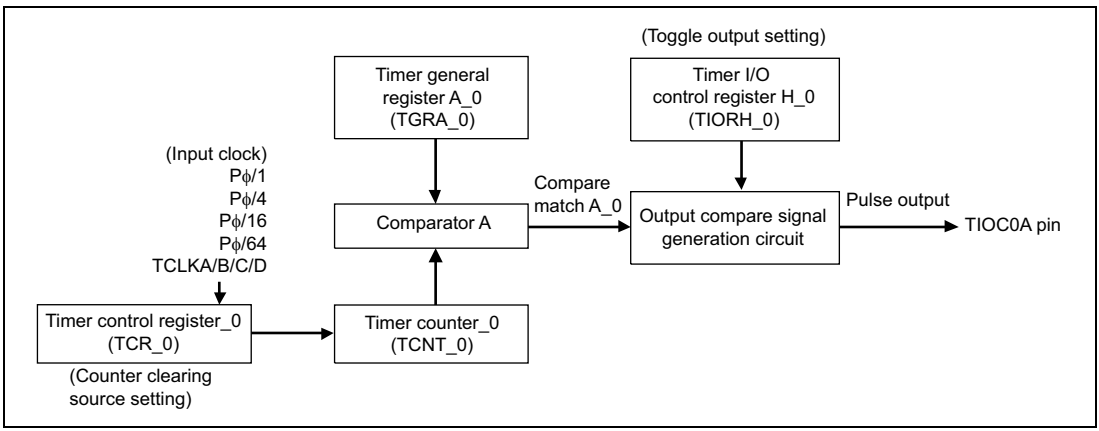


Figure 2.5 Block Diagram of MTU/ch0

(2) Table 2.2 shows the function assignments used in this sample task. Pulses are output by assigning MTU functions as shown in the table.

Table 2.2 Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC0A	Pin	Pulse output pin
TCR_0	Register	Selection of counter clearing source and input clock
TIORH_0	Register	Pulse output level setting
TGRA_0	Register	Pulse 1/2 period setting

Operation

(1) Figure 2.6 illustrates the principles of operation of this sample task. Pulses are output by SH7046 hardware and software processing as shown in the figure.

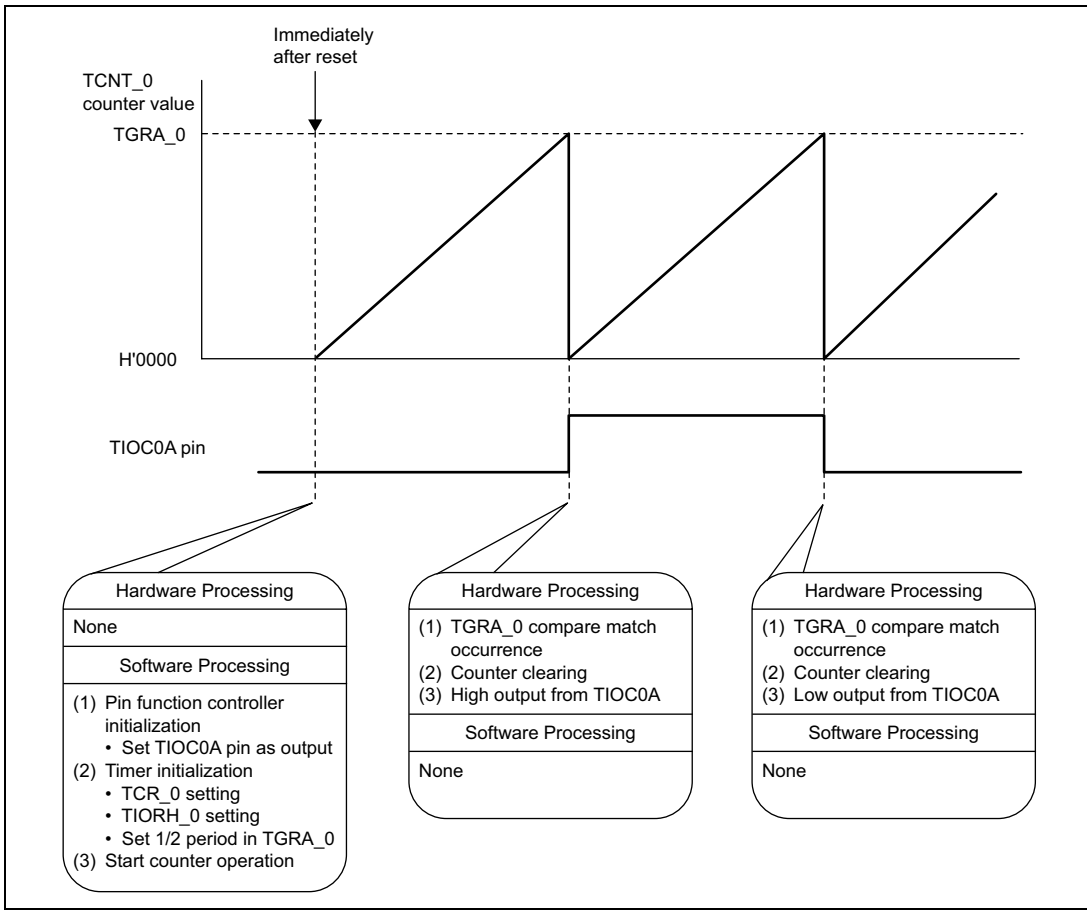


Figure 2.6 Principles of Operation of Pulse Output

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	puls_out	PFC and pulse output setting

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
pul_cyc	Used to set timer value for pulse 1/2 period Pulse period is calculated using following equation: Pulse period (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz)	1 word	Main routine	Input

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_PORTE.PECRL	Sets TIOC0A pin as output	H'FFFF83B4	H'0001
P_PORTE.PECRL2	Used to set multiplex pin as TIOC0A output	H'FFFF83BA	H'0001
P_MTU0.TCR_0	Sets TGRA_0 compare match as counter clearing source Sets Pϕ/1 as input clock	H'FFFF8260	H'20
P_MTU0.TIORH_0	TIOC0A initial output 0, output toggled on compare match	H'FFFF8262	H'03
P_MTU0.TGRA_0	Output pulse 1/2 period setting	H'FFFF8268	pul_cyc
P_MTU0.TMDR_0	Sets MTU/ch0 to normal mode	H'FFFF8261	H'c0
P_STBY.MSTCR2	MTU module standby mode clearing	H'FFFF861E	H'd2fd

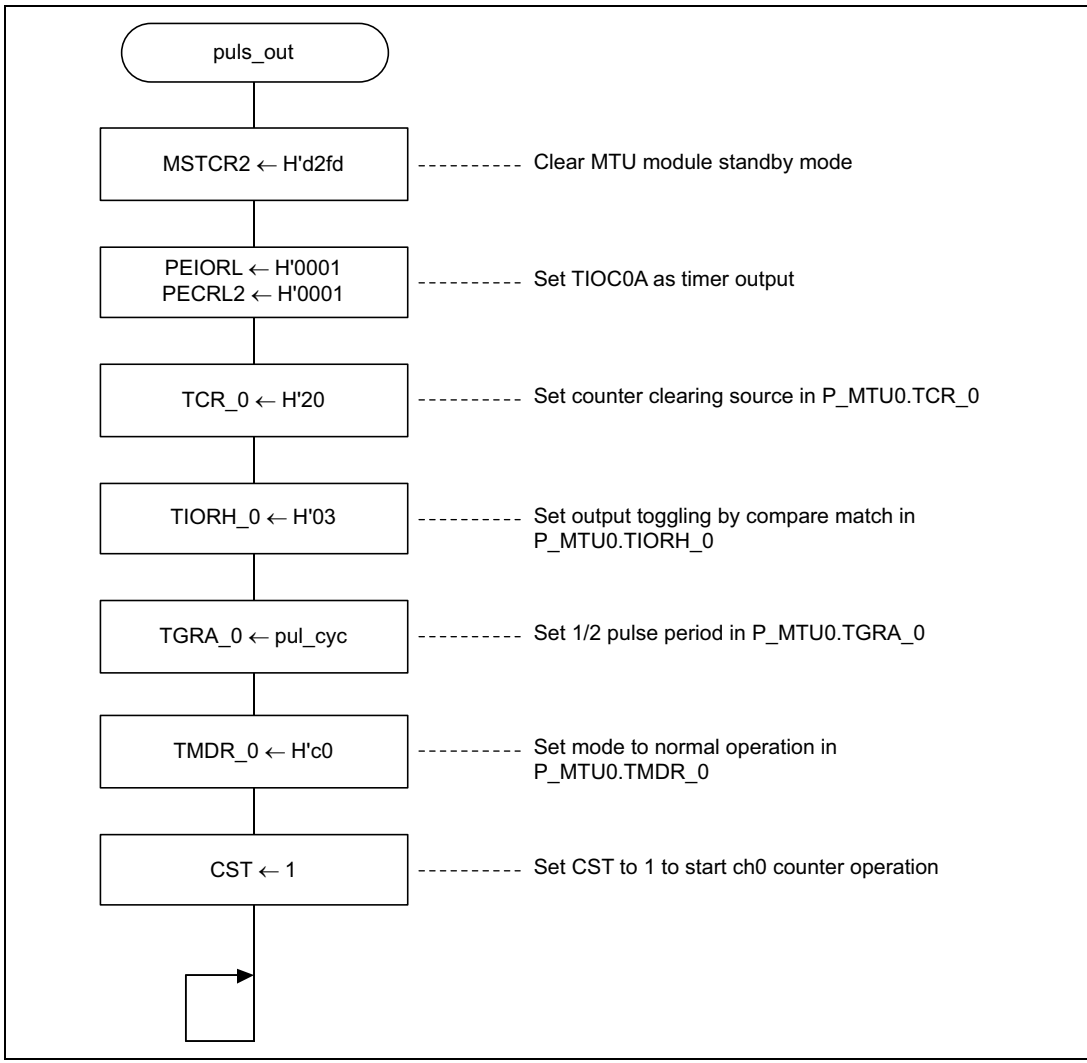
(4) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/************************************************************************/
/*                                INCLUDE FILE                          */
/************************************************************************/
#include<machine.h>
#include"iodefine_7046.h"
/************************************************************************/
/*                                PROTOTYPE                             */
/************************************************************************/
void puls_out(void);
/************************************************************************/
/*                               RAM ALLOCATION                         */
/************************************************************************/
#define  pul_cyc  (*(unsigned short *)0xffffd000)
/************************************************************************/
/*                               MAIN PROGRAM                           */
/************************************************************************/
void puls_out(void)
 {
  P_STBY.MSTCR2.WORD = 0xd2fd;
  P_PORTE.PEIORL.WORD = 0x0001;    /* TIOC0A = Output */
  P_PORTE.PECRL2.WORD = 0x0001;    /* PE0 function = TIOC0A */

  P_MTU0.TCR_0.BYTE = 0x20;        /* Counter cleare by TGRA */
  P_MTU0.TIORH_0.BYTE = 0x03;      /* toggle output */
  P_MTU0.TGRA_0 = pul_cyc;         /* 1/2 period */
  P_MTU0.TCNT_0 = 0x0000;          /* Cleara timer counter */
  P_MTU0.TMDR_0.BYTE = 0xc0;       /* Set mode */
  P_MTU34.TSTR.CST.BIT = 1;        /* Start timer counter */
  while(1);
 }
```

2.3 PWM 4-Phase Output

PWM 4-Phase Output	MCU: SH7046/47	Functions Used: MTU (PWM Mode 1)

Specifications

(1) Using MTU PWM mode 1, 4-phase PWM output is performed based on a set duty value and period.

(2) In PWM mode 1, an arbitrary period can be set for each channel. Two outputs are possible for each of ch0, ch3, and ch4, and one output for each of ch1 and ch2. Thus for ch0, ch3, and ch4, waveforms can be generated with a different high width within the same period.

(3) A duty of 0% to 100% can be set with a 1/65,535 resolution.

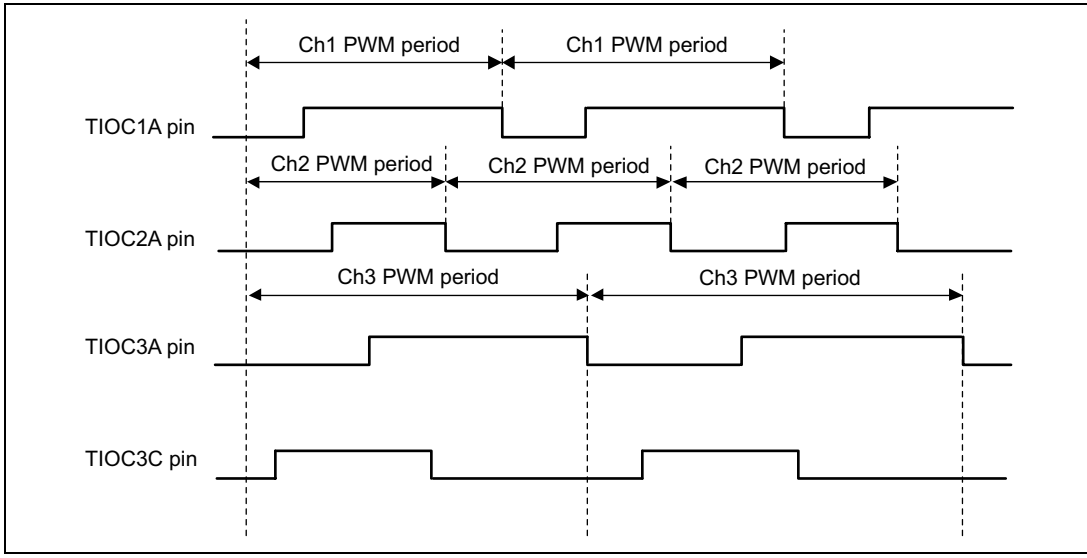


Figure 2.7 Example of PWM Output

Functions Used

(1) In this sample task, 4-phase PWM output is performed using MTU ch1 to ch3.

In PWM mode 1, PWM output is generated with TGRA paired with TGRB, and TGRC paired with TGRD. By using ch0 to ch4, a maximum of 8-phase PWM output is possible.

(a) Figure 2.8 shows a block diagram of the MTU as used in this sample task.

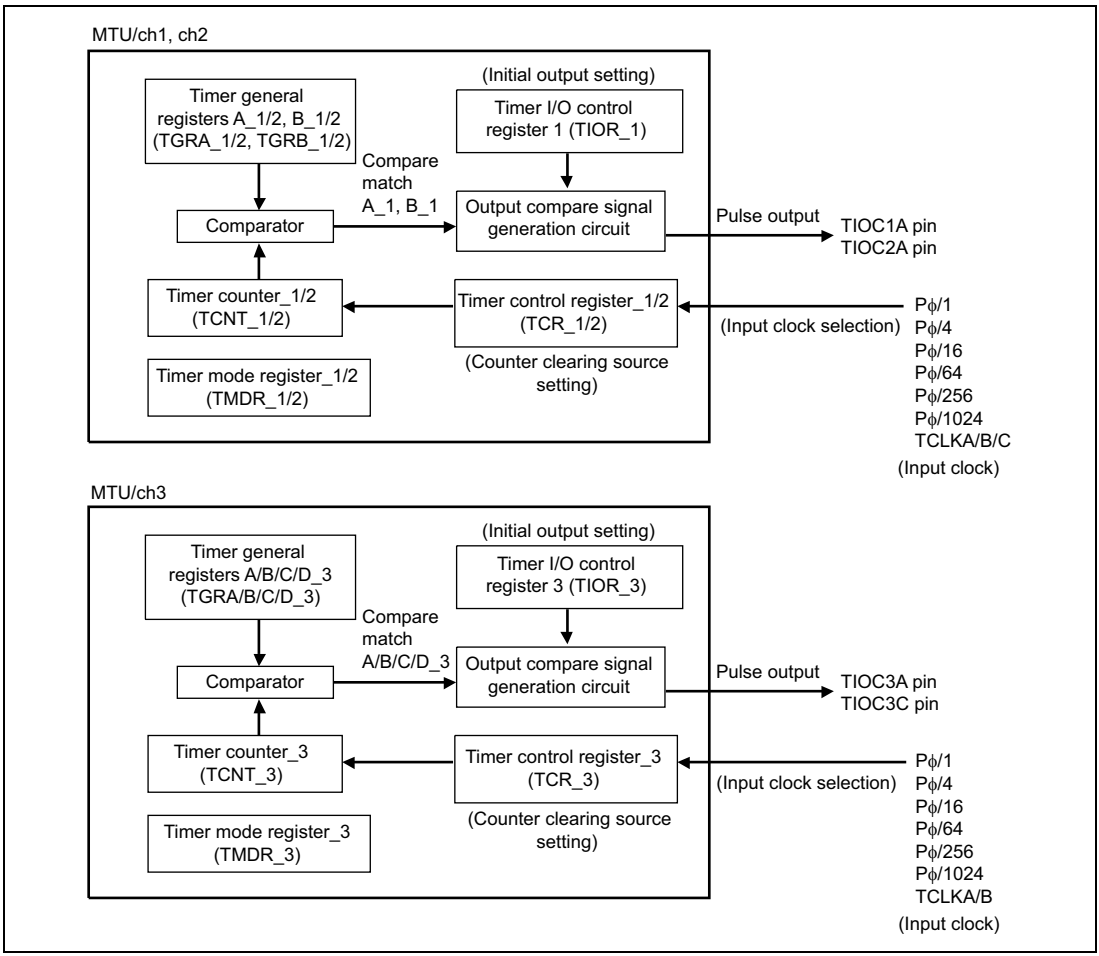


Figure 2.8 Block Diagram of MTU/ch1, ch2, ch3

(2) Table 2.3 shows the function assignments used in this sample task. PWM pulses are output by assigning MTU functions as shown in the table.

Table 2.3 Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC1A TIOC2A TIOC3A TIOC3C	Pins	PWM pulse output pins
TCR_1 TCR_2 TCR_3	Registers	Selection of ch1 to ch3 timer counter clearing sources and input clocks
TMDR_1 TMDR_2 TMDR_3	Registers	Operation of ch1 to ch3 in PWM mode 1
TGRA_1 TGRA_2 TGRA_3	Registers	PWM period setting
TGRB_1 TGRB_2 TGRB_3 TGRC_3 TGRD_3	Registers	Duty value setting

Operation

(1) Figure 2.9 illustrates the principles of operation of this sample task. Four-phase PWM output is performed from the ch1 to ch3 PWM output pins (TIOC1A, TIOC2A, TIOC3A/C) by SH7046 hardware and software processing as shown in the figure.

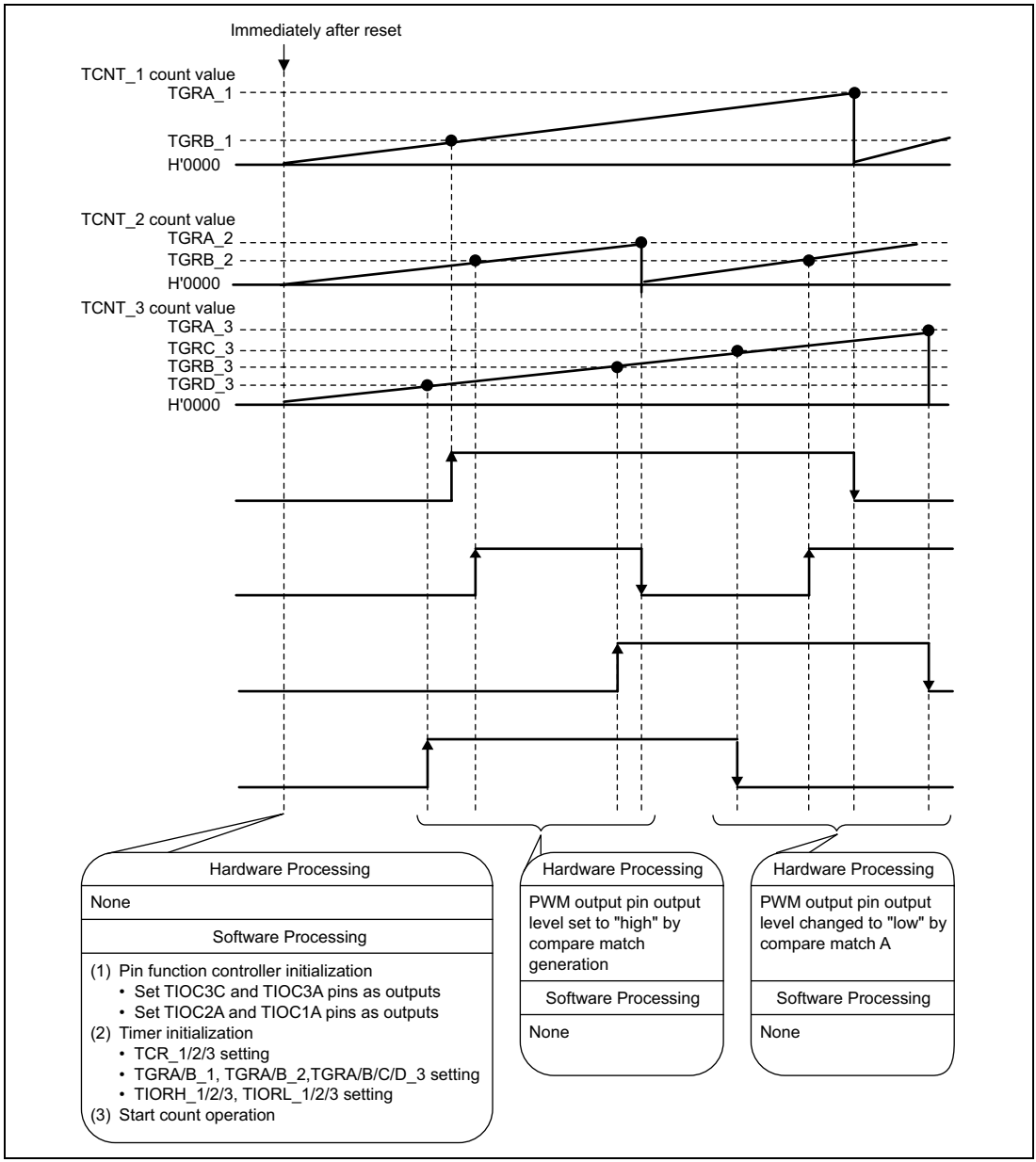


Figure 2.9 Principles of Operation of PWM Waveforms

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	pwm_1	PFC and PWM output setting

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
pul_cyc1 pul_cyc2 pul_cyc3	Used to set timer value for pulse period Pulse period is calculated using following equation: Pulse period (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz operation)	1 word	Main routine	Input
pul_duty1b pul_duty2b pul_duty3b pul_duty3c pul_duty3d	Used to set TIOC pin output waveform transition timing			

(3) Internal Registers Used

Register Name	Function Assignment	Address	Set Value
P_STBY.MSTCR2	MTU module standby mode clearing	H'FFFF861E	H'd2fd
P_PORTE.PEIORL P_PORTE.PECRL1 P_PORTE.PECRL2	Multiplex pins set as timer output pins TIOC1A, TIOC2A, TIOC3A, TIOC3C	H'FFFF83B4 H'FFFF83B8 H'FFFF83BA	H'0550 H'0011 H'1100
P_MTU1.TCR_1 P_MTU2.TCR_2 P_MTU3.TCR_3	Timer counter clearing sources cleared by TGRA_1, TGRA_2, TGRA_3 compare matches Pϕ/1 selected as input clock	H'FFFF8280 H'FFFF82A0 H'FFFF8200	H'20 H'20 H'20
P_MTU1.TGRA_1	Channel 1 PWM period setting	H'FFFF8288	pul_cyc1
P_MTU1.TGRB_1	Used to set timer counter value causing high output from TIOC1A	H'FFFF828A	pul_duty1b
P_MTU2.TGRA_2	Channel 2 PWM period setting	H'FFFF82A8	pul_cyc2
P_MTU2.TGRB_2	Used to set timer counter value causing high output from TIOC2A	H'FFFF82AA	pul_duty2b

Register Name	Function Assignment	Address	Set Value
P_MTU34.TGRA_3	Channel 3 PWM period setting	H'FFFF8218	pul_cyc3
P_MTU34.TGRB_3	Used to set timer counter value causing high output from TIOC3A	H'FFFF821A	pul_duty3b
P_MTU34.TGRC_3	Used to set timer counter value causing low output from TIOC3C	H'FFFF8224	pul_duty3c
P_MTU34.TGRD_3	Used to set timer counter value causing high output from TIOC3C	H'FFFF8226	pul_duty3d
P_MTU1.TIOR_1	Sets TGRA_1 initial output 0, 0 output on output compare, TGRB_1 initial output 0, 1 output on output compare	H'FFFF8282	H'02
P_MTU2.TIOR_2	Sets TGRA_2 initial output 0, 0 output on output compare, TGRB_2 initial output 0, 1 output on output compare	H'FFFF82A2	H'02
P_MTU34.TIORH_3	Sets TGRA_3 initial output 0, 0 output on output compare, TGRB_3 initial output 0, 1 output on output compare	H'FFFF8204	H'02
P_MTU34.TIORL_3	Sets TGRC_3 initial output 0, 0 output on output compare, TGRD_3 initial output 0, 1 output on output compare	H'FFFF8205	H'21
P_MTU1.TMDR_1 P_MTU2.TMDR_2 P_MTU34.TMDR_3	Used to set PWM mode 1 as operating mode	H'FFFF8281 H'FFFF82A1 H'FFFF8202	H'c2 H'c2 H'c2

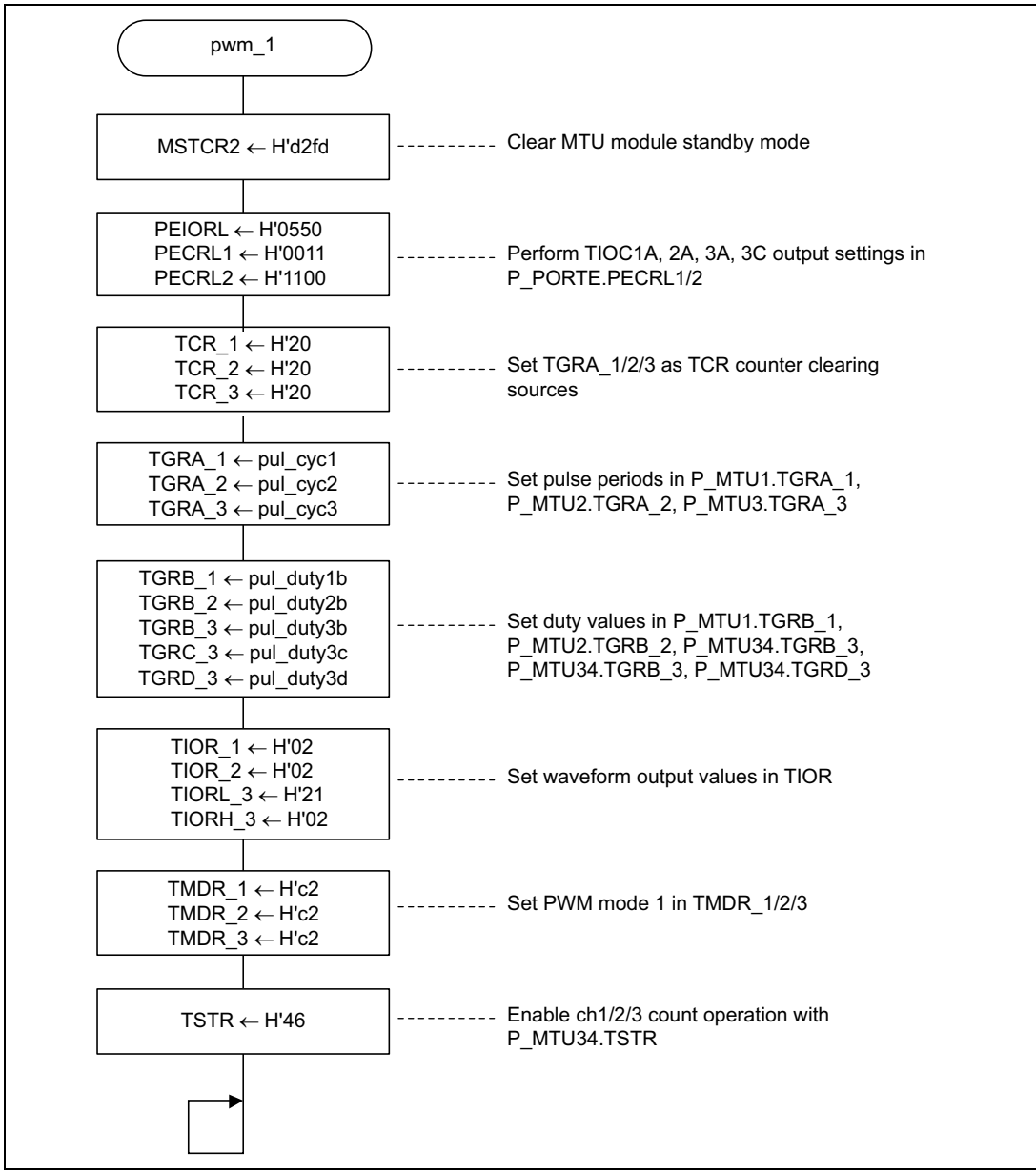
(4) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/*****************************************************************************/
/*                                INCLUDE FILE                               */
/*****************************************************************************/
#include<machine.h>
#include"iodefine_7046.h"
/*****************************************************************************/
/*                                 PROTOTYPE                                 */
/*****************************************************************************/
void pwm_1(void);
/*****************************************************************************/
/*                                RAM ALLOCATION                             */
/*****************************************************************************/
#define  pul_cyc1     (*(unsigned short  *)0xffffd000)
#define  pul_duty1b   (*(unsigned short  *)0xffffd002)
#define  pul_cyc2     (*(unsigned short  *)0xffffd004)
#define  pul_duty2b   (*(unsigned short  *)0xffffd006)
#define  pul_cyc3     (*(unsigned short  *)0xffffd008)
#define  pul_duty3b   (*(unsigned short  *)0xffffd00a)
#define  pul_duty3c   (*(unsigned short  *)0xffffd00c)
#define  pul_duty3d   (*(unsigend short  *)0xffffd00e)
/*****************************************************************************/
/*                                MAIN PROGRAM                               */
/*****************************************************************************/
void pwm_1(void)
 {
    P_STBY.MSTCR2.WORD = 0xd2fd;        /* Clear module standby mode */
    P_PORTE.PEIORL.WORD = 0x0550;       /* TIOC1A/2A/3A/3C = output */
    P_PORTE.PECRL1.WORD = 0x0011;
    P_PORTE.PECRL2.WORD = 0x1100;

    P_MTU1.TCR_1.BYTE = 0x20;           /* Counter clear by TGRA */
    P_MTU1.TGRA_1 = pul_cyc1;           /* set period */
    P_MTU1.TGRB_1 = pul_duty1b;         /* set duty */
    P_MTU1.TIOR_1.BYTE = 0x02;
    P_MTU1.TMDR_1.BYTE = 0xc2;          /* PWM mode1 */
    P_MTU1.TCNT_1 = 0x0000;

    P_MTU2.TCR_2.BYTE = 0x20;
    P_MTU2.TGRA_2 = pul_cyc2;
    P_MTU2.TGRB_2 = pul_duty2b;
    P_MTU2.TIOR_2.BYTE = 0x02;
    P_MTU2.TMDR_2.BYTE =0xc2;
    P_MTU2.TCNT_2 = 0x0000;

    P_MTU34.TCR_3.BYTE = 0x20;
    P_MTU34.TGRA_3 = pul_cyc3;
    P_MTU34.TGRB_3 = pul_duty3b;
    P_MTU34.TGRC_3 = pul_duty3c;
    P_MTU34.TGRD_3 = pul_duty3d;
```

```
 P_MTU34.TIORL_3.BYTE = 0x21;
 P_MTU34.TIORH_3.BYTE = 0x02;
 P_MTU34.TMDR_3.BYTE = 0xc2;
 P_MTU34.TCNT_3 = 0x0000;

 P_MTU34.TSTR.BYTE = 0x46;         /* Timer counter start */
 while(1);
}
```

2.4 PWM 7-Phase Output

PWM 7-Phase Output	MCU: SH7046/47	Functions Used: MTU (PWM Mode 2)

Specifications

(1) Seven-phase PWM output allowing the pulse high width and duty to be varied is performed as shown in figure 2.10

(2) When operating with on-chip peripheral clock Pϕ = 20.0 MHz, the output PWM period can be set arbitrarily in the range 100 ns to 3.27 ms.

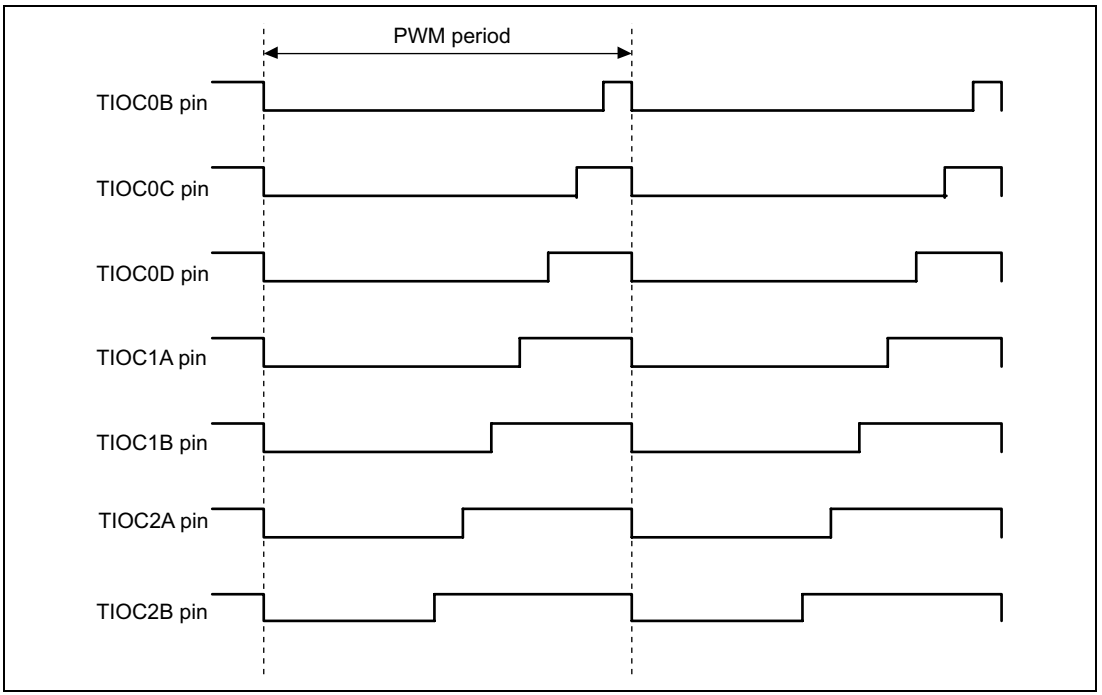


Figure 2.10 Example of PWM Output

Functions Used

(1) In this sample task, 7-phase PWM output is performed by synchronous operation of MTU ch0 to ch2.

(a) Figure 2.11 shows a block diagram of the MTU as used in this sample task.

This sample task uses the following MTU functions.

- A function that outputs pulses automatically by hardware without software intervention (output compare)
- A function that clears a counter when a compare match occurs (counter clearing)
- A function that reverses output each time a compare match occurs (toggle output)

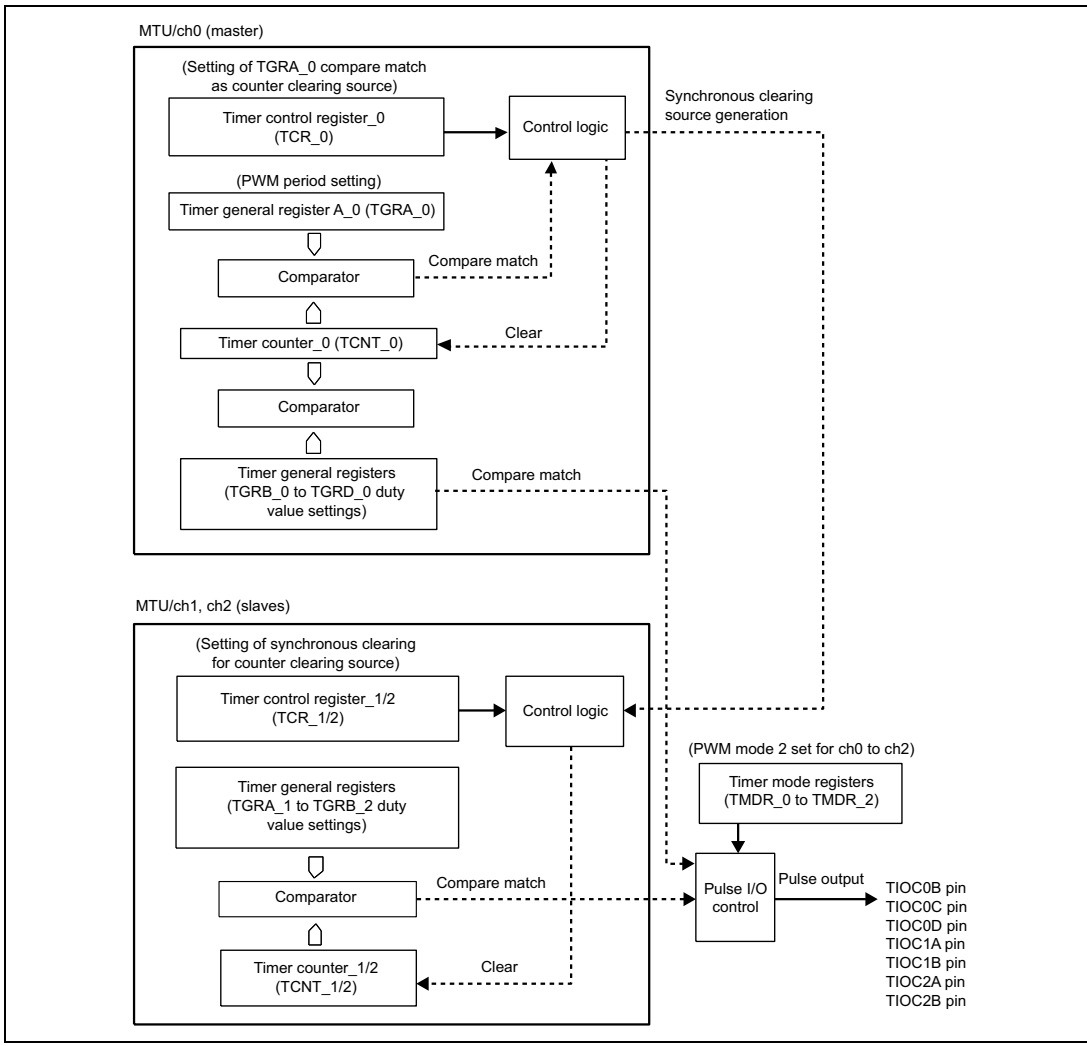


Figure 2.11 Block Diagram of Synchronous Clearing

(2) Table 2.4 shows the function assignments used in this task. PWM pulses are output by assigning MTU functions as shown in the table.

Table 2.4 MTU Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC0B TIOC0C TIOC0D TIOC1A TIOC1B TIOC2A TIOC2B	Pins	PWM pulse output pins
TSYR	Register	Ch0/1/2 synchronous operation
TCR_0/1/2	Register	Selection of ch0/1/2 timer counter clearing sources and input clocks
TGRA_0	Register	PWM period setting
TGRB_0 TGRC_0 TGRD_0 TGRA_1 TGRB_1 TGRA_2 TGRB_2	Registers	Duty value setting
TMDR_0/1/2	Register	Operation of ch0/1/2 in PWM Mode 2

Operation

(1) Figure 2.12 illustrates the principles of operation of this sample task. Seven-phase PWM output is performed from the ch0/1/2 PWM output pins (TIOC0B/C/D, TIOC1A/B, TIOC2A/B) by SH7046 hardware and software processing as shown in the figure.

