NEC

User's Manual

μPD780822B Subseries

8-bit Single-Chip Microcontroller

Hardware

μPD780821B (A)
μPD780822B (A)
μPD78F0822B (A)

Document No. U17681EE2V0UD00
Date Published July 2007

© NEC Electronics Corporation 2007
Printed in Germany

NOTES FOR CMOS DEVICES

① **VOLTAGE APPLICATION WAVEFORM AT INPUT PIN**

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (MAX) and V_{IH} (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (MAX) and V_{IH} (MIN).

② **HANDLING OF UNUSED INPUT PINS**

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

③ **PRECAUTION AGAINST ESD**

A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

④ **STATUS BEFORE INITIALIZATION**

Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

⑤ **INPUT OF SIGNAL DURING POWER OFF STATE**

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

All (other) product, brand, or trade names used in this pamphlet are the trademarks or registered trademarks of their respective owners.
Product specifications are subject to change without notice. To ensure that you have the latest product data, please contact your local NEC Electronics sales office.

- **The information in this document is current as of July, 2006. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC 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 NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

  "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.

  "Special": 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, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)

(1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.

(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02. 11-1

For further information,
please contact:

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

[America]

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

[Europe]

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

Hanover Office
Podbielski Strasse 166 B
30177 Hanover
Tel: 0 511 33 40 2-0

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

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

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

Succursale Française
9, rue Paul Dautier, B.P. 52180
78142 Velizy-Villacoublay Cédex
France
Tel: 01-3067-5800

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

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

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

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

[Asia & Oceania]

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

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

NEC Electronics Hong Kong Ltd.
12/F., Cityplaza 4,
12 Taikoo Wan Road, Hong Kong
Tel: 2886-9318
http://www.hk.necel.com/

Seoul Branch
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku,
Seoul, 135-080, Korea
Tel: 02-558-3737

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

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

G06.7A

Preface

Readers

This manual has been prepared for engineers who want to understand the functions of the µPD780822B Subseries and design and develop its application systems and programs.

µPD780822B Subseries:
µPD780821B(A), µPD780822B(A), µPD78F0822B(A)

Purpose

This manual is intended for users to understand the functions of the µPD780822B Subseries.

Organization

The µPD780822B Subseries manual is separated into two parts: this manual and the instruction edition (common to the 78K/0 series).

µPD780822B Subseries This Manual	**78K/0 series User's Manual Instruction**
• Pin functions • Internal block functions • Interrupt • Other on-chip peripheral functions	• CPU functions • Instruction set • Explanation of each instruction

How to Read This Manual

Before reading this manual, you should have general knowledge of electric and logic circuits and microcontrollers.

- When using this manual as the manual for (A) products:
  → Read the part number as follows:
  µPD780821B → µPD780821B(A)
  µPD780822B → µPD780822B(A)
  µPD78F0822B → µPD78F0822B(A)
- When you want to understand the function in general:
  → Read this manual in the order of the contents.
- How to interpret the register format:
  → For the bit number enclosed in square, the bit name is defined as a reserved word in RA78K/0, and in CC78K/0 and defined in the header file of hte IAR compiler.
- To make sure the details of the registers when you know the register name.
  → Refer to **Appendix C.**

Related Documents

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

- **Related documents for µPD780822B Subseries**

Document name	Document No.	
	Japanese	English
µPD780822B Subseries User's Manual	Planned	U16985E
78K/0 Series User's Manual-Instruction	IEU-849	U12326E
78K/0 Series Instruction Table	U10903J	-
78K/0 Series Instruction Set	U10904J	-

- **Related documents for development tools (User's Manuals)**

Document name		Document No.	
		Japanese	English
RA78K Series Assembler Package	Operation	EEU-809	EEU-1399
	Language	EEU-815	EEU-1404
RA78K Series Structured Assembler Preprocessor		EEU-817	EEU-1402
CC78K Series C Compiler	Operation	EEU-656	EEU-1280
	Language	EEU-655	EEU-1284
CC78K/0 C Compiler	Operation	U11517J	-
	Language	U11518J	-
CC78K/0 C Compiler Application Note	Programming Note	EEA-618	EEA-1208
CC78K Series Library Source File		EEU-777	-
QB-780822		Planned	U17172E
SM78K0 System Simulator Windows™ Base	Reference	U15373J	U15373E
SM78K0 Series System Simulator	External part user open Interface	U15802J	U15802E
ID78K0-NS Integrated Debugger		U15185J	U15185E

- **Related documents for embedded software (User's Manual)**

Document name		Document No.	
		Japanese	English
78K/0 Series Real-Time OS	Basics	U11537J	-
	Installation	U11536J	-
	Technical	U11538J	-
78K/0 Series OS MX78K0	Basics	EEU-5010	-
Fuzzy Knowledge Data Creation Tool		EEU-829	EEU1438
78K/0, 78K/II, 87AD Series Fuzzy Inference Development Support System-Translator		EEU-862	EEU-1444
78K/0 Series Fuzzy Inference Development Support System- Fuzzy Inference Module		EEU-858	EEU-1441
78K/0 Series Fuzzy Inference Development Support System- Fuzzy Inference Debugger		EEU-921	EEU-1458