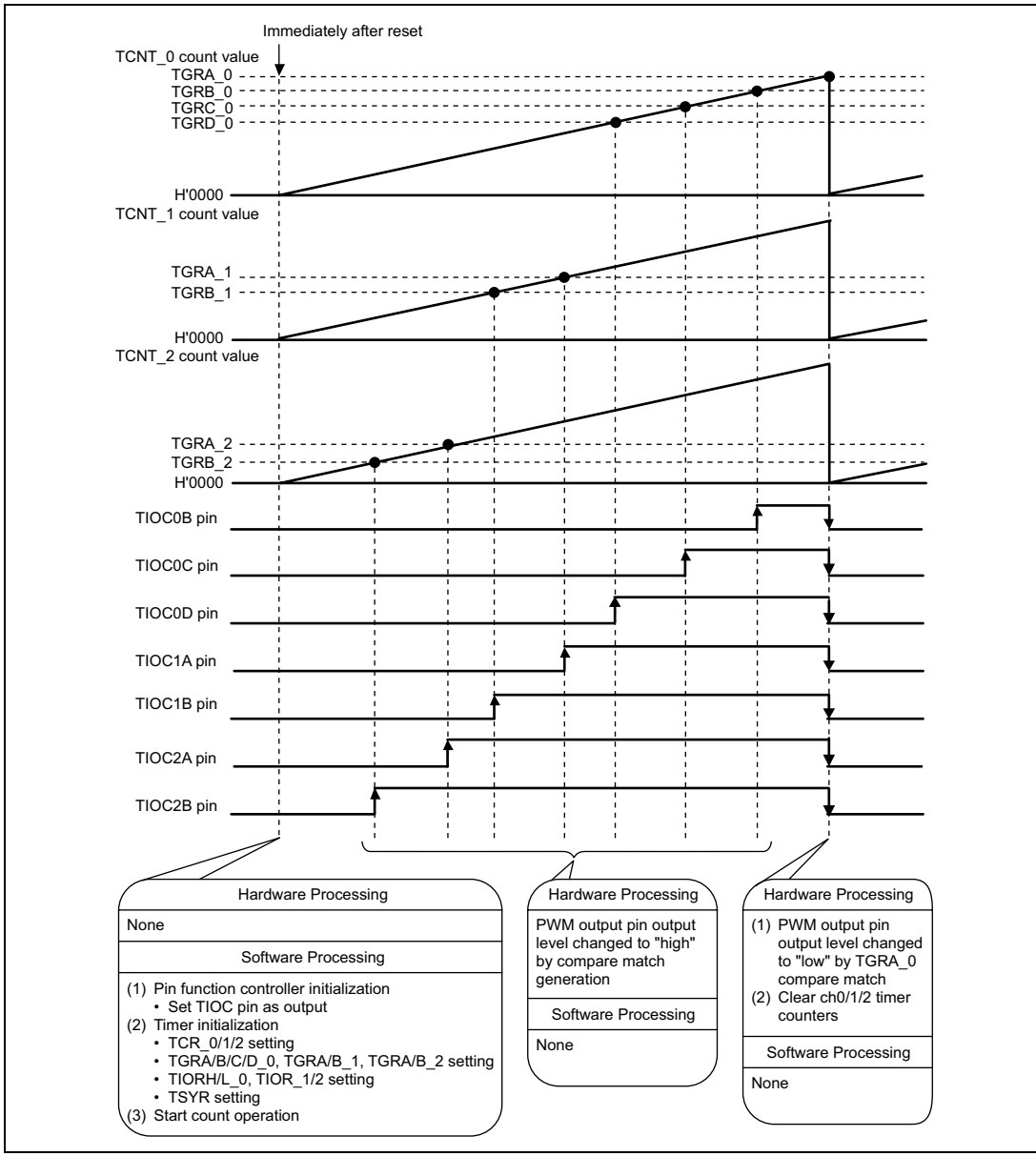


Figure 2.12 Principles of Operation of PWM Output (7-Phase) Using Sawtooth Waveform Generation

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	pwm_2	PFC and PWM output setting

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
pul_cyc0a	Used to set timer value for pulse period Pulse period is calculated using following equation: Pulse period (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz operation)	1 word	Main routine	Input
pul_duty0b pul_duty0c pul_duty0d pul_duty1a pul_duty1b pul_duty2a pul_duty2b	Used to set TIOC pin output waveform transition timing			

(3) Internal Registers Used

Register Name	Function Assignment	Address	Set Value
P_STBY.MSTCR2	Module standby mode clearing	H'FFFF861E	H'd2fd
P_PORTE.PEIORL P_PORTE.PECRL2	Used to set multiplex pins as timer output pins TIOC0B/C/D, TIOC1A/B, TIOC2A/B	H'FFFF83B4 H'FFFF83BA	H'00fe H'5554
P_MTU34.TSYR	Synchronous operation set for timer counters 0/1/2	H'FFFF8241	H'07
P_MTU0.TCR_0 P_MTU1.TCR_1 P_MTU2.TCR_2	Used to select TGRA_0 compare match set as timer counter clearing source, and Pϕ/1 as input clock	H'FFFF8260 H'FFFF8280 H'FFFF82A0	H'20 H'60 H'60
P_MTU0.TGRA_0	PWM period setting	H'FFFF8268	pul_cyc0
P_MTU0.TGRB_0	Used to set timer counter value causing high output from TIOC0B	H'FFFF826A	pul_duty0b
P_MTU0.TGRC_0	Used to set timer counter value causing high output from TIOC0C	H'FFFF826C	pul_duty0c

Register Name	Function Assignment	Address	Set Value
P_MTU0.TGRD_0	Used to set timer counter value causing high output from TIOC0D	H'FFFF826E	pul_duty0d
P_MTU1.TGRA_1	Used to set timer counter value causing high output from TIOC1A	H'FFFF8288	pul_duty1a
P_MTU1.TGRB_1	Used to set timer counter value causing high output from TIOC1B	H'FFFF828A	pul_duty1b
P_MTU2.TGRA_2	Used to set timer counter value causing high output from TIOC2A	H'FFFF82A8	pul_duty2a
P_MTU2.TGRB_2	Used to set timer counter value causing high output from TIOC2B	H'FFFF82AA	pul_duty2b
P_MTU0.TIORH_0	Sets TGRA_0 initial output 0, 0 output on output compare, TGRB_0 initial output 0, 1 output on output compare	H'FFFF8262	H'20
P_MTU0.TIORL_0	Sets TGRC_0 initial output 0, 1 output on output compare, TGRD_0 initial output 0, 1 output on output compare	H'FFFF8263	H'22
P_MTU1.TIOR_1	Sets TGRA_1 initial output 0, 1 output on output compare, TGRB_1 initial output 0, 1 output on output compare	H'FFFF8282	H'22
P_MTU1.TIOR_2	Sets TGRA_2 initial output 0, 1 output on output compare, TGRB_2 initial output 0, 1 output on output compare	H'FFFF82A2	H'22
P_MTU0.TMDR_0 P_MTU1.TMDR_1 P_MTU2.TMDR_2	Used to set PWM Mode 2 as operating mode of each channel	H'FFFF8261 H'FFFF8281 H'FFFF82A1	H'c3 H'c3 H'c3

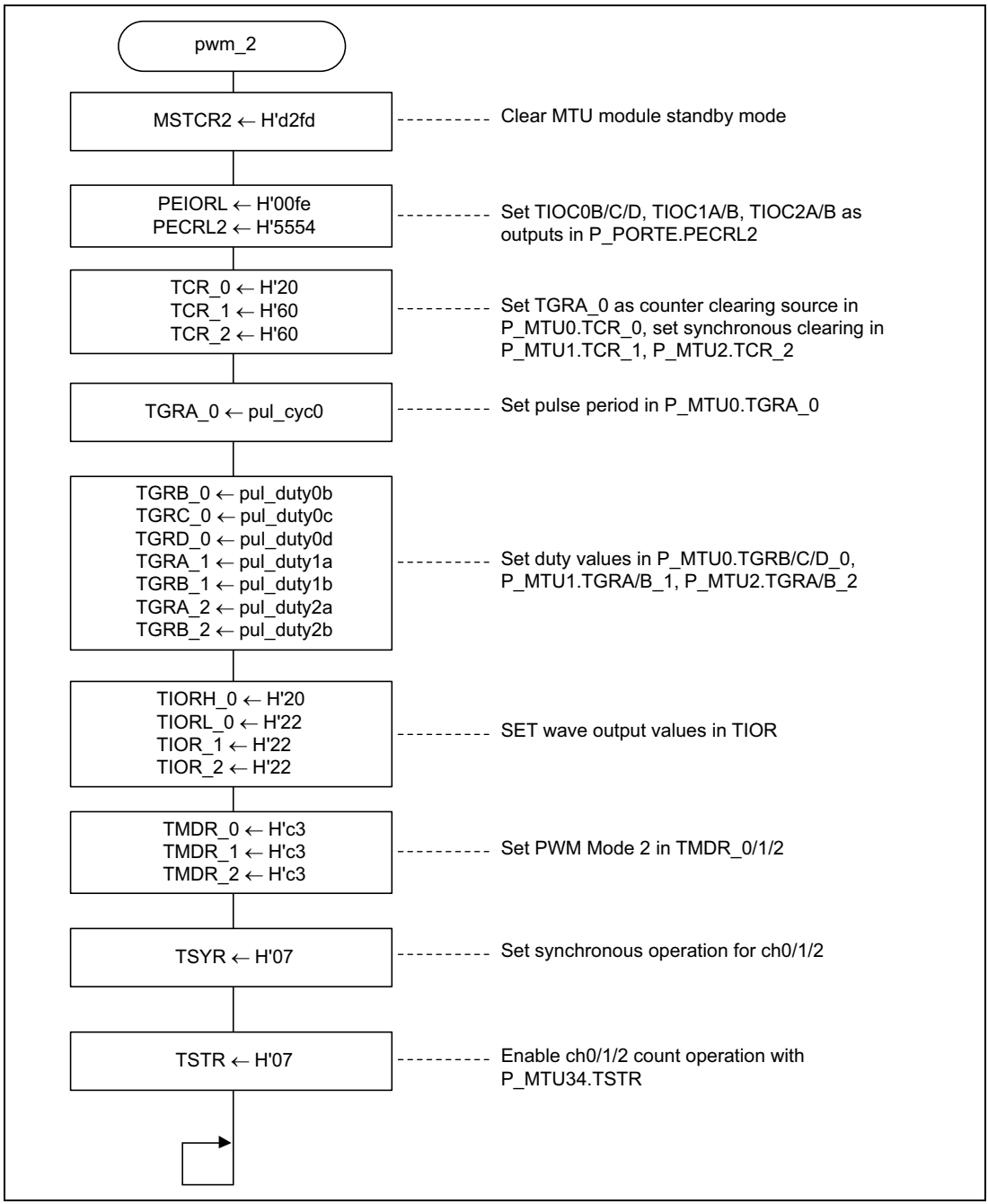
(4) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/************************************************************************/
/*                              INCLUDE  FILE                           */
/************************************************************************/
#include<machine.h>
#include"iodefine_7046.h"
/************************************************************************/
/*                               PROTOTYPE                              */
/************************************************************************/
void pwm_2(void);
/************************************************************************/
/*                            RAM ALLOCATION                            */
/************************************************************************/
#define pul_cyc0  (*(unsigned short *)0xffffd000)
#define pul_duty0b  (*(unsigned short *)0xffffd002)
#define pul_duty0c  (*(unsigned short *)0xffffd004)
#define pul_duty0d  (*(unsigned short *)0xffffd006)
#define pul_duty1a  (*(unsigned short *)0xffffd008)
#define pul_duty1b  (*(unsigned short *)0xffffd00a)
#define pul_duty2a  (*(unsigned short *)0xffffd00c)
#define pul_duty2b   (*(unsigned short *)0xffffd00e)
/************************************************************************/
/*                              MAIN PROGRAM                            */
/************************************************************************/
void pwm_2(void)
 {
   P_STBY.MSTCR2.WORD =0xd2fd;         /* Clear module standby mode */
   P_PORTE.PEIORL.WORD = 0x00fe;       /* TIOC0B/C/D,TIOC1A/B,TIOC2A/B output */
   P_PORTE.PECRL2.WORD = 0x5554;

   P_MTU0.TCR_0.BYTE = 0x20;           /* Counter clear by TGRA_0 */
   P_MTU0.TIORH_0.BYTE = 0x20;
   P_MTU0.TIORL_0.BYTE = 0x22;
   P_MTU0.TCNT_0 = 0x0000;
   P_MTU0.TGRA_0 = pul_cyc0;           /* Set general register */
   P_MTU0.TGRB_0 = pul_duty0b;
   P_MTU0.TGRC_0 = pul_duty0c;
   P_MTU0.TGRD_0 = pul_duty0d;
   P_MTU0.TMDR_0.BYTE = 0xc3;          /* PWM mode2 */

   P_MTU1.TCR_1.BYTE = 0x60;           /* Counter clear by TGRA_0 */
   P_MTU1.TIOR_1.BYTE = 0x22;
   P_MTU1.TCNT_1 = 0x0000;
   P_MTU1.TGRA_1 = pul_duty1a;         /* Set general register */
   P_MTU1.TGRB_1 = pul_duty1b;
   P_MTU1.TMDR_1.BYTE = 0xc3;          /* PWM mode2 */

   P_MTU2.TCR_2.BYTE =0x60;            /* Counter clear by TGRA_0 */
   P_MTU2.TIOR_2.BYTE = 0x22;
   P_MTU2.TCNT_2 = 0x0000;
```

```
 P_MTU2.TGRA_2 = pul_duty2a;          /* Set general register */
 P_MTU2.TGRB_2 = pul_duty2b;
 P_MTU2.TMDR_2.BYTE = 0xc3;           /* PWM mode2 */

 P_MTU34.TSYR.BYTE = 0x07;
 P_MTU34.TSTR.BYTE = 0x07;            /* Start timer counter */
 while(1);
}
```

2.5 Positive-Phase/Negative Phase PWM 3-Phase Output

Positive-Phase/Negative Phase PWM 3-Phase Output	MCU: SH7046/47	Functions Used: MTU (Reset-Synchronized PWM Mode)

Specifications

(1) Positive-phase and negative-phase 3-phase pulse (duty pulse) output is performed that allows the pulse high width and duty to be varied, as shown in figure 2.13.

(2) When operating with on-chip peripheral clock Pϕ = 20.0 MHz, the output pulse period can be set arbitrarily in the range 100.0 ns to 3.27 ms.

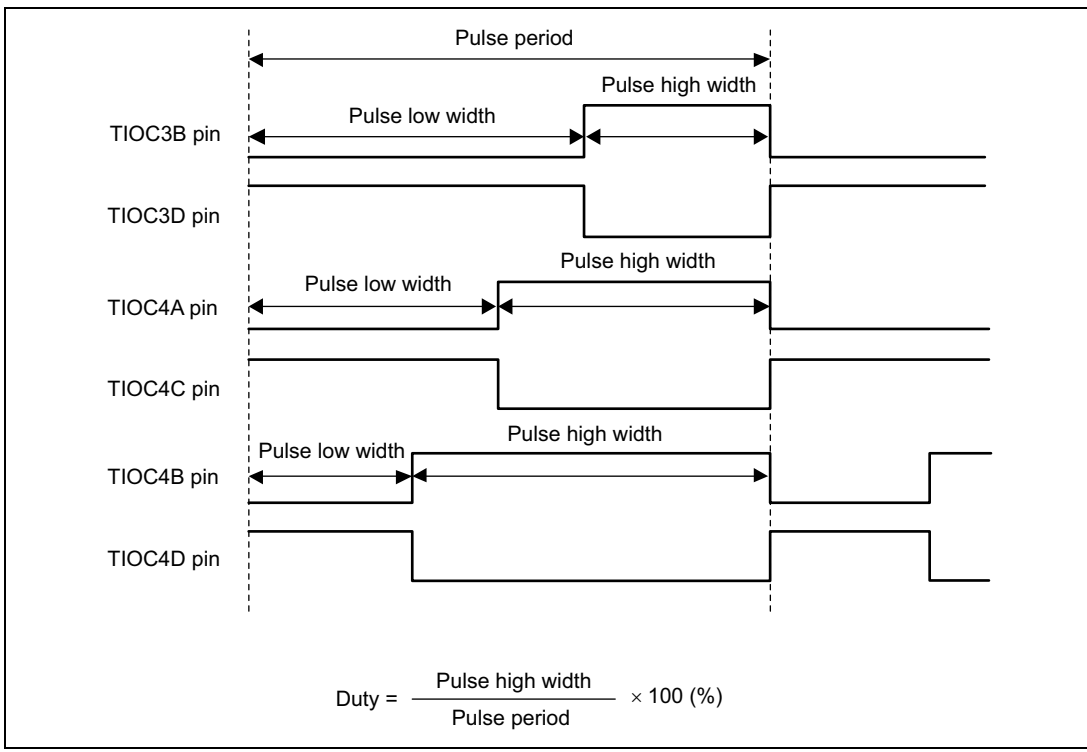


Figure 2.13 Positive-Phase/Negative-Phase PWM 3-Phase Output Waveforms

Functions Used

(1) In this sample task, MTU ch3 and ch4 are used in combination, and 3-phase PWM waveform output is performed with one common transition point in the relationship between the positive phase and negative phase.

In reset-synchronized PWM mode, PWM waveforms are generated using buffer operation, with TGRA and TGRC operating as a pair, and TRGB and TGRD operating as a pair.

(a) Figure 2.14 shows a block diagram of the MTU as used in this sample task.

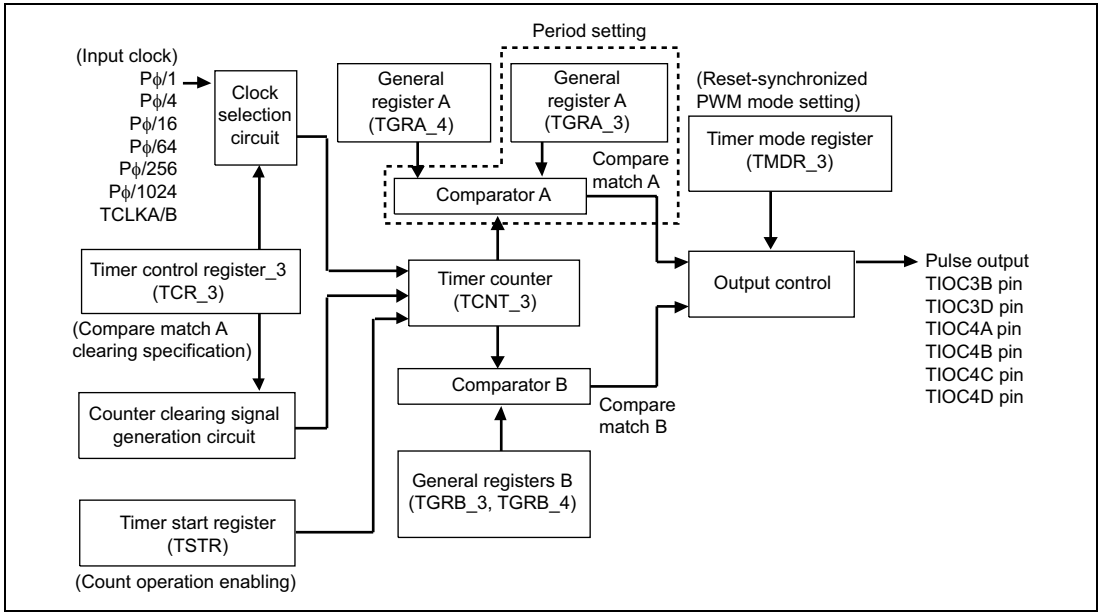


Figure 2.14 Block Diagram of MTU/ch3, ch4

(2) Table 2.5 shows the function assignments used in this task. PWM pulses are output by assigning MTU functions as shown in the table.

Table 2.5 Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC3B	Pin	PWM output 1
TIOC3D	Pin	Negative-phase waveform of PWM output 1
TIOC4A	Pin	PWM output 2
TIOC4B	Pin	PWM output 3
TIOC4C	Pin	Negative-phase waveform of PWM output 2
TIOC4D	Pin	Negative-phase waveform of PWM output 3
TCR_3	Register	Selection of ch3 timer counter clearing source and input clock
TMDR_3	Register	Ch3 set to operate in reset-synchronized PWM mode
TGRA_3	Register	PWM period setting
TGRB_3 TGRA_4 TGRB_4	Registers	Duty value setting

Operation

(1) Figure 2.15 illustrates the principles of operation of this sample task. Three-phase PWM waveforms are output from the PWM output pins (TIOC3B/D, TIOC4A/B/C/D) by SH7046 hardware and software processing as shown in the figure.

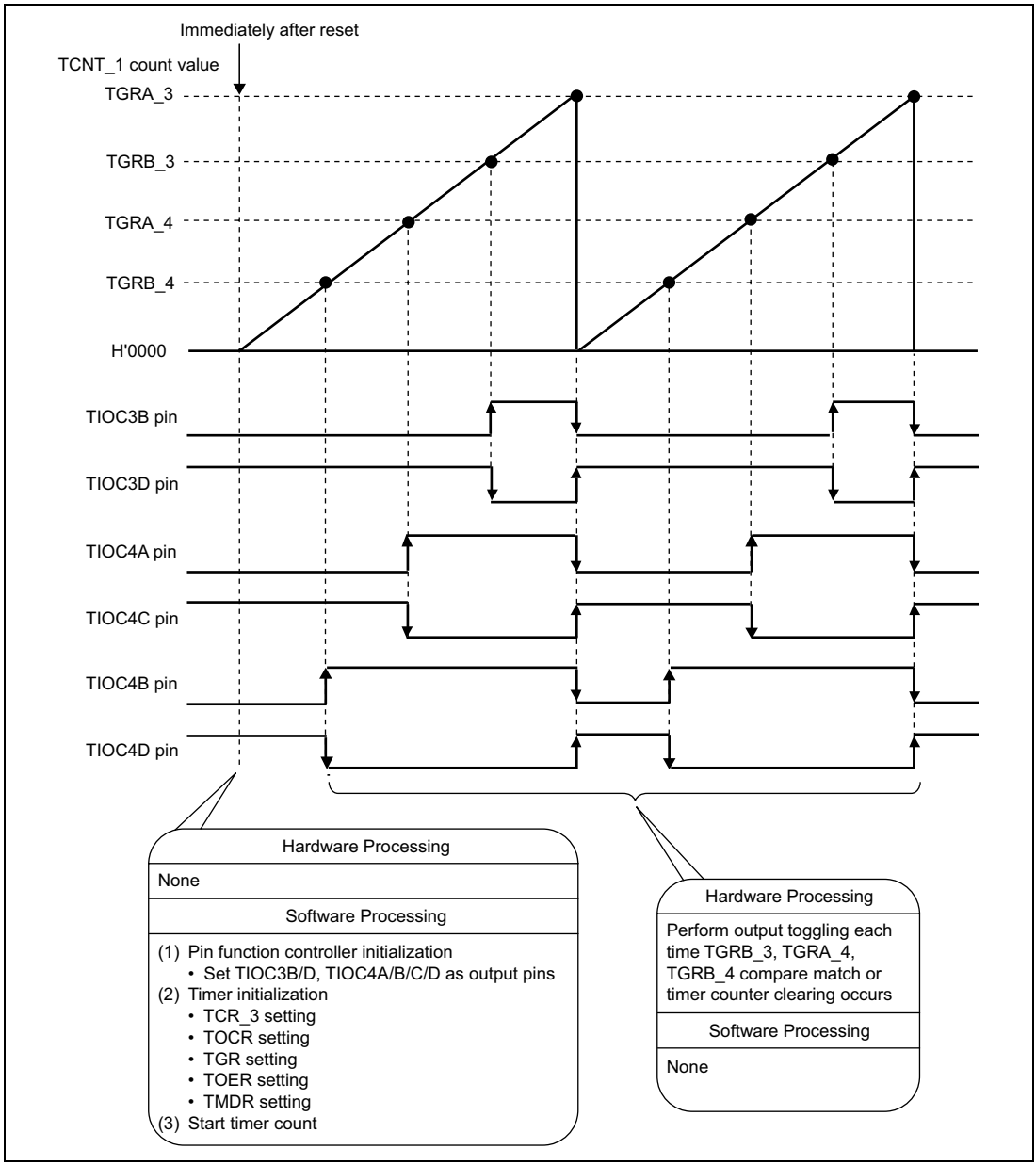


Figure 2.15 Principles of Operation of Reset-Synchronized PWM Waveforms

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	rst_pwm	PFC and PWM output setting

(2) Arguments

Label or Register Name	Function	Data Length	Module	Input/ Output
pul_cyc1	Used to set timer value for pulse period Pulse period is calculated using following equation: Pulse period (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz operation)	1 word	Main routine	Input
pul_duty3b pul_duty4a pul_duty4b	Used to set TIOC pin output waveform transition timing			

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_PORTE.PEIORL P_PORTE.PECRL1	Used to set multiplex insert as timer output pins TIOC3B/D, TIOC4A/B/C/D	H'FFFF83B4 H'FFFF83B8	H'fa00 H'5544
P_MTU34.TCR_3	Used to select TGRA_3 compare match as timer counter clearing source, and Pϕ/1 as input clock	H'FFFF8200	H'20
P_MTU34.TOCR	Enabling of toggle output synchronized with PWM period, and positive-phase/negative-phase output level setting	H'FFFF820B	H'43
P_MTU34.TGRA_3	PWM period setting	H'FFFF8218	pul_cyc1
P_MTU34.TGRB_3	Used to set timer counter value for toggle output from TIOC3B/D	H'FFFF821A	pul_duty3b
P_MTU34.TGRA_4	Used to set timer counter value for toggle output from TIOC4A/C	H'FFFF821C	pul_duty4a
P_MTU34.TGRB_4	Used to set timer counter value for toggle output from TIOC4B/D	H'FFFF821E	pul_duty4b
P_MTU34.TOER	Sets enabling of reset-synchronized PWM output	H'FFFF821E	H'ff
P_MTU34.TMDR_3	Sets reset-synchronized PWM mode	H'FFFF8202	H'c8
P_STBY.MSTCR2	Module standby mode clearing	H'FFFF861E	H'd2fd

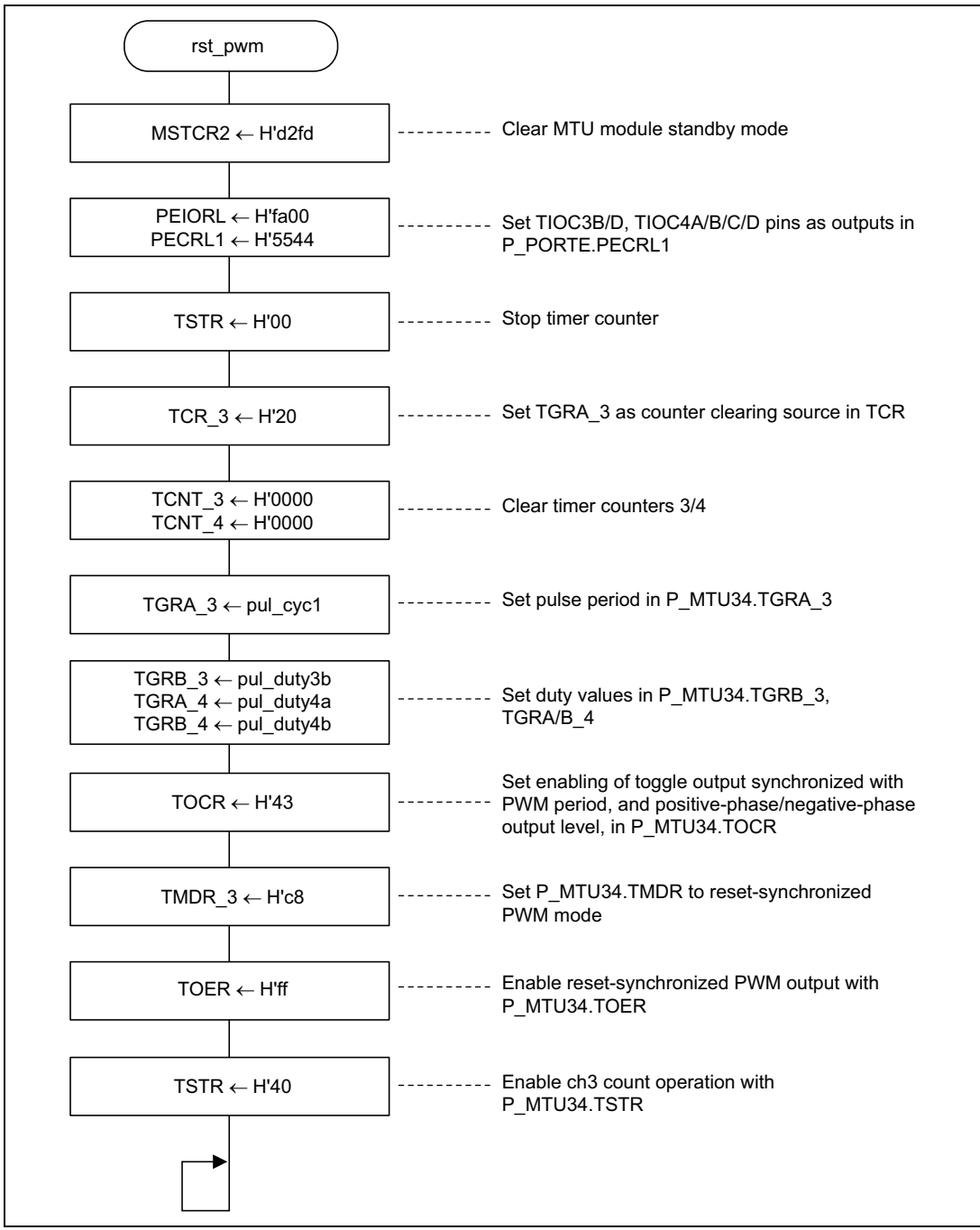
(4) RAM Used

This sample application does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/******************************************************************************/
/*                                INCLUDE FILE                                */
/******************************************************************************/
#include<machine.h>
#include"iodefine_7046.h"
/******************************************************************************/
/*                                 PROTOTYPE                                  */
/******************************************************************************/
void rst_pwm(void);
/******************************************************************************/
/*                               RAM ALLOCATION                               */
/******************************************************************************/
#define pul_cyc1  (*(unsigned short *)0xffffd000)
#define pul_duty3b  (*(unsigned short *)0xffffd002)
#define pul_duty4a  (*(unsigned short *)0xffffd004)
#define pul_duty4b  (*(unsigned short *)0xffffd006)
/******************************************************************************/
/*                                MAIN PROGRAM                                */
/******************************************************************************/
void rst_pwm(void)
 {
   P_STBY.MSTCR2.WORD = 0xd2fd;      /* Clear module standby mode */
   P_PORTE.PEIORL.WORD = 0xfa00;     /* TIOC3B/D,TIOC4A/B/C/D output */
   P_PORTE.PECRL1.WORD = 0x5544;
   P_MTU34.TSTR.BYTE = 0x00;
   P_MTU34.TCR_3.BYTE = 0x20;        /* Counter clear by TGRA_3 */
   P_MTU34.TCNT_3 = 0x0000;          /* Clear timer counter3 */
   P_MTU34.TCNT_4 = 0x0000;          /* Clear timer counter4 */
   P_MTU34.TGRA_3 = pul_cyc1;        /* Set period */
   P_MTU34.TGRB_3 = pul_duty3b;      /* Set duty */
   P_MTU34.TGRA_4 = pul_duty4a;
   P_MTU34.TGRB_4 = pul_duty4b;
   P_MTU34.TOCR.BYTE = 0x43;         /* Set timer output control register */
   P_MTU34.TMDR_3.BYTE = 0xc8;       /* Reset synchronized PWM mode */
   P_MTU34.TOER.BYTE = 0xff;         /* Timer output enable */
   P_MTU34.TSTR = 0x40;              /* Start timer counter */
   while(1);
 }
```

2.6 Complementary PWM 3-Phase Output

Complementary PWM 3-Phase Output	MCU: SH7046/47	Functions Used: MTU (Complementary PWM Mode)

Specifications

(1) Three-phase PWM waveform output is performed with a non-overlapping relationship between positive and negative phases, as shown in figure 2.16.

(2) The duty can be changed between 0% and 100% by setting an arbitrary value in RAM.

$$\text{Duty} = \frac{\text{Pulse high width}}{\text{Pulse period}} \times 100\ (\%)$$

(3) Toggle waveform output is performed synchronized with the period.

(4) When operating with on-chip peripheral clock Pϕ = 20.0 MHz, the output pulse period can be set arbitrarily in the range 100.0 ns to 3.27 ms.

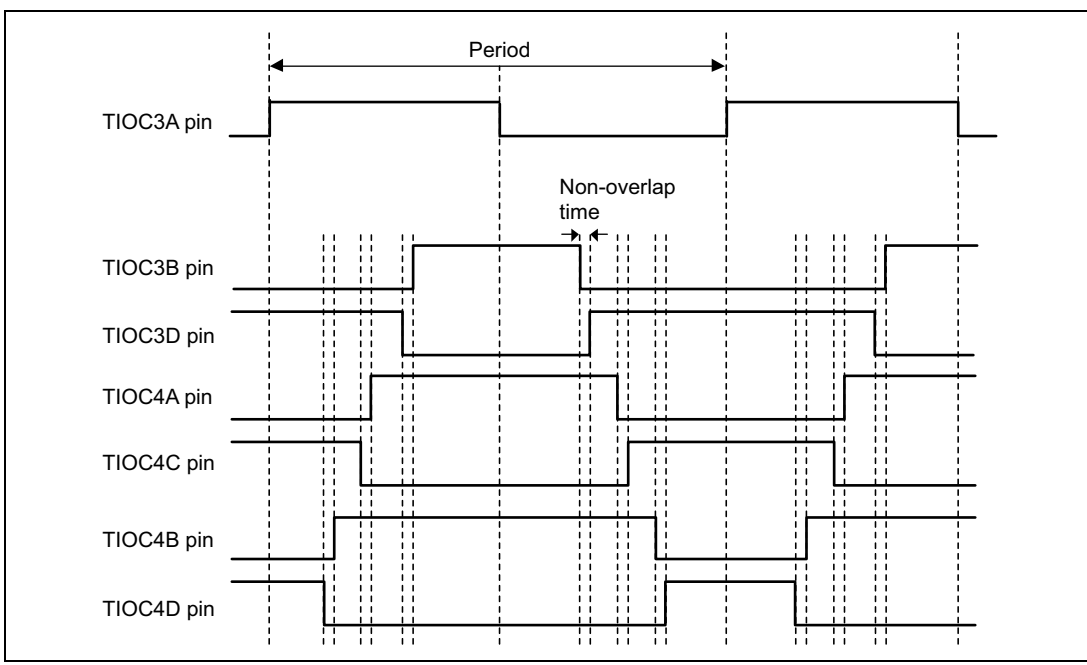


Figure 2.16 Complementary PWM 3-Phase Output Waveforms

Functions Used

(1) In this sample task, 3-phase PWM waveform output with a non-overlapping relationship between positive and negative phases is performed using MTU channels 3 and 4.

(a) Figure 2.17 shows a block diagram of MTU/ch3 and ch4 as used in this sample task.
This sample task uses the following functions.

- A function that performs 3-phase PWM waveform output with a non-overlapping relationship between positive and negative phases (complementary PWM mode)
- A function that transfers buffer register (TGRC/D_3, TGRC/D_4) contents to compare registers (TGRA/B_3, TGRA/B_4) when a compare match occurs
- A function that outputs a toggle waveform synchronized with the PWM waveform period

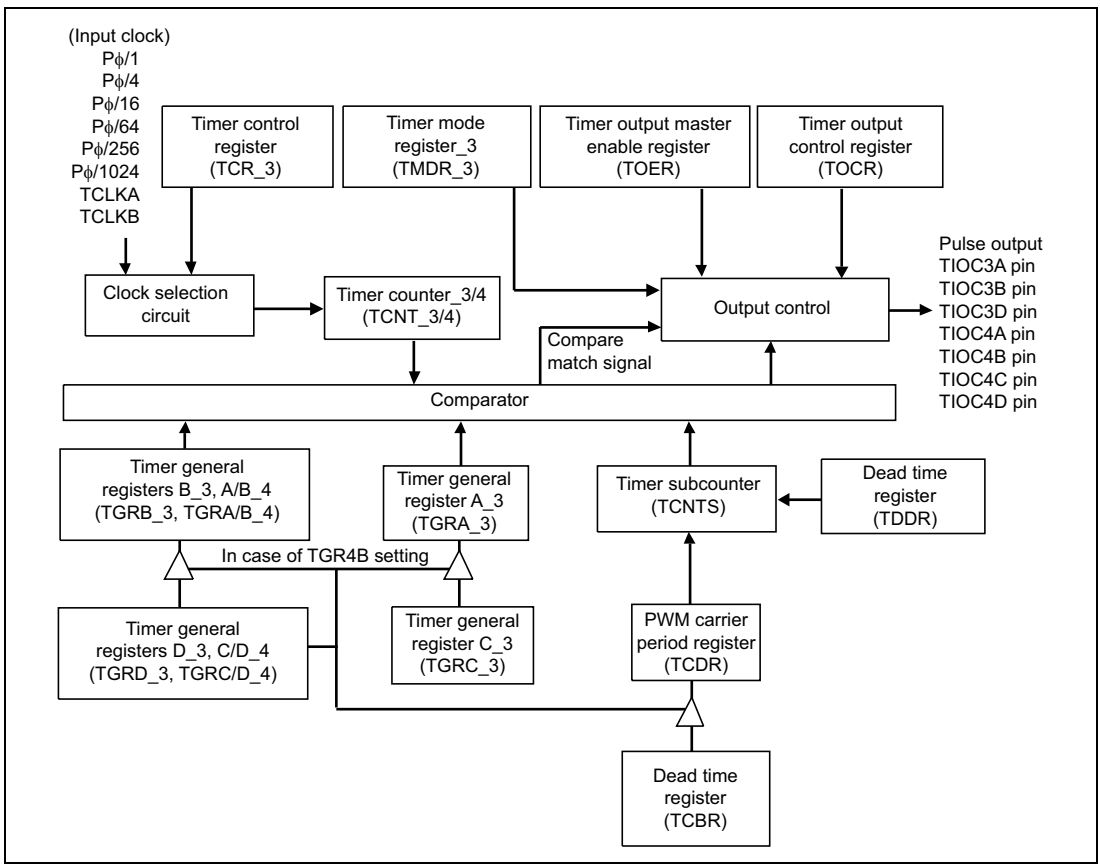


Figure 2.17 Block Diagram of MTU/ch3, ch4

(2) Table 2.6 shows the function assignments used in this task. PWM pulses are output by assigning MTU functions as shown in the table.

Table 2.6 Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC3A	Pin	Toggle output synchronized with PWM period
TIOC3C	Pin	PWM output 1
TIOC3D	Pin	Negative-phase waveform in non-overlapping relationship with PWM output 1
TIOC4A	Pin	PWM output 2
TIOC4B	Pin	PWM output 3
TIOC4C	Pin	Negative-phase waveform in non-overlapping relationship with PWM output 2
TIOC4D	Pin	Negative-phase waveform in non-overlapping relationship with PWM output 3
TOCR	Register	Enabling/disabling of toggle output synchronized with PWM period
TOER	Register	Complementary PWM output pin signal output enabling/disabling
TCR_3	Register	Selection of ch3 timer counter clearing source and input clock
TMDR_3	Register	Ch3, ch4 set to complementary PWM mode operation
TGRA_3	Register	Used to set value of 1/2 PWM period + dead time
TGRC_3	Register	TGRA_3 buffer register
TGRB_3	Registers	Output pulse transition point setting (compare register)
TGRA_4		
TGRB_4		
TGRD_3	Register	TGRB_3 buffer register
TGRC_4	Register	TGRA_4 buffer register
TGRD_4	Register	TGRB_4 buffer register
TDDR	Register	Dead time setting
TCDR	Register	Setting of 1/2 period
TCBR	Register	TCDR buffer register

Operation

(1) Figure 2.18 illustrates the principles of operation. Complementary PWM waveform output is performed by SH7046 hardware and software processing.

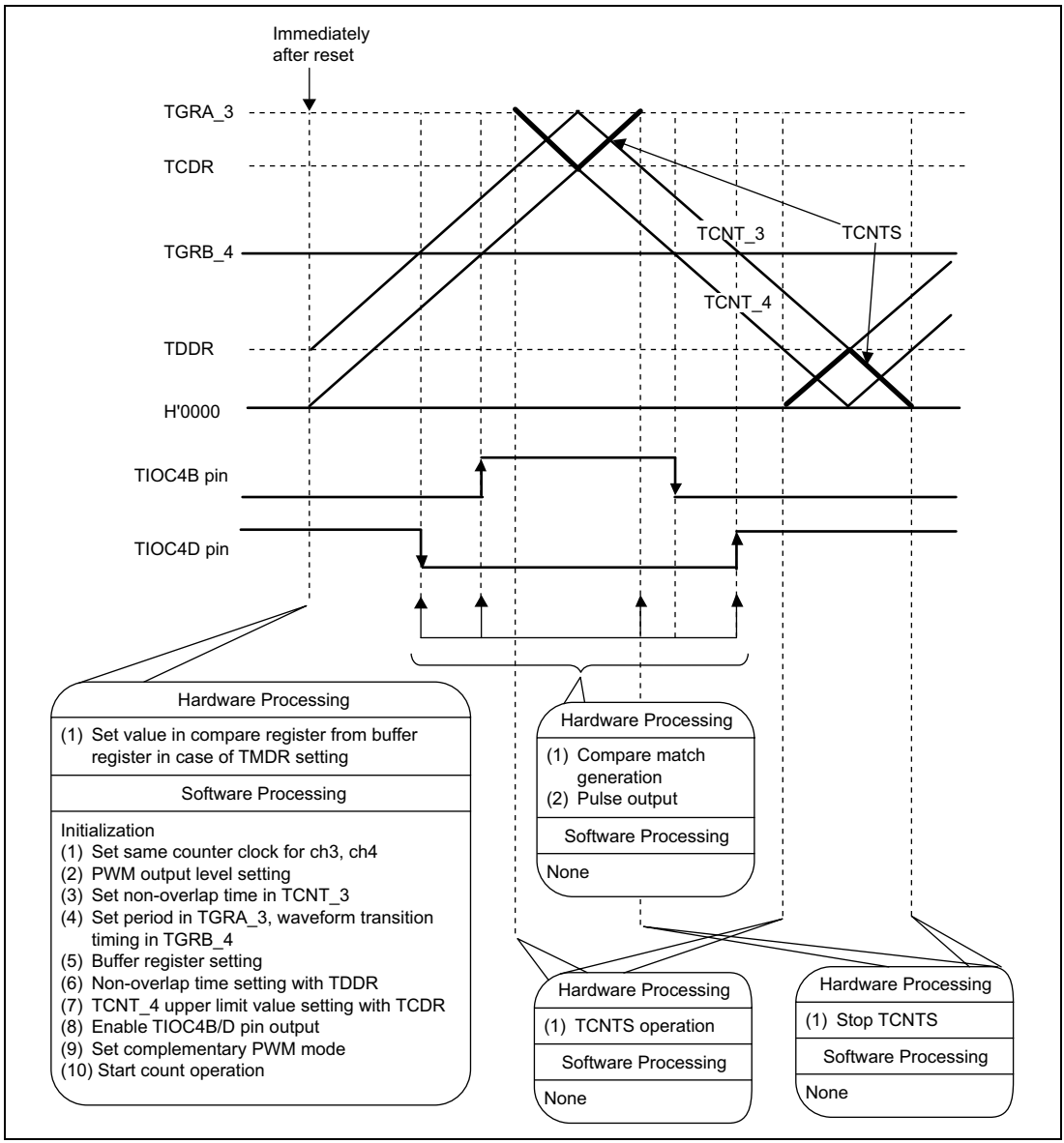


Figure 2.18 Principles of Operation of Complementary PWM Single-Phase Waveform Output

(2) Figure 2.19 shows the PWM waveform output method. When complementary PWM mode is set, the following rules apply to data transfer and compare operations.

Data Transfer

- In period Ta, data written to a buffer register (at the point at which data is set in TGRD_4) is transferred to a temporary register.
- In period Tb1, when the transfer mode is set to transfer at the peak, data is not transferred from a buffer register to a temporary register. In period Tb2, the operation is the same as in period Ta. Similarly, when a trough setting is made, data is not transferred in period Tb2.
- Data transfer to a buffer register can be performed arbitrarily.
- When period Tb ends, a value transferred to a temporary register is transferred to a compare match register. This transfer timing can be selected with timer mode register (TMDR) bits MD3 to MD0.

Compare Match

- In period Tb, two registers—the temporary register and compare register—and three counters—TCNT_3/4 and TCNTS—are compared, and the PWM waveform is controlled.
- In area (a), pre-change data and compare matches (3) and (4) have priority.
- In area (b), post-change data and compare matches (1) and (2) have priority.

Generation of a compare match whereby the output waveform goes to the active level (compare match (1) or (3)) occurs only after generation of a compare match whereby the respective output waveform goes to the positive level (compare match (4) or (2)).

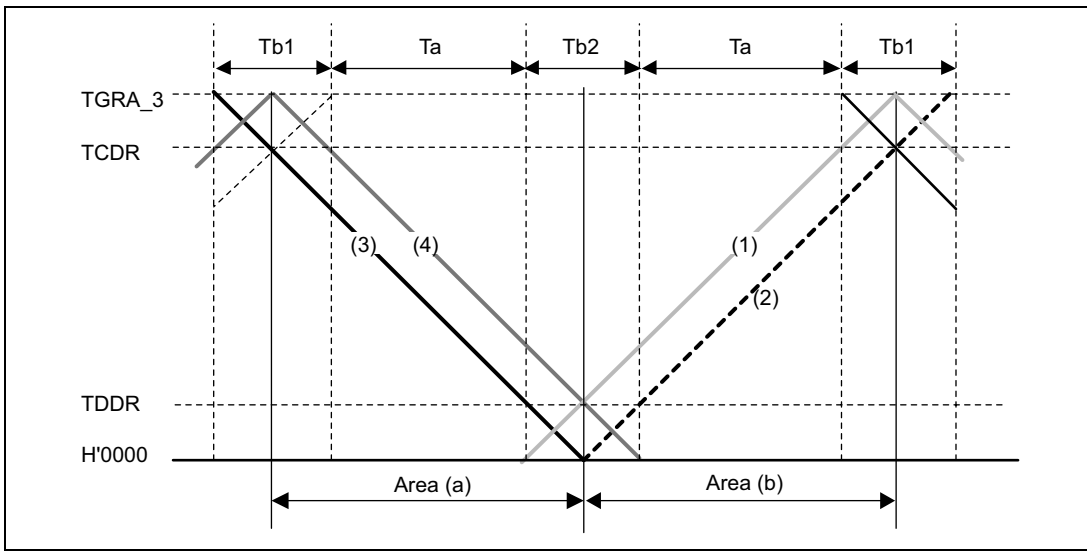


Figure 2.19 Principles of Operation of PWM Waveform Output Method

(3) Figure 2.20 illustrates the principles of operation. Complementary PWM waveform output is performed by SH7046 hardware and software processing. The transfer mode selected in this sample task is the mode in which data is changed at a peak.

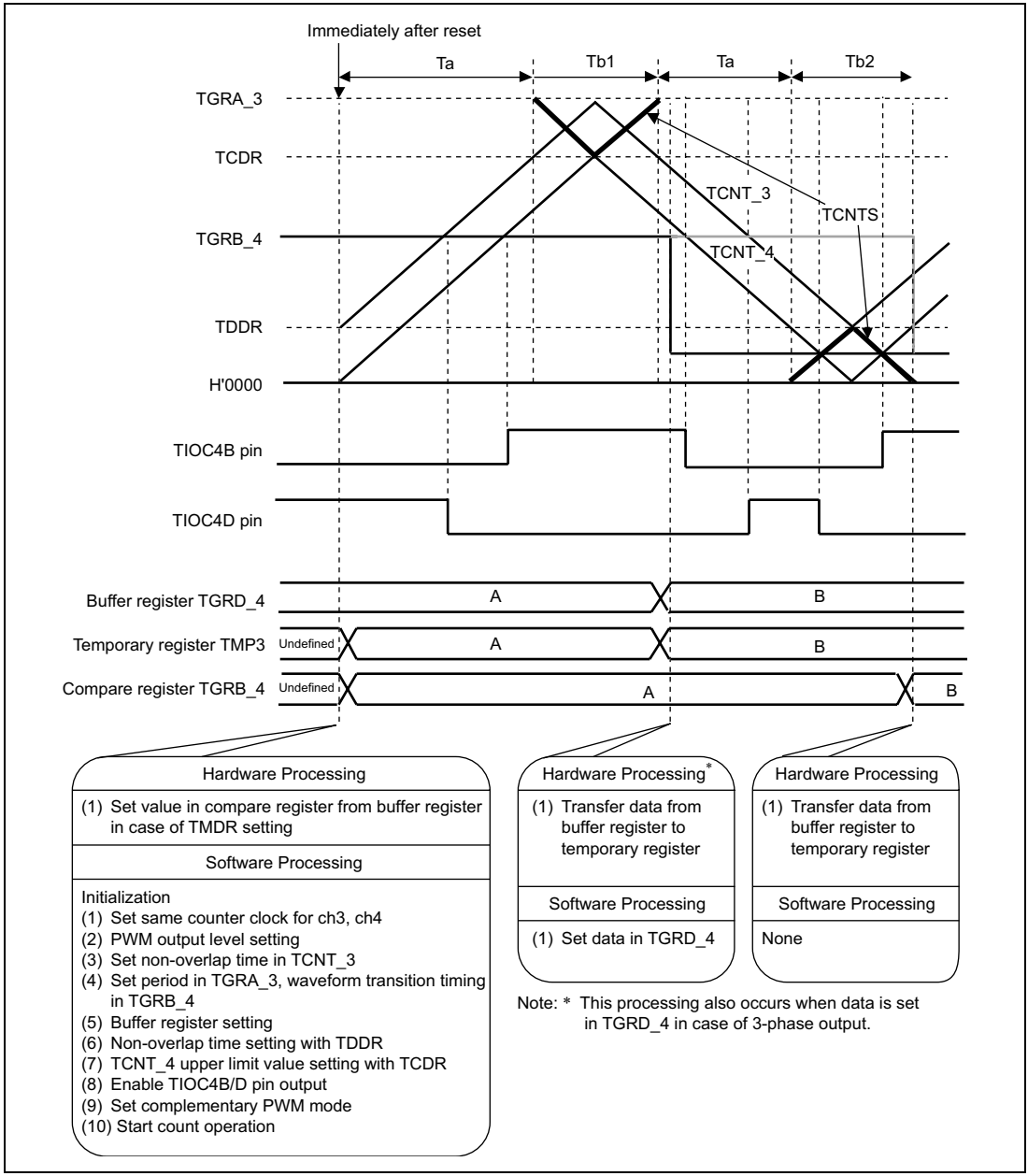


Figure 2.20 Principles of Operation of Complementary PWM Single-Phase Waveform Output

(4) Figure 2.21 illustrates the principles of operation. Three-phase PWM output is performed from the ch3 and ch4 PWM output pins (TIOC3B/D, TIOC4A/B/C/D) by SH7046 hardware and software processing as shown in the figure.

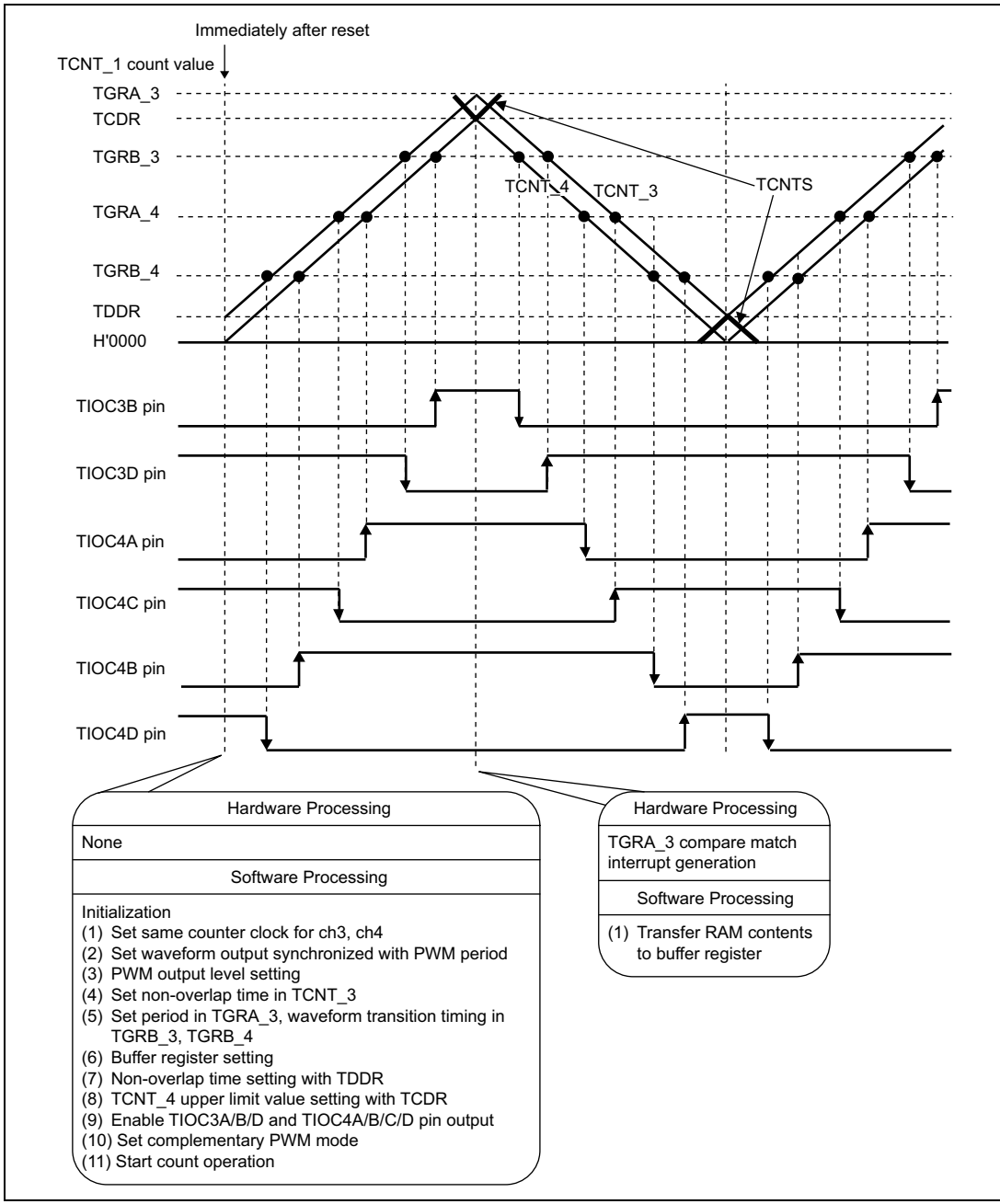


Figure 2.21 Principles of Operation of PWM Waveforms

(5) Figure 2.22 illustrates the principles of operation. Toggle output synchronized with the PWM period is performed by SH7046 hardware and software processing.

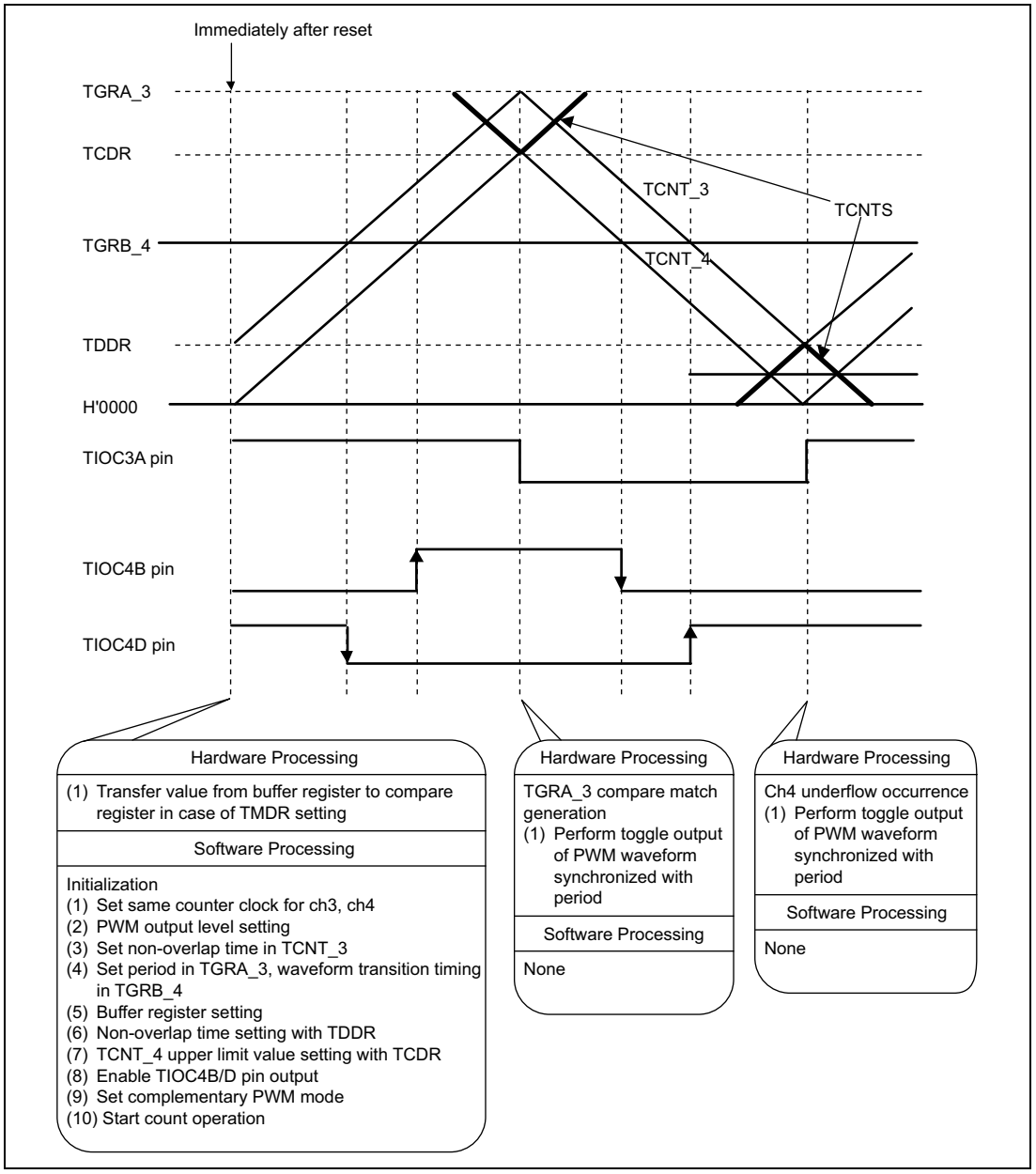


Figure 2.22 Principles of Operation of Toggle Waveform Output Synchronized with PWM Period

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	comple	Complementary PWM output setting
Data setting	setdata	Sets waveform transition timing in buffer register

(2) Arguments

Label or Register Name	Function	Data Length	Module	Input/ Output
pul_cyc1	Used to set pulse 1/2 period + dead time value Pulse period is calculated using following equation: Pulse period (ns) = timer value × ϕ period (50.0 ns at 20.0 MHz operation)	1 word	Main routine	Input
pul_duty3d	Used to set TIOC pin output waveform transition timing			
pul_duty4c				
pul_duty4d				
c_cyc	PWM carrier period register value setting			
dead_time	Non-overlap time setting		Main routine Data setting	

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	MTU module standby mode clearing, and setting of MTU to operational status	H'FFFF861E	H'd2fd
P_PORTE.PECRH P_PORTE.PECRL1 P_PORTE.PECRL2 P_PORTE.PEIORH P_PORTE.PEIORL	Used to set multiplex pins as MTU timer output pins TIOC3A, TIOC3B, TIOC3D, TIOC4A, TIOC4B, TIOC4C, TIOC4D	H'FFFF83BC H'FFFF83B8 H'FFFF83BA H'FFFF83B6 H'FFFF83B4	H'0000 H'5545 H'0000 H'0000 H'fb00
P_MTU34.TCR_3	Selects timer counter clearing source and input clock	H'FFFF8200	H'00
P_MTU34.TCR_4	Selects timer counter clearing source and input clock	H'FFFF8201	H'00
P_MTU34.TIER_3	Enables TGR3A interrupt	H'FFFF8208	H'01

Register Name	Function	Address	Set Value
P_MTU34.TGRA_3	Used to set 1/2 carrier period + dead time register value	H'FFFF8218	pul_cyc1
P_MTU34.TGRC_3	Used to set 1/2 carrier period + dead time register value	H'FFFF8224	pul_cyc1
P_MTU34.TGRB_3	Setting of PWM duty value of waveform output from TIOC3B, TIOC3D	H'FFFF821A	pul_duty3d
P_MTU34.TGRD_3	Setting of PWM duty value of waveform output from TIOC3B, TIOC3D	H'FFFF8226	pul_duty3d
P_MTU34.TGRA_4	Setting of PWM duty value of waveform output from TIOC4A, TIOC4C	H'FFFF821C	pul_duty4c
P_MTU34.TGRC_4	Setting of PWM duty value of waveform output from TIOC4A, TIOC4C	H'FFFF821C	pul_duty4c
P_MTU34.TGRB_4	Setting of PWM duty value of waveform output from TIOC4B, TIOC4D	H'FFFF821E	pul_duty4d
P_MTU34.TGRD_4	Setting of PWM duty value of waveform output from TIOC4B, TIOC4D	H'FFFF821E	pul_duty4d
P_MTU34.TCNT_3	Dead time value setting	H'FFFF8210	dead_time
P_MTU34.TDDR	Dead time value setting	H'FFFF8216	dead_time
P_MTU34.TCDR	Setting of upper limit value of timer counter TCNT_4 (1/2 carrier period)	H'FFFF8214	c_cyc
P_MTU34.TCBR	Setting of upper limit value of timer counter TCNT_4 (1/2 carrier period)	H'FFFF8222	c_cyc
P_MTU34.TOCR	Enabling of toggle output synchronized with PWM period, and positive-phase/negative phase output level setting	H'FFFF820B	H'43
P_MTU34.TOER	Complementary PWM output enabling setting	H'FFFF820A	H'ff
P_MTU34.TMDR_3	Complementary PWM mode setting	H'FFFF8202	H'ff
P_INTC.IPRE	Sets 15 as MTU channel 3 interrupt priority level	H'FFFF8350	H'00f0

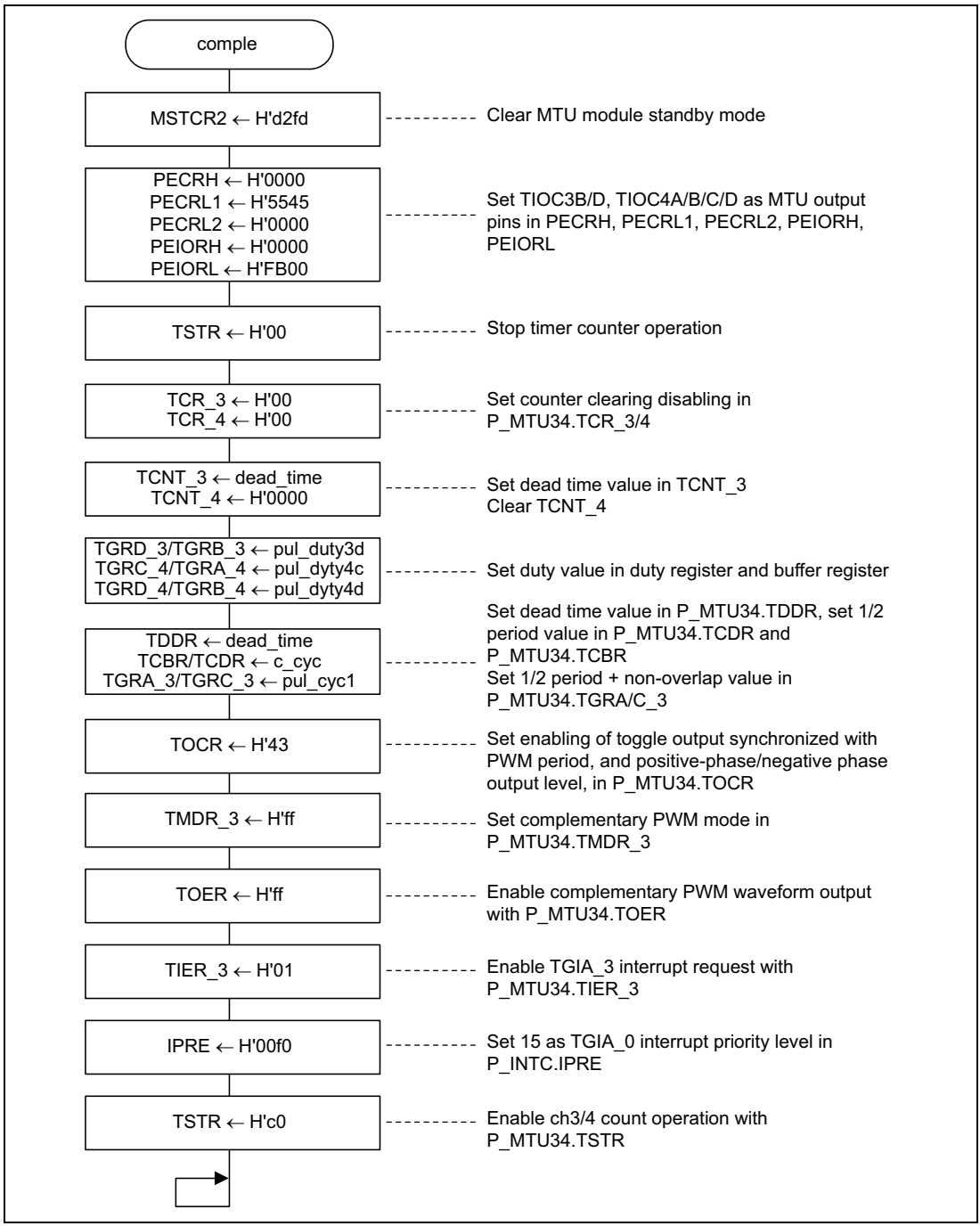
(4) RAM Used

This sample application does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/*------------------------------------------------------------------------*/
/*                              INCLUDE FILE                              */
/*------------------------------------------------------------------------*/
#include <machine.h>
#include "iodefine_7046.h"
/*------------------------------------------------------------------------*/
/*                                PROTOTYPE                               */
/*------------------------------------------------------------------------*/
extern void comple(void);
#pragma interrupt(setdata)
/*------------------------------------------------------------------------*/
/*                             RAM ALLOCATION                             */
/*------------------------------------------------------------------------*/
#define pul_cyc1             (*(unsigned short *)0xffffd000)
#define pul_duty3d           (*(unsigned short *)0xffffd002)
#define pul_duty4c           (*(unsigned short *)0xffffd004)
#define pul_duty4d           (*(unsigned short *)0xffffd006)
#define c_cyc                (*(unsigned short *)0xffffd008)
#define dead_time            (*(unsigned short *)0xffffd00a)
/*------------------------------------------------------------------------*/
/*                              MAIN PROGRAM                              */
/*------------------------------------------------------------------------*/
void comple(void)
 {
  P_STBY.MSTCR2 = 0xd2fd;            /* MTU module stop mode clear */
  P_PORTE.PECRH.WORD = 0x0000;       /* TIOC3A/B/D,TIOC4A/B/C/D output */
  P_PORTE.PECRL1.WORD = 0x5545;
  P_PORTE.PECRL2.WORD = 0x0000;
  P_PORTE.PEIORH.WORD = 0x0000;
  P_PORTE.PEIORL.WORD = 0xFB00;
  P_MTU34.TSTR.BYTE = 0x00;
  P_MTU34.TCR_3.BYTE = 0x00;         /* not clear */
  P_MTU34.TCR_4.BYTE = 0x00;         /* not clear */
  P_MTU34.TCNT_3 = dead_time;        /* initial data */
  P_MTU34.TCNT_4 = 0x0000;
  P_MTU34.TGRD_3 = pul_duty3d;       /* TGRD_3 buffer register */
  P_MTU34.TGRB_3 = pul_duty3d;       /* PWM output1 compare register */
  P_MTU34.TGRC_4 = pul_duty4c;       /* TGRA_4 buffer register */
  P_MTU34.TGRA_4 = pul_duty4c;       /* PWM output2 compare register */
  P_MTU34.TGRD_4 = pul_duty4d;       /* TGRB_4 buffer register */
  P_MTU34.TGRB_4 = pul_duty4d;       /* PWM output3 compare register */
  P_MTU34.TDDR = dead_time;          /* dead time set */
  P_MTU34.TCDR = c_cyc;              /* 1/2 carrer period */
  P_MTU34.TCBR = c_cyc;              /* TCDR buffer register */
  P_MTU34.TGRA_3 = pul_cyc1;         /* 1/2 carrer period + dead time */
  P_MTU34.TGRC_3 = pul_cyc1;         /* TGRA_3 buffer register */
  P_MTU34.TOCR.BYTE = 0x43;          /* timer output control register */
  P_MTU34.TMDR_3.BYTE = 0xff;        /* complementary-pwm mode */
  P_MTU34.TOER.BYTE = 0xff;          /* timer output enable register */
```

```
  P_MTU34.TIER_3.BYTE = 0x01;         /* timer interrupt enable register */
  P_INTC.IPRE.WORD = 0x00f0;          /* set interrput level = 15 */
  set_imask(0x0);                     /* set imask level = 0 */
  P_MTU34.TSTR.BYTE = 0xc0;           /* timer start */
  while(1);                           /* loop */
 }
void setdata()
 {
   P_MTU34.TSR_3.BYTE &= 0xfe;        /* interrupt flag clear */
   P_MTU34.TCBR = c_cyc;
   P_MTU34.TGRC_3 = pul_cyc1;
   P_MTU34.TGRD_3 = pul_duty3d;
   P_MTU34.TGRC_4 = pul_duty4c;
   P_MTU34.TGRD_4 = pul_duty4d;
 }
```

2.7 2-Phase Encoder Count

2-Phase Encoder Count	MCU: SH7046/47	Functions Used: MTU (Phase Counting Mode)

Specifications

(1) Two external clocks are input to channel 1 (ch1), and a counter is incremented or decremented according to the phase difference of the pulses, as shown in figure 2.23. The ch1 count is measured in synchronization with measurement times set in ch0 (measurement times 1 and 2), and the result is set in RAM.

(2) H'0000 is set as the timer counter initial value, and counting can be performed from -2,147,483,648 to 2,147,483,647 using a software counter.

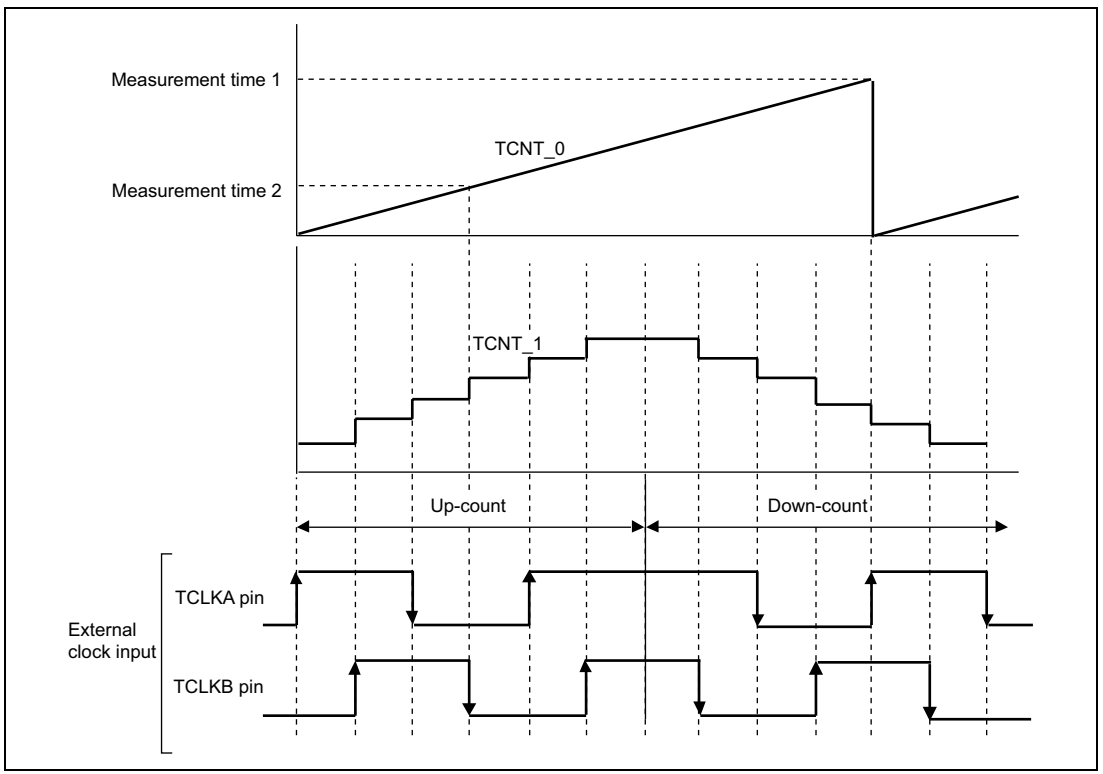


Figure 2.23 2-Phase Encoder Counter Capture

Functions Used

(1) In this sample task, measurement times are set in TGRA/B_0 using an MTU ch1 up/down-counter.
Using a TGRA/B_0 output compare as a trigger, the TCNT_1 value for the control period is captured by ch1 input capture. In addition, the ch1 counter input clock width is captured using ch0 input capture.

(a) Figure 2.24 shows a block diagram of ch0. In ch0, a ch1 input capture trigger is output every measurement time using the following functions. In ch1, the TCNT_1 value is measured when an input capture signal is input.

- A function that outputs pulses automatically by hardware without software intervention (output compare)
- A function that performs pulse input edge detection, and captures a timer value in an internal register (input capture)

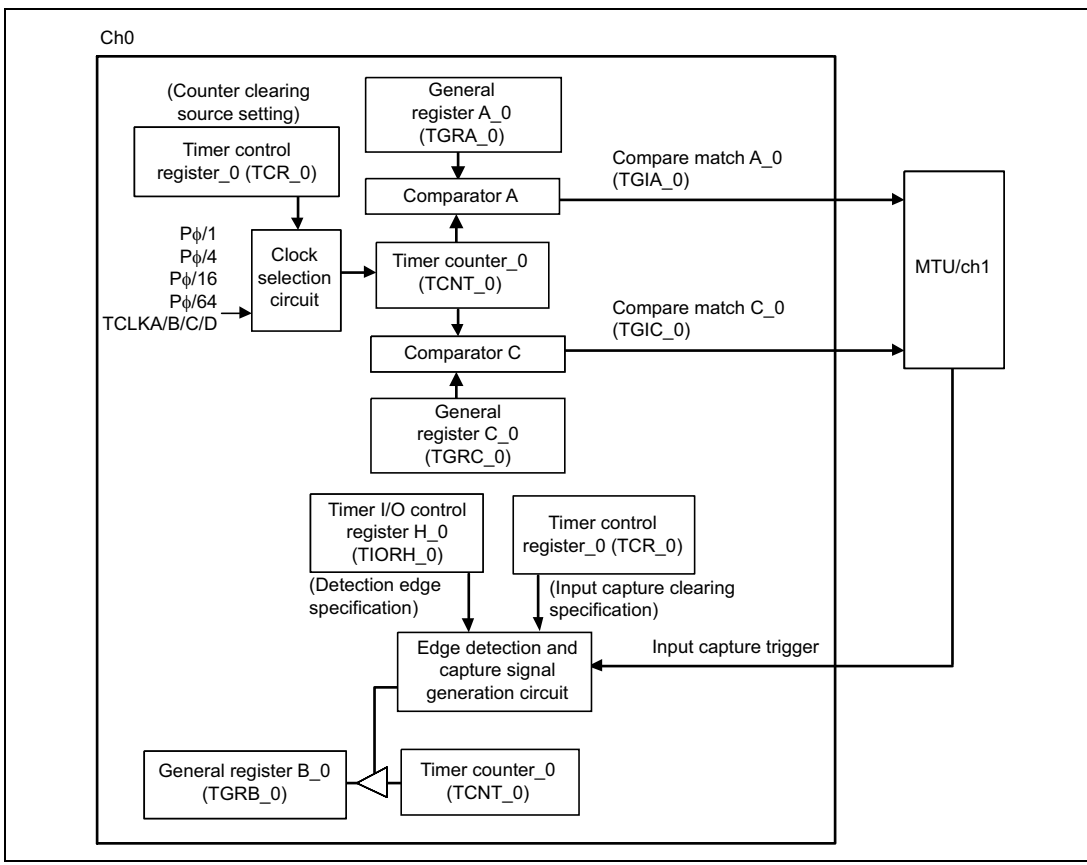


Figure 2.24 Block Diagram of MTU/ch0

(b) Figure 2.25 shows a block diagram of ch1. In ch1, a timer counter is incremented/decremented using the following functions. The counter value when an input capture signal rising edge is detected is taken as the measurement result.

- A function that detects the phase difference between two external clocks, and increments/decrements a timer counter (phase counting mode)
- A function that performs pulse input edge detection, and captures the timer value at that point in an internal register (input capture)
- A function that initiates interrupt handling when input capture occurs
- A function that clears the timer counter when a pulse input edge is detected
- A function that initiates interrupt handling when timer counter overflow or underflow is detected

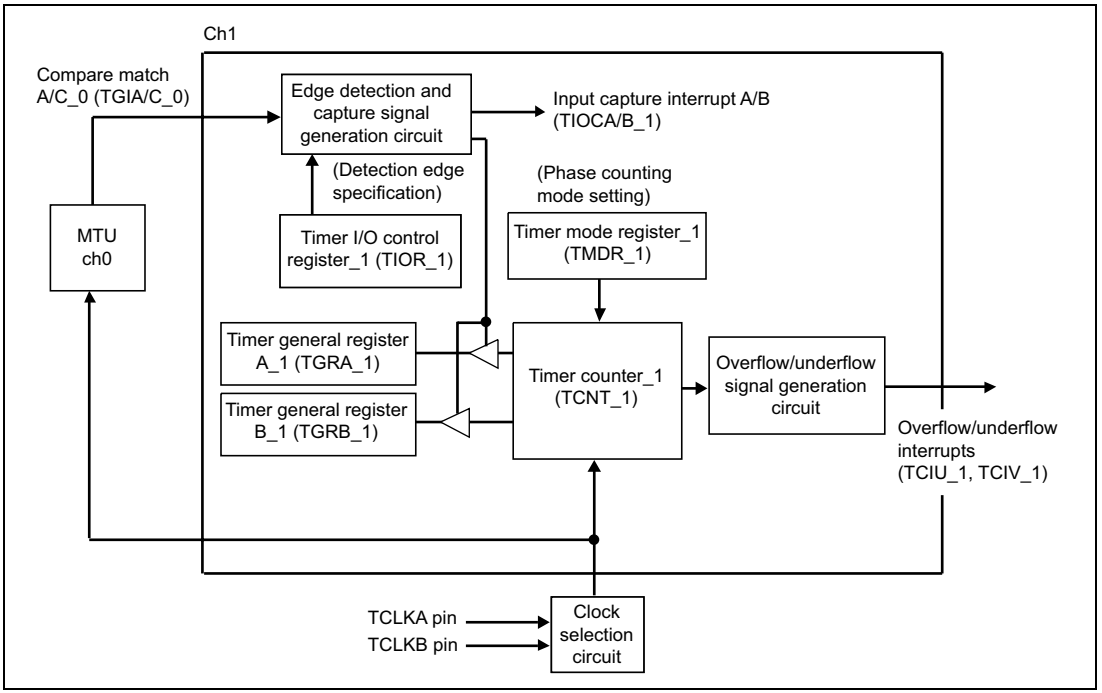


Figure 2.25 Block Diagram of MTU/ch1

(2) Table 2.7 shows the function assignments used in this sample task. MTU functions are assigned as shown in the table to detect the phase difference between two 2-phase encoder pulses, and increment/decrement a counter.

Table 2.7 Function Assignments

Pin or Register Name	Function	Function Assignment
TCLKA	Pin	External clock input pins
TCLKB	Pin	
TSTR	Register	Enabling/disabling of ch0, ch1 timer counter operation
TCR_0	Register	Selection of counter clock and counter clearing source
TIORH_0	Register	TIOC0A output compare setting. Setting of TIOC0B for input capture on ch0 output compare occurrence
TIORL_0	Register	TIOC0C output compare setting
TGRA_0	Register	Measurement time 1 setting
TGRB_0	Register	Count result stored on input capture B
TGRC_0	Register	Measurement time 2 setting
TMDR_1	Register	Phase counting mode setting
TCR_1	Register	Selection of counter clock and counter clearing source
TIOR_1	Register	Setting of TIOC1A/C for input capture on ch1 output compare occurrence
TIER_1	Register	Enables TIOC1A/B, TCIU_1, TCIV_1 interrupts
TGRA_1	Register	Count result storage on input capture A
TGRB_1	Register	

Operation

(1) Figure 2.26 illustrates the principles of operation. A counter is incremented or decremented by SH7046 hardware and software processing.

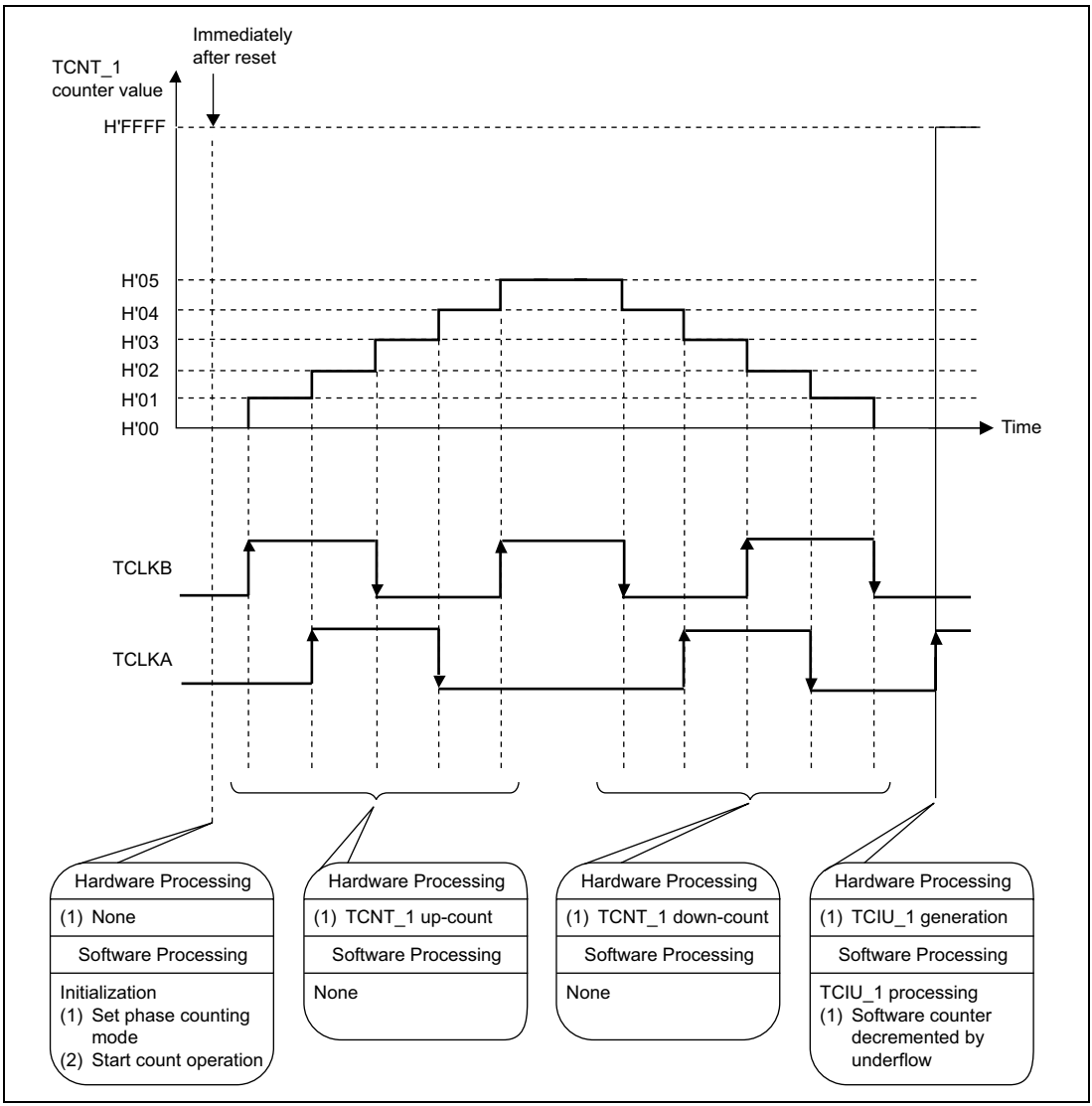


Figure 2.26 Principles of Operation in Phase Counting Mode (1)

(2) Interrupt handling is executed on external event occurrence by means of SH7046 hardware and software processing as shown in figure 2.27.

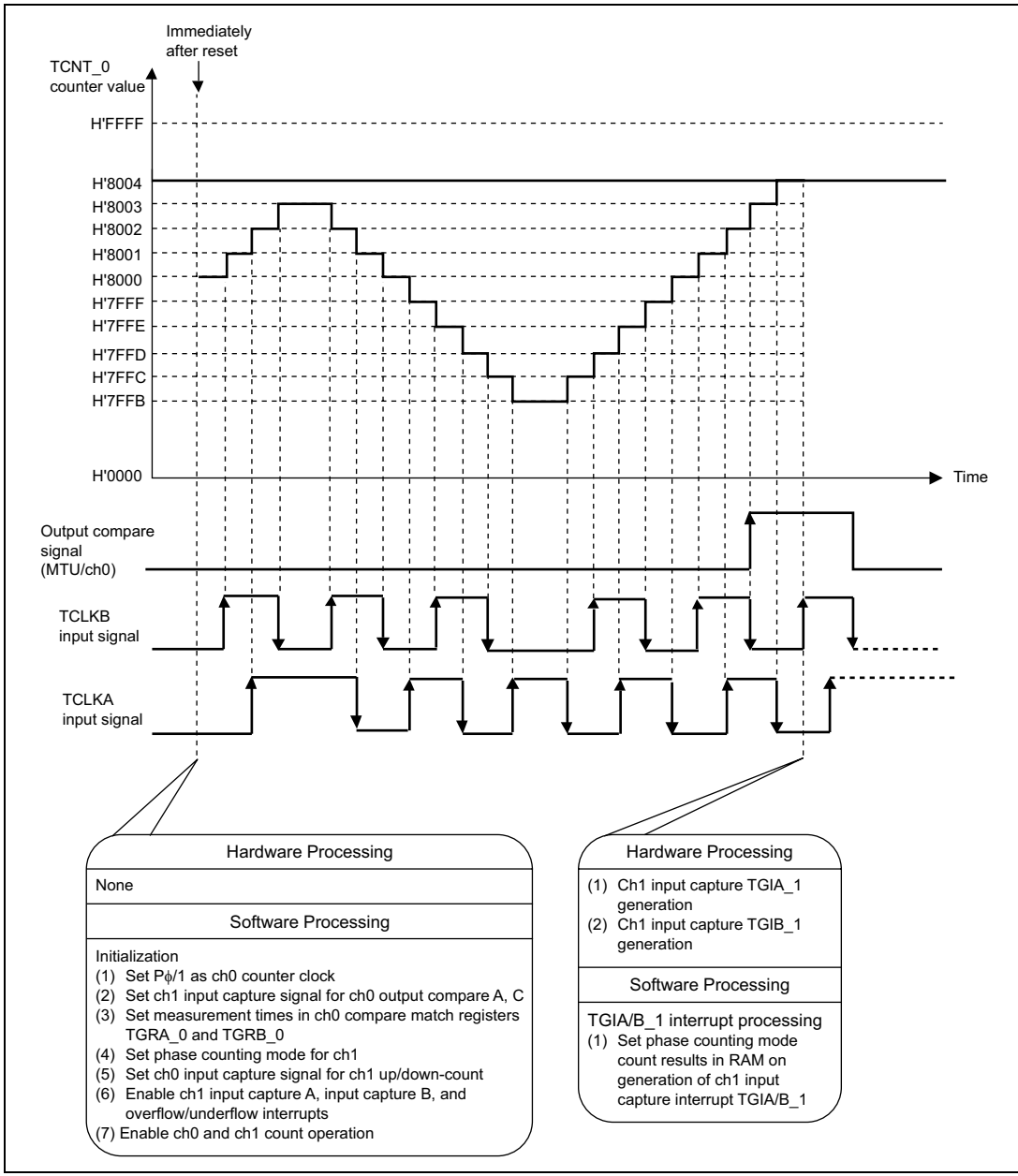


Figure 2.27 Principles of Operation in Phase Counting Mode (2)

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	en2	Initialization of MTU, etc.
Counter value measurement 1	phacnt1	Initiated by TGIA_1. Sets up/down-count result in RAM based on TGRA value. Sets counter period result in RAM based on TGRC value
Counter value measurement 2	phacnt2	Initiated by TGIB_1. Sets up/down-count result in RAM based on TGRB value
Overflow	ovf1	Initiated by TCIV_1. Software counter incrementing
Underflow	unf1	Initiated by TCIU_1. Software counter decrementing

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
msr_tim1 msr_tim2	Used to set timer value for counter measurement time Measurement time is calculated using following equation: Measurement time (ns) = timer value $\times$ ϕ period (50.0 ns at 20.0 MHz operation)	Word	Main routine	Input
cnt_data1 cnt_data2	Used to set up/down-count results	Longword	Counter value measurement 1 Counter value measurement 2	Output
p_cycle	Used to set count period result	Word	Counter value measurement 2	

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	MTU module standby mode clearing, and setting of MTU to operational status	H'FFFF861E	H'd2fd
P_PORTA.PACRL2 P_PORTA.PACRL3	Used to set multiplex pins as timer pins TCLKA, TCLKB	H'FFFF838E H'FFFF838A	H'5000 H'0000
P_MTU0.TCR_0	Selection of counter clock and counter clearing source	H'FFFF8260	H'20
P_MTU0.TIORH_0	TIOC0A output compare setting. Setting of TIOBC0B for input capture on ch0 output compare	H'FFFF8262	H'f0
P_MTU0.TIORL_0	TIOC0C output compare setting	H'FFFF8263	H'00
P_MTU0.TGRA_0	Measurement time 1 setting	H'FFFF8268	msr_tim1
P_MTU0.TGRC_0	Measurement time 2 setting	H'FFFF826C	msr_tim2
P_MTU1.TMDR_1	Phase counting mode setting	H'FFFF8281	H'04
P_MTU0.TMDR_0	Sets buffer operation for GRD	H'FFFF8261	H'20
P_MTU1.TIOR_1	Setting of TIOC0A/C for input capture on ch1 output compare occurrence	H'FFFF8282	H'ff
P_MTU1.TIER_1	Enables interrupts by TGIA/B_1, TCIU_1, TCIV_1	H'FFFF8284	H'33
P_MTU34.TSTR	Starts ch0, ch1 timer count	H'FFFF8240	H'03
P_INTC.IPRD	Sets 15 as MTU0, MTU1 interrupt priority level	H'FFFF834E	H'00ff

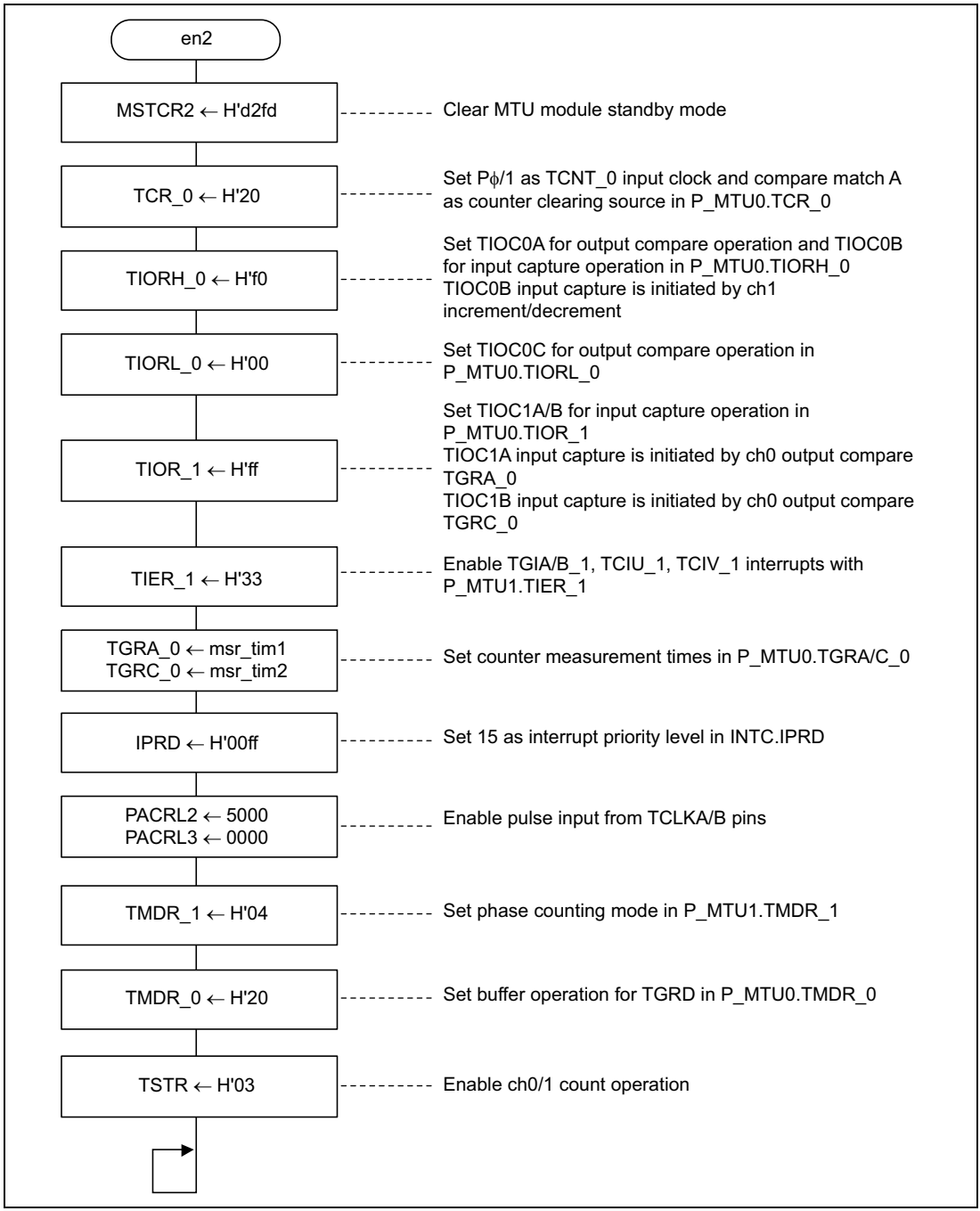
(4) RAM Used

Module	Label	Function Assignment
Counter value measurement 1, 2	wrk	Used as work area for data setting
All modules	cnt	Software counter

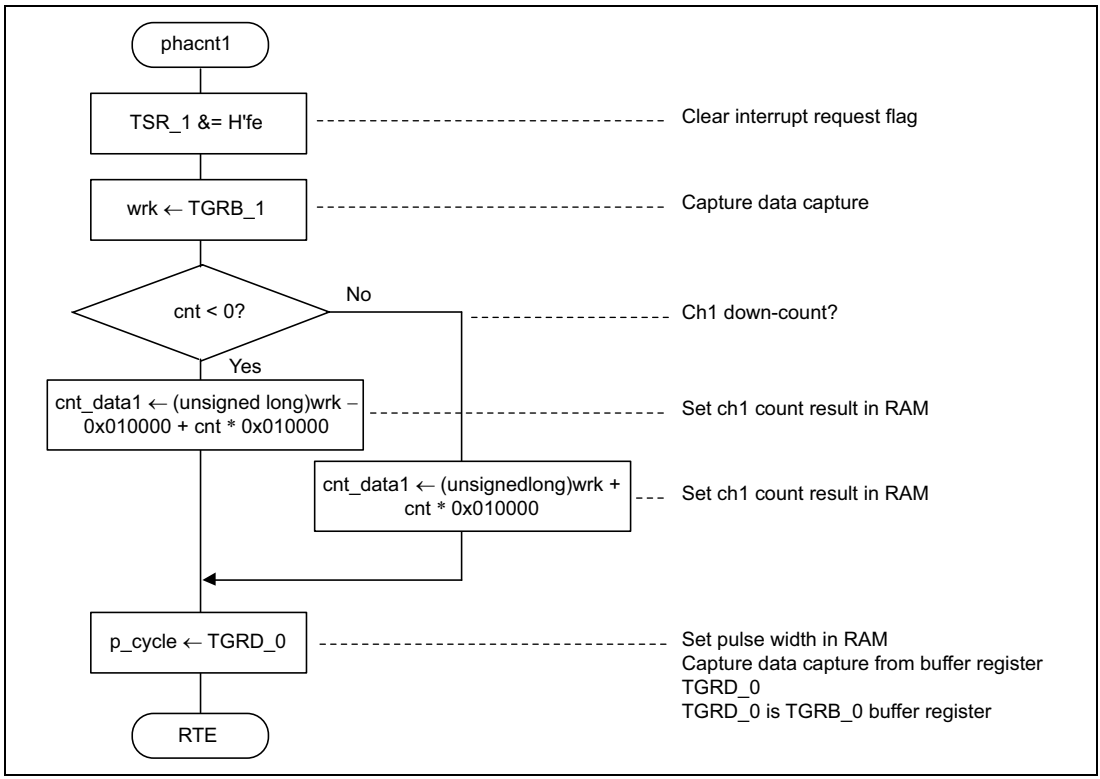
Note: SH7046 header file names are used for register label names.

Flowcharts

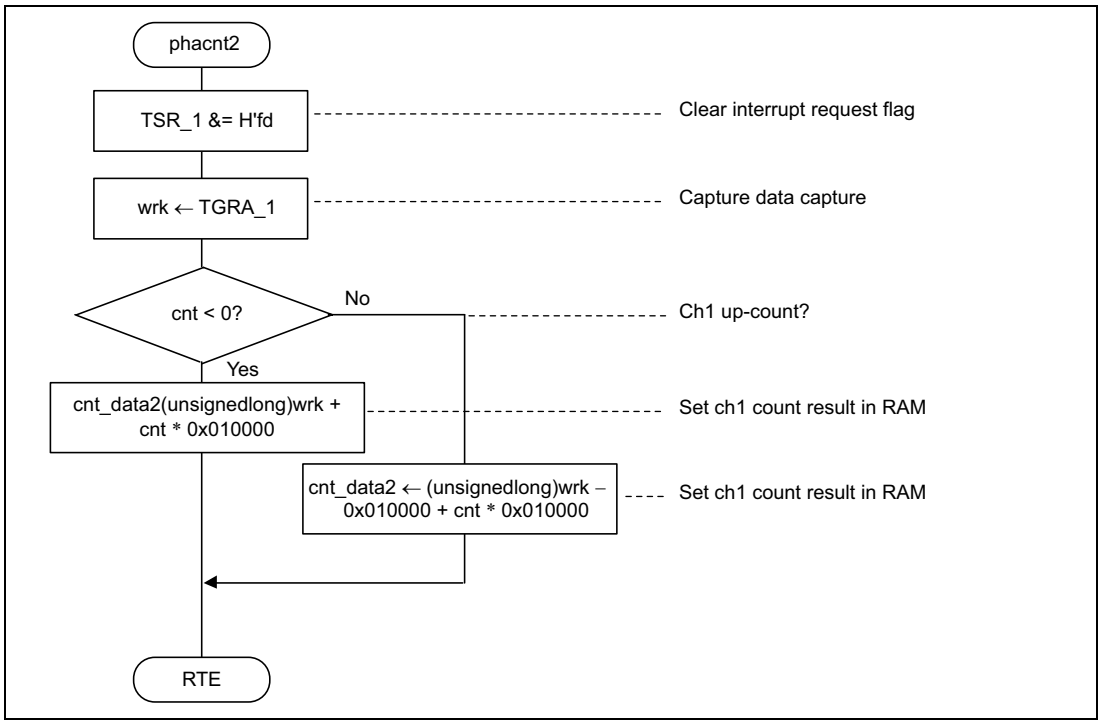
(1) Main routine



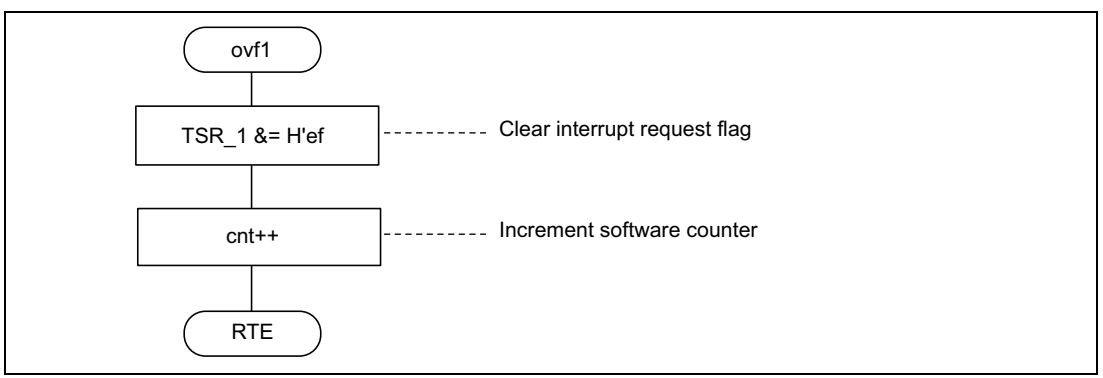
(2) Counter value measurement 1 (ch1 input capture B interrupt)



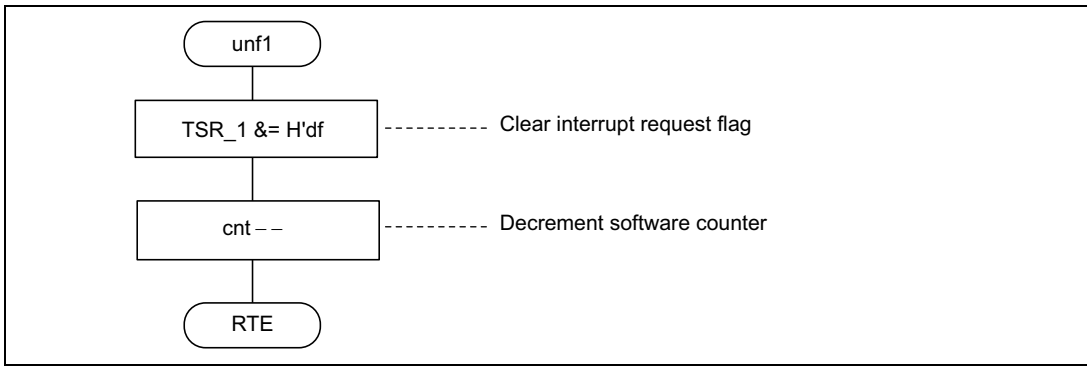
(3) Counter value measurement 2 (ch1 input capture B interrupt)



(4) Ch1 overflow interrupt



(5) Ch1 underflow interrupt



Program Listing