- **Other Documents**

Document name	Document No.	
	Japanese	English
IC Package Manual	C10943X	IEI-1213
Semiconductor Device Mounting Technology Manual	C10535J	C10535E
Quality Grade on NEC Semiconductor Devices	C11531J	C11531E
Reliability Quality Control on NEC Semiconductor Devices	C10983J	C10983E
Electric Static Discharge (ESD) Test	MEM-539	-
Semiconductor Devices Quality Assurance Guide	MEI-603	MEI-1202
Microcontroller Related Product Guide - Third Party Manufacturers	U11416J	-

Caution: The above documents are subject to change without prior notice. Be sure to use the latest version document when starting design.

Legend

Symbols and notation are used as follows:

Weight in data notation	:	Left is high-order column, right is low order column
Active low notation	:	$\overline{xxx}$ (pin or signal name is over-scored) or /xxx (slash before signal name)
Memory map address:	:	High order at high stage and low order at low stage
Note	:	Explanation of (Note) in the text
Caution	:	Item deserving extra attention
Remark	:	Supplementary explanation to the text
Numeric notation	:	Binary . . . xxxx or xxxB Decimal . . . xxxx Hexadecimal . . . xxxxH or 0x xxxx

Prefixes representing powers of 2 (address space, memory capacity)

K (kilo): $2^{10} = 1024$

M (mega): $2^{20} = 1024^2 = 1,048,576$

G (giga): $2^{30} = 1024^3 = 1,073,741,824$

Table of Contents

Preface 5

Chapter 1 Outline (µPD780822B Subseries) 29
1.1 Features 29
1.2 Application fields 29
1.3 Ordering Information 29
1.4 Quality Grade 30
1.5 Pin Configuration (Top View) 31
1.6 78K/0 CAN Products Expansion 34
1.7 Block Diagram 36
1.8 Overview of Functions 37

Chapter 2 Pin Functions (µPD780822B Subseries) 39
2.1 List of Pin Functions 39
2.2 Non-port Pins 42
2.3 Description of Pin Functions 44
2.3.1 P00 to P03 (Port 0) 44
2.3.2 P10 to P17 (Port 1) 45
2.3.3 P20 to P27 (Port 2) 46
2.3.4 P30 to P33 (Port 3) 47
2.3.5 P70 to P77 (Port 7) 48
2.3.6 P80 to P87 (Port 8) 49
2.3.7 P90 to P97 (Port 9) 50
2.3.8 P100 to P105 (Port 10) 51
2.3.9 P110 to P113 (Port 11) 52
2.3.10 P120 (Port 12) 53
2.3.11 P130 (Port 13) 53
2.3.12 P140 to P147 (Port 14) 54
2.3.13 P150 to P157 (Port 15) 55
2.3.14 COM0 to COM3 56
2.3.15 V_{LCD} 56
2.3.16 $\overline{RESET}$ 56
2.3.17 X1 and X2 56
2.3.18 XT1, XT2 56
2.3.19 V_{DD0} to V_{DD2} 56
2.3.20 V_{SS0} to V_{SS2} 56
2.3.21 AV_{DD} 56
2.3.22 AV_{SS} 56
2.3.23 AV_{REF} 56
2.3.24 SMV_{DD0}, SMV_{DD1} 56
2.3.25 SMV_{SS0}, SMV_{SS1} 57
2.3.26 FLMD0, FLMD1 (Flash ROM version only) 57
2.3.27 IC (Mask ROM version only) 57
2.4 Pin I/O Circuits and Recommended Connection of Unused Pins 58

Chapter 3 CPU Architecture (µPD780822B Subseries) 63
3.1 Memory Space 63
3.1.1 Internal program memory space 66
3.1.2 Internal data memory space 71
3.1.3 Special function register (SFR) area 71
3.1.4 Data memory addressing 72
3.2 Processor Registers 73
3.2.1 Control registers 73
3.2.2 General registers 76
3.2.3 Special function register (SFR) 77

3.3 Instruction Address Addressing 83
3.3.1 Relative addressing 83
3.3.2 Immediate addressing 84
3.3.3 Table indirect addressing 85
3.3.4 Register addressing 86
3.4 Operand Addressing 87
3.4.1 Implied addressing 87
3.4.2 Register addressing 88
3.4.3 Direct addressing 89
3.4.4 Short direct addressing 90
3.4.5 Special function register (SFR) addressing 91
3.4.6 Register indirect addressing 92
3.4.7 Based addressing 93
3.4.8 Based indexed addressing 94
3.4.9 Stack addressing 95