```
/*---------------------------------------------------------------------*/
/*                           INCLUDE FILE                              */
/*---------------------------------------------------------------------*/
#include <machine.h>
#include "iodefine_7046.h"
/*---------------------------------------------------------------------*/
/*                            PROTOTYPE                                */
/*---------------------------------------------------------------------*/
void en2(void);
#pragma interrupt(phacnt1,phacnt2,ovf1,unf1)
/*---------------------------------------------------------------------*/
/*                          RAM ALLOCATION                             */
/*---------------------------------------------------------------------*/
#define msr_tim1      (*(unsigned short *)0xffffd000)
#define msr_tim2      (*(unsigned short *)0xffffd002)
#define cnt_data2     (*(signed long *)0xffffd004)
#define cnt_data1     (*(signed long *)0xffffd008)
#define p_cycle       (*(unsigned long *)0xffffd00c)
#define cnt           (*(signed long *)0xffffd010)
#define wrk           (*(unsigned short *)0xffffd014)
/*---------------------------------------------------------------------*/
/*                          MAIN PROGRAM                               */
/*---------------------------------------------------------------------*/
void en2(void)
 {
    P_STBY.MSTCR2.WORD = 0xd2fd;      /* MTU module stop mode clear */

    P_MTU0.TCR_0.BYTE = 0x20;         /* timer clear output compare TGRA_0 */
    P_MTU0.TIORH_0.BYTE = 0xf0;       /* output compare TIOC0A */
                                      /* input capture TIOC0B */
    P_MTU0.TIORL_0.BYTE = 0x00;       /* output compare TIOC0C */
    P_MTU1.TIOR_1.BYTE = 0xff;        /* input capture TIOC1A,B */

    P_MTU1.TIER_1.BYTE = 0x33;        /* interrupt TIOC1A,TIOC1B,TCIU1,TCIV1 */
    P_MTU0.TGRC_0 = msr_tim2;         /* set position cycle */
    P_MTU0.TGRA_0 = msr_tim1;         /* set speed cycle */
    INTC.IPRD.WORD = 0x00ff;          /* set interrupt level=15 */

    P_PORTA.PACRL2.WORD = 0x5000;     /* TCLKA,TCLKB sellect */
    P_PORTA.PACRL3.WORD = 0x0000;

    P_MTU1.TMDR_1.BYTE = 0x04;        /* set phase counting mode1 */
    P_MTU0.TMDR_0.BYTE = 0x20;        /* TGRD buffer mode  */
    P_MTU34.TSTR.BYTE = 0x03;         /* start timer 0,1*/

    set_imask(0x0);                   /* set imask level=0 */
    while(1);                         /* loop*/
 }
```

```
void ovf1(void)
 {
   P_MTU1.TSR_1.BYTE &= 0xef;                                  /* clear flag */
   cnt++;                                                      /* count up */
 }

void unf1(void)
 {
   P_MTU1.TSR_1.BYTE &= 0xdf;                                  /* clear flag */
   cnt--;                                                      /* count down */
 }

void phacnt1(void)
 {
   P_MTU1.TSR_1.BYTE &= 0xfe;                                  /* clear flag */
   wrk = P_MTU1.TGRB_1;
   if(cnt < 0)                                                 /* count < 0 */
    cnt_data1 = (unsigned long)wrk-0x010000+cnt*0x010000;      /* set sp */
   else
   cnt_data1 = (unsigned long)wrk+cnt*0x010000;                /* set sp */
    p_cycle = P_MTU0.TGRD_0;                                   /* set width pulse */

 }

void phacnt2(void)
 {

   P_MTU1.TSR_1.BYTE &= 0xfd;                                  /* clear flag */
   wrk = P_MTU1.TGRA_1;
   if(cnt < 0)
    cnt_data2 = (unsigned long)wrk+cnt*0x010000;               /* set po */
     else
     cnt_data2 = (unsigned long)wrk-0x010000+cnt*0x010000;     /* set po */
 }
```

2.8 Externally Triggered Timer Waveform Cutoff

Externally Triggered Timer Waveform Cutoff	MCU: SH7046/47	Functions Used: MTU, POE

Specifications

(1) Timer output waveform cutoff is performed by driving timer output waveforms to the high-impedance state in synchronization with the falling edge of an external signal, as shown in figure 2.28.

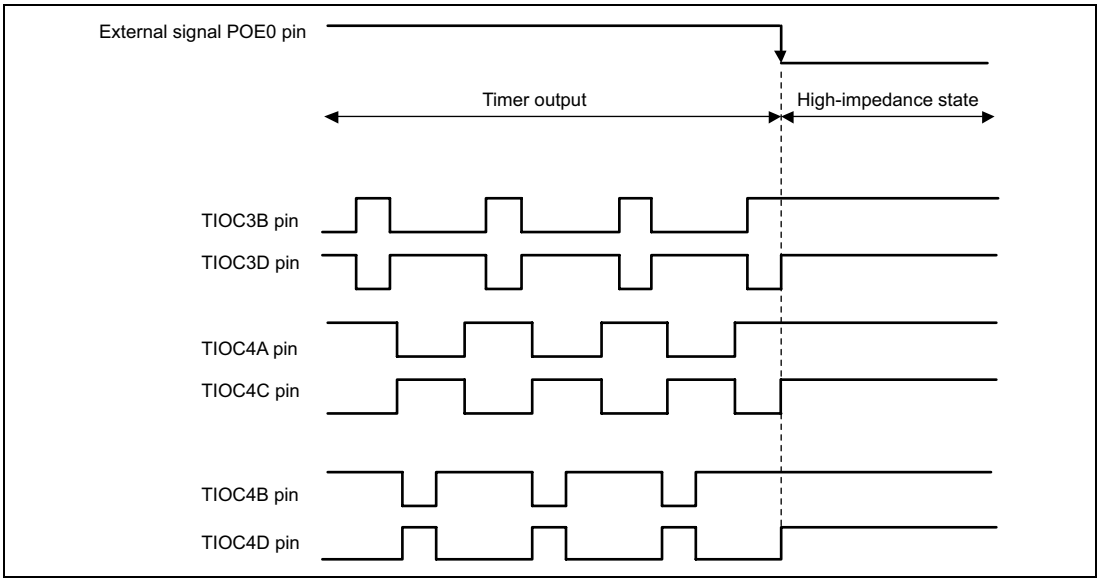


Figure 2.28 Example of Externally Triggered Waveform Cutoff

Functions Used

(1) In this sample task, waveforms output by MTU ch3/4 (reset-synchronized PWM mode) are cut by being driven to the high-impedance state in synchronization with the falling edge of an external signal.

(a) Figure 2.29 shows a block diagram of MTU/ch3 and ch4, and the POE.

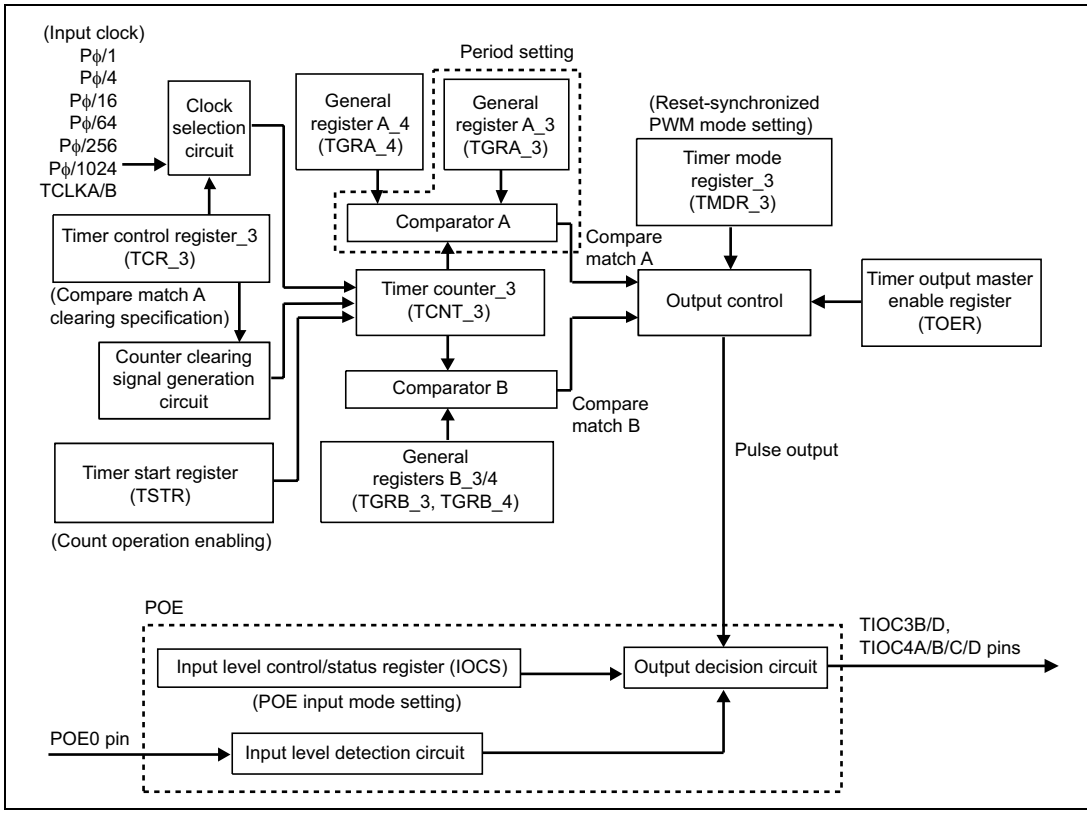


Figure 2.29 Block Diagram of MTU/ch3, ch4, and POE

(2) Table 2.8 shows the function assignments used in this task. Waveform cutoff is performed by assigning MTU and POE functions as shown in the table.

Table 2.8 Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC3B	Pins	Pulse output pins
TIOC3D		
TIOC4A		
TIOC4B		
TIOC4C		
TIOC4D		
POE0	Pin	Waveform cutoff external signal input pin
TSTR_3	Register	Enabling/disabling of ch3 timer counter operation
TCR_3	Register	Selection of ch3 timer counter clearing source and input clock
TMDR_3	Register	Sets reset-synchronized PWM mode for ch3, ch4
TGRA_3	Register	PWM period setting
TGRB_3	Registers	Output waveform transition timing setting
TGRA_4		
TGRB_4		
TOER	Register	Enabling/disabling of TIOC3B/D and TIOC4A/B/C/D pin timer output
ICSR	Register	POE input mode selection

Operation

(1) Figure 2.30 illustrates the principles of operation of this sample task. Waveform cutoff is performed automatically by hardware. (See section 2.5, Positive-Phase/Negative Phase PWM 3-Phase Output, in this Application Note for the principles of reset-synchronized PWM operation.)

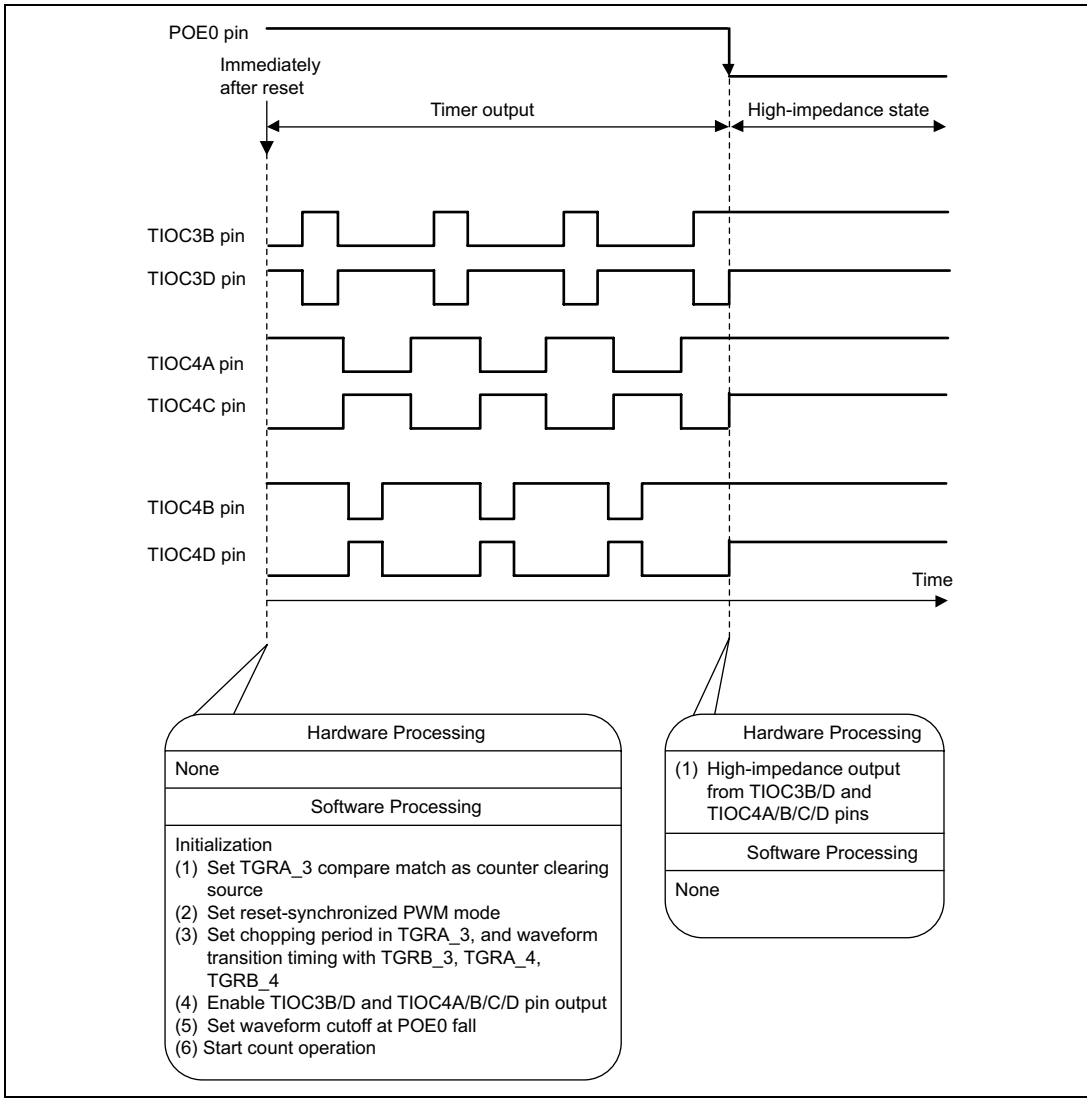


Figure 2.30 Principles of Operation of Externally Triggered Waveform Cutoff

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	down	DC motor control waveform generation

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
cycle	PWM period setting	1 word	Main routine	Input
duk1	Used to set TIOC3B/D output waveform transition timing			
duk2	Used to set TIOC4A/C output waveform transition timing			
duk3	Used to set TIOC4B/D output waveform transition timing			

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	MTU module standby mode clearing, and setting of MTU to operational status	H'FFFF861E	H'd2fd
P_PORTE.PEIORL	Sets TIOC3B/D, TIOC4A/B/C/D as output pins	H'FFFF83B4	H'fa00
P_PORTE.PECRL1	Sets TIOC3B/D, TIOC4A/B/C/D as MTU output pins	H'FFFF83B8	H'5544
P_PORTB.PBCR2	Sets POE0 pin	H'FFFF839A	H'0020
P_MTU34.TCR_3	Selection of timer counter clearing source and input clock	H'FFFF8200	H'20
P_MTU34.TOCR	Enabling of toggle output synchronized with PWM period, and positive-phase/negative-phase output level setting	H'FFFF820B	H'00
P_MTU34.TGRA_3	PWM period setting	H'FFFF8218	cycle
P_MTU34.TGRB_3	Used to set TIOC3B, TIOC3D output waveform transition timing	H'FFFF821A	duk1
P_MTU34.TGRA_4	Used to set TIOC4A, TIOC4C output waveform transition timing	H'FFFF821C	duk2
P_MTU34.TGRB_4	Used to set TIOC4B, TIOC4D output waveform transition timing	H'FFFF821E	duk3
P_MTU34.TOER	Sets TIOC3B/D, TIOC4A/B/C/D as MTU output pins	H'FFFF820A	H'ff
P_MTU34.TMDR_3	Sets reset-synchronized PWM mode	H'FFFF8202	H'c8
P_MTU.ICSR1	Sets high-impedance output synchronized with falling edge of POE0 pin input signal	H'FFFF83C0	H'0000

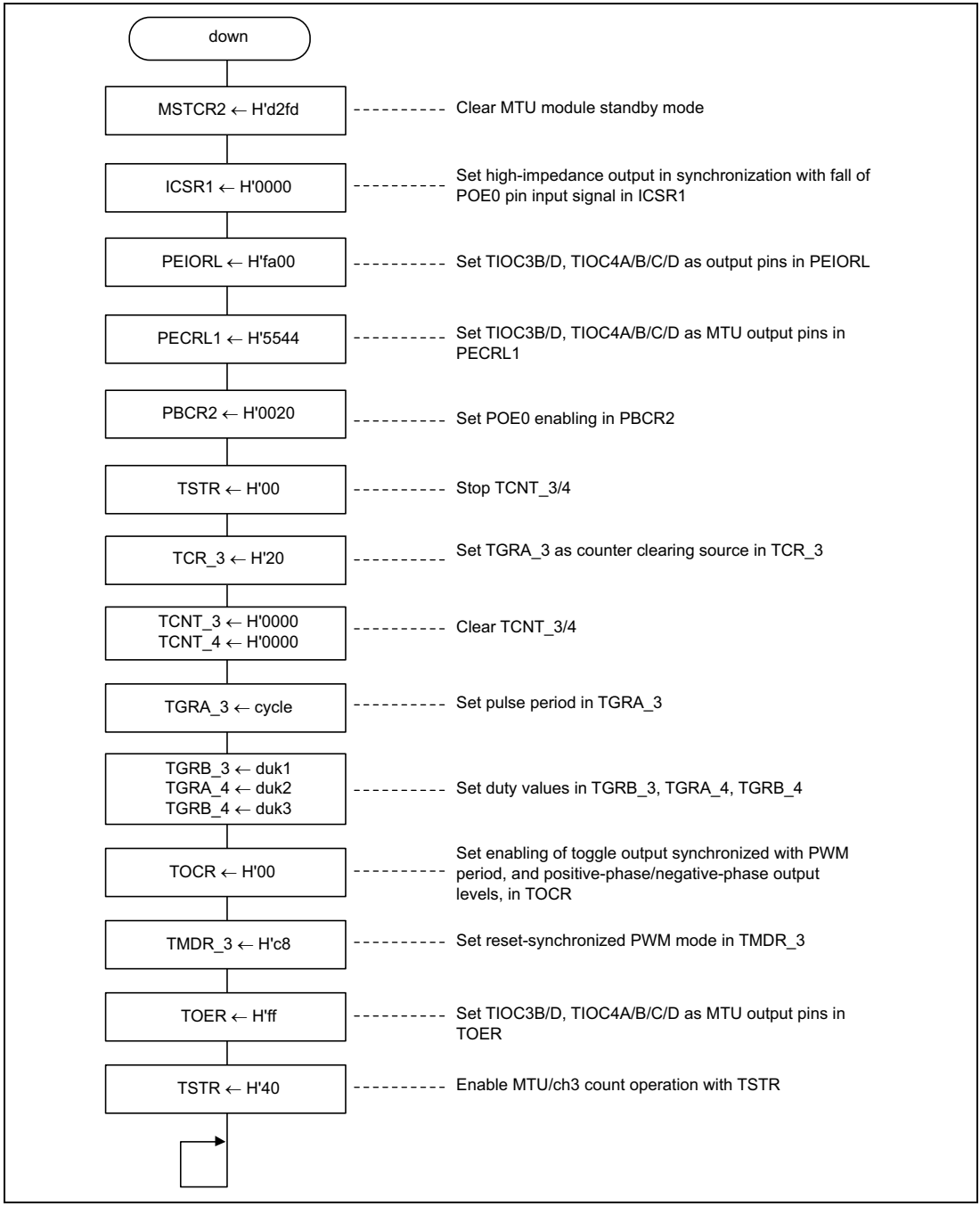
(4) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/*---------------------------------------------------------------------*/
/*                            INCLUDE FILE                             */
/*---------------------------------------------------------------------*/
#include <machine.h>
#include "iodefine_7046.h"
/*---------------------------------------------------------------------*/
/*                            PROTOTYPE                                */
/*---------------------------------------------------------------------*/
void down(void);
/*---------------------------------------------------------------------*/
/*                         RAM ALLOCATION                              */
/*---------------------------------------------------------------------*/
#define cycle   (*(unsigned short *)0xffffd000)
#define duk1    (*(unsigned short *)0xffffd002)
#define duk2    (*(unsigned short *)0xffffd004)
#define duk3    (*(unsigned short *)0xffffd006)
/*---------------------------------------------------------------------*/
/*                            MAIN PROGRAM                             */
/*---------------------------------------------------------------------*/
void down(void)
{
  P_STBY.MSTCR2.WORD = 0xd2fd;        /* MTU module stop mode clear */

  P_PORTE.PEIORL.WORD = 0xfa00;       /* TIOC3B/D,TIOC4A/B/C/D output */
  P_PORTE.PECRL1.WORD = 0x5544;       /* TIOC3B/D,TIOC4A/B/C/D output */
  P_PORTB.PBIOR.WORD = 0x0000;        /* POE enable */
  P_PORTB.PBCR1.WORD = 0x0000;        /* POE enable */
  P_PORTB.PBCR2.WORD = 0x0020;        /* POE enable */

  P_MTU.ICSR1.WORD = 0x0000;          /* stop timer POE0 falling edge */
  P_MTU.OCSR.WORD = 0x0000;

  P_MTU34.TSTR.BYTE = 0x00;
  P_MTU34.TCR_3.BYTE = 0x20;          /* timer clear input caputure TGRA_3 */

  P_MTU34.TCNT_3 = 0x0000;            /* set timer counter3 0x0000 */
  P_MTU34.TCNT_4 = 0x0000;            /* set timer counter4 0x0000 */
  P_MTU34.TGRA_3 = cycle;             /* period set */
  P_MTU34.TGRB_3 = duk1;              /* duty set */
  P_MTU34.TGRA_4 = duk2;
  P_MTU34.TGRB_4 = duk3;

  P_MTU34.TOCR.BYTE = 0x00;           /* set output level */
  P_MTU34.TMDR_3.BYTE = 0xc8;         /* reset-synchronized pwm mode */
  P_MTU34.TOER.BYTE = 0xff;           /* set timer3,4 output */
  P_MTU34.TSTR.BYTE = 0x40;           /* start timer3 */

  while(1);                           /* loop */
 }
```

2.9 DC Motor Control Signal Output

DC Motor Control Signal Output	MCU: SH7046/47	Functions Used: MTU (Gate Control Register)

Specifications

(1) Waveforms necessary for DC brushless motor control are output as shown in figure 2.31. The output waveforms are output by chopping the respective pin gate signals and reset-synchronized PWM output.

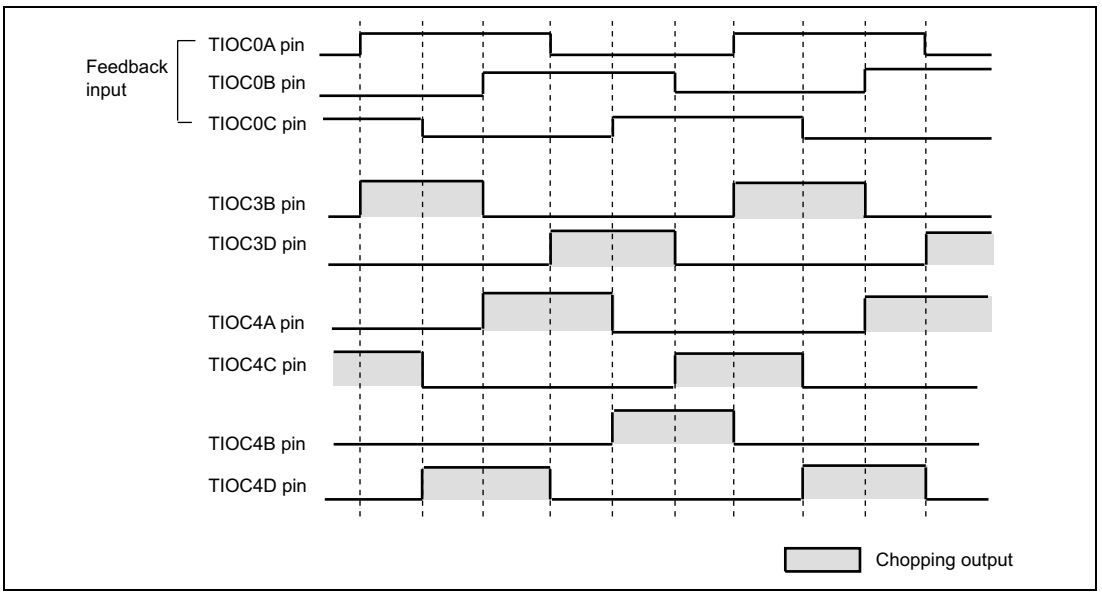


Figure 2.31 Example of DC Brushless Motor Control Signal Output

Functions Used

(1) In this sample task, MTU channels 3 and 4 are used in combination, and 3-phase PWM waveform output is performed with one common transition point in the relationship between the positive phase and negative phase. Gate signals generated from the generated waveforms and feedback input are chopped and output.

(a) Figure 2.32 shows a block diagram of the MTU as used in this sample task.

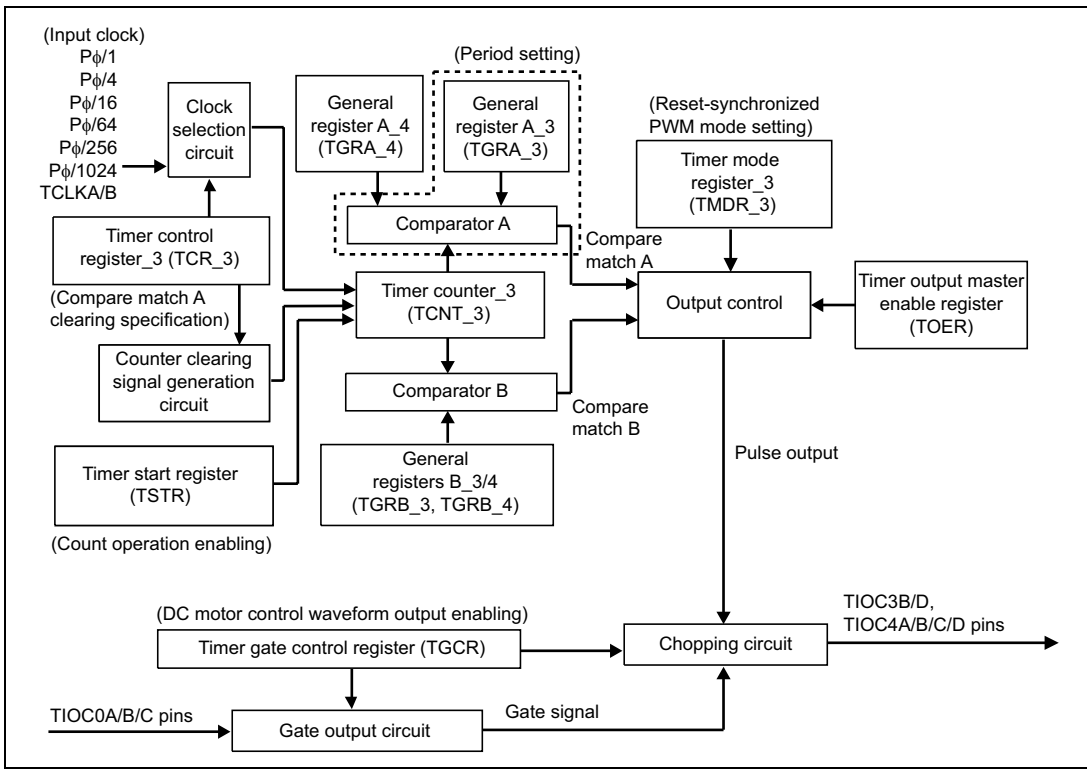


Figure 2.32 Block Diagram of MTU/ch3, ch4

(2) Table 2.9 shows the function assignments used in this task. DC motor control waveform output is performed by assigning MTU functions as shown in the table.

Table 2.9 Function Assignments

Pin or Register Name	Function	Function Assignment
TIOC3B	Pins	Pulse output pins
TIOC3D		
TIOC4A		
TIOC4B		
TIOC4C		
TIOC4D		
TIOC0A	Pins	Feedback signal input pins
TIOC0B		
TIOC0C		
TSTR	Register	Enabling/disabling of ch3, ch4 timer counter operation
TCR_3	Register	Selection of ch3 timer counter clearing source and input clock
TMDR_3	Register	Ch3, ch4 set to operate in reset-synchronized PWM mode
TGRA_3	Register	PWM period setting
TGRB_3	Registers	Output waveform transition timing setting
TGRA_4		
TGRB_4		
TOER	Register	Enabling/disabling of TIOC3B/D and TIOC4A/B/C/D pin timer output
TGCR	Register	Enabling/disabling of DC motor control waveform output

Operation

(1) Figure 2.33 illustrates the principles of operation of this sample task. DC motor control waveform output is performed automatically by hardware. (See section 2.5, Positive-Phase/Negative Phase PWM 3-Phase Output, in this Application Note for the principles of reset-synchronized PWM operation.)

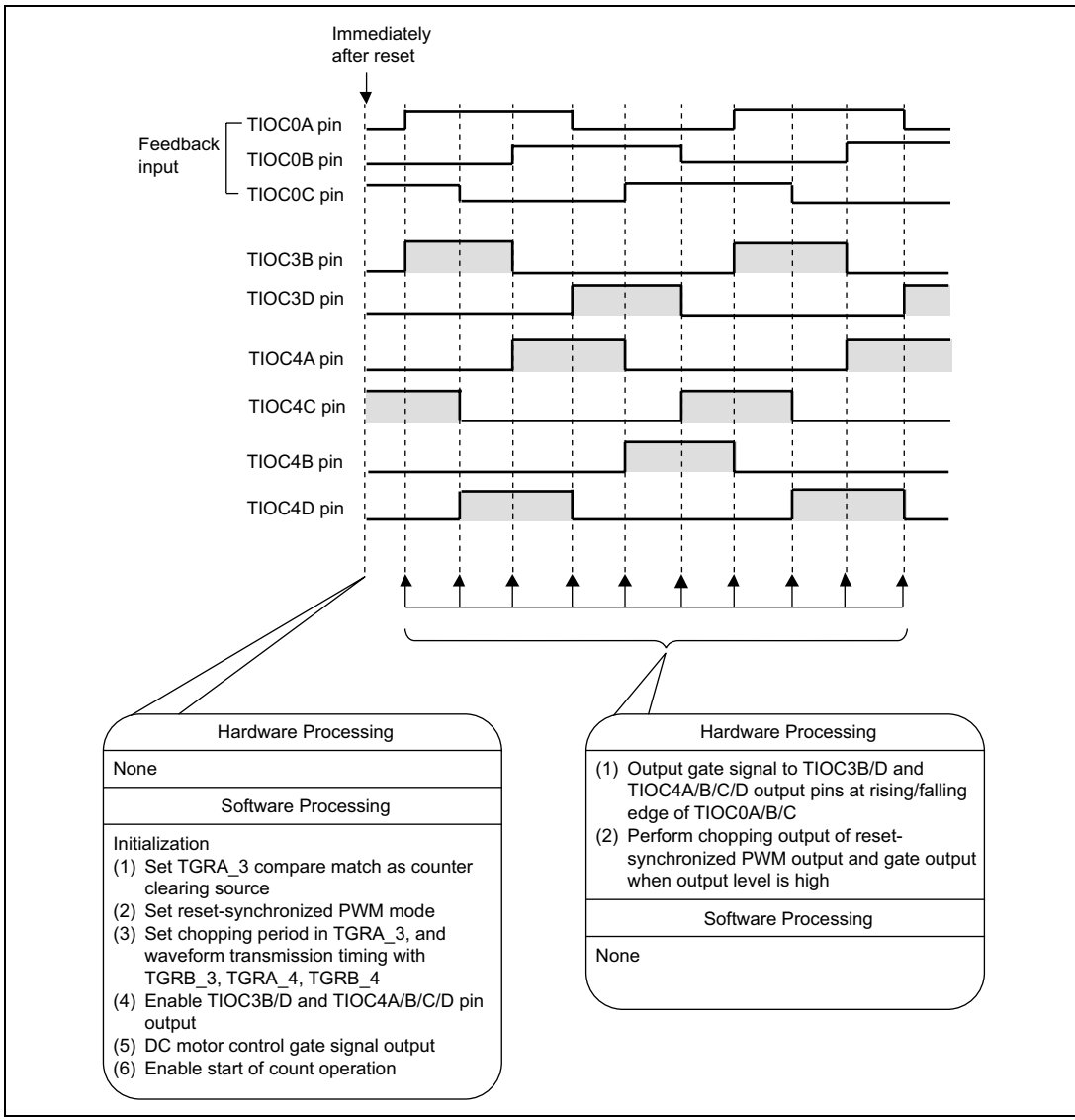


Figure 2.33 Principles of Operation of DC Motor Control Signal Output

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	dc_3	DC motor control waveform generation

(2) Arguments

Label or Register Name	Function Assignment	Data Length	Module	Input/ Output
cycle	Used to set timer value for PWM pulse period	1 word	Main routine	Input
duk1	Used to set TIOC3B/3D output waveform transition timing			
duk2	Used to set TIOC4A/4C output waveform transition timing			
duk3	Used to set TIOC4B/4D output waveform transition timing			

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	MTU module standby mode clearing, and setting of MTU to operational status	H'FFFF861E	H'd2fd
P_PORTE.PEIORL	Sets TIOC3B/D, TIOC4A/B/C/D as output pins	H'FFFF83B4	H'fa00
P_PORTE.PECRL1	Sets TIOC3B/D, TIOC4A/B/C/D as MTU input/output pins	H'FFFF83B8	H'5544
P_MTU34.TCR_3	Selection of timer counter clearing source and input clock	H'FFFF8200	H'20
P_MTU34.TOCR	Enabling of toggle output synchronized with PWM period, and positive-phase/negative-phase output level setting	H'FFFF820B	H'03
P_MTU34.TGRA_3	PWM period setting	H'FFFF8218	cycle
P_MTU34.TGRB_3	Used to set TIOC3B, TIOC3D output waveform transition timing	H'FFFF821A	duk1
P_MTU34.TGRA_4	Used to set TIOC4A, TIOC4C output waveform transition timing	H'FFFF821C	duk2
P_MTU34.TGRB_4	Used to set TIOC4B, TIOC4D output waveform transition timing	H'FFFF821E	duk3
P_MTU34.TOER	Sets TIOC3B/3D, TIOC4A/4B/4C/4D as MTU output pins	H'FFFF820A	H'ff
P_MTU34.TMDR_3	Sets reset-synchronized PWM mode	H'FFFF8202	H'c8
P_MTU34.TGCR	Enables DC motor control waveform output	H'FFFF820D	H'70

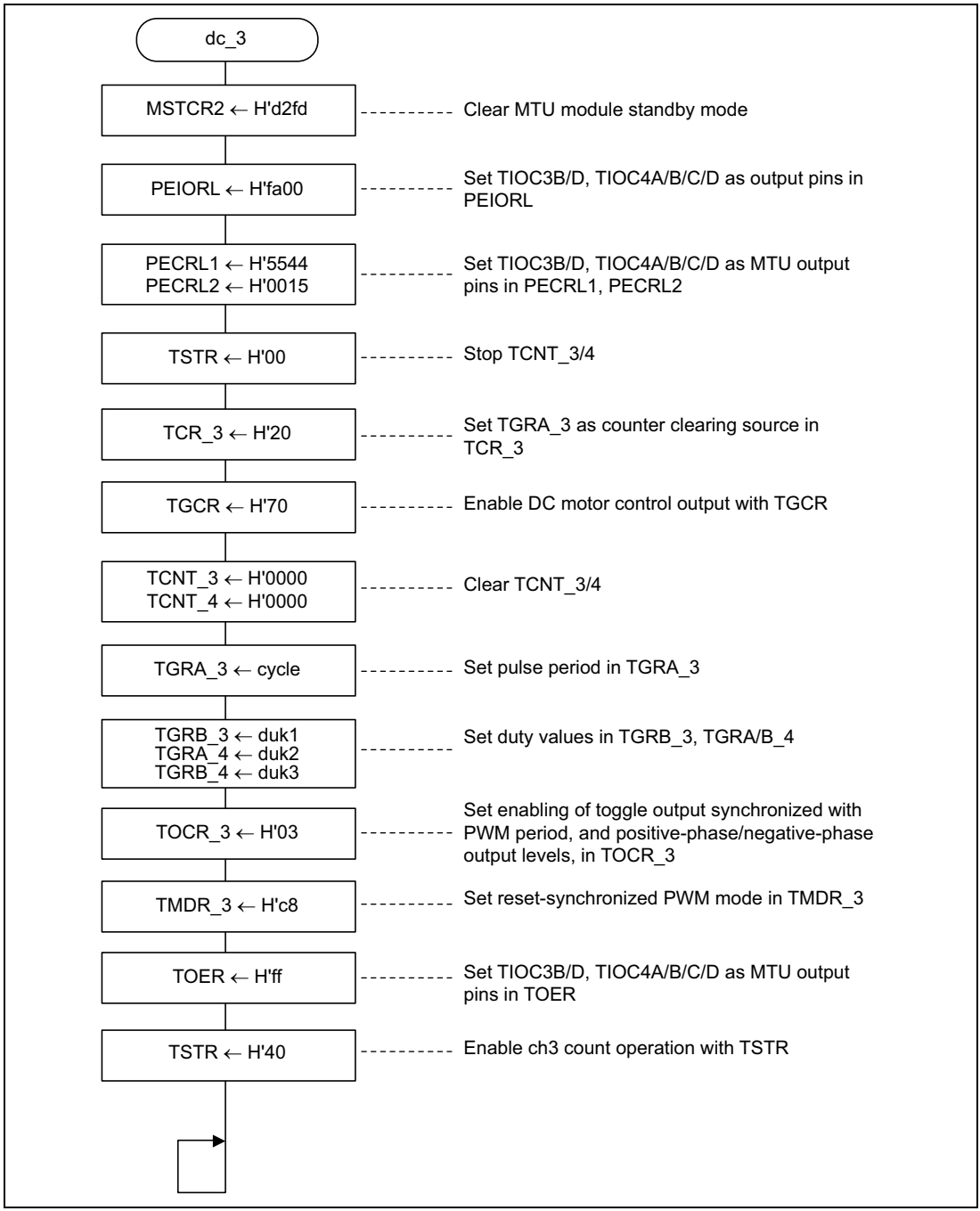
(4) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/*------------------------------------------------------------------------*/
/*                              INCLUDE FILE                              */
/*------------------------------------------------------------------------*/
#include <machine.h>
#include "iodefine_7046.h"
/*------------------------------------------------------------------------*/
/*                               PROTOTYPE                                */
/*------------------------------------------------------------------------*/
void dc_3(void);
/*------------------------------------------------------------------------*/
/*                             RAM ALLOCATION                             */
/*------------------------------------------------------------------------*/
#define cycle          (*(unsigned short *)0xffffd000)
#define duk1           (*(unsigned short *)0xffffd002)
#define duk2           (*(unsigned short *)0xffffd004)
#define duk3           (*(unsigned short *)0xffffd006)
/*------------------------------------------------------------------------*/
/*                               MAIN PROGRAM                             */
/*------------------------------------------------------------------------*/
void dc_3(void)
{
  P_STBY.MSTCR2.WORD = 0xd2fd;        /* MTU module stop mode clear */

  P_PORTE.PEIORL.WORD = 0xfa00;       /* TIOC3B/D,TIOC4A/B/C/D output */

  P_PORTE.PECRL1.WORD = 0x5544;       /* TIOC3B/D,TIOC4A/B/C/D output */
  P_PORTE.PECRL2.WORD = 0x0015;       /* TIOC0A/B/C input */

  P_MTU34.TSRT.BYTE = 0x00;
  P_MTU34.TCR_3.BYTE = 0x20;          /* timer clear input capture TGRA_3 */
  P_MTU34.TGCR.BYTE = 0x70;           /* set DC motor output */

  P_MTU34.TCNT_3 = 0x0000;            /* set timer counter3 0x0000 */
  P_MTU34.TCNT_4 = 0x0000;            /* set timer counter4 0x0000 */

  P_MTU34.TGRA_3 = cycle;             /* period set */
  P_MTU34.TGRB_3 = duk1;              /* duty set */
  P_MTU34.TGRA_4 = duk2;
  P_MTU34.TGRB_4 = duk3;

  P_MTU34.TOCR.BYTE = 0x03;           /* set output level */
  P_MTU34.TMDR_3.BYTE = 0xc8;         /* reset-synchronized pwm mode */
  P_MTU34.TOER.BYTE = 0xff;           /* set timer3,4 output */
  P_MTU34.TSTR.BYTE = 0x40;           /* start timer3 */
  while(1);                           /* loop */
 }
```

2.10 Start of A/D Conversion by MTU

Start of A/D Conversion by MTU	MCU: SH7046/47	Functions Used: MTU, A/D Converter

Specifications

(1) Four channel voltages are input and subjected to A/D conversion as shown in figure 2.34.

(2) Single-cycle scan mode and 4-channel scan mode are used for A/D conversion, with A/D conversion performed consecutively on channels 8 to 11.

(3) A/D converter activation is performed by an MTU/ch0 TGRA_0 compare match.

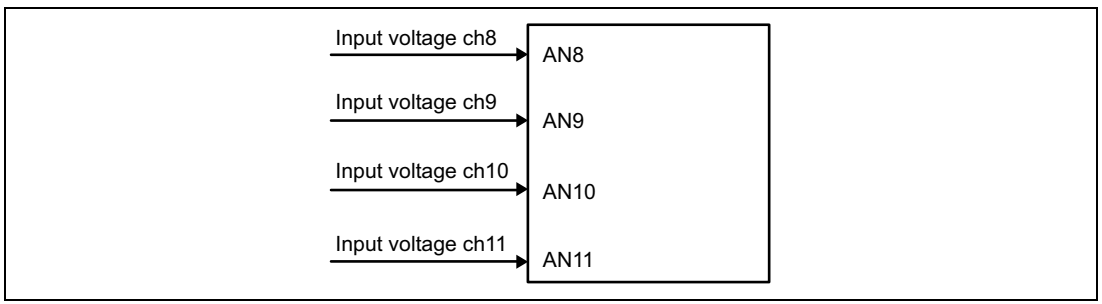


Figure 2.34 Block Diagram of Voltage Measurement by SH7046

Functions Used

(1) In this sample task, A/D conversion is started by an MTU compare match.

(a) Figure 2.35 shows a block diagram of ch0. In ch0, the A/D converter is activated using the following functions.

- A function that starts A/D conversion by means of an MTU compare match, without software intervention
- A function that outputs pulses automatically by hardware without software intervention (output compare)

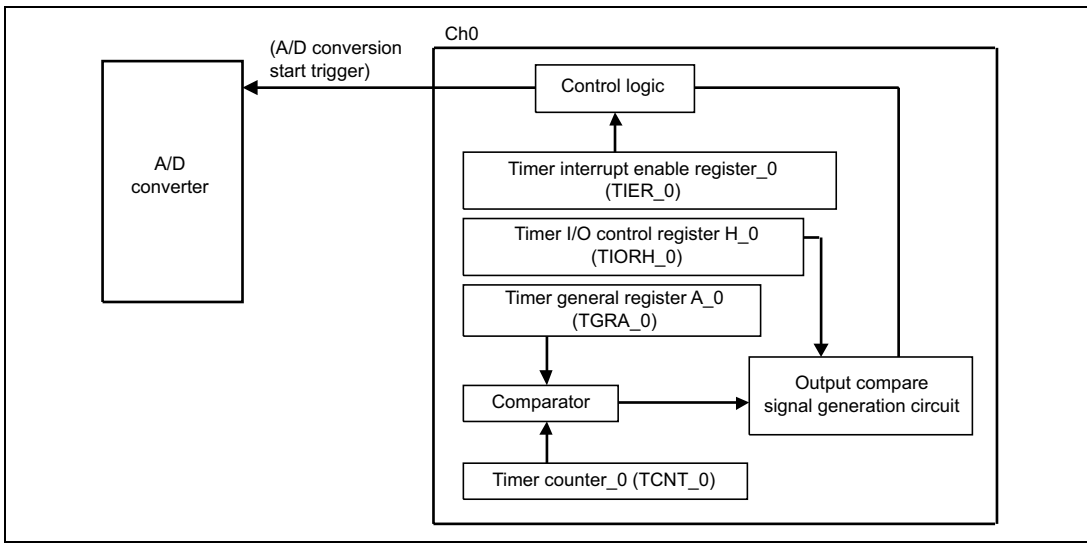


Figure 2.35 Block Diagram of SH7046 ch0

(b) Figure 2.36 shows a block diagram of the A/D converter. The A/D converter performs conversion from analog to digital form using the following function.

- A function that performs A/D conversion once on a number of channels (ch8 to ch11) (4-channel, single-cycle scan mode)

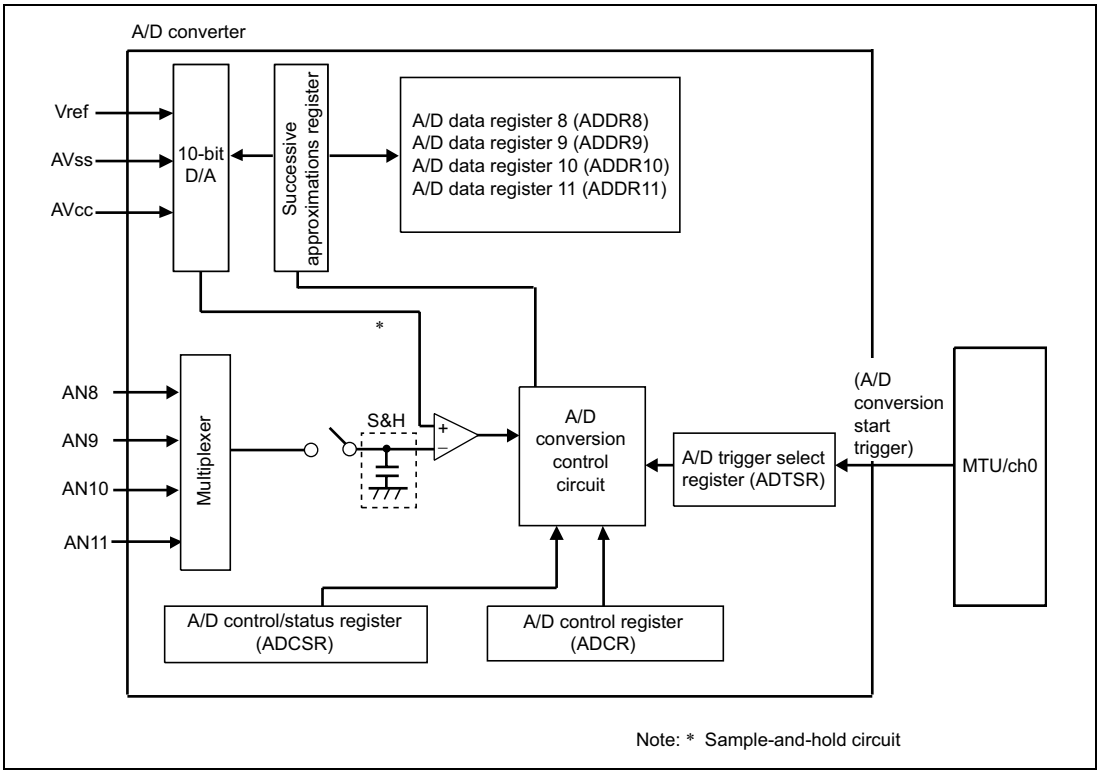


Figure 2.36 Block Diagram of Voltage Measurement by SH7046

(2) Table 2.10 shows the function assignments used in this sample task.

Table 2.10 Function Assignments

Pin or Register Name	Function	Function Assignment
AN8 to AN11	Pins	Analog measurement pins
TCR_0	Register	Selection of counter clearing source
TIER_0	Register	Enables A/D conversion start request generation
TIORH_0	Register	Timer pin function setting
TGRA_0	Register	Sampling period setting
ADCR	Register	A/D conversion mode and measurement pin setting
ADCSR	Register	Selection of conversion time and activation source
ADTSR	Register	Enables start of A/D0 module conversion by MTU trigger signal
ADDR8 to ADDR11	Registers	Storage of A/D conversion results

Operation

(1) Figure 2.37 illustrates the principles of operation of this sample task. As shown in the figure, the A/D converter is activated by a TGRA_0 compare match and sequentially measures voltages input to AN8 through AN11.

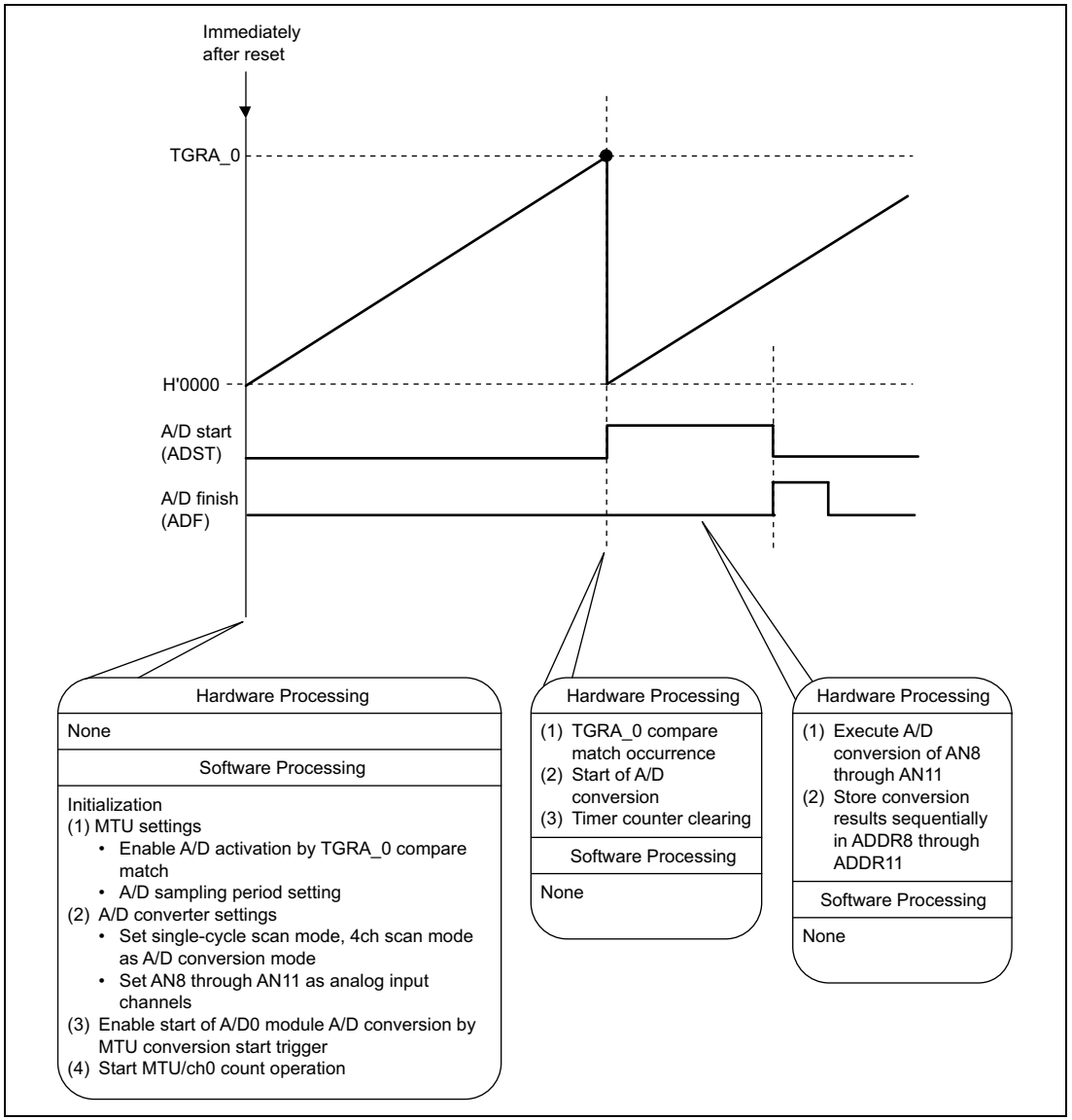


Figure 2.37 Principles of Operation of A/D Converter Activation by MTU

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	main	A/D converter activation by MTU

(2) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	Module standby mode clearing (MTU, A/D)	H'FFFF861E	H'd2ed
P_MTU0.TCR_0	Selection of TCNT counter clock, and setting of output compare A as TCNT_0 counter clearing source	H'FFFF8260	H'00
P_MTU0.TIORH_0	Sets TGRA_0 for output compare	H'FFFF8262	H'00
P_MTU0.TIER_0	Enables A/D conversion start request generation	H'FFFF8264	H'c1
P_MTU0.TGRA_0	Sets A/D conversion sampling period	H'FFFF8268	H'1000
P_AD.ADCR_0	Sets MTU conversion start trigger as A/D conversion mode (single-cycle scan mode) activation source	H'FFFF8488	H'87
P_AD.ADCSR_0	Setting of A/D conversion mode (4ch scan mode), conversion channels (AN8 to AN11), and conversion time, and enabling of A/D conversion end interrupt	H'FFFF8480	H'5f
P_AD.ADTSR	Enables start of A/D0 module conversion by MTU conversion start trigger signal	H'FFFF87F4	H'02

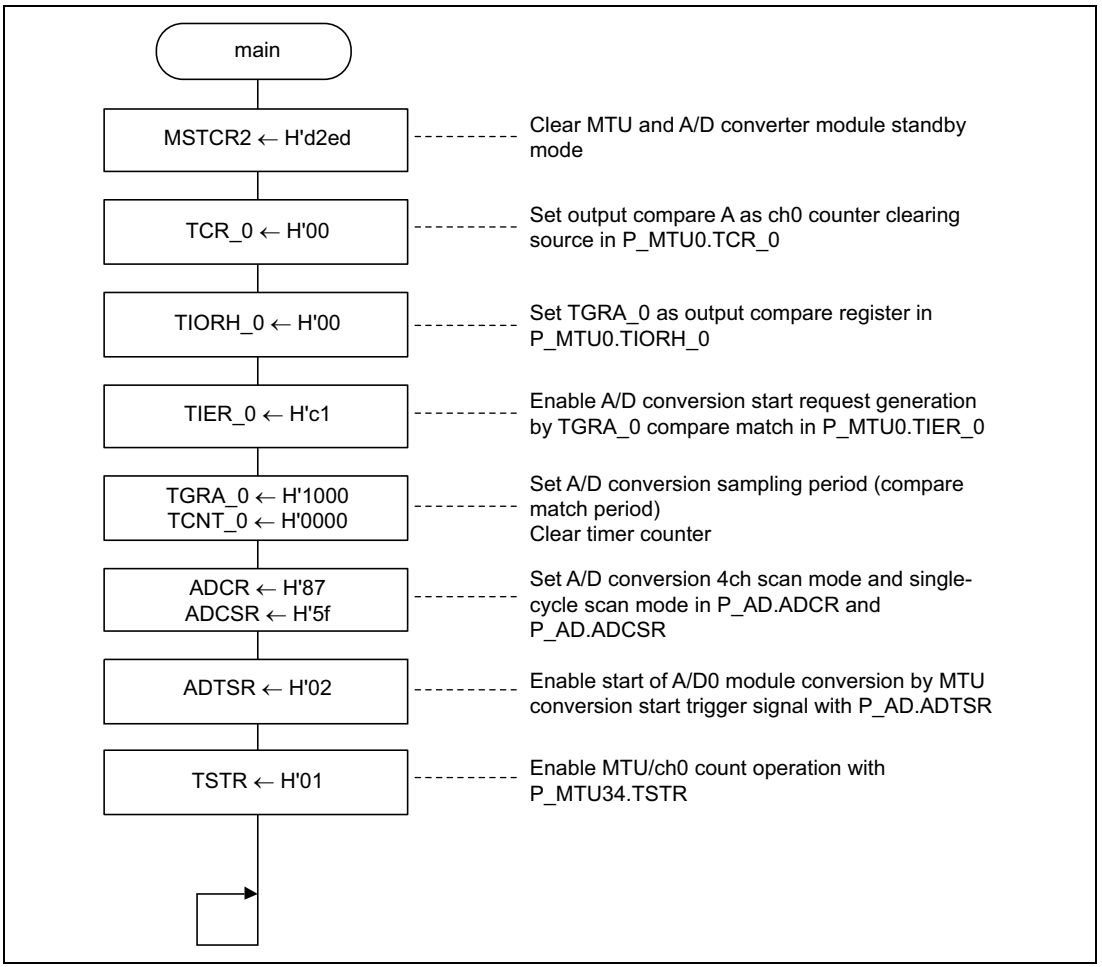
(3) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/************************************************************************/
/*                                INCLUDE FILE                          */
/************************************************************************/
#include <machine.h>
#include "iodefine_7046.h"
/************************************************************************/
/*                                 PROTOTYPE                            */
/************************************************************************/
void main(void);
/************************************************************************/
/*                                MAIN PROGRAM                          */
/************************************************************************/
void main(void)
{
    P_STBY.MSTCR2.WORD = 0xd2ed;      /* Clear Module standby mode */

    P_MTU0.TCR_0.BYTE = 0x00;         /* clock=Pϕ/1 */
    P_MTU0.TIORH_0.BYTE = 0x00;
    P_MTU0.TIER_0.BYTE = 0xc1;        /* enable TGIA interrupt */
    P_MTU0.TGRA_0 = 0x1000;
    P_MTU0.TCNT_0 = 0x0000;

    P_AD.ADCR_0.BYTE = 0x87;          /* 1-cycle scan mode */
    P_AD.ADCSR_0.BYTE = 0x5f;         /* 4-channel scan mode */
    P_AD.ADTSR.BYTE = 0x02;           /* A/D start by MTU */
    P_MTU34.TSTR.BYTE = 0x01;         /* Start timer counter */

    set_imask(0x0);
    while(1);
 }
```

2.11 Complementary PWM 3-Phase Output

Complementary PWM 3-Phase Output	MCU: SH7046/47	Functions Used: MMT

Specifications

(1) Three-phase complementary PWM waveform output is performed with a non-overlap time (dead time) between positive and negative phases, as shown in figure 2.38.

(2) In this task, duty values are set in a data table, and the duty ratio can be changed by interrupt handling.

(3) In this task, 2.0 ms is set for the period, and 0.1 ms for the dead time.

$$\text{Duty} = \frac{\text{Pulse high width}}{\text{Pulse period}} \times 100\ (\%)$$

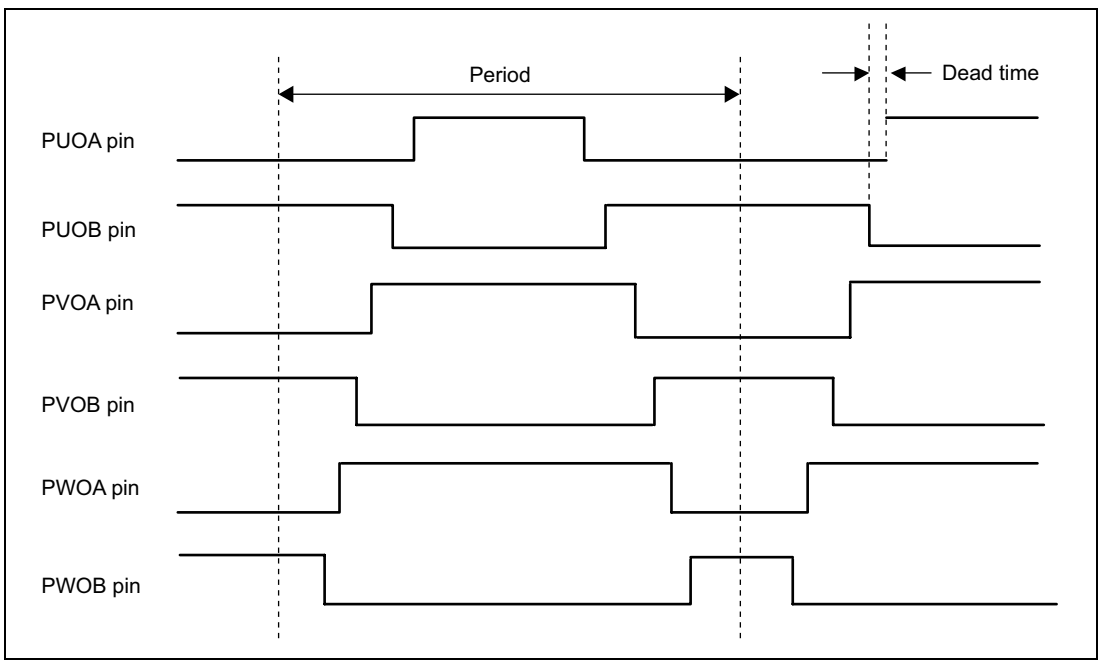


Figure 2.38 Complementary PWM 3-Phase Output Waveforms

Functions Used

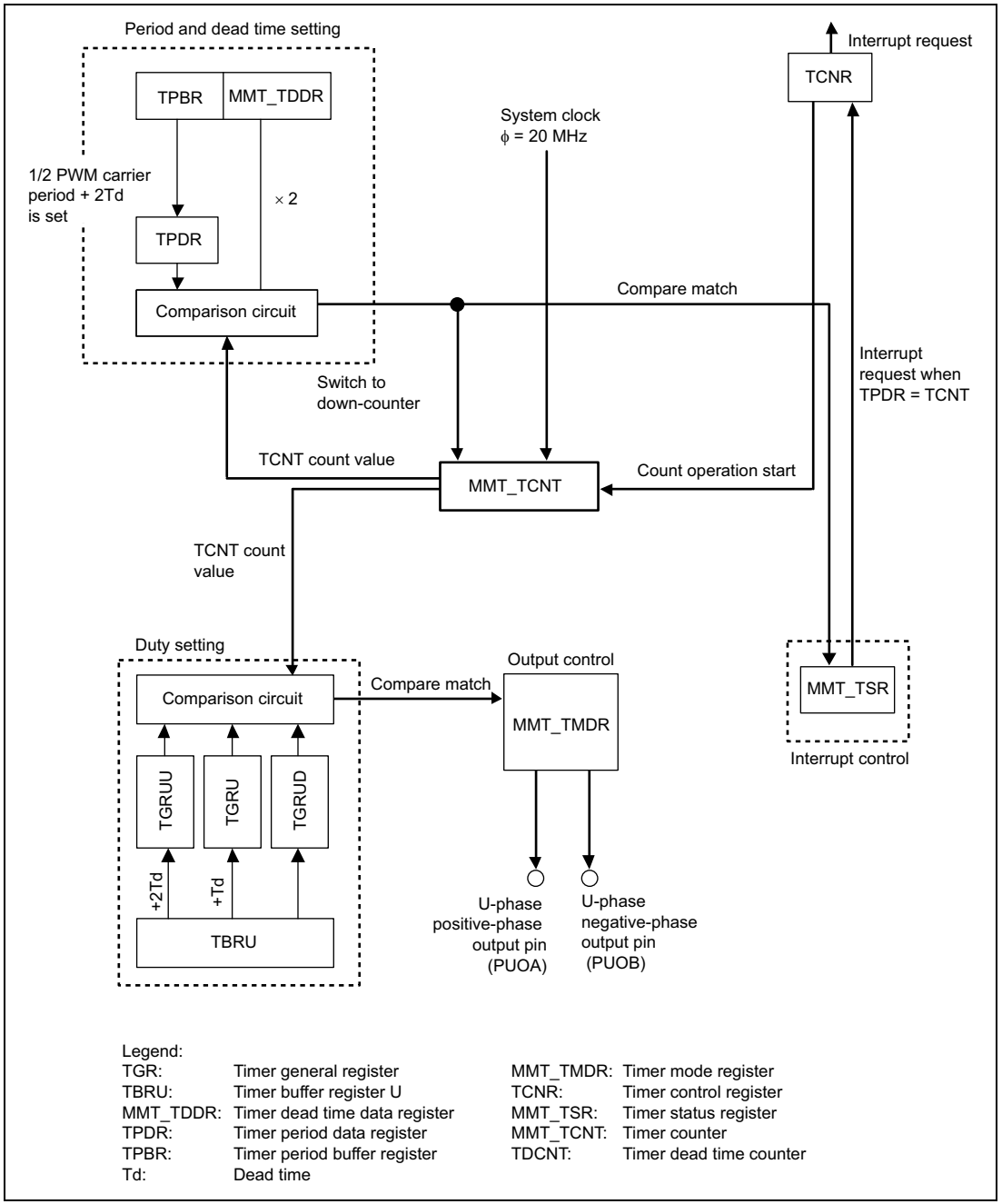
(1) In this sample task, 3-phase PWM waveforms with a non-overlap time are output from pins PU0A, PU0B, PV0A, PV0B, PW0A, and PW0B, using the complementary PWM waveform output function.

(a) Figure 2.39 shows a block diagram of the complementary PWM waveform output function for the U-phase.

The block diagram of the U-phase complementary PWM waveform output function is described below.

- The timer counter (MMT_TCNT) is a 16-bit counter that counts up/down on an input clock.
- The timer period buffer register (TPBR) is a 16-bit readable/writable register that functions as a buffer register for the timer period data register. A register value of 1/2 PWM carrier period is set. In this task, a setting of 2.0 ms is used.
- The timer period data register (TPDR) is a 16-bit read-only register that is compared with MMT_CNT in the operating mode. When the TPDR value matches the MMT_CNT value, MMT_CNT switches direction from up-counting to down-counting, and the TGFM bit in MMT_TSR is set to 1. The TPDR value is [TPBR value + 2Td].
- The timer dead time data register (MMT_TDDR) is a 16-bit readable/writable register that is used to set the non-overlap time (dead time) between positive and negative phases. In this task, a dead time setting of 0.1 ms is used.
- The timer mode register (MMT_TMDR) is an 8-bit readable/writable register that is used for positive-phase/negative-phase output level selection and operating mode setting.
- The timer control register (TCNR) is an 8-bit readable/writable register that selects operation/halting of MMT_CNT, and, when the TGFM bit in MMT_TSR is set to 1, enables/disables interrupt requests.
- Timer buffer register U is a 16-bit readable/writable register that functions as the TGR buffer register. A value written to TBR is transferred to TGR at the timing set by MD1 and MD0 in MMT_TMDR. However, a value written to the TBR free-operation register is transferred to TGR immediately. In this task, the free-operation register is used.
- Timer general register UD (TGRUD) is a 16-bit read-only register to which the TBRU value is transferred. MMT_CNT is compared with TGRUD when counting down.
- Timer general register U (TGRU) is a 16-bit read-only register to which the value of TBRU+Td is transferred. TGRU is constantly compared with MMT_CNT.
- Timer general register UU (TGRUU) is a 16-bit read-only register to which the value of TBRU+2Td is transferred. MMT_CNT is compared with TGRUU when counting up.
- The PWM U-phase (positive-phase) output pin (PU0A) outputs the U-phase positive-phase waveform.

- The PWM U-phase (negative-phase) output pin (PU0B) outputs the U-phase negative-phase waveform.



Legend:

TGR:	Timer general register	MMT_TMDR:	Timer mode register
TBRU:	Timer buffer register U	TCNR:	Timer control register
MMT_TDDR:	Timer dead time data register	MMT_TSR:	Timer status register
TPDR:	Timer period data register	MMT_TCNT:	Timer counter
TPBR:	Timer period buffer register	TDCNT:	Timer dead time counter
Td:	Dead time		

Figure 2.39 MTU U-Phase Block Diagram

(2) Table 2.11 shows the function assignments used in this task.

Table 2.11 Function Assignments

Pin or Register Name	Function	Function Assignment
PUOA	Pin	PWM U-phase output (positive phase)
PUOB	Pin	PWM U-phase output (negative phase)
TBRU	Register	TGRUD, TGRU, TGRUU buffer register
MMT_TDDR	Register	Setting of non-overlap time (Td: dead time) between positive and negative phases
TPBR	Register	TPDR buffer register. Value of 1/2 PWM carrier period is set
TPDR	Register	1/2 PWM period + 2Td
MMT_TSR	Register	Indicates TCNT/TPDR, 2Td compare match occurrence
TCNR	Register	Interrupt request enabling/disabling control Register access enabling/disabling selection Counter operation/halting selection
MMT_TMDR	Register	Operating mode setting, PWM output level selection

Operation

(1) Figure 2.40 illustrates the principles of operation of complementary PWM waveform output by SH7046 hardware and software processing.

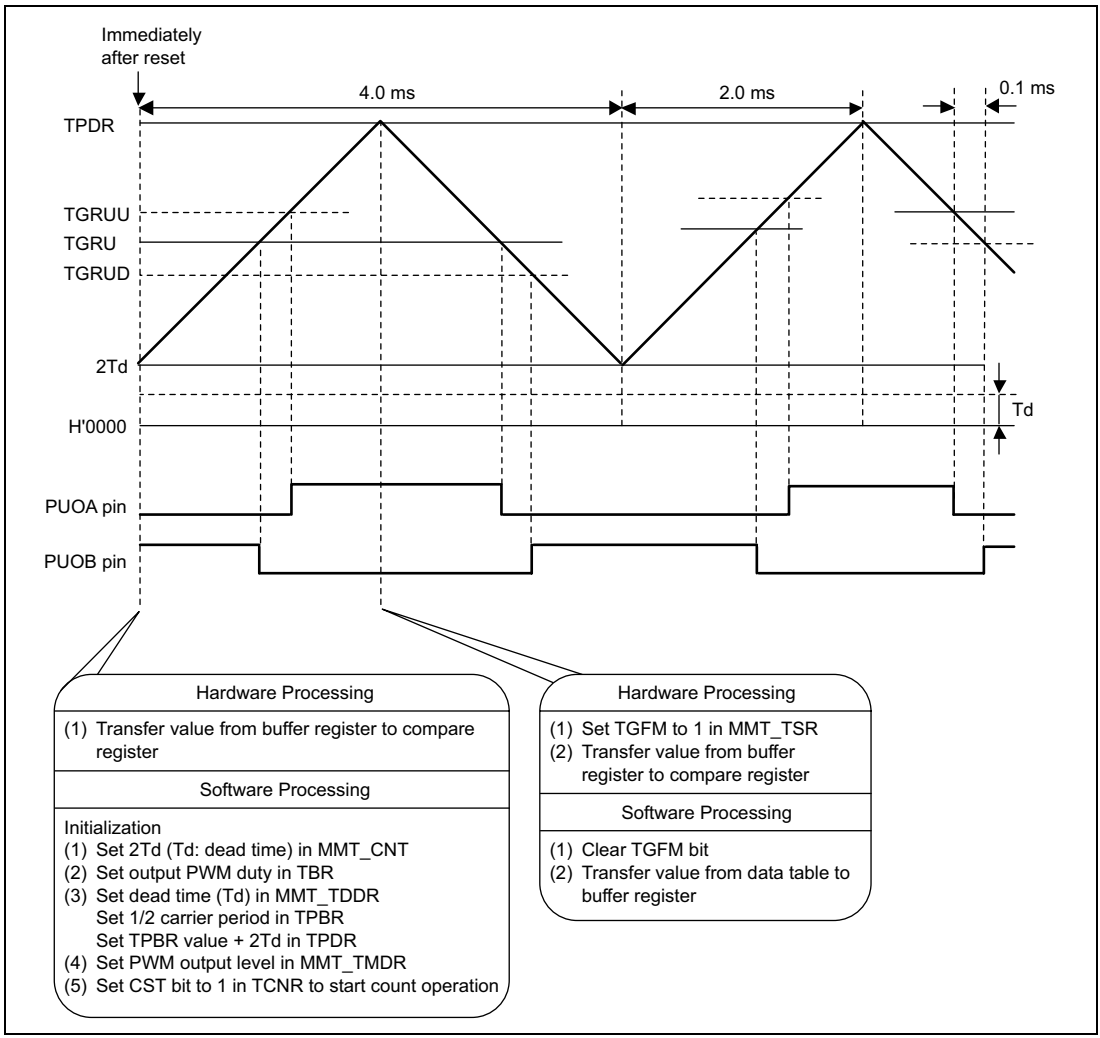


Figure 2.40 Principles of Operation of Complementary PWM Waveform Output

(2) Figure 2.41 shows the PWM waveform output method. When complementary PWM mode is set, the following rules apply to data transfer and compare operations.

Data Transfer

- In the operating mode, a buffer register is used when updating compare register data. The update data is fetched from a data table.
- Regarding the timing of data transfer from the data table to the buffer register, transfer is performed by interrupt handling when TGFM is set to 1 by a compare match between MMT_TCNT and TPDR.
- In this sample task TBRU free operation addresses are used, and therefore buffer register data is transferred to the compare register immediately.

Compare Output Waveform (for U-Phase)

Regarding the compare output waveform, MMT_TCNT is compared with TGRU, TGRUU, and TGRUD, and a PWM waveform is generated.

U-phase A

- In period T1 (during TCNT up-counting), MMT_TCNT and TGRUU are compared.
- In period T2 (during TCNT down-counting), MMT_TCNT and TGRU are compared.

U-phase B

- In period T1 (during TCNT up-counting), MMT_TCNT and TGRU are compared.
- In period T2 (during TCNT down-counting), MMT_TCNT and TGRUD are compared.

Period Setting

In case of 20 MHz operation:

Set while 1/2 period (TPBR) = H'0000 to H'FFFF (3.27675 ms) – 4Td.

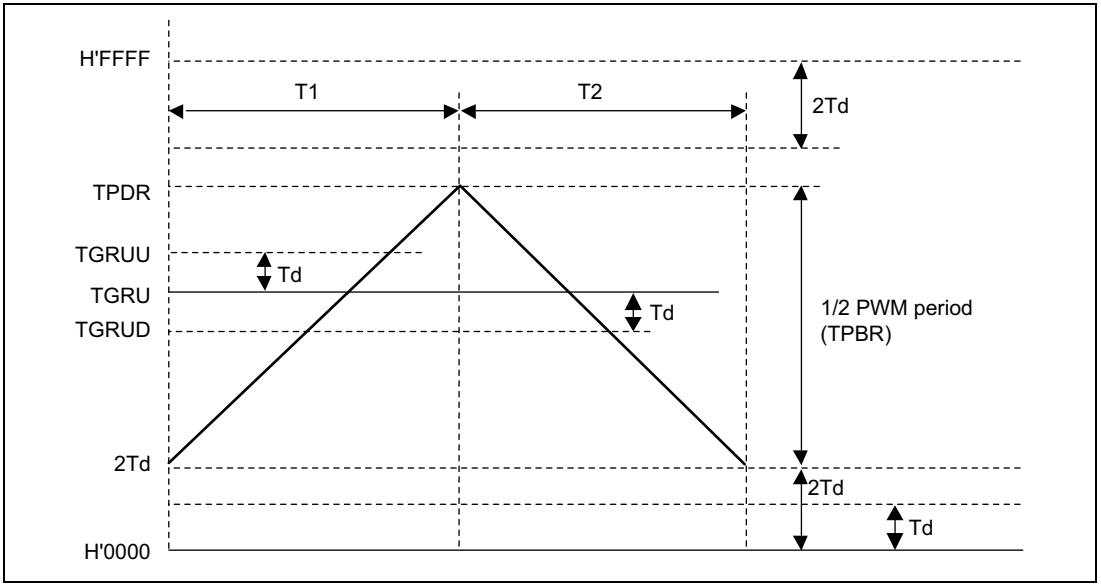


Figure 2.41 Principles of Operation of PWM Waveform Output Method

(3) Figure 2.42 illustrates the principles of operation. Complementary PWM waveform output is performed by SH7046 hardware and software processing. In this sample task TBRU free operation addresses are used.

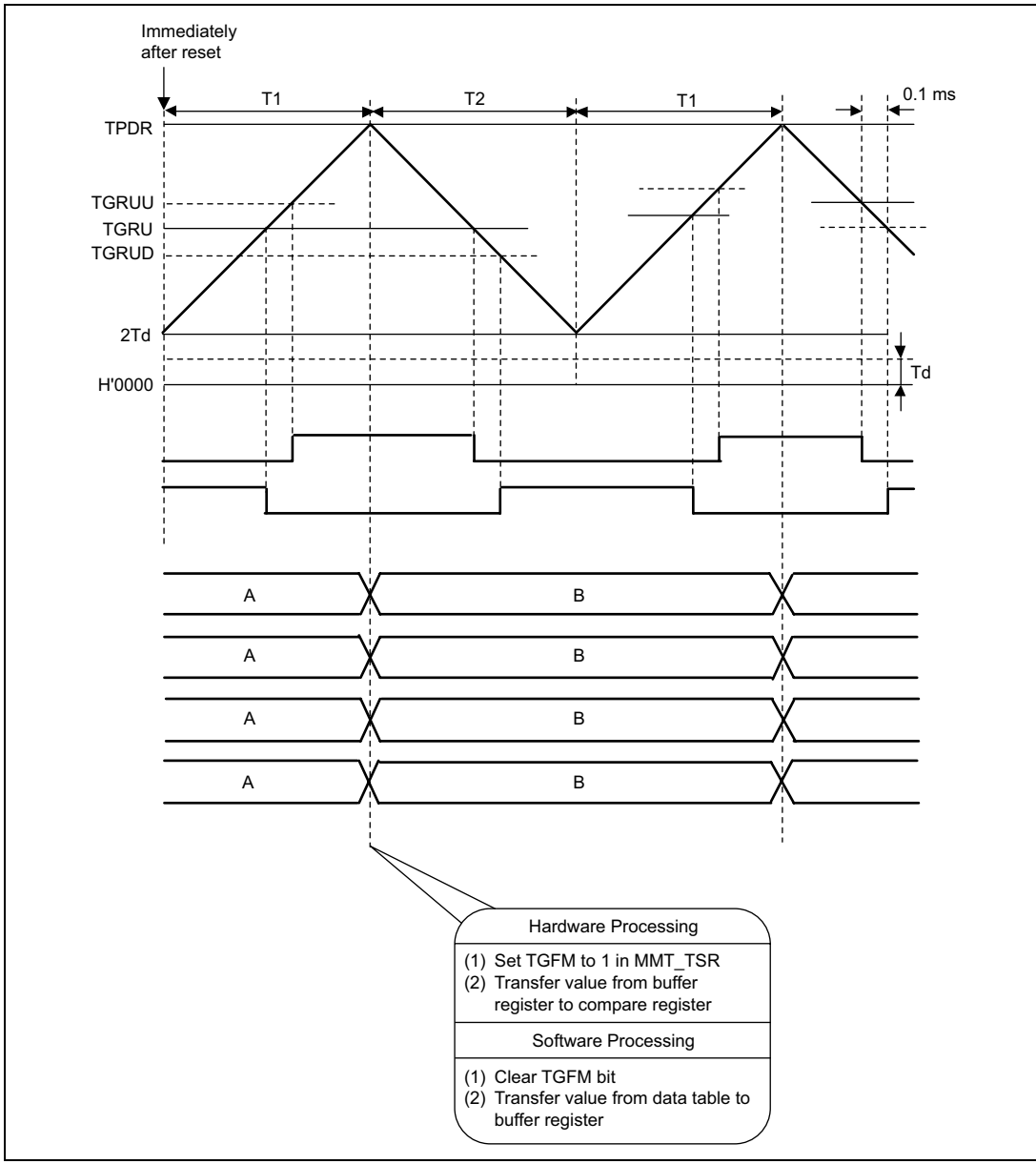


Figure 2.42 Principles of Operation of Complementary PWM U-Phase Waveform

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	mmt	MMT initialization
Duty change routine call	UP	Calls U-phase/V-phase/W-phase duty switching routine when TGFM interrupt occurs
U-phase duty change	set_u	Changes U-phase duty ratio each time TGFM interrupt occurs
V-phase duty change	set_v	Changes V-phase duty ratio each time TGFM interrupt occurs
W-phase duty change	set_w	Changes W-phase duty ratio each time TGFM interrupt occurs

(2) Arguments

This sample task does not use any arguments.