Chapter 4 Port Functions (μPD780822B Subseries) 97
4.1 Port Functions 97
4.2 Port Configuration 100
4.2.1 Port 0 101
4.2.2 Port 1 102
4.2.3 Port 2 104
4.2.4 Port 3 105
4.2.5 Port 7 106
4.2.6 Port 8 107
4.2.7 Port 9 108
4.2.8 Port 10 110
4.2.9 Port 11 112
4.2.10 Port 12 114
4.2.11 Port 13 115
4.2.12 Port 14 116
4.2.13 Port 15 117
4.3 Port Function Control Registers 118
4.4 Port Function Operations 122
4.4.1 Writing to input/output port 122
4.4.2 Reading from input/output port 122
4.4.3 Operations on input/output port 123

Chapter 5 Clock Generator (μPD780822B Subseries) 125
5.1 Clock Generator Functions 125
5.2 Clock Generator Configuration 125
5.3 Clock Generator Control Register 127
5.4 System Clock Oscillator 136
5.4.1 Main system clock oscillator 136
5.4.2 Subsystem clock oscillator 137
5.4.3 When subsystem clock is not used 141
5.4.4 Internal low-speed oscillator 141
5.4.5 Prescaler 141
5.5 Clock Generator Operations 142
5.6 Time Required to Switch Between Internal Low-speed Oscillator Clock and X1 Input Clock 149
5.7 Time Required for CPU Clock Switchover 150
5.8 Clock Switching Flowchart and Register Setting 151
5.8.1 Switching from internal low-speed oscillator clock to X1 input clock 151
5.8.2 Switching from X1 input clock to internal low-speed oscillator clock 152
5.8.3 Switching from X1 input clock to subsystem clock 153
5.8.4 Switching from subsystem clock to X1 input clock 154
5.8.5 Register settings 155

Chapter 6 16-Bit Timer 2 (µPD780822B Subseries) 157
6.1 16-Bit Timer 2 Functions 157
6.2 16-Bit Timer 2 Configuration 159
6.3 16-Bit Timer 2 Control Registers 161
6.4 16-Bit Timer 2 Operations 166
6.4.1 Pulse width measurement operations 166
6.5 16-Bit Timer 2 Precautions 170

Chapter 7 16-Bit Timer/Event Counter P (µPD780822B Subseries) 173
7.1 Features 173
7.2 Functional Outline 173
7.3 Configuration 174
7.4 Control Registers 179
7.5 Operation 186
7.5.1 Anytime write and reload 186
7.5.2 Interval timer mode (TPnMD2 to TPnMD0 = 000) 191
7.5.3 One-shot pulse mode (TPnMD2 to TPnMD0 = 011) 194
7.5.4 PWM mode (TPnMD2 to TPnMD0 = 100) 197
7.5.5 Free-running mode (TPnMD2 to TPnMD0 = 101) 202
7.5.6 Pulse width measurement mode (TPnMD2 to TPnMD0 = 110) 207
7.6 Timer Synchronized Operation Function 209

Chapter 8 8-Bit Timer/Event Counters 50 and 51 (µPD780822B Subseries) 213
8.1 8-Bit Timer/Event Counters 50 and 51 Functions 213
8.2 8-Bit Timer/Event Counters 50 and 51 Configurations 215
8.3 8-Bit Timer/Event Counters 50 and 51 Control Registers 218
8.4 8-Bit Timer/Event Counters 50 and 51 Operations 224
8.4.1 Interval timer operations (8-bit timer/event counter mode) 224
8.4.2 External event counter operation 229
8.4.3 Square-wave output 230
8.4.4 PWM output operations 232
8.5 Cautions on 8-Bit Timer/Event Counters 50 and 51 235

Chapter 9 Watch Timer (µPD780822B Subseries) 237
9.1 Watch Timer Functions 237
9.2 Watch Timer Configuration 239
9.2.1 Input Switch Control Register (ISC) 239
9.2.2 Watch Timer Mode Register (WTM) 240
9.3 Watch Timer Operations 242
9.3.1 Watch timer operation 242
9.3.2 Interval timer operation 243
9.4 Cautions for Watch Timer 245

Chapter 10 Watchdog Timer (µPD780822B Subseries) 247
10.1 Watchdog Timer Functions 247
10.2 Watchdog Timer Configuration 248
10.3 Watchdog Timer Control Registers 250
10.4 Watchdog Timer Operations 254
10.4.1 Interval timer operation 254
10.4.2 Watchdog timer mode 1 operation 256
10.4.3 Watchdog timer mode 2 operation when “Internal low-speed oscillator cannot be stopped” is selected by mask option 258
10.4.4 Watchdog timer mode 2 operation when “Internal low-speed oscillator can be stopped by software” is selected by mask option 259

Chapter 11 Clock Output / Buzzer