(3) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	MTU module standby mode clearing	H'FFFF861E	H'b2fd
P_PORTE.PECRH	Sets port E pins as MMT output pins	H'FFFF83BC	H'0555
P_PORTE.PEIORH	Sets port E pins as output pins	H'FFFF83B6	H'003f
P_MMT.MMT_TCNT	2Td (Td: dead time) is set	H'FFFF8A06	H'0fa0
P_MMT.TBRU_F	Used to set U-phase PWM duty (PWM duty – Td)	H'FFFF8A1C	H'2710
P_MMT.TBRV_F	Used to set V-phase PWM duty (PWM duty – Td)	H'FFFF8A2C	H'55f0
P_MMT.TBRW_F	Used to set W-phase PWM duty (PWM duty – Td)	H'FFFF8A3C	H'84b0
P_MMT.MMT_TDDR	Dead time setting	H'FFFF8A0C	H'07d0
P_MMT.TPBR	Setting of 1/2 PWM carrier period	H'FFFF8A0A	H'9c40
P_MMT.MMT_TMDR	Operating mode setting	H'FFFF8A00	H'0e
P_MMT.TCNR	Enables TGFM interrupts	H'FFFF8A02	H'41

(4) RAM Used

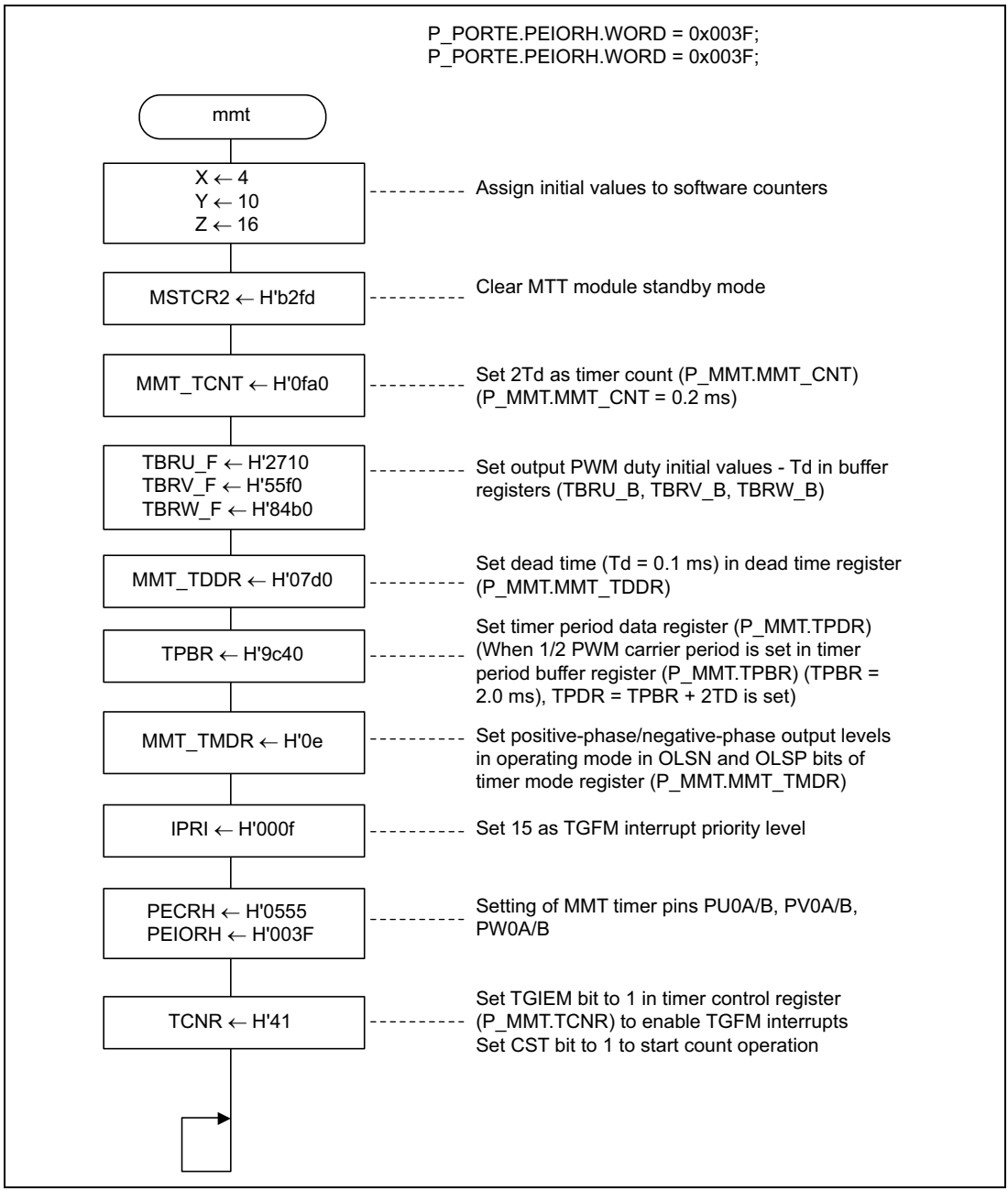
Label	Function	Address	Module
X	U-phase duty change counter	H'FFFFD000	set_u
Y	V-phase duty change counter	H'FFFFD001	set_v
Y	W-phase duty change counter	H'FFFFD002	set_w

(5) Data Table

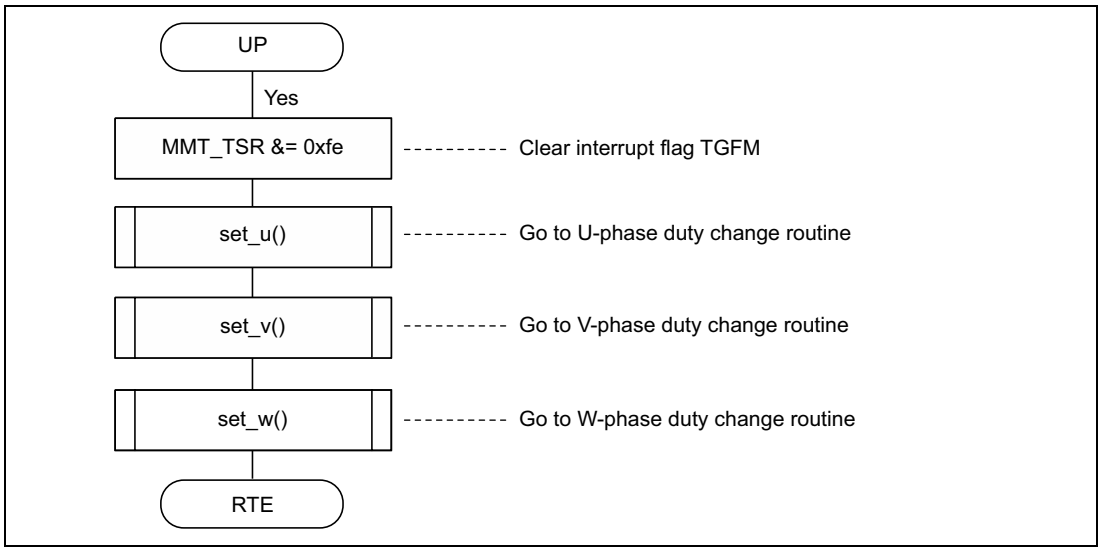
In this task, a data table (t_data) is referenced and the PWM duty ratio of each phase is changed by interrupt handling.

Flowcharts

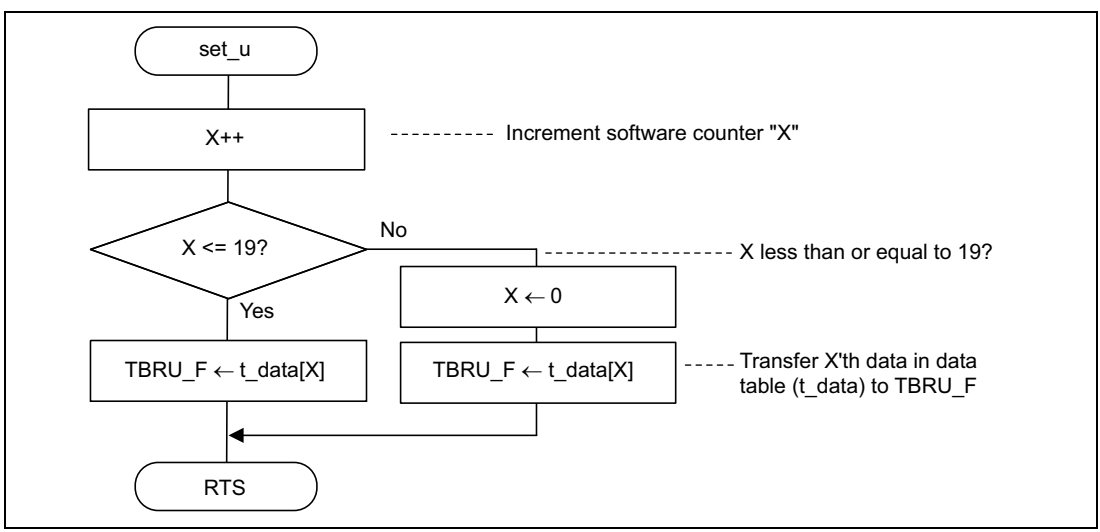
(1) Main routine



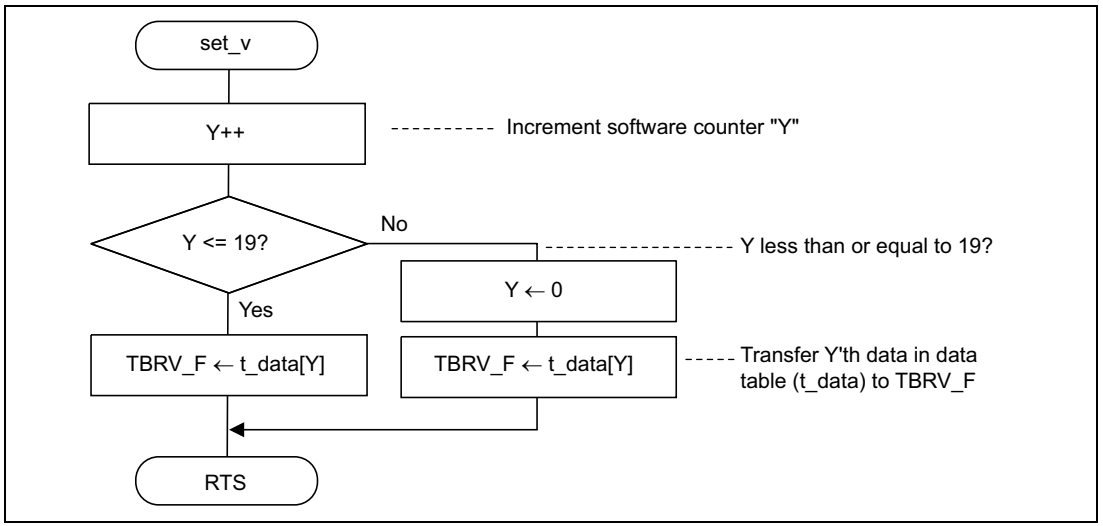
(2) Interrupt routine



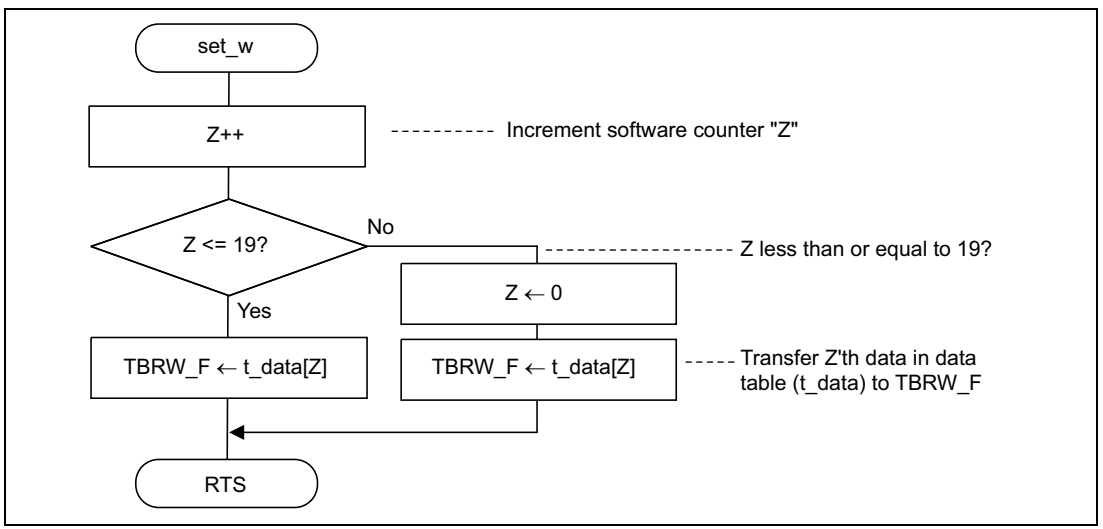
(3) U-phase output



(4) V-phase output



(5) W-phase output



Program Listing

```
/*---------------------------------------------------------------------*/
/*                          INCLUDE FILE                               */
/*---------------------------------------------------------------------*/
#include <machine.h>
#include "iodefine_7046.h"
/*---------------------------------------------------------------------*/
/*                            PROTOTYPE                                */
/*---------------------------------------------------------------------*/
void mmt(void);
void set_u(void);
void set_v(void);
void set_w(void);
#pragma interrupt(UP)
/*---------------------------------------------------------------------*/
/*                            MAIN PROGRAM                             */
/*---------------------------------------------------------------------*/
const short t_data[20] = {0x07d0,0x1770,0x2710,0x36b0,
                          0x4650,0x55f0,0x6590,0x7530,
                          0x84d0,0x9470,0xa410,0x9470,
                          0x84d0,0x7530,0x6590,0x55f0,
                          0x4650,0x36b0,0x2710,0x1770
                          };

  unsigned char X ;
  unsigned char Y ;
  unsigned char Z ;

void mmt(void)
{
   X=4 ;
   Y=10 ;
   Z=16 ;
   P_STBY.MSTCR2.WORD = 0xb2fd;   /* MMT module stop mode clear */

   P_MMT.MMT_TCNT = 0x0FA0;
   P_MMT.TBRU_F = 0x2710;
   P_MMT.TBRV_F = 0x55F0;
   P_MMT.TBRW_F = 0x84B0;
   P_MMT.MMT_TDDR = 0x07D0;
   P_MMT.TPBR = 0x9C40;
   P_MMT.MMT_TMDR.BYTE = 0x0E;    /* output level High, mode2 */
   P_INTC.IPRI.WORD = 0x000f;     /* set interrupt level=15 */
   P_PORTE.PECRH.WORD = 0x0555;   /* PUOA/B,PVOA/B,PWOA/B output */
   P_PORTE.PEIORH.WORD = 0x003F;  /* PUOA/B,PVOA/B,PWOA/B output */
   P_MMT.TCNR.BYTE = 0x41;        /* timer counter start, TGFM interrupt enable */
   set_imask(0x0);                /* set imask level=0 */
   while(1);                      /* loop */
 }
```

```
void UP()
{
   P_MMT.MMT_TSR.BYTE &= 0xfe;              /* TGFM flag clear */

   set_u();                                 /* change duty Phase U */
   set_v();                                 /* change duty Phase V */
   set_w();                                 /* change duty Phase W */
}

void set_u()
{
      X++;                                  /* increment software counter X */
      if(X <= 19){                          /* X<=19? */
              P_MMT.TBRU_F = t_data[X];     /* Phase U duty = t_data[X] */
              }
      else{
              X = 0;                        /* Clear software counter X */
              P_MMT.TBRU_F = t_data[X];     /* Phase U duty = t_data[X] */
              }

}
void set_v()
{
      Y++;                                  /* increment software counter Y */
      if(Y <= 19){                          /* Y<=19? */
              P_MMT.TBRV_F = t_data[Y];     /* Phase V duty = t_data[Y] */
              }
      else{
              Y = 0;                        /* Clear software counter Y */
              P_MMT.TBRV_F = t_data[Y];     /* Phase V duty = t_data[Y] */
              }
}

void set_w()
{
     Z++;                                   /* increment software counter Z */
     if(Z <= 19){                           /* Z<=19? */
             P_MMT.TBRW_F = t_data[Z];      /* Phase W duty = t_data[Z] */
             }
     else{
             Z = 0;                         /* Clear software counter Z */
             P_MMT.TBRW_F = t_data[Z];      /* Phase W duty = t_data[Z] */
             }
}
```

2.12 Start of A/D Conversion by MMT

Start of A/D Conversion by MMT	MCU: SH7046/47	Functions Used: MMT, A/D Converter

Specifications

(1) Four channel voltages are input and subjected to A/D conversion as shown in figure 2.43.

(2) Single-cycle scan mode and 4-channel scan mode are used for A/D conversion, with A/D conversion performed consecutively on channels 8 to 11.

(3) A/D converter activation is performed by a compare match between MTT TCNT and TPDR.

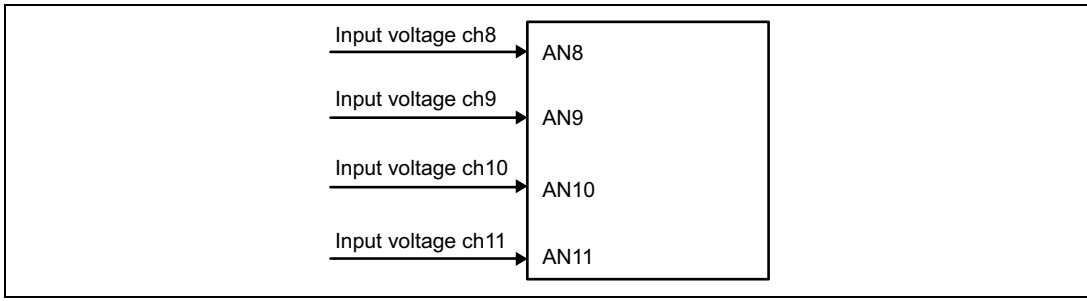


Figure 2.43 Block Diagram of Voltage Measurement by SH7046

Functions Used

(1) In this sample task, A/D conversion is started by an MMT compare match.

(a) Figure 2.44 shows a block diagram of ch0. In ch0, the A/D converter is activated using the following functions.

- A function that starts A/D conversion by means of an MMT compare match, without software intervention
- A function that outputs pulses automatically by hardware without software intervention (output compare)

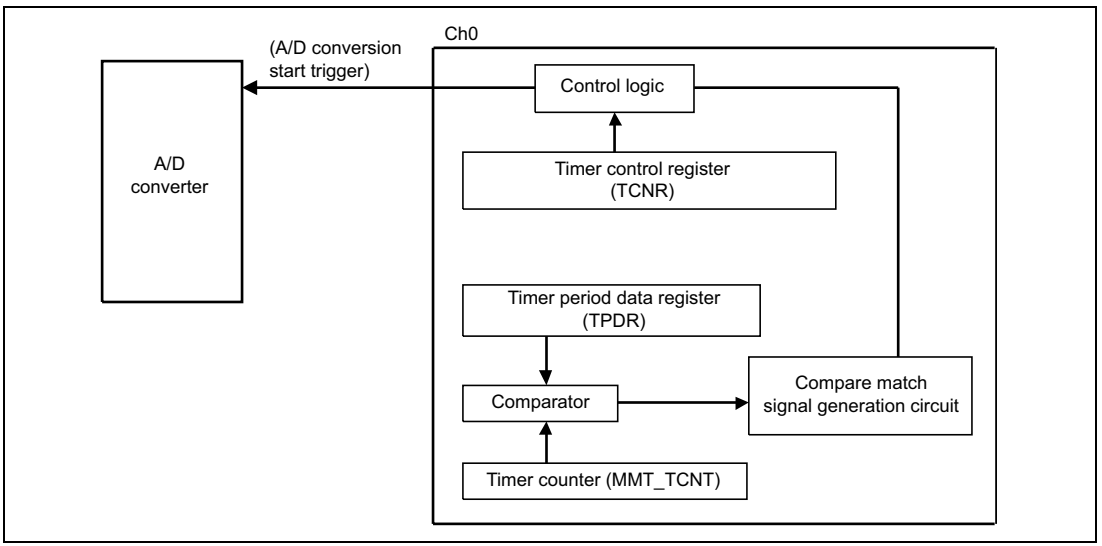


Figure 2.44 Block Diagram of SH7046 ch0

(b) Figure 2.45 shows a block diagram of the A/D converter. The A/D converter performs conversion from analog to digital form using the following function.

- A function that performs A/D conversion once on a number of channels (ch8 to ch11) (4-channel, single-cycle scan mode)

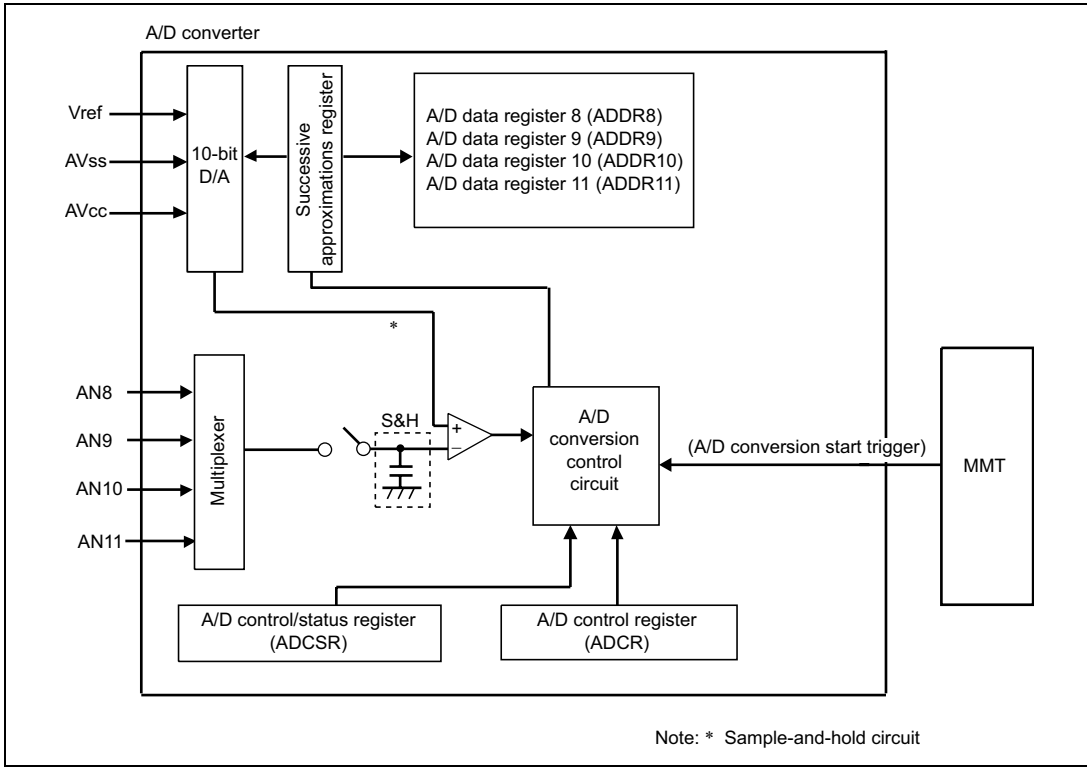


Figure 2.45 Block Diagram of Voltage Measurement by SH7046

(2) Table 2.12 shows the function assignments used in this sample task.

Table 2.12 Function Assignments

Pin or Register Name	Function	Function Assignment
AN8 to AN11	Pins	Analog measurement pins
ADDR8 to ADDR11	Registers	Storage of A/D conversion results
TCNR	Register	Enables A/D conversion start request generation
TPDR	Register	Sampling period setting
ADCR	Register	A/D conversion mode and measurement pin setting
ADCSR	Register	Selection of conversion time and activation source

Operation

(1) Figure 2.46 illustrates the principles of operation of this sample task. As shown in the figure, the A/D converter is activated by a compare match between MMT_TCNT and TPDR, and sequentially measures voltages input to AN8 through AN11.

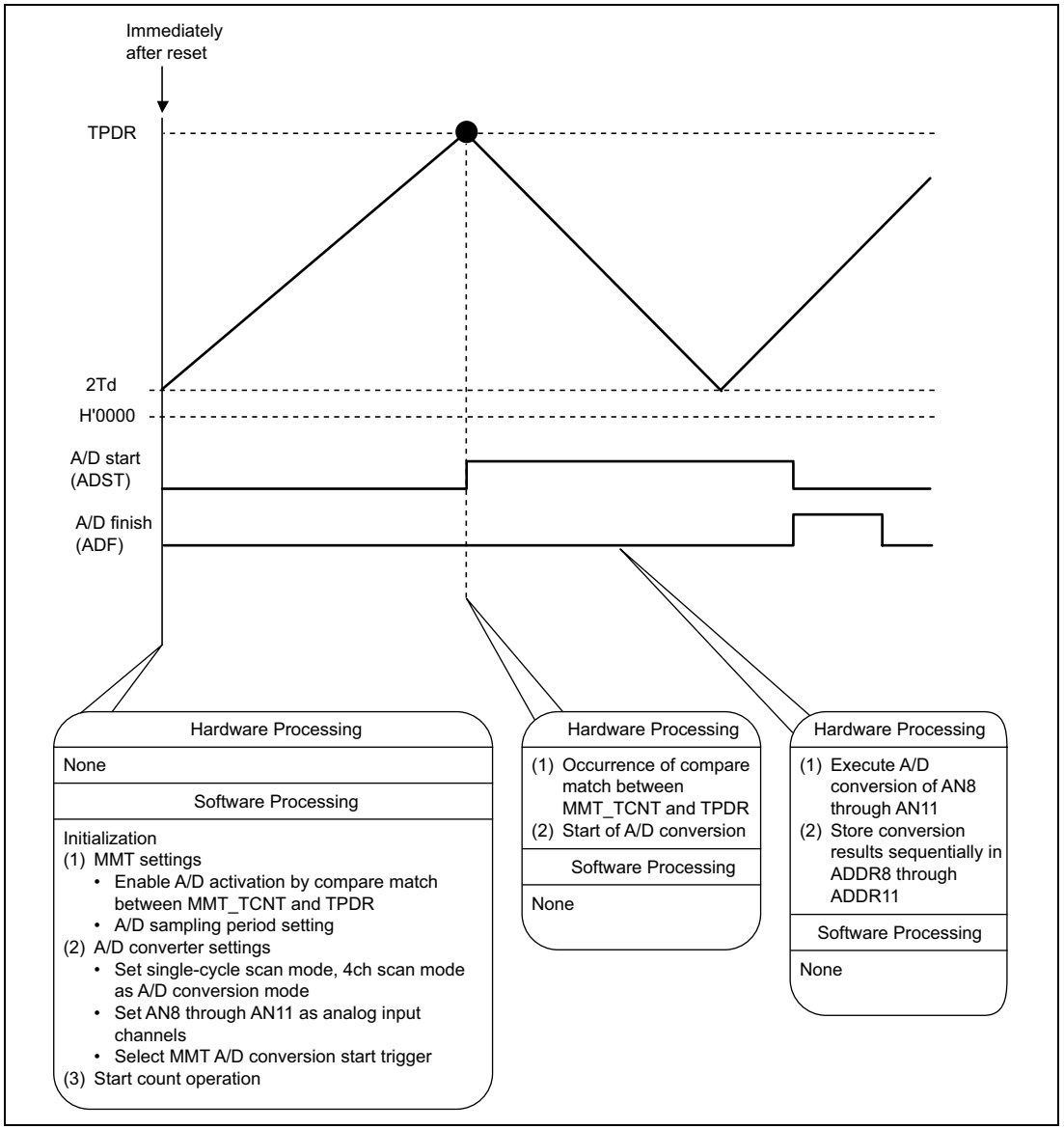


Figure 2.46 Principles of Operation of A/D Converter Activation by MMT

Software

(1) Modules

Module Name	Label	Function Assignment
Main routine	main	A/D converter activation by MMT

(2) Internal Registers Used

Register Name	Function	Address	Set Value
P_STBY.MSTCR2	Module standby mode clearing (MMT, A/D)	H'FFFF861E	H'b2ed
P_MMT.TCNR	Enables A/D conversion start request generation by compare match between TPBR and MMT_TCNT	H'FFFF8A02	H'c1
P_MMT.TPBR	Sets A/D conversion sampling period	H'FFFF8A0A	H'61a8
P_AD.ADCR_0	Sets MMT conversion start trigger as A/D conversion mode (single-cycle scan mode) activation source	H'FFFF8488	H'87
P_AD.ADCSR_0	Setting of A/D conversion mode (4ch scan mode), conversion channels (AN8 to AN11), and conversion time, and enabling of A/D conversion end interrupt	H'FFFF8480	H'5f
P_AD.ADTSR	Selects start of A/D conversion by MMT trigger signal	H'FFFF87F4	H'3f

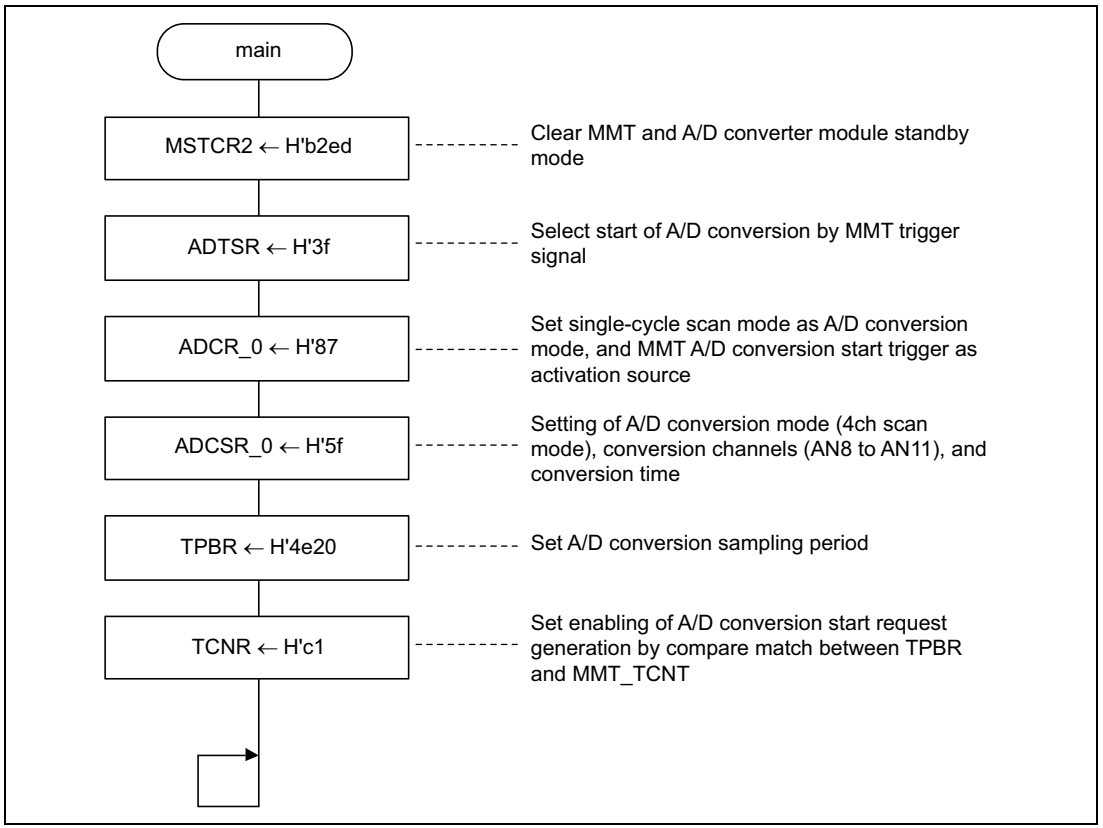
(3) RAM Used

This sample task does not use any RAM apart from the arguments.

Note: SH7046 header file names are used for register label names.

Flowcharts

(1) Main routine



Program Listing

```
/************************************************************************/
/*                              INCLUDE FILE                            */
/************************************************************************/
#include <machine.h>
#include "iodefine_7046.h"
/************************************************************************/
/*                               PROTOTYPE                              */
/************************************************************************/
void main(void);
/************************************************************************/
/*                              MAIN PROGRAM                            */
/************************************************************************/
void main(void)
{
    P_STBY.MSTCR2.WORD = 0xb2ed;      /* Clear Module standby mode */

    P_AD.ADTSR.BYTE = 0x3f;           /* A/D start by MMT */
    P_AD.ADCR_0.BYTE = 0x87;          /* 1-cycle scan mode */
    P_AD.ADCSR_0.BYTE = 0x5f;         /* 4ch scan mode */

    P_MMT.TCNR.BYTE = 0x00;
    P_MMT.TPBR = 0x4e20;              /* Sampling period=1ms */

    P_MMT.TCNR.BYTE = 0xc1;           /* Start timer counter */
    set_imask(0x0);
    while(1);
}
```

Header File	MCU: SH7046/47	Functions Used: ——

Program Listing

```
/************************************************************************/
/*      7046/47 Include File                                            */
/************************************************************************/
struct st_sci {                                         /* struct SCI   */
       union {                                          /* SMR          */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char CA:1;                 /*    C/A       */
                    unsigned char CHR:1;                /*    CHR       */
                    unsigned char PE:1;                 /*    PE        */
                    unsigned char OE:1;                 /*    O/E       */
                    unsigned char STOP:1;               /*    STOP      */
                    unsigned char MP:1;                 /*    MP        */
                    unsigned char CKS:2;                /*    CKS       */
                    } BIT;                              /*              */
             } SMR;                                     /*              */
       unsigned char BRR;                               /* BRR          */
       union {                                          /* SCR          */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char TIE:1;                /*    TIE       */
                    unsigned char RIE:1;                /*    RIE       */
                    unsigned char TE:1;                 /*    TE        */
                    unsigned char RE:1;                 /*    RE        */
                    unsigned char MPIE:1;               /*    MPIE      */
                    unsigned char TEIE:1;               /*    TEIE      */
                    unsigned char CKE:2;                /*    CKE       */
                    } BIT;                              /*              */
             } SCR;                                     /*              */
       unsigned char TDR;                               /* TDR          */
       union {                                          /* SSR          */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char TDRE:1;               /*    TDRE      */
                    unsigned char RDRF:1;               /*    RDRF      */
                    unsigned char ORER:1;               /*    ORER      */
                    unsigned char FER:1;                /*    FER       */
                    unsigned char PER:1;                /*    PER       */
                    unsigned char TEND:1;               /*    TEND      */
                    unsigned char MPB:1;                /*    MPB       */
                    unsigned char MPBT:1;               /*    MPBT      */
                    } BIT;                              /*              */
```

```
            } SSR;                                  /*              */
      unsigned char RDR;                            /* RDR          */
      union {                                       /* SDCR         */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char :4;                 /*              */
                  unsigned char DIR:1;              /*    DIR       */
                  unsigned char :3;                 /*              */
                  } BIT;                            /*              */
            } SDCR;                                 /*              */
};                                                  /*              */
struct st_mtu34 {                                   /* struct MTU34 */
      union {                                       /* TCR_3        */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char CCLR:3;             /*    CCLR      */
                  unsigned char CKEG:2;             /*    CKEG      */
                  unsigned char TPSC:3;             /*    TPSC      */
                  } BIT;                            /*              */
            } TCR_3;                                /*              */
      union {                                       /* TCR_4        */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char CCLR:3;             /*    CCLR      */
                  unsigned char CKEG:2;             /*    CKEG      */
                  unsigned char TPSC:3;             /*    TPSC      */
                  } BIT;                            /*              */
            } TCR_4;                                /*              */
      union {                                       /* TMDR_3       */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char :2;                 /*              */
                  unsigned char BFB:1;              /*    BFB       */
                  unsigned char BFA:1;              /*    BFA       */
                  unsigned char MD:4;               /*    MD        */
                  } BIT;                            /*              */
            } TMDR_3;                               /*              */
      union {                                       /* TMDR_4       */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char :2;                 /*              */
                  unsigned char BFB:1;              /*    BFB       */
                  unsigned char BFA:1;              /*    BFA       */
                  unsigned char MD:4;               /*    MD        */
                  } BIT;                            /*              */
            } TMDR_4;                               /*              */
      union {                                       /* TIORH_3      */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char IOB:4;              /*    IOB       */
                  unsigned char IOA:4;              /*    IOA       */
```

```
            } BIT;                                  /*              */
      } TIORH_3;                                    /*              */
union {                                             /* TIORL_3      */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char IOD:4;                    /*    IOD       */
            unsigned char IOC:4;                    /*    IOC       */
            } BIT;                                  /*              */
      } TIORL_3;                                    /*              */
union {                                             /* TIORH_4      */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char IOB:4;                    /*    IOB       */
            unsigned char IOA:4;                    /*    IOA       */
            } BIT;                                  /*              */
      } TIORH_4;                                    /*              */
union {                                             /* TIORL_4      */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char IOD:4;                    /*    IOD       */
            unsigned char IOC:4;                    /*    IOC       */
            } BIT;                                  /*              */
      } TIORL_4;                                    /*              */
union {                                             /* TIER_3       */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char TTGE:1;                   /*    TTGE      */
            unsigned char :2;                       /*              */
            unsigned char TCIEV:1;                  /*    TCIEV     */
            unsigned char TGIED:1;                  /*    TGIED     */
            unsigned char TGIEC:1;                  /*    TGIEC     */
            unsigned char TGIEB:1;                  /*    TGIEB     */
            unsigned char TGIEA:1;                  /*    TGIEA     */
            } BIT;                                  /*              */
      } TIER_3;                                     /*              */
union {                                             /* TIER_4       */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char TTGE:1;                   /*    TTGE      */
            unsigned char :2;                       /*              */
            unsigned char TCIEV:1;                  /*    TCIEV     */
            unsigned char TGIED:1;                  /*    TGIED     */
            unsigned char TGIEC:1;                  /*    TGIEC     */
            unsigned char TGIEB:1;                  /*    TGIEB     */
            unsigned char TGIEA:1;                  /*    TGIEA     */
            } BIT;                                  /*              */
      } TIER_4;                                     /*              */
union {                                             /* TOER         */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char :2;                       /*              */
```

```
                unsigned char OE4D:1;           /*    OE4D      */
                unsigned char OE4C:1;           /*    OE4C      */
                unsigned char OE3D:1;           /*    OE3D      */
                unsigned char OE4B:1;           /*    OE4B      */
                unsigned char OE4A:1;           /*    OE4A      */
                unsigned char OE3B:1;           /*    OE3B      */
                } BIT;                          /*              */
        } TOER;                                 /*              */
  union {                                       /* TOCR         */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
                unsigned char :1;               /*              */
                unsigned char PSYE:1;           /*    PSYE      */
                unsigned char :4;               /*              */
                unsigned char OLSN:1;           /*    OLSN      */
                unsigned char OLSP:1;           /*    OLSP      */
                } BIT;                          /*              */
        } TOCR;                                 /*              */
  unsigned char wk0[1];                         /*              */
  union {                                       /* TGCR         */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
                unsigned char :1;               /*              */
                unsigned char BDC:1;            /*    BDC       */
                unsigned char N:1;              /*    N         */
                unsigned char P:1;              /*    P         */
                unsigned char FB:1;             /*    FB        */
                unsigned char WF:1;             /*    WF        */
                unsigned char VF:1;             /*    VF        */
                unsigned char UF:1;             /*    UF        */
                } BIT;                          /*              */
        } TGCR;                                 /*              */
  unsigned char wk1[2];                         /*              */
  unsigned short TCNT_3;                        /* TCNT_3       */
  unsigned short TCNT_4;                        /* TCNT_4       */
  unsigned short TCDR;                          /* TCDR         */
  unsigned short TDDR;                          /* TDDR         */
  unsigned short TGRA_3;                        /* TGRA_3       */
  unsigned short TGRB_3;                        /* TGRB_3       */
  unsigned short TGRA_4;                        /* TGRA_4       */
  unsigned short TGRB_4;                        /* TGRB_4       */
  unsigned short TCNTS;                         /* TCNTS        */
  unsigned short TCBR;                          /* TCBR         */
  unsigned short TGRC_3;                        /* TGRC_3       */
  unsigned short TGRD_3;                        /* TGRD_3       */
  unsigned short TGRC_4;                        /* TGRC_4       */
  unsigned short TGRD_4;                        /* TGRD_4       */
  union {                                       /* TSR_3        */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
                unsigned char TDFD:1;           /*    TDFD      */
```

```
                    unsigned char :2;                   /*              */
                    unsigned char TCFV:1;               /*    TCFV      */
                    unsigned char TGFD:1;               /*    TGFD      */
                    unsigned char TGFC:1;               /*    TGFC      */
                    unsigned char TGFB:1;               /*    TGFB      */
                    unsigned char TGFA:1;               /*    TGFA      */
                    } BIT;                              /*              */
             } TSR_3;                                   /*              */
       union {                                          /* TSR_4        */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char TDFD:1;               /*    TDFD      */
                    unsigned char :2;                   /*              */
                    unsigned char TCFV:1;               /*    TCFV      */
                    unsigned char TGFD:1;               /*    TGFD      */
                    unsigned char TGFC:1;               /*    TGFC      */
                    unsigned char TGFB:1;               /*    TGFB      */
                    unsigned char TGFA:1;               /*    TGFA      */
                    } BIT;                              /*              */
             } TSR_4;                                   /*              */
       unsigned char wk2[18];                           /*              */
       union {                                          /* TSTR         */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char CST4:1;               /*    CST4      */
                    unsigned char CST3:1;               /*    CST3      */
                    unsigned char :3;                   /*              */
                    unsigned char CST:3;                /*    CST       */
                    } BIT;                              /*              */
             } TSTR;                                    /*              */
       union {                                          /* TSYR         */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char SYNC4:1;              /*    SYNC4     */
                    unsigned char SYNC3:1;              /*    SYNC3     */
                    unsigned char :3;                   /*              */
                    unsigned char SYNC2:1;              /*    SYNC2     */
                    unsigned char SYNC1:1;              /*    SYNC1     */
                    unsigned char SYNC0:1;              /*    SYNC0     */
                    } BIT;                              /*              */
             } TSYR;                                    /*              */
};                                                      /*              */
struct st_mtu0 {                                        /* struct MTU0  */
       union {                                          /* TCR_0        */
             unsigned char BYTE;                        /*  Byte Access */
             struct {                                   /*  Bit Access  */
                    unsigned char CCLR:3;               /*    CCLR      */
                    unsigned char CKEG:2;               /*    CKEG      */
                    unsigned char TPSC:3;               /*    TPSC      */
                    } BIT;                              /*              */
             } TCR_0;                                   /*              */
```

```
union {                                 /* TMDR_0       */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char :2;          /*              */
             unsigned char BFB:1;       /*    BFB       */
             unsigned char BFA:1;       /*    BFA       */
             unsigned char MD:4;        /*    MD        */
             } BIT;                     /*              */
      } TMDR_0;                         /*              */
union {                                 /* TIORH_0      */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char IOB:4;       /*    IOB       */
             unsigned char IOA:4;       /*    IOA       */
             } BIT;                     /*              */
      } TIORH_0;                        /*              */
union {                                 /* TIORL_0      */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char IOD:4;       /*    IOD       */
             unsigned char IOC:4;       /*    IOC       */
             } BIT;                     /*              */
      } TIORL_0;                        /*              */
union {                                 /* TIER_0       */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char TTGE:1;      /*    TTGE      */
             unsigned char :2;          /*              */
             unsigned char TCIEV:1;     /*    TCIEV     */
             unsigned char TGIED:1;     /*    TGIED     */
             unsigned char TGIEC:1;     /*    TGIEC     */
             unsigned char TGIEB:1;     /*    TGIEB     */
             unsigned char TGIEA:1;     /*    TGIEA     */
             } BIT;                     /*              */
      } TIER_0;                         /*              */
union {                                 /* TSR_0        */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char :3;          /*              */
             unsigned char TCFV:1;      /*    TCFV      */
             unsigned char TGFD:1;      /*    TGFD      */
             unsigned char TGFC:1;      /*    TGFC      */
             unsigned char TGFB:1;      /*    TGFB      */
             unsigned char TGFA:1;      /*    TGFA      */
             } BIT;                     /*              */
      } TSR_0;                          /*              */
unsigned short TCNT_0;                  /* TCNT_0       */
unsigned short TGRA_0;                  /* TGRA_0       */
unsigned short TGRB_0;                  /* TGRB_0       */
unsigned short TGRC_0;                  /* TGRC_0       */
unsigned short TGRD_0;                  /* TGRD_0       */
```

```
};                                                          /*              */
struct st_mtu1 {                                            /* struct MTU1  */
       union {                                              /* TCR_1        */
             unsigned char BYTE;                            /*  Byte Access */
             struct {                                       /*  Bit Access  */
                    unsigned char :1;                       /*              */
                    unsigned char CCLR:2;                   /*    CCLR      */
                    unsigned char CKEG:2;                   /*    CKEG      */
                    unsigned char TPSC:3;                   /*    TPSC      */
                    } BIT;                                  /*              */
             } TCR_1;                                       /*              */
       union {                                              /* TMDR_1       */
             unsigned char BYTE;                            /*  Byte Access */
             struct {                                       /*  Bit Access  */
                    unsigned char :4;                       /*              */
                    unsigned char MD:4;                     /*    MD        */
                    } BIT;                                  /*              */
             } TMDR_1;                                      /*              */
       union {                                              /* TIOR_1       */
             unsigned char BYTE;                            /*  Byte Access */
             struct {                                       /*  Bit Access  */
                    unsigned char IOB:4;                    /*    IOB       */
                    unsigned char IOA:4;                    /*    IOA       */
                    } BIT;                                  /*              */
             } TIOR_1;                                      /*              */
       unsigned char wk0[1];                                /*              */
       union {                                              /* TIER_1       */
             unsigned char BYTE;                            /*  Byte Access */
             struct {                                       /*  Bit Access  */
                    unsigned char TTGE:1;                   /*    TTGE      */
                    unsigned char :1;                       /*              */
                    unsigned char TCIEU:1;                  /*    TCIEU     */
                    unsigned char TCIEV:1;                  /*    TCIEV     */
                    unsigned char :2;                       /*              */
                    unsigned char TGIEB:1;                  /*    TGIEB     */
                    unsigned char TGIEA:1;                  /*    TGIEA     */
                    } BIT;                                  /*              */
             } TIER_1;                                      /*              */
       union {                                              /* TSR_1        */
             unsigned char BYTE;                            /*  Byte Access */
             struct {                                       /*  Bit Access  */
                    unsigned char TCFD:1;                   /*    TCFD      */
                    unsigned char :1;                       /*              */
                    unsigned char TCFU:1;                   /*    TCFU      */
                    unsigned char TCFV:1;                   /*    TCFV      */
                    unsigned char :2;                       /*              */
                    unsigned char TGFB:1;                   /*    TGFB      */
                    unsigned char TGFA:1;                   /*    TGFA      */
                    } BIT;                                  /*              */
             } TSR_1;                                       /*              */
       unsigned short TCNT_1;                               /* TCNT_1       */
```

```
        unsigned short TGRA_1;                  /* TGRA_1       */
        unsigned short TGRB_1;                  /* TGRB_1       */
};                                              /*              */
struct st_mtu2 {                                /* struct MTU2  */
        union {                                 /* TCR_2        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char :1;          /*              */
                     unsigned char CCLR:2;      /*    CCLR      */
                     unsigned char CKEG:2;      /*    CKEG      */
                     unsigned char TPSC:3;      /*    TPSC      */
                     } BIT;                     /*              */
              } TCR_2;                          /*              */
        union {                                 /* TMDR_2       */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char :4;          /*              */
                     unsigned char MD:4;        /*    MD        */
                     } BIT;                     /*              */
              } TMDR_2;                         /*              */
        union {                                 /* TIOR_2       */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char IOB:4;       /*    IOB       */
                     unsigned char IOA:4;       /*    IOA       */
                     } BIT;                     /*              */
              } TIOR_2;                         /*              */
        unsigned char wk0[1];                   /*              */
        union {                                 /* TIER_2       */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char TTGE:1;      /*    TTGE      */
                     unsigned char :1;          /*              */
                     unsigned char TCIEU:1;     /*    TCIEU     */
                     unsigned char TCIEV:1;     /*    TCIEV     */
                     unsigned char :2;          /*              */
                     unsigned char TGIEB:1;     /*    TGIEB     */
                     unsigned char TGIEA:1;     /*    TGIEA     */
                     } BIT;                     /*              */
              } TIER_2;                         /*              */
        union {                                 /* TSR_2        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char TCFD:1;      /*    TCFD      */
                     unsigned char :1;          /*              */
                     unsigned char TCFU:1;      /*    TCFU      */
                     unsigned char TCFV:1;      /*    TCFV      */
                     unsigned char :2;          /*              */
                     unsigned char TGFB:1;      /*    TGFB      */
                     unsigned char TGFA:1;      /*    TGFA      */
                     } BIT;                     /*              */
```

```
            } TSR_2;                                    /*             */
      unsigned short TCNT_2;                            /* TCNT_2      */
      unsigned short TGRA_2;                            /* TGRA_2      */
      unsigned short TGRB_2;                            /* TGRB_2      */
};                                                      /*             */
struct st_intc {                                        /* struct INTC */
      union {                                           /* IPRA        */
            unsigned short WORD;                        /*  Word Access*/
            struct {                                    /*  Bit Access */
                  unsigned short IRQ0:4;                /*    IRQ0     */
                  unsigned short IRQ1:4;                /*    IRQ1     */
                  unsigned short IRQ2:4;                /*    IRQ2     */
                  unsigned short IRQ3:4;                /*    IRQ3     */
                  } BIT;                                /*             */
            } IPRA;                                     /*             */
      unsigned char wk0[4];                             /*             */
      union {                                           /* IPRD        */
            unsigned short WORD;                        /*  Word Access*/
            struct {                                    /*  Bit Access */
                  unsigned short TGI_0:4;               /*    TGI_0    */
                  unsigned short TCI_0:4;               /*    TCI_0    */
                  unsigned short TGI_1:4;               /*    TGI_1    */
                  unsigned short TCI_1:4;               /*    TCI_1    */
                  } BIT;                                /*             */
            } IPRD;                                     /*             */
      union {                                           /* IPRE        */
            unsigned short WORD;                        /*  Word Access*/
            struct {                                    /*  Bit Access */
                  unsigned short TGI_2:4;               /*    TGI_2    */
                  unsigned short TCI_2:4;               /*    TCI_2    */
                  unsigned short TGI_3:4;               /*    TGI_3    */
                  unsigned short TCI_3:4;               /*    TCI_3    */
                  } BIT;                                /*             */
            } IPRE;                                     /*             */
      union {                                           /* IPRF        */
            unsigned short WORD;                        /*  Word Access*/
            struct {                                    /*  Bit Access */
                  unsigned short TGI_4:4;               /*    TGI_4    */
                  unsigned short TCI_4:4;               /*    TCI_4    */
                  unsigned short :8;                    /*             */
                  } BIT;                                /*             */
            } IPRF;                                     /*             */
      union {                                           /* IPRG        */
            unsigned short WORD;                        /*  Word Access*/
            struct {                                    /*  Bit Access */
                  unsigned short AD01:4;                /*    A/D0,1   */
                  unsigned short DTC:4;                 /*    DTC      */
                  unsigned short CMT0:4;                /*    CMT0     */
                  unsigned short CMT1:4;                /*    CMT1     */
                  } BIT;                                /*             */
            } IPRG;                                     /*             */
```

```
union {                                 /* IPRH         */
      unsigned short WORD;              /*  Word Access */
      struct {                          /*  Bit Access  */
            unsigned short WDT:4;       /*    WDT       */
            unsigned short IOMTU:4;     /*    I/O(MTU)  */
            unsigned short :8;          /*              */
            } BIT;                      /*              */
      } IPRH;                           /*              */
union {                                 /* ICR1         */
      unsigned short WORD;              /*  Word Access */
      struct {                          /*  Bit Access  */
            unsigned short NMIL:1;      /*    NMIL      */
            unsigned short :6;          /*              */
            unsigned short NMIE:1;      /*    NMIE      */
            unsigned short IRQ0S:1;     /*    IRQ0S     */
            unsigned short IRQ1S:1;     /*    IRQ1S     */
            unsigned short IRQ2S:1;     /*    IRQ2S     */
            unsigned short IRQ3S:1;     /*    IRQ3S     */
            unsigned short :4;          /*              */
            } BIT;                      /*              */
      } ICR1;                           /*              */
union {                                 /* ISR          */
      unsigned short WORD;              /*  Word Access */
      struct {                          /*  Bit Access  */
            unsigned short :8;          /*              */
            unsigned short IRQ0F:1;     /*    IRQ0F     */
            unsigned short IRQ1F:1;     /*    IRQ1F     */
            unsigned short IRQ2F:1;     /*    IRQ2F     */
            unsigned short IRQ3F:1;     /*    IRQ3F     */
            unsigned short :4;          /*              */
            } BIT;                      /*              */
      } ISR;                            /*              */
union {                                 /* IPRI         */
      unsigned short WORD;              /*  Word Access */
      struct {                          /*  Bit Access  */
            unsigned short SCI2:4;      /*    SCI2      */
            unsigned short SCI3:4;      /*    SCI3      */
            unsigned short SCI4:4;      /*    SCI4      */
            unsigned short MMT:4;       /*    MMT       */
            } BIT;                      /*              */
      } IPRI;                           /*              */
union {                                 /* IPRJ         */
      unsigned short WORD;              /*  Word Access */
      struct {                          /*  Bit Access  */
            unsigned short AD2:4;       /*    A/D2      */
            unsigned short :12;         /*              */
            } BIT;                      /*              */
      } IPRJ;                           /*              */
union {                                 /* IPRK         */
      unsigned short WORD;              /*  Word Access */
      struct {                          /*  Bit Access  */
```

```
                    unsigned short IOMMT:4;          /*    I/O(MMT)  */
                    unsigned short :4;               /*              */
                    unsigned short HCAN2:4;          /*    HCAN1     */
                    unsigned short :4;               /*              */
                    } BIT;                           /*              */
              } IPRK;                                /*              */
        unsigned char wk1[4];                        /*              */
        union {                                      /* ICR2         */
              unsigned short WORD;                   /*  Word Access */
              struct {                               /*  Bit Access  */
                    unsigned short IRQ0ES:2;         /*    IRQ0ES    */
                    unsigned short IRQ1ES:2;         /*    IRQ1ES    */
                    unsigned short IRQ2ES:2;         /*    IRQ2ES    */
                    unsigned short IRQ3ES:2;         /*    IRQ3ES    */
                    unsigned short :8;               /*              */
                    } BIT;                           /*              */
              } ICR2;                                /*              */
};                                                   /*              */
struct st_porta {                                    /* struct PORTA */
        union {                                      /* PADRL        */
              unsigned short WORD;                   /*  Word Access */
              struct {                               /*  Bit Access  */
                    unsigned short PA15DR:1;         /*    PA15DR    */
                    unsigned short PA14DR:1;         /*    PA14DR    */
                    unsigned short PA13DR:1;         /*    PA13DR    */
                    unsigned short PA12DR:1;         /*    PA12DR    */
                    unsigned short PA11DR:1;         /*    PA11DR    */
                    unsigned short PA10DR:1;         /*    PA10DR    */
                    unsigned short PA9DR:1;          /*    PA9DR     */
                    unsigned short PA8DR:1;          /*    PA8DR     */
                    unsigned short PA7DR:1;          /*    PA7DR     */
                    unsigned short PA6DR:1;          /*    PA6DR     */
                    unsigned short PA5DR:1;          /*    PA5DR     */
                    unsigned short PA4DR:1;          /*    PA4DR     */
                    unsigned short PA3DR:1;          /*    PA3DR     */
                    unsigned short PA2DR:1;          /*    PA2DR     */
                    unsigned short PA1DR:1;          /*    PA1DR     */
                    unsigned short PA0DR:1;          /*    PA0DR     */
                    } BIT;                           /*              */
              } PADRL;                               /*              */
        unsigned char wk0[2];                        /*              */
        union {                                      /* PAIORL       */
              unsigned short WORD;                   /*  Word Access */
              struct {                               /*  Bit Access  */
                    unsigned short PA15IOR:1;        /*    PA15IOR   */
                    unsigned short PA14IOR:1;        /*    PA14IOR   */
                    unsigned short PA13IOR:1;        /*    PA13IOR   */
                    unsigned short PA12IOR:1;        /*    PA12IOR   */
                    unsigned short PA11IOR:1;        /*    PA11IOR   */
                    unsigned short PA10IOR:1;        /*    PA10IOR   */
                    unsigned short PA9IOR:1;         /*    PA9IOR    */
```

```
                unsigned short PA8IOR:1;          /*    PA8IOR    */
                unsigned short PA7IOR:1;          /*    PA7IOR    */
                unsigned short PA6IOR:1;          /*    PA6IOR    */
                unsigned short PA5IOR:1;          /*    PA5IOR    */
                unsigned short PA4IOR:1;          /*    PA4IOR    */
                unsigned short PA3IOR:1;          /*    PA3IOR    */
                unsigned short PA2IOR:1;          /*    PA2IOR    */
                unsigned short PA1IOR:1;          /*    PA1IOR    */
                unsigned short PA0IOR:1;          /*    PA0IOR    */
                } BIT;                            /*              */
        } PAIORL;                                 /*              */
    unsigned char wk1[2];                         /*              */
    union {                                       /* PACRL3       */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short PA15MD2:1;         /*    PA15MD2   */
                unsigned short PA14MD2:1;         /*    PA14MD2   */
                unsigned short PA13MD2:1;         /*    PA13MD2   */
                unsigned short PA12MD2:1;         /*    PA12MD2   */
                unsigned short PA11MD2:1;         /*    PA11MD2   */
                unsigned short PA10MD2:1;         /*    PA10MD2   */
                unsigned short PA9MD2:1;          /*    PA9MD2    */
                unsigned short PA8MD2:1;          /*    PA8MD2    */
                unsigned short PA7MD2:1;          /*    PA7MD2    */
                unsigned short PA6MD2:1;          /*    PA6MD2    */
                unsigned short PA5MD2:1;          /*    PA5MD2    */
                unsigned short PA4MD2:1;          /*    PA4MD2    */
                unsigned short PA3MD2:1;          /*    PA3MD2    */
                unsigned short PA2MD2:1;          /*    PA2MD2    */
                unsigned short PA1MD2:1;          /*    PA1MD2    */
                unsigned short PA0MD2:1;          /*    PA0MD2    */
                } BIT;                            /*              */
        } PACRL3;                                 /*              */
    union {                                       /* PACRL1       */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short PA15MD:2;          /*    PA15MD    */
                unsigned short PA14MD:2;          /*    PA14MD    */
                unsigned short PA13MD:2;          /*    PA13MD    */
                unsigned short PA12MD:2;          /*    PA12MD    */
                unsigned short PA11MD:2;          /*    PA11MD    */
                unsigned short PA10MD:2;          /*    PA10MD    */
                unsigned short PA9MD:2;           /*    PA9MD     */
                unsigned short PA8MD:2;           /*    PA8MD     */
                } BIT;                            /*              */
        } PACRL1;                                 /*              */
    union {                                       /* PACRL2       */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short PA7MD:2;           /*    PA7MD     */
                unsigned short PA6MD:2;           /*    PA6MD     */
```

```
                unsigned short PA5MD:2;          /*    PA5MD     */
                unsigned short PA4MD:2;          /*    PA4MD     */
                unsigned short PA3MD:2;          /*    PA3MD     */
                unsigned short PA2MD:2;          /*    PA2MD     */
                unsigned short PA1MD:2;          /*    PA1MD     */
                unsigned short PA0MD:2;          /*    PA0MD     */
                } BIT;                           /*              */
          } PACRL2;                              /*              */
};                                               /*              */
struct st_portb {                                /* struct PORTB */
      union {                                    /* PBDR         */
            unsigned short WORD;                 /*  Word Access */
            struct {                             /*  Bit Access  */
                unsigned short :10;              /*              */
                unsigned short PB5DR:1;          /*    PB5DR     */
                unsigned short PB4DR:1;          /*    PB4DR     */
                unsigned short PB3DR:1;          /*    PB3DR     */
                unsigned short PB2DR:1;          /*    PB2DR     */
                unsigned short PB1DR:1;          /*    PB1DR     */
                unsigned short PB0DR:1;          /*    PB0DR     */
                } BIT;                           /*              */
            } PBDR;                              /*              */
      unsigned char wk0[2];                      /*              */
      union {                                    /* PBIOR        */
            unsigned short WORD;                 /*  Word Access */
            struct {                             /*  Bit Access  */
                unsigned short :10;              /*              */
                unsigned short PB5IOR:1;         /*    PB5IOR    */
                unsigned short PB4IOR:1;         /*    PB4IOR    */
                unsigned short PB3IOR:1;         /*    PB3IOR    */
                unsigned short PB2IOR:1;         /*    PB2IOR    */
                unsigned short PB1IOR:1;         /*    PB1IOR    */
                unsigned short PB0IOR:1;         /*    PB0IOR    */
                } BIT;                           /*              */
            } PBIOR;                             /*              */
      unsigned char wk1[2];                      /*              */
      union {                                    /* PBCR1        */
            unsigned short WORD;                 /*  Word Access */
            struct {                             /*  Bit Access  */
                unsigned short :2;               /*              */
                unsigned short PB5MD2:1;         /*    PB5MD2    */
                unsigned short PB4MD2:1;         /*    PB4MD2    */
                unsigned short PB3MD2:1;         /*    PB3MD2    */
                unsigned short PB2MD2:1;         /*    PB2MD2    */
                unsigned short PB1MD2:1;         /*    PB1MD2    */
                unsigned short :9;               /*              */
                } BIT;                           /*              */
            } PBCR1;                             /*              */
      union {                                    /* PBCR2        */
            unsigned short WORD;                 /*  Word Access */
            struct {                             /*  Bit Access  */
```

```
                unsigned short :4;              /*              */
                unsigned short PB5MD:2;         /*    PB5MD     */
                unsigned short PB4MD:2;         /*    PB4MD     */
                unsigned short PB3MD:2;         /*    PB3MD     */
                unsigned short PB2MD:2;         /*    PB2MD     */
                unsigned short PB1MD:2;         /*    PB1MD     */
                unsigned short PB0MD:2;         /*    PB0MD     */
                } BIT;                          /*              */
          } PBCR2;                              /*              */
};                                              /*              */
struct st_portd {                               /* struct PORTD */
      union {                                   /* PDDRL        */
          unsigned short WORD;                  /*  Word Access */
          struct {                              /*  Bit Access  */
                unsigned short :7;              /*              */
                unsigned short PD8DR:1;         /*    PD8DR     */
                unsigned short PD7DR:1;         /*    PD7DR     */
                unsigned short PD6DR:1;         /*    PD6DR     */
                unsigned short PD5DR:1;         /*    PD5DR     */
                unsigned short PD4DR:1;         /*    PD4DR     */
                unsigned short PD3DR:1;         /*    PD3DR     */
                unsigned short PD2DR:1;         /*    PD2DR     */
                unsigned short PD1DR:1;         /*    PD1DR     */
                unsigned short PD0DR:1;         /*    PD0DR     */
                } BIT;                          /*              */
          } PDDRL;                              /*              */
      unsigned char wk0[2];                     /*              */
      union {                                   /* PDIORL       */
          unsigned short WORD;                  /*  Word Access */
          struct {                              /*  Bit Access  */
                unsigned short :7;              /*              */
                unsigned short PD8IOR:1;        /*    PD8IOR    */
                unsigned short PD7IOR:1;        /*    PD7IOR    */
                unsigned short PD6IOR:1;        /*    PD6IOR    */
                unsigned short PD5IOR:1;        /*    PD5IOR    */
                unsigned short PD4IOR:1;        /*    PD4IOR    */
                unsigned short PD3IOR:1;        /*    PD3IOR    */
                unsigned short PD2IOR:1;        /*    PD2IOR    */
                unsigned short PD1IOR:1;        /*    PD1IOR    */
                unsigned short PD0IOR:1;        /*    PD0IOR    */
                } BIT;                          /*              */
          } PDIORL;                             /*              */
      unsigned char wk1[4];                     /*              */
      union {                                   /* PDCRL1       */
          unsigned short WORD;                  /*  Word Access */
          struct {                              /*  Bit Access  */
                unsigned short :7;              /*              */
                unsigned short PD8MD0:1;        /*    PD8MD0    */
                unsigned short PD7MD0:1;        /*    PD7MD0    */
                unsigned short PD6MD0:1;        /*    PD6MD0    */
                unsigned short PD5MD0:1;        /*    PD5MD0    */
```

```
                unsigned short PD4MD0:1;          /*    PD4MD0    */
                unsigned short PD3MD0:1;          /*    PD3MD0    */
                unsigned short PD2MD0:1;          /*    PD2MD0    */
                unsigned short PD1MD0:1;          /*    PD1MD0    */
                unsigned short PD0MD0:1;          /*    PD0MD0    */
                } BIT;                            /*              */
          } PDCRL1;                               /*              */
      union {                                     /* PDCRL2       */
          unsigned short WORD;                    /*  Word Access */
          struct {                                /*  Bit Access  */
                unsigned short :7;                /*              */
                unsigned short PD8MD1:1;          /*    PD8MD1    */
                unsigned short PD7MD1:1;          /*    PD7MD1    */
                unsigned short PD6MD1:1;          /*    PD6MD1    */
                unsigned short PD5MD1:1;          /*    PD5MD1    */
                unsigned short PD4MD1:1;          /*    PD4MD1    */
                unsigned short PD3MD1:1;          /*    PD3MD1    */
                unsigned short PD2MD1:1;          /*    PD2MD1    */
                unsigned short PD1MD1:1;          /*    PD1MD1    */
                unsigned short PD0MD1:1;          /*    PD0MD1    */
                } BIT;                            /*              */
          } PDCRL2;                               /*              */
};                                                /*              */
struct st_porte {                                 /* struct PORTE */
      union {                                     /* PEDRL        */
          unsigned short WORD;                    /*  Word Access */
          struct {                                /*  Bit Access  */
                unsigned short PE15DR:1;          /*    PE15DR    */
                unsigned short PE14DR:1;          /*    PE14DR    */
                unsigned short PE13DR:1;          /*    PE13DR    */
                unsigned short PE12DR:1;          /*    PE12DR    */
                unsigned short PE11DR:1;          /*    PE11DR    */
                unsigned short PE10DR:1;          /*    PE10DR    */
                unsigned short PE9DR:1;           /*    PE9DR     */
                unsigned short PE8DR:1;           /*    PE8DR     */
                unsigned short PE7DR:1;           /*    PE7DR     */
                unsigned short PE6DR:1;           /*    PE6DR     */
                unsigned short PE5DR:1;           /*    PE5DR     */
                unsigned short PE4DR:1;           /*    PE4DR     */
                unsigned short PE3DR:1;           /*    PE3DR     */
                unsigned short PE2DR:1;           /*    PE2DR     */
                unsigned short PE1DR:1;           /*    PE1DR     */
                unsigned short PE0DR:1;           /*    PE0DR     */
                } BIT;                            /*              */
          } PEDRL;                                /*              */
      unsigned char wk0[2];                       /*              */
      union {                                     /* PEIORL       */
          unsigned short WORD;                    /*  Word Access */
          struct {                                /*  Bit Access  */
                unsigned short PE15IOR:1;         /*    PE15IOR   */
                unsigned short PE14IOR:1;         /*    PE14IOR   */
```

```
                unsigned short PE13IOR:1;          /*    PE13IOR   */
                unsigned short PE12IOR:1;          /*    PE12IOR   */
                unsigned short PE11IOR:1;          /*    PE11IOR   */
                unsigned short PE10IOR:1;          /*    PE10IOR   */
                unsigned short PE9IOR:1;           /*    PE9IOR    */
                unsigned short PE8IOR:1;           /*    PE8IOR    */
                unsigned short PE7IOR:1;           /*    PE7IOR    */
                unsigned short PE6IOR:1;           /*    PE6IOR    */
                unsigned short PE5IOR:1;           /*    PE5IOR    */
                unsigned short PE4IOR:1;           /*    PE4IOR    */
                unsigned short PE3IOR:1;           /*    PE3IOR    */
                unsigned short PE2IOR:1;           /*    PE2IOR    */
                unsigned short PE1IOR:1;           /*    PE1IOR    */
                unsigned short PE0IOR:1;           /*    PE0IOR    */
                } BIT;                             /*              */
        } PEIORL;                                  /*              */
  union {                                          /* PEIORH       */
        unsigned short WORD;                       /*  Word Access */
        struct {                                   /*  Bit Access  */
                unsigned short :10;                /*              */
                unsigned short PE21IOR:1;          /*    PE21IOR   */
                unsigned short PE20IOR:1;          /*    PE20IOR   */
                unsigned short PE19IOR:1;          /*    PE19IOR   */
                unsigned short PE18IOR:1;          /*    PE18IOR   */
                unsigned short PE17IOR:1;          /*    PE17IOR   */
                unsigned short PE16IOR:1;          /*    PE16IOR   */
                } BIT;                             /*              */
        } PEIORH;                                  /*              */
  union {                                          /* PECRL1       */
        unsigned short WORD;                       /*  Word Access */
        struct {                                   /*  Bit Access  */
                unsigned short PE15MD:2;           /*    PE15MD    */
                unsigned short PE14MD:2;           /*    PE14MD    */
                unsigned short PE13MD:2;           /*    PE13MD    */
                unsigned short PE12MD:2;           /*    PE12MD    */
                unsigned short PE11MD:2;           /*    PE11MD    */
                unsigned short PE10MD:2;           /*    PE10MD    */
                unsigned short PE9MD:2;            /*    PE9MD     */
                unsigned short PE8MD:2;            /*    PE8MD     */
                } BIT;                             /*              */
        } PECRL1;                                  /*              */
  union {                                          /* PECRL2       */
        unsigned short WORD;                       /*  Word Access */
        struct {                                   /*  Bit Access  */
                unsigned short PE7MD:2;            /*    PE7MD     */
                unsigned short PE6MD:2;            /*    PE6MD     */
                unsigned short PE5MD:2;            /*    PE5MD     */
                unsigned short PE4MD:2;            /*    PE4MD     */
                unsigned short PE3MD:2;            /*    PE3MD     */
                unsigned short PE2MD:2;            /*    PE2MD     */
                unsigned short PE1MD:2;            /*    PE1MD     */
```

```
                unsigned short PE0MD:2;          /*    PE0MD     */
                } BIT;                           /*              */
        } PECRL2;                                /*              */
        union {                                  /* PECRH        */
                unsigned short WORD;             /*  Word Access */
                struct {                         /*  Bit Access  */
                unsigned short :4;               /*              */
                unsigned short PE21MD:2;         /*    PE21MD    */
                unsigned short PE20MD:2;         /*    PE20MD    */
                unsigned short PE19MD:2;         /*    PE19MD    */
                unsigned short PE18MD:2;         /*    PE18MD    */
                unsigned short PE17MD:2;         /*    PE17MD    */
                unsigned short PE16MD:2;         /*    PE16MD    */
                } BIT;                           /*              */
        } PECRH;                                 /*              */
        union {                                  /* PEDRH        */
                unsigned short WORD;             /*  Word Access */
                struct {                         /*  Bit Access  */
                unsigned short :10;              /*              */
                unsigned short PE21DR:1;         /*    PE21DR    */
                unsigned short PE20DR:1;         /*    PE20DR    */
                unsigned short PE19DR:1;         /*    PE19DR    */
                unsigned short PE18DR:1;         /*    PE18DR    */
                unsigned short PE17DR:1;         /*    PE17DR    */
                unsigned short PE16DR:1;         /*    PE16DR    */
                } BIT;                           /*              */
        } PEDRH;                                 /*              */
};                                               /*              */
struct st_portf {                                /* struct PORTF */
        union {                                  /* PFDR         */
                unsigned short WORD;             /*  Word Access */
                struct {                         /*  Bit Access  */
                unsigned short PF15DR:1;         /*    PF15DR    */
                unsigned short PF14DR:1;         /*    PF14DR    */
                unsigned short PF13DR:1;         /*    PF13DR    */
                unsigned short PF12DR:1;         /*    PF12DR    */
                unsigned short PF11DR:1;         /*    PF11DR    */
                unsigned short PF10DR:1;         /*    PF10DR    */
                unsigned short PF9DR:1;          /*    PF9DR     */
                unsigned short PF8DR:1;          /*    PF8DR     */
                unsigned short PF7DR:1;          /*    PF7DR     */
                unsigned short PF6DR:1;          /*    PF6DR     */
                unsigned short PF5DR:1;          /*    PF5DR     */
                unsigned short PF4DR:1;          /*    PF4DR     */
                unsigned short PF3DR:1;          /*    PF3DR     */
                unsigned short PF2DR:1;          /*    PF2DR     */
                unsigned short PF1DR:1;          /*    PF1DR     */
                unsigned short PF0DR:1;          /*    PF0DR     */
                } BIT;                           /*              */
        } PFDR;                                  /*              */
};                                               /*              */
```

```
struct st_mtu {                                 /* struct MTU   */
       union {                                  /* ICSR1        */
             unsigned short WORD;               /*  Word Access */
             struct {                           /*  Bit Access  */
                    unsigned short POE3F:1;     /*    POE3F     */
                    unsigned short POE2F:1;     /*    POE2F     */
                    unsigned short POE1F:1;     /*    POE1F     */
                    unsigned short POE0F:1;     /*    POE0F     */
                    unsigned short :3;          /*              */
                    unsigned short PIE:1;       /*    PIE       */
                    unsigned short POE3M:2;     /*    POE3M     */
                    unsigned short POE2M:2;     /*    POE2M     */
                    unsigned short POE1M:2;     /*    POE1M     */
                    unsigned short POE0M:2;     /*    POE0M     */
                    } BIT;                      /*              */
             } ICSR1;                           /*              */
       union {                                  /* OCSR         */
             unsigned short WORD;               /*  Word Access */
             struct {                           /*  Bit Access  */
                    unsigned short OSF:1;       /*    OSF       */
                    unsigned short :5;          /*              */
                    unsigned short OCE:1;       /*    OCE       */
                    unsigned short OIE:1;       /*    OIE       */
                    unsigned short :8;          /*              */
                    } BIT;                      /*              */
             } OCSR;                            /*              */
};                                              /*              */
struct st_mmt {                                 /* struct MMT   */
       union {                                  /* ICSR2        */
             unsigned short WORD;               /*  Word Access */
             struct {                           /*  Bit Access  */
                    unsigned short :1;          /*              */
                    unsigned short POE6F:1;     /*    POE6F     */
                    unsigned short POE5F:1;     /*    POE5F     */
                    unsigned short POE4F:1;     /*    POE4F     */
                    unsigned short :3;          /*              */
                    unsigned short PIE:1;       /*    PIE       */
                    unsigned short :2;          /*              */
                    unsigned short POE6M:2;     /*    POE6M     */
                    unsigned short POE5M:2;     /*    POE5M     */
                    unsigned short POE4M:2;     /*    POE4M     */
                    } BIT;                      /*              */
             } ICSR2;                           /*              */
       unsigned char wk0[1594];                 /*              */
       union {                                  /* MMT_TMDR     */
             unsigned char BYTE;                /*  Byte Access */
             struct {                           /*  Bit Access  */
                    unsigned char CKS:4;        /*    CKS       */
                    unsigned char OLSN:1;       /*    OLSN      */
                    unsigned char OLSP:1;       /*    OLSP      */
                    unsigned char MD:2;         /*    MD        */
```

```
            } BIT;                                  /*              */
      } MMT_TMDR;                                   /*              */
unsigned char wk1[1];                               /*              */
union {                                             /* TCNR         */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char TTGE:1;                   /*    TTGE      */
            unsigned char CST:1;                    /*    CST       */
            unsigned char RPRO:1;                   /*    RPRO      */
            unsigned char :3;                       /*              */
            unsigned char TGIEN:1;                  /*    TGIEN     */
            unsigned char TGIEM:1;                  /*    TGIEM     */
            } BIT;                                  /*              */
      } TCNR;                                       /*              */
unsigned char wk2[1];                               /*              */
union {                                             /* MMT_TSR      */
      unsigned char BYTE;                           /*  Byte Access */
      struct {                                      /*  Bit Access  */
            unsigned char TCFD:1;                   /*    TCFD      */
            unsigned char :5;                       /*              */
            unsigned char TGFN:1;                   /*    TGFN      */
            unsigned char TGFM:1;                   /*    TGFM      */
            } BIT;                                  /*              */
      } MMT_TSR;                                    /*              */
unsigned char wk3[1];                               /*              */
unsigned short MMT_TCNT;                            /* MMT_TCNT     */
unsigned short TPDR;                                /* TPDR         */
unsigned short TPBR;                                /* TPBR         */
unsigned short MMT_TDDR;                            /* MMT_TDDR     */
unsigned char wk4[2];                               /*              */
unsigned short TBRU_B;                              /* TBRU_B       */
unsigned short TGRUU;                               /* TGRUU        */
unsigned short TGRU;                                /* TGRU         */
unsigned short TGRUD;                               /* TGRUD        */
unsigned short TDCNT0;                              /* TDCNT0       */
unsigned short TDCNT1;                              /* TDCNT1       */
unsigned short TBRU_F;                              /* TBRU_F       */
unsigned char wk5[2];                               /*              */
unsigned short TBRV_B;                              /* TBRV_B       */
unsigned short TGRVU;                               /* TGRVU        */
unsigned short TGRV;                                /* TGRV         */
unsigned short TGRVD;                               /* TGRVD        */
unsigned short TDCNT2;                              /* TDCNT2       */
unsigned short TDCNT3;                              /* TDCNT3       */
unsigned short TBRV_F;                              /* TBRV_F       */
unsigned char wk6[2];                               /*              */
unsigned short TBRW_B;                              /* TBRW_B       */
unsigned short TGRWU;                               /* TGRWU        */
unsigned short TGRW;                                /* TGRW         */
unsigned short TGRWD;                               /* TGRWD        */
unsigned short TDCNT4;                              /* TDCNT4       */
```

```
        unsigned short TDCNT5;                  /* TDCNT5       */
        unsigned short TBRW_F;                  /* TBRW_F       */
};                                              /*              */
struct st_portg {                               /* struct PORTG */
        union {                                 /* PGDR         */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char :4;          /*              */
                     unsigned char PG3DR:1;     /*    PG3DR     */
                     unsigned char PG2DR:1;     /*    PG2DR     */
                     unsigned char PG1DR:1;     /*    PG1DR     */
                     unsigned char PG0DR:1;     /*    PG0DR     */
                     } BIT;                     /*              */
              } PGDR;                           /*              */
};                                              /*              */
struct st_cmt {                                 /* struct CMT   */
        union {                                 /* CMSTR        */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                     unsigned short :14;        /*              */
                     unsigned short STR:2;      /*    STR       */
                     } BIT;                     /*              */
              } CMSTR;                          /*              */
        union {                                 /* CMCSR_0      */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                     unsigned short :8;         /*              */
                     unsigned short CMF:1;      /*    CMF       */
                     unsigned short CMIE:1;     /*    CMIE      */
                     unsigned short :4;         /*              */
                     unsigned short CKS:2;      /*    CKS       */
                     } BIT;                     /*              */
              } CMCSR_0;                        /*              */
        unsigned short CMCNT_0;                 /* CMCNT_0      */
        unsigned short CMCOR_0;                 /* CMCOR_0      */
        union {                                 /* CMCSR_1      */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                     unsigned short :8;         /*              */
                     unsigned short CMF:1;      /*    CMF       */
                     unsigned short CMIE:1;     /*    CMIE      */
                     unsigned short :4;         /*              */
                     unsigned short CKS:2;      /*    CKS       */
                     } BIT;                     /*              */
              } CMCSR_1;                        /*              */
        unsigned short CMCNT_1;                 /* CMCNT_1      */
        unsigned short CMCOR_1;                 /* CMCOR_1      */
};                                              /*              */
struct st_ad {                                  /* struct AD    */
        union {                                 /* ADDR0        */
              unsigned short WORD;              /*  Word Access */
```

```
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR0;                                /*              */
union {                                         /* ADDR1        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR1;                                /*              */
union {                                         /* ADDR2        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR2;                                /*              */
union {                                         /* ADDR3        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR3;                                /*              */
union {                                         /* ADDR4        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
```

```
                } BIT;                          /*              */
        } ADDR4;                                /*              */
union {                                         /* ADDR5        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
                unsigned char ADH;              /*    AD H      */
                unsigned char wk;               /*              */
                } BYTE;                         /*              */
        struct {                                /*  Bit Access  */
                unsigned short AD:10;           /*    AD        */
                unsigned short :6;              /*              */
                } BIT;                          /*              */
        } ADDR5;                                /*              */
union {                                         /* ADDR6        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
                unsigned char ADH;              /*    AD H      */
                unsigned char wk;               /*              */
                } BYTE;                         /*              */
        struct {                                /*  Bit Access  */
                unsigned short AD:10;           /*    AD        */
                unsigned short :6;              /*              */
                } BIT;                          /*              */
        } ADDR6;                                /*              */
union {                                         /* ADDR7        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
                unsigned char ADH;              /*    AD H      */
                unsigned char wk;               /*              */
                } BYTE;                         /*              */
        struct {                                /*  Bit Access  */
                unsigned short AD:10;           /*    AD        */
                unsigned short :6;              /*              */
                } BIT;                          /*              */
        } ADDR7;                                /*              */
union {                                         /* ADDR8        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
                unsigned char ADH;              /*    AD H      */
                unsigned char wk;               /*              */
                } BYTE;                         /*              */
        struct {                                /*  Bit Access  */
                unsigned short AD:10;           /*    AD        */
                unsigned short :6;              /*              */
                } BIT;                          /*              */
        } ADDR8;                                /*              */
union {                                         /* ADDR9        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
                unsigned char ADH;              /*    AD H      */
                unsigned char wk;               /*              */
```

```
            } BYTE;                         /*              */
      struct {                              /*  Bit Access  */
            unsigned short AD:10;           /*    AD        */
            unsigned short :6;              /*              */
            } BIT;                          /*              */
      } ADDR9;                              /*              */
union {                                     /* ADDR10       */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Byte Access */
            unsigned char ADH;              /*    AD H      */
            unsigned char wk;               /*              */
            } BYTE;                         /*              */
      struct {                              /*  Bit Access  */
            unsigned short AD:10;           /*    AD        */
            unsigned short :6;              /*              */
            } BIT;                          /*              */
      } ADDR10;                             /*              */
union {                                     /* ADDR11       */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Byte Access */
            unsigned char ADH;              /*    AD H      */
            unsigned char wk;               /*              */
            } BYTE;                         /*              */
      struct {                              /*  Bit Access  */
            unsigned short AD:10;           /*    AD        */
            unsigned short :6;              /*              */
            } BIT;                          /*              */
      } ADDR11;                             /*              */
union {                                     /* ADDR12       */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Byte Access */
            unsigned char ADH;              /*    AD H      */
            unsigned char wk;               /*              */
            } BYTE;                         /*              */
      struct {                              /*  Bit Access  */
            unsigned short AD:10;           /*    AD        */
            unsigned short :6;              /*              */
            } BIT;                          /*              */
      } ADDR12;                             /*              */
union {                                     /* ADDR13       */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Byte Access */
            unsigned char ADH;              /*    AD H      */
            unsigned char wk;               /*              */
            } BYTE;                         /*              */
      struct {                              /*  Bit Access  */
            unsigned short AD:10;           /*    AD        */
            unsigned short :6;              /*              */
            } BIT;                          /*              */
      } ADDR13;                             /*              */
union {                                     /* ADDR14       */
```

```
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR14;                               /*              */
  union {                                       /* ADDR15       */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR15;                               /*              */
  union {                                       /* ADDR16       */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR16;                               /*              */
  union {                                       /* ADDR17       */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
               unsigned short :6;               /*              */
               } BIT;                           /*              */
        } ADDR17;                               /*              */
  union {                                       /* ADDR18       */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Byte Access */
               unsigned char ADH;               /*    AD H      */
               unsigned char wk;                /*              */
               } BYTE;                          /*              */
        struct {                                /*  Bit Access  */
               unsigned short AD:10;            /*    AD        */
```

```
            unsigned short :6;              /*             */
            } BIT;                          /*             */
      } ADDR18;                             /*             */
union {                                     /* ADDR19      */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Byte Access */
            unsigned char ADH;              /*    AD H      */
            unsigned char wk;               /*              */
            } BYTE;                         /*              */
      struct {                              /*  Bit Access  */
            unsigned short AD:10;           /*    AD        */
            unsigned short :6;              /*              */
            } BIT;                          /*              */
      } ADDR19;                             /*              */
unsigned char wk0[56];                      /*              */
union {                                     /* ADCSR_0     */
      unsigned char BYTE;                   /*  Byte Access */
      struct {                              /*  Bit Access  */
            unsigned char ADF:1;            /*    ADF       */
            unsigned char ADIE:1;           /*    ADIE      */
            unsigned char ADM:2;            /*    ADM       */
            unsigned char :1;               /*              */
            unsigned char CH:3;             /*    CH        */
            } BIT;                          /*              */
      } ADCSR_0;                            /*              */
union {                                     /* ADCSR_1     */
      unsigned char BYTE;                   /*  Byte Access */
      struct {                              /*  Bit Access  */
            unsigned char ADF:1;            /*    ADF       */
            unsigned char ADIE:1;           /*    ADIE      */
            unsigned char ADM:2;            /*    ADM       */
            unsigned char :1;               /*              */
            unsigned char CH:3;             /*    CH        */
            } BIT;                          /*              */
      } ADCSR_1;                            /*              */
union {                                     /* ADCSR_2     */
      unsigned char BYTE;                   /*  Byte Access */
      struct {                              /*  Bit Access  */
            unsigned char ADF:1;            /*    ADF       */
            unsigned char ADIE:1;           /*    ADIE      */
            unsigned char ADM:2;            /*    ADM       */
            unsigned char :1;               /*              */
            unsigned char CH:3;             /*    CH        */
            } BIT;                          /*              */
      } ADCSR_2;                            /*              */
unsigned char wk1[5];                       /*              */
union {                                     /* ADCR_0      */
      unsigned char BYTE;                   /*  Byte Access */
      struct {                              /*  Bit Access  */
            unsigned char TRGE:1;           /*    TRGE      */
            unsigned char CKS:2;            /*    CKS       */
```

```
                unsigned char ADST:1;           /*    ADST      */
                unsigned char ADCS:1;           /*    ADCS      */
                unsigned char :3;               /*              */
                } BIT;                          /*              */
          } ADCR_0;                             /*              */
    union {                                     /* ADCR_1       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char TRGE:1;           /*    TRGE      */
                unsigned char CKS:2;            /*    CKS       */
                unsigned char ADST:1;           /*    ADST      */
                unsigned char ADCS:1;           /*    ADCS      */
                unsigned char :3;               /*              */
                } BIT;                          /*              */
          } ADCR_1;                             /*              */
    union {                                     /* ADCR_2       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char TRGE:1;           /*    TRGE      */
                unsigned char CKS:2;            /*    CKS       */
                unsigned char ADST:1;           /*    ADST      */
                unsigned char ADCS:1;           /*    ADCS      */
                unsigned char :3;               /*              */
                } BIT;                          /*              */
          } ADCR_2;                             /*              */
    unsigned char wk2[873];                     /*              */
    union {                                     /* ADTSR        */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char :2;               /*              */
                unsigned char TRG2S:2;          /*    TRG2S     */
                unsigned char TRG1S:2;          /*    TRG1S     */
                unsigned char TRG0S:2;          /*    TRG0S     */
                } BIT;                          /*              */
          } ADTSR;                              /*              */
};                                              /*              */
struct st_flash {                               /* struct FLASH */
    union {                                     /* FLMCR1       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char FWE:1;            /*    FWE       */
                unsigned char SWE:1;            /*    SWE       */
                unsigned char ESU:1;            /*    ESU       */
                unsigned char PSU:1;            /*    PSU       */
                unsigned char EV:1;             /*    EV        */
                unsigned char PV:1;             /*    PV        */
                unsigned char E:1;              /*    E         */
                unsigned char P:1;              /*    P         */
                } BIT;                          /*              */
          } FLMCR1;                             /*              */
    union {                                     /* FLMCR2       */
```

```
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char FLER:1;             /*    FLER      */
                  unsigned char :7;                 /*              */
                  } BIT;                            /*              */
            } FLMCR2;                               /*              */
      union {                                       /* EBR1         */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char EB:8;               /*    EB        */
                  } BIT;                            /*              */
            } EBR1;                                 /*              */
      union {                                       /* EBR2         */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char :4;                 /*              */
                  unsigned char EB11:1;             /*    EB11      */
                  unsigned char EB10:1;             /*    EB10      */
                  unsigned char EB9:1;              /*    EB9       */
                  unsigned char EB8:1;              /*    EB8       */
                  } BIT;                            /*              */
            } EBR2;                                 /*              */
      unsigned char wk0[164];                       /*              */
      union {                                       /* RAMER        */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                  unsigned short :12;               /*              */
                  unsigned short RAMS:1;            /*    RAMS      */
                  unsigned short RAM:3;             /*    RAM       */
                  } BIT;                            /*              */
            } RAMER;                                /*              */
};                                                  /*              */
struct st_ubc {                                     /* struct UBC   */
      unsigned short UBARH;                         /* UBARH        */
      unsigned short UBARL;                         /* UBARL        */
      union {                                       /* UBAMRH       */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                  unsigned short UBM31:1;           /*    UBM31     */
                  unsigned short UBM30:1;           /*    UBM30     */
                  unsigned short UBM29:1;           /*    UBM29     */
                  unsigned short UBM28:1;           /*    UBM28     */
                  unsigned short UBM27:1;           /*    UBM27     */
                  unsigned short UBM26:1;           /*    UBM26     */
                  unsigned short UBM25:1;           /*    UBM25     */
                  unsigned short UBM24:1;           /*    UBM24     */
                  unsigned short UBM23:1;           /*    UBM23     */
                  unsigned short UBM22:1;           /*    UBM22     */
                  unsigned short UBM21:1;           /*    UBM21     */
                  unsigned short UBM20:1;           /*    UBM20     */
                  unsigned short UBM19:1;           /*    UBM19     */
```

```
                unsigned short UBM18:1;          /*    UBM18     */
                unsigned short UBM17:1;          /*    UBM17     */
                unsigned short UBM16:1;          /*    UBM16     */
                } BIT;                           /*              */
          } UBAMRH;                              /*              */
      union {                                    /* UBAMRL       */
          unsigned short WORD;                   /*  Word Access */
          struct {                               /*  Bit Access  */
                unsigned short UBM15:1;          /*    UBM15     */
                unsigned short UBM14:1;          /*    UBM14     */
                unsigned short UBM13:1;          /*    UBM13     */
                unsigned short UBM12:1;          /*    UBM12     */
                unsigned short UBM11:1;          /*    UBM11     */
                unsigned short UBM10:1;          /*    UBM10     */
                unsigned short UBM9:1;           /*    UBM9      */
                unsigned short UBM8:1;           /*    UBM8      */
                unsigned short UBM7:1;           /*    UBM7      */
                unsigned short UBM6:1;           /*    UBM6      */
                unsigned short UBM5:1;           /*    UBM5      */
                unsigned short UBM4:1;           /*    UBM4      */
                unsigned short UBM3:1;           /*    UBM3      */
                unsigned short UBM2:1;           /*    UBM2      */
                unsigned short UBM1:1;           /*    UBM1      */
                unsigned short UBM0:1;           /*    UBM0      */
                } BIT;                           /*              */
          } UBAMRL;                              /*              */
      union {                                    /* UBBR         */
          unsigned short WORD;                   /*  Word Access */
          struct {                               /*  Bit Access  */
                unsigned short :8;               /*              */
                unsigned short CP:2;             /*    CP        */
                unsigned short ID:2;             /*    ID        */
                unsigned short RW:2;             /*    RW        */
                unsigned short SZ:2;             /*    SZ        */
                } BIT;                           /*              */
          } UBBR;                                /*              */
      union {                                    /* UBCR         */
          unsigned short WORD;                   /*  Word Access */
          struct {                               /*  Bit Access  */
                unsigned short :13;              /*              */
                unsigned short CKS:2;            /*    CKS       */
                unsigned short UBID:1;           /*    UBID      */
                } BIT;                           /*              */
          } UBCR;                                /*              */
};                                               /*              */
struct st_wdt {                                  /* struct WDT   */
      union {                                    /* TCSR         */
          unsigned char BYTE;                    /*  Byte Access */
          struct {                               /*  Bit Access  */
                unsigned char OVF:1;             /*    OVF       */
                unsigned char WTIT:1;            /*    WT/IT     */
```

```
                unsigned char TME:1;            /*    TME       */
                unsigned char :2;               /*              */
                unsigned char CKS:3;            /*    CKS       */
                } BIT;                          /*              */
          } TCSR;                               /*              */
    unsigned char TCNT;                         /* TCNT         */
    union {                                     /* RSTCSR       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char WOVF:1;           /*    WOVF      */
                unsigned char RSTE:1;           /*    RSTE      */
                unsigned char RSTS:1;           /*    RSTS      */
                unsigned char :5;               /*              */
                } BIT;                          /*              */
          } RSTCSR;                             /*              */
};                                              /*              */
struct st_stby {                                /* struct STBY  */
    union {                                     /* SBYCR        */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char SSBY:1;           /*    SSBY      */
                unsigned char HIZ:1;            /*    HIZ       */
                unsigned char :5;               /*              */
                unsigned char IRQEL:1;          /*    IRQEL     */
                } BIT;                          /*              */
          } SBYCR;                              /*              */
    unsigned char wk0[3];                       /*              */
    union {                                     /* SYSCR        */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char :6;               /*              */
                unsigned char AUDSRST:1;        /*    AUDSRST   */
                unsigned char RAME:1;           /*    RAME      */
                } BIT;                          /*              */
          } SYSCR;                              /*              */
    unsigned char wk1[3];                       /*              */
    union {                                     /* MSTCR1       */
          unsigned short WORD;                  /*  Word Access */
          struct {                              /*  Bit Access  */
                unsigned short :4;              /*              */
                unsigned short MSTP27:1;        /*    MSTP27    */
                unsigned short MSTP26:1;        /*    MSTP26    */
                unsigned short MSTP25:1;        /*    MSTP25    */
                unsigned short MSTP24:1;        /*    MSTP24    */
                unsigned short :3;              /*              */
                unsigned short MSTP20:1;        /*    MSTP20    */
                unsigned short MSTP19:1;        /*    MSTP19    */
                unsigned short MSTP18:1;        /*    MSTP18    */
                unsigned short :2;              /*              */
                } BIT;                          /*              */
          } MSTCR1;                             /*              */
```

```
        union {                                 /* MSTCR2       */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short :1;          /*              */
                    unsigned short MSTP14:1;    /*    MSTP14    */
                    unsigned short MSTP13:1;    /*    MSTP13    */
                    unsigned short MSTP12:1;    /*    MSTP12    */
                    unsigned short :2;          /*              */
                    unsigned short MSTP9:1;     /*    MSTP9     */
                    unsigned short :2;          /*              */
                    unsigned short MSTP6:1;     /*    MSTP6     */
                    unsigned short MSTP5:1;     /*    MSTP5     */
                    unsigned short MSTP4:1;     /*    MSTP4     */
                    unsigned short MSTP3:1;     /*    MSTP3     */
                    unsigned short MSTP2:1;     /*    MSTP2     */
                    unsigned short :1;          /*              */
                    unsigned short MSTP0:1;     /*    MSTP0     */
                    } BIT;                      /*              */
              } MSTCR2;                         /*              */
};                                              /*              */
struct st_bsc {                                 /* struct BSC   */
        union {                                 /* BCR1         */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short :1;          /*              */
                    unsigned short MMTRWE:1;    /*    MMTRWE    */
                    unsigned short MTURWE:1;    /*    MTURWE    */
                    unsigned short :12;         /*              */
                    unsigned short A0SZ:1;      /*    A0SZ      */
                    } BIT;                      /*              */
              } BCR1;                           /*              */
        union {                                 /* BCR2         */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short :6;          /*              */
                    unsigned short IW:2;        /*    IW        */
                    unsigned short :3;          /*              */
                    unsigned short CW0:1;       /*    CW0       */
                    unsigned short :3;          /*              */
                    unsigned short SW0:1;       /*    SW0       */
                    } BIT;                      /*              */
              } BCR2;                           /*              */
        union {                                 /* WCR1         */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short :12;         /*              */
                    unsigned short W:4;         /*    W         */
                    } BIT;                      /*              */
              } WCR1;                           /*              */
};                                              /*              */
struct st_dtc {                                 /* struct DTC   */
```

```
union {                                 /* DTEA         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char TGI4A:1;     /*              */
             unsigned char TGI4B:1;     /*              */
             unsigned char TGI4C:1;     /*              */
             unsigned char TGI4D:1;     /*              */
             unsigned char TGI4V:1;     /*              */
             unsigned char TGI3A:1;     /*              */
             unsigned char TGI3B:1;     /*              */
             unsigned char TGI3C:1;     /*              */
             } BIT;                     /*              */
      } DTEA;                           /*              */
union {                                 /* DTEB         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char TGI3D:1;     /*              */
             unsigned char TGI2A:1;     /*              */
             unsigned char TGI2B:1;     /*              */
             unsigned char TGI1A:1;     /*              */
             unsigned char TGI1B:1;     /*              */
             unsigned char TGI0A:1;     /*              */
             unsigned char TGI0B:1;     /*              */
             unsigned char TGI0C:1;     /*              */
             } BIT;                     /*              */
      } DTEB;                           /*              */
union {                                 /* DTEC         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char TGI0D:1;     /*              */
             unsigned char ADI0:1;      /*              */
             unsigned char IRQ0:1;      /*              */
             unsigned char IRQ1:1;      /*              */
             unsigned char IRQ2:1;      /*              */
             unsigned char IRQ3:1;      /*              */
             unsigned char b1:1;        /*              */
             unsigned char b0:1;        /*              */
             } BIT;                     /*              */
      } DTEC;                           /*              */
union {                                 /* DTED         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char b7:1;        /*              */
             unsigned char b6:1;        /*              */
             unsigned char CMI0:1;      /*              */
             unsigned char CMI1:1;      /*              */
             unsigned char b3:1;        /*              */
             unsigned char b2:1;        /*              */
             unsigned char b1:1;        /*              */
             unsigned char b0:1;        /*              */
             } BIT;                     /*              */
```

```
            } DTED;                                 /*              */
      unsigned char wk0[2];                         /*              */
      union {                                       /* DTCSR        */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                  unsigned short :5;                /*              */
                  unsigned short NMIF:1;            /*    NMIF      */
                  unsigned short AE:1;              /*    AE        */
                  unsigned short SWDTE:1;           /*    SWDTE     */
                  unsigned char DTVEC7:1;           /*    DTVEC7    */
                  unsigned char DTVEC6:1;           /*    DTVEC6    */
                  unsigned char DTVEC5:1;           /*    DTVEC5    */
                  unsigned char DTVEC4:1;           /*    DTVEC4    */
                  unsigned char DTVEC3:1;           /*    DTVEC3    */
                  unsigned char DTVEC2:1;           /*    DTVEC2    */
                  unsigned char DTVEC1:1;           /*    DTVEC1    */
                  unsigned char DTVEC0:1;           /*    DTVEC0    */
                  } BIT;                            /*              */
            } DTCSR;                                /*              */
      unsigned short DTBR;                          /* DTBR         */
      unsigned char wk1[6];                         /*              */
      union {                                       /* DTEE         */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char b7:1;               /*              */
                  unsigned char b6:1;               /*              */
                  unsigned char ADI1:1;             /*              */
                  unsigned char ADI2:1;             /*              */
                  unsigned char RXI_2:1;            /*              */
                  unsigned char TXI_2:1;            /*              */
                  unsigned char RXI_3:1;            /*              */
                  unsigned char TXI_3:1;            /*              */
                  } BIT;                            /*              */
            } DTEE;                                 /*              */
      union {                                       /* DTEF         */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char RXI_4:1;            /*              */
                  unsigned char TXI_4:1;            /*              */
                  unsigned char TGN:1;              /*              */
                  unsigned char TGM:1;              /*              */
                  unsigned char b3:1;               /*              */
                  unsigned char RM1:1;              /*              */
                  unsigned char b1:1;               /*              */
                  unsigned char b0:1;               /*              */
                  } BIT;                            /*              */
            } DTEF;                                 /*              */
};                                                  /*              */
struct st_hudi {                                    /* struct HUDI  */
      union {                                       /* SDIR         */
            unsigned short WORD;                    /*  Word Access */
```

```
            struct {                                    /*  Bit Access  */
                   unsigned short TS:4;                 /*    TS        */
                   unsigned short :12;                  /*              */
                   } BIT;                               /*              */
            } SDIR;                                     /*              */
      union {                                           /* SDSR         */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                   unsigned short :15;                  /*              */
                   unsigned short SDTRF:1;              /*    SDTRF     */
                   } BIT;                               /*              */
            } SDSR;                                     /*              */
      unsigned short SDDRH;                             /* SDDRH        */
      unsigned short SDDRL;                             /* SDDRL        */
};                                                      /*              */
struct st_hcan2 {                                       /* struct HCAN2 */
      union {                                           /* MCR          */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                   unsigned short :8;                   /*              */
                   unsigned short MCR7:1;               /*    MCR7      */
                   unsigned short :1;                   /*              */
                   unsigned short MCR5:1;               /*    MCR5      */
                   unsigned short :2;                   /*              */
                   unsigned short MCR2:1;               /*    MCR2      */
                   unsigned short MCR1:1;               /*    MCR1      */
                   unsigned short MCR0:1;               /*    MCR0      */
                   } BIT;                               /*              */
            } MCR;                                      /*              */
      union {                                           /* GSR          */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                   unsigned short :10;                  /*              */
                   unsigned short GSR5:1;               /*    GSR5      */
                   unsigned short GSR4:1;               /*    GSR4      */
                   unsigned short GSR3:1;               /*    GSR3      */
                   unsigned short GSR2:1;               /*    GSR2      */
                   unsigned short GSR1:1;               /*    GSR1      */
                   unsigned short GSR0:1;               /*    GSR0      */
                   } BIT;                               /*              */
            } GSR;                                      /*              */
      union {                                           /* HCAN2_BCR1   */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                   unsigned short TSG1:4;               /*    TSG1      */
                   unsigned short :1;                   /*              */
                   unsigned short TSG2:3;               /*    TSG2      */
                   unsigned short :2;                   /*              */
                   unsigned short SJW:2;                /*    SJW       */
                   unsigned short :3;                   /*              */
                   unsigned short BSP:1;                /*    BSP       */
```

```
                } BIT;                                  /*              */
        } HCAN2_BCR1;                                   /*              */
    union {                                             /* HCAN2_BCR0   */
        unsigned short WORD;                            /*  Word Access */
        struct {                                        /*  Bit Access  */
                unsigned short :8;                      /*              */
                unsigned short BRP:8;                   /*    BRP       */
                } BIT;                                  /*              */
        } HCAN2_BCR0;                                   /*              */
    union {                                             /* IRR          */
        unsigned short WORD;                            /*  Word Access */
        struct {                                        /*  Bit Access  */
                unsigned short IRR15:1;                 /*    IRR15     */
                unsigned short IRR14:1;                 /*    IRR14     */
                unsigned short IRR13:1;                 /*    IRR13     */
                unsigned short IRR12:1;                 /*    IRR12     */
                unsigned short :2;                      /*              */
                unsigned short IRR9:1;                  /*    IRR9      */
                unsigned short IRR8:1;                  /*    IRR8      */
                unsigned short IRR7:1;                  /*    IRR7      */
                unsigned short IRR6:1;                  /*    IRR6      */
                unsigned short IRR5:1;                  /*    IRR5      */
                unsigned short IRR4:1;                  /*    IRR4      */
                unsigned short IRR3:1;                  /*    IRR3      */
                unsigned short IRR2:1;                  /*    IRR2      */
                unsigned short IRR1:1;                  /*    IRR1      */
                unsigned short IRR0:1;                  /*    IRR0      */
                } BIT;                                  /*              */
        } IRR;                                          /*              */
    union {                                             /* IMR          */
        unsigned short WORD;                            /*  Word Access */
        struct {                                        /*  Bit Access  */
                unsigned short IMR15:1;                 /*    IMR15     */
                unsigned short IMR14:1;                 /*    IMR14     */
                unsigned short IMR13:1;                 /*    IMR13     */
                unsigned short IMR12:1;                 /*    IMR12     */
                unsigned short :2;                      /*              */
                unsigned short IMR9:1;                  /*    IMR9      */
                unsigned short IMR8:1;                  /*    IMR8      */
                unsigned short IMR7:1;                  /*    IMR7      */
                unsigned short IMR6:1;                  /*    IMR6      */
                unsigned short IMR5:1;                  /*    IMR5      */
                unsigned short IMR4:1;                  /*    IMR4      */
                unsigned short IMR3:1;                  /*    IMR3      */
                unsigned short IMR2:1;                  /*    IMR2      */
                unsigned short IMR1:1;                  /*    IMR1      */
                unsigned short :1;                      /*              */
                } BIT;                                  /*              */
        } IMR;                                          /*              */
    unsigned char TEC;                                  /* TEC          */
    unsigned char REC;                                  /* REC          */
```

```
unsigned char wk0[18];                          /*              */
union {                                         /* TXPR1        */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short TXPR31:1;           /*    TXPR31    */
             unsigned short TXPR30:1;           /*    TXPR30    */
             unsigned short TXPR29:1;           /*    TXPR29    */
             unsigned short TXPR28:1;           /*    TXPR28    */
             unsigned short TXPR27:1;           /*    TXPR27    */
             unsigned short TXPR26:1;           /*    TXPR26    */
             unsigned short TXPR25:1;           /*    TXPR25    */
             unsigned short TXPR24:1;           /*    TXPR24    */
             unsigned short TXPR23:1;           /*    TXPR23    */
             unsigned short TXPR22:1;           /*    TXPR22    */
             unsigned short TXPR21:1;           /*    TXPR21    */
             unsigned short TXPR20:1;           /*    TXPR20    */
             unsigned short TXPR19:1;           /*    TXPR19    */
             unsigned short TXPR18:1;           /*    TXPR18    */
             unsigned short TXPR17:1;           /*    TXPR17    */
             unsigned short TXPR16:1;           /*    TXPR16    */
             } BIT;                             /*              */
      } TXPR1;                                  /*              */
union {                                         /* TXPR0        */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short TXPR15:1;           /*    TXPR15    */
             unsigned short TXPR14:1;           /*    TXPR14    */
             unsigned short TXPR13:1;           /*    TXPR13    */
             unsigned short TXPR12:1;           /*    TXPR12    */
             unsigned short TXPR11:1;           /*    TXPR11    */
             unsigned short TXPR10:1;           /*    TXPR10    */
             unsigned short TXPR9:1;            /*    TXPR9     */
             unsigned short TXPR8:1;            /*    TXPR8     */
             unsigned short TXPR7:1;            /*    TXPR7     */
             unsigned short TXPR6:1;            /*    TXPR6     */
             unsigned short TXPR5:1;            /*    TXPR5     */
             unsigned short TXPR4:1;            /*    TXPR4     */
             unsigned short TXPR3:1;            /*    TXPR3     */
             unsigned short TXPR2:1;            /*    TXPR2     */
             unsigned short TXPR1:1;            /*    TXPR1     */
             unsigned short :1;                 /*              */
             } BIT;                             /*              */
      } TXPR0;                                  /*              */
unsigned char wk1[4];                           /*              */
union {                                         /* TXCR1        */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short TXCR31:1;           /*    TXCR31    */
             unsigned short TXCR30:1;           /*    TXCR30    */
             unsigned short TCR29:1;            /*    TCR29     */
             unsigned short TXCR28:1;           /*    TXCR28    */
```

```
                unsigned short TXCR27:1;          /*    TXCR27    */
                unsigned short TSCR26:1;          /*    TSCR26    */
                unsigned short TXCR25:1;          /*    TXCR25    */
                unsigned short TXCR24:1;          /*    TXCR24    */
                unsigned short TXCR23:1;          /*    TXCR23    */
                unsigned short TXCR22:1;          /*    TXCR22    */
                unsigned short TXCR21:1;          /*    TXCR21    */
                unsigned short TXCR20:1;          /*    TXCR20    */
                unsigned short TXCR19:1;          /*    TXCR19    */
                unsigned short TXCR18:1;          /*    TXCR18    */
                unsigned short TXCR17:1;          /*    TXCR17    */
                unsigned short TXCR16:1;          /*    TXCR16    */
                } BIT;                            /*              */
        } TXCR1;                                  /*              */
  union {                                         /* TXCR0        */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short TXCR15:1;          /*    TXCR15    */
                unsigned short TXCR14:1;          /*    TXCR14    */
                unsigned short TCR13:1;           /*    TCR13     */
                unsigned short TXCR12:1;          /*    TXCR12    */
                unsigned short TXCR11:1;          /*    TXCR11    */
                unsigned short TSCR10:1;          /*    TSCR10    */
                unsigned short TXCR9:1;           /*    TXCR9     */
                unsigned short TXCR8:1;           /*    TXCR8     */
                unsigned short TXCR7:1;           /*    TXCR7     */
                unsigned short TXCR6:1;           /*    TXCR6     */
                unsigned short TXCR5:1;           /*    TXCR5     */
                unsigned short TXCR4:1;           /*    TXCR4     */
                unsigned short TXCR3:1;           /*    TXCR3     */
                unsigned short TXCR2:1;           /*    TXCR2     */
                unsigned short TXCR1:1;           /*    TXCR1     */
                unsigned short :1;                /*              */
                } BIT;                            /*              */
        } TXCR0;                                  /*              */
  unsigned char wk2[4];                           /*              */
  union {                                         /* TXACK1       */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short TXACK31:1;         /*    TXACK31   */
                unsigned short TXACK30:1;         /*    TXACK30   */
                unsigned short TXACK29:1;         /*    TXACK29   */
                unsigned short TXACK28:1;         /*    TXACK28   */
                unsigned short TXACK27:1;         /*    TXACK27   */
                unsigned short TXACK26:1;         /*    TXACK26   */
                unsigned short TXACK25:1;         /*    TXACK25   */
                unsigned short TXACK24:1;         /*    TXACK24   */
                unsigned short TXACK23:1;         /*    TXACK23   */
                unsigned short TXACK22:1;         /*    TXACK22   */
                unsigned short TXACK21:1;         /*    TXACK21   */
                unsigned short TXACK20:1;         /*    TXACK20   */
```

```
            unsigned short TXACK19:1;          /*    TXACK19   */
            unsigned short TXACK18:1;          /*    TXACK18   */
            unsigned short TXACK17:1;          /*    TXACK17   */
            unsigned short TXACK16:1;          /*    TXACK16   */
            } BIT;                             /*              */
      } TXACK1;                                /*              */
union {                                        /* TXACK0       */
      unsigned short WORD;                     /*  Word Access */
      struct {                                 /*  Bit Access  */
            unsigned short TXACK15:1;          /*    TXACK15   */
            unsigned short TXACK14:1;          /*    TXACK14   */
            unsigned short TXACK13:1;          /*    TXACK13   */
            unsigned short TXACK12:1;          /*    TXACK12   */
            unsigned short TXACK11:1;          /*    TXACK11   */
            unsigned short TXACK10:1;          /*    TXACK10   */
            unsigned short TXACK9:1;           /*    TXACK9    */
            unsigned short TXACK8:1;           /*    TXACK8    */
            unsigned short TXACK7:1;           /*    TXACK7    */
            unsigned short TXACK6:1;           /*    TXACK6    */
            unsigned short TXACK5:1;           /*    TXACK5    */
            unsigned short TXACK4:1;           /*    TXACK4    */
            unsigned short TXACK3:1;           /*    TXACK3    */
            unsigned short TXACK2:1;           /*    TXACK2    */
            unsigned short TXACK1:1;           /*    TXACK1    */
            unsigned short :1;                 /*              */
            } BIT;                             /*              */
      } TXACK0;                                /*              */
unsigned char wk3[4];                          /*              */
union {                                        /* ABACK1       */
      unsigned short WORD;                     /*  Word Access */
      struct {                                 /*  Bit Access  */
            unsigned short ABACK31:1;          /*    ABACK31   */
            unsigned short ABACK30:1;          /*    ABACK30   */
            unsigned short ABACK29:1;          /*    ABACK29   */
            unsigned short ABACK28:1;          /*    ABACK28   */
            unsigned short ABACK27:1;          /*    ABACK27   */
            unsigned short ABACK26:1;          /*    ABACK26   */
            unsigned short ABACK25:1;          /*    ABACK25   */
            unsigned short ABACK24:1;          /*    ABACK24   */
            unsigned short ABACK23:1;          /*    ABACK23   */
            unsigned short ABACK22:1;          /*    ABACK22   */
            unsigned short ABACK21:1;          /*    ABACK21   */
            unsigned short ABACK20:1;          /*    ABACK20   */
            unsigned short ABACK19:1;          /*    ABACK19   */
            unsigned short ABACK18:1;          /*    ABACK18   */
            unsigned short ABACK17:1;          /*    ABACK17   */
            unsigned short ABACK16:1;          /*    ABACK16   */
            } BIT;                             /*              */
      } ABACK1;                                /*              */
union {                                        /* ABACK0       */
      unsigned short WORD;                     /*  Word Access */
```

```
        struct {                                /*  Bit Access  */
               unsigned short ABACK15:1;        /*    ABACK15   */
               unsigned short ABACK14:1;        /*    ABACK14   */
               unsigned short ABACK13:1;        /*    ABACK13   */
               unsigned short ABACK12:1;        /*    ABACK12   */
               unsigned short ABACK11:1;        /*    ABACK11   */
               unsigned short ABACK10:1;        /*    ABACK10   */
               unsigned short ABACK9:1;         /*    ABACK9    */
               unsigned short ABACK8:1;         /*    ABACK8    */
               unsigned short ABACK7:1;         /*    ABACK7    */
               unsigned short ABACK6:1;         /*    ABACK6    */
               unsigned short ABACK5:1;         /*    ABACK5    */
               unsigned short ABACK4:1;         /*    ABACK4    */
               unsigned short ABACK3:1;         /*    ABACK3    */
               unsigned short ABACK2:1;         /*    ABACK2    */
               unsigned short ABACK1:1;         /*    ABACK1    */
               unsigned short :1;               /*              */
               } BIT;                           /*              */
        } ABACK0;                               /*              */
    unsigned char wk4[4];                       /*              */
    union {                                     /* RXPR1        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Bit Access  */
               unsigned short RXPR31:1;         /*    RXPR31    */
               unsigned short RXPR30:1;         /*    RXPR30    */
               unsigned short RXPR29:1;         /*    RXPR29    */
               unsigned short RXPR28:1;         /*    RXPR28    */
               unsigned short RXPR27:1;         /*    RXPR27    */
               unsigned short RXPR26:1;         /*    RXPR26    */
               unsigned short RXPR25:1;         /*    RXPR25    */
               unsigned short RXPR24:1;         /*    RXPR24    */
               unsigned short RXPR23:1;         /*    RXPR23    */
               unsigned short RXPR22:1;         /*    RXPR22    */
               unsigned short RXPR21:1;         /*    RXPR21    */
               unsigned short RXPR20:1;         /*    RXPR20    */
               unsigned short RXPR19:1;         /*    RXPR19    */
               unsigned short RXPR18:1;         /*    RXPR18    */
               unsigned short RXPR17:1;         /*    RXPR17    */
               unsigned short RXPR16:1;         /*    RXPR16    */
               } BIT;                           /*              */
        } RXPR1;                                /*              */
    union {                                     /* RXPR0        */
        unsigned short WORD;                    /*  Word Access */
        struct {                                /*  Bit Access  */
               unsigned short RXPR15:1;         /*    RXPR15    */
               unsigned short RXPR14:1;         /*    RXPR14    */
               unsigned short RXPR13:1;         /*    RXPR13    */
               unsigned short RXPR12:1;         /*    RXPR12    */
               unsigned short RXPR11:1;         /*    RXPR11    */
               unsigned short RXPR10:1;         /*    RXPR10    */
               unsigned short RXPR9:1;          /*    RXPR9     */
```

```
            unsigned short RXPR8:1;          /*    RXPR8     */
            unsigned short RXPR7:1;          /*    RXPR7     */
            unsigned short RXPR6:1;          /*    RXPR6     */
            unsigned short RXPR5:1;          /*    RXPR5     */
            unsigned short RXPR4:1;          /*    RXPR4     */
            unsigned short RXPR3:1;          /*    RXPR3     */
            unsigned short RXPR2:1;          /*    RXPR2     */
            unsigned short RXPR1:1;          /*    RXPR1     */
            unsigned short RXPR0:1;          /*    RXPR0     */
            } BIT;                           /*              */
      } RXPR0;                               /*              */
unsigned char wk5[4];                        /*              */
union {                                      /* RFPR1        */
      unsigned short WORD;                   /*  Word Access */
      struct {                               /*  Bit Access  */
            unsigned short RFPR31:1;         /*    RFPR31    */
            unsigned short RFPR30:1;         /*    RFPR30    */
            unsigned short RFPR29:1;         /*    RFPR29    */
            unsigned short RFPR28:1;         /*    RFPR28    */
            unsigned short RFPR27:1;         /*    RFPR27    */
            unsigned short RFPR26:1;         /*    RFPR26    */
            unsigned short RFPR25:1;         /*    RFPR25    */
            unsigned short RFPR24:1;         /*    RFPR24    */
            unsigned short RFPR23:1;         /*    RFPR23    */
            unsigned short RFPR22:1;         /*    RFPR22    */
            unsigned short RFPR21:1;         /*    RFPR21    */
            unsigned short RFPR20:1;         /*    RFPR20    */
            unsigned short RFPR19:1;         /*    RFPR19    */
            unsigned short RFPR18:1;         /*    RFPR18    */
            unsigned short RFPR17:1;         /*    RFPR17    */
            unsigned short RFPR16:1;         /*    RFPR16    */
            } BIT;                           /*              */
      } RFPR1;                               /*              */
union {                                      /* RFPR0        */
      unsigned short WORD;                   /*  Word Access */
      struct {                               /*  Bit Access  */
            unsigned short RFPR15:1;         /*    RFPR15    */
            unsigned short RFPR14:1;         /*    RFPR14    */
            unsigned short RFPR13:1;         /*    RFPR13    */
            unsigned short RFPR12:1;         /*    RFPR12    */
            unsigned short RFPR11:1;         /*    RFPR11    */
            unsigned short RFPR10:1;         /*    RFPR10    */
            unsigned short RFPR9:1;          /*    RFPR9     */
            unsigned short RFPR8:1;          /*    RFPR8     */
            unsigned short RFPR7:1;          /*    RFPR7     */
            unsigned short RFPR6:1;          /*    RFPR6     */
            unsigned short RFPR5:1;          /*    RFPR5     */
            unsigned short RFPR4:1;          /*    RFPR4     */
            unsigned short RFPR3:1;          /*    RFPR3     */
            unsigned short RFPR2:1;          /*    RFPR2     */
            unsigned short RFPR1:1;          /*    RFPR1     */
```

```
                unsigned short RFPR0:1;          /*    RFPR0     */
                } BIT;                           /*              */
        } RFPR0;                                 /*              */
    unsigned char wk6[4];                        /*              */
    union {                                      /* MBIMR1       */
        unsigned short WORD;                     /*  Word Access */
        struct {                                 /*  Bit Access  */
                unsigned short MBIMR31:1;        /*    MBIMR31   */
                unsigned short MBIMR30:1;        /*    MBIMR30   */
                unsigned short MBIMR29:1;        /*    MBIMR29   */
                unsigned short MBIMR28:1;        /*    MBIMR28   */
                unsigned short MBIMR27:1;        /*    MBIMR27   */
                unsigned short MBIMR26:1;        /*    MBIMR26   */
                unsigned short MBIMR25:1;        /*    MBIMR25   */
                unsigned short MBIMR24:1;        /*    MBIMR24   */
                unsigned short MBIMR23:1;        /*    MBIMR23   */
                unsigned short MBIMR22:1;        /*    MBIMR22   */
                unsigned short MBIMR21:1;        /*    MBIMR21   */
                unsigned short MBIMR20:1;        /*    MBIMR20   */
                unsigned short MBIMR19:1;        /*    MBIMR19   */
                unsigned short MBIMR18:1;        /*    MBIMR18   */
                unsigned short MBIMR17:1;        /*    MBIMR17   */
                unsigned short MBIMR16:1;        /*    MBIMR16   */
                } BIT;                           /*              */
        } MBIMR1;                                /*              */
    union {                                      /* MBIMR0       */
        unsigned short WORD;                     /*  Word Access */
        struct {                                 /*  Bit Access  */
                unsigned short MBIMR15:1;        /*    MBIMR15   */
                unsigned short MBIMR14:1;        /*    MBIMR14   */
                unsigned short MBIMR13:1;        /*    MBIMR13   */
                unsigned short MBIMR12:1;        /*    MBIMR12   */
                unsigned short MBIMR11:1;        /*    MBIMR11   */
                unsigned short MBIMR10:1;        /*    MBIMR10   */
                unsigned short MBIMR9:1;         /*    MBIMR9    */
                unsigned short MBIMR8:1;         /*    MBIMR8    */
                unsigned short MBIMR7:1;         /*    MBIMR7    */
                unsigned short MBIMR6:1;         /*    MBIMR6    */
                unsigned short MBIMR5:1;         /*    MBIMR5    */
                unsigned short MBIMR4:1;         /*    MBIMR4    */
                unsigned short MBIMR3:1;         /*    MBIMR3    */
                unsigned short MBIMR2:1;         /*    MBIMR2    */
                unsigned short MBIMR1:1;         /*    MBIMR1    */
                unsigned short MBIMR0:1;         /*    MBIMR0    */
                } BIT;                           /*              */
        } MBIMR0;                                /*              */
    unsigned char wk7[4];                        /*              */
    union {                                      /* UMSR1        */
        unsigned short WORD;                     /*  Word Access */
        struct {                                 /*  Bit Access  */
                unsigned short UMSR31:1;         /*    UMSR31    */
```

```
                unsigned short UMSR30:1;          /*    UMSR30    */
                unsigned short UMSR29:1;          /*    UMSR29    */
                unsigned short UMSR28:1;          /*    UMSR28    */
                unsigned short UMSR27:1;          /*    UMSR27    */
                unsigned short UMSR26:1;          /*    UMSR26    */
                unsigned short UMSR25:1;          /*    UMSR25    */
                unsigned short UMSR24:1;          /*    UMSR24    */
                unsigned short UMSR23:1;          /*    UMSR23    */
                unsigned short UMSR22:1;          /*    UMSR22    */
                unsigned short UMSR21:1;          /*    UMSR21    */
                unsigned short UMSR20:1;          /*    UMSR20    */
                unsigned short UMSR19:1;          /*    UMSR19    */
                unsigned short UMSR18:1;          /*    UMSR18    */
                unsigned short UMSR17:1;          /*    UMSR17    */
                unsigned short UMSR16:1;          /*    UMSR16    */
                } BIT;                            /*              */
        } UMSR1;                                  /*              */
  union {                                         /* UMSR0        */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short UMSR15:1;          /*    UMSR15    */
                unsigned short UMSR14:1;          /*    UMSR14    */
                unsigned short UMSR13:1;          /*    UMSR13    */
                unsigned short UMSR12:1;          /*    UMSR12    */
                unsigned short UMSR11:1;          /*    UMSR11    */
                unsigned short UMSR10:1;          /*    UMSR10    */
                unsigned short UMSR9:1;           /*    UMSR9     */
                } BIT;                            /*              */
        } UMSR0;                                  /*              */
  unsigned char wk8[36];                          /*              */
  unsigned short TCNTR;                           /* TCNTR        */
  union {                                         /* TCR          */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short TCR15:1;           /*    TCR15     */
                unsigned short TCR14:1;           /*    TCR14     */
                unsigned short TCR13:1;           /*    TCR13     */
                unsigned short TCR12:1;           /*    TCR12     */
                unsigned short TCR11:1;           /*    TCR11     */
                unsigned short TCR10:1;           /*    TCR10     */
                unsigned short TCR9:1;            /*    TCR9      */
                unsigned short TCR8:1;            /*    TCR8      */
                unsigned short TCR7:1;            /*    TCR7      */
                unsigned short :1;                /*              */
                unsigned short TPSC:6;            /*    TPSC      */
                } BIT;                            /*              */
        } TCR;                                    /*              */
  union {                                         /* TSR          */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short :13;               /*              */
```

```
                unsigned short TSR2:1;          /*    TSR2      */
                unsigned short TSR1:1;          /*    TSR1      */
                unsigned short TSR0:1;          /*    TSR0      */
                } BIT;                          /*              */
          } TSR;                                /*              */
     unsigned short TDCR;                       /* TDCR         */
     unsigned short LOSR;                       /* LOSR         */
     unsigned char wk9[2];                      /*              */
     unsigned short HCAN2_ICR0;                 /* HCAN2_ICR0   */
     unsigned short HCAN2_ICR1;                 /* HCAN2_ICR1   */
     unsigned short TCMR0;                      /* TCMR0        */
     unsigned short TCMR1;                      /* TCMR1        */
     unsigned char wk10[108];                   /*              */
struct st_mb {
     union {                                     /* MB0          */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char :1;               /*              */
                unsigned char STDID10:1;        /*    STDID10   */
                unsigned char STDID9:1;         /*    STDID9    */
                unsigned char STDID8:1;         /*    STDID8    */
                unsigned char STDID7:1;         /*    STDID7    */
                unsigned char STDID6:1;         /*    STDID6    */
                unsigned char STDID5:1;         /*    STDID5    */
                unsigned char STDID4:1;         /*    STDID4    */
                } BIT;                          /*              */
          } MB0;                                /*              */
     union {                                    /* MB1          */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char STDID:4;          /*    STDID     */
                unsigned char RTR:1;            /*    RTR       */
                unsigned char IDE:1;            /*    IDE       */
                unsigned char EXTID17:1;        /*    EXTID17   */
                unsigned char EXTID16:1;        /*    EXTID16   */
                } BIT;                          /*              */
          } MB1;                                /*              */
     union {                                    /* MB2          */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char EXTID15:1;        /*    EXTID15   */
                unsigned char EXTID14:1;        /*    EXTID14   */
                unsigned char EXTID13:1;        /*    EXTID13   */
                unsigned char EXTID12:1;        /*    EXTID12   */
                unsigned char EXTID11:1;        /*    EXTID11   */
                unsigned char EXTID10:1;        /*    EXTID10   */
                unsigned char EXTID9:1;         /*    EXTID9    */
                unsigned char EXTID8:1;         /*    EXTID8    */
                } BIT;                          /*              */
          } MB2;                                /*              */
     union {                                    /* MB3          */
```

```
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char EXTID7:1;           /*    EXTID7    */
                  unsigned char EXTID6:1;           /*    EXTID6    */
                  unsigned char EXTID5:1;           /*    EXTID5    */
                  unsigned char EXTID4:1;           /*    EXTID4    */
                  unsigned char EXTID3:1;           /*    EXTID3    */
                  unsigned char EXTID2:1;           /*    EXTID2    */
                  unsigned char EXTID1:1;           /*    EXTID1    */
                  unsigned char EXTID0:1;           /*    EXTID0    */
                  } BIT;                            /*              */
            } MB3;                                  /*              */
      union {                                       /* MB4          */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char CCM:1;              /*    CCM       */
                  unsigned char TTE:1;              /*    TTE       */
                  unsigned char NMC:1;              /*    NMC       */
                  unsigned char ATX:1;              /*    ATX       */
                  unsigned char DART:1;             /*    DART      */
                  unsigned char MBC:3;              /*    MBC       */
                  } BIT;                            /*              */
            } MB4;                                  /*              */
      union {                                       /* MB5          */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char PTE:1;              /*    PTE       */
                  unsigned char TCT:1;              /*    TCT       */
                  unsigned char CBE:1;              /*    CBE       */
                  unsigned char :1;                 /*              */
                  unsigned char DLC:4;              /*    DLC       */
                  } BIT;                            /*              */
            } MB5;                                  /*              */
      unsigned char TIME_STAMP;                     /* TIME_STAMP   */
      unsigned char wk11[1];                        /*              */
      unsigned char MSG_DATA[8];                    /* MSG_DATA     */
      unsigned char LAFM0[2];                       /* LAFM0        */
      unsigned char LAFM1[2];                       /* LAFM1        */
      unsigned char wk12[12];                       /*              */
    }mb[32];
};                                                  /*              */

#define P_SCI2 (*(volatile struct st_sci *)0xFFFF81C0)   /* SCI2 Address */
#define P_SCI3 (*(volatile struct st_sci *)0xFFFF81D0)   /* SCI3 Address */
#define P_SCI4 (*(volatile struct st_sci *)0xFFFF81E0)   /* SCI4 Address */
#define P_MTU34 (*(volatile struct st_mtu34 *)0xFFFF8200)/* MTU34 Address */
#define P_MTU0 (*(volatile struct st_mtu0 *)0xFFFF8260)  /* MTU0 Address */
#define P_MTU1 (*(volatile struct st_mtu1 *)0xFFFF8280)  /* MTU1 Address */
#define P_MTU2 (*(volatile struct st_mtu2 *)0xFFFF82A0)  /* MTU2 Address */
#define P_INTC (*(volatile struct st_intc *)0xFFFF8348)  /* INTC Address */
#define P_PORTA (*(volatile struct st_porta *)0xFFFF8382)/* PORTA Address */
```

```
#define P_PORTB (*(volatile struct st_portb *)0xFFFF8390)/* PORTB Address */
#define P_PORTD (*(volatile struct st_portd *)0xFFFF83A2)/* PORTD Address */
#define P_PORTE (*(volatile struct st_porte *)0xFFFF83B0)/* PORTE Address */
#define P_PORTF (*(volatile struct st_portf *)0xFFFF83B2)/* PORTF Address */
#define P_MTU (*(volatile struct st_mtu *)0xFFFF83C0)    /* MTU Address  */
#define P_MMT (*(volatile struct st_mmt *)0xFFFF83C4)    /* MMT Address  */
#define P_PORTG (*(volatile struct st_portg *)0xFFFF83CD)/* PORTG Address */
#define P_CMT (*(volatile struct st_cmt *)0xFFFF83D0)    /* CMT Address  */
#define P_AD (*(volatile struct st_ad *)0xFFFF8420)      /* AD Address   */
#define P_FLASH (*(volatile struct st_flash *)0xFFFF8580)/* FLASH Address */
#define P_UBC (*(volatile struct st_ubc *)0xFFFF8600)    /* UBC Address  */
#define P_WDT (*(volatile struct st_wdt *)0xFFFF8610)    /* WDT Address  */
#define P_STBY (*(volatile struct st_stby *)0xFFFF8614)  /* STBY Address */
#define P_BSC (*(volatile struct st_bsc *)0xFFFF8620)    /* BSC Address  */
#define P_DTC (*(volatile struct st_dtc *)0xFFFF8700)    /* DTC Address  */
#define P_HUDI (*(volatile struct st_hudi *)0xFFFF8A50)  /* HUDI Address */
#define P_HCAN2 (*(volatile struct st_hcan2 *)0xFFFFB000)/* HCAN2 Address */
```

Note: The header file shown here is the file generated automatically by Renesas Technology Integrated Development Environment (High-performance Embedded Workshop) C/C++ Compiler Ver. 6. Please refer to development environment upgrade information for details of modifications and amendments.

SH7046 Group On-Chip I/O Volume Application Note

Publication Date: 1st Edition, July 10, 2003
Published by: Sales Strategic Planning Div.
Renesas Technology Corp.
Edited by: Technical Documentation & Information Department
Renesas Kodaira Semiconductor Co., Ltd.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

http://www.renesas.com

SH7046 Group On-Chip I/O Volume Application Note

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ05B0076-0100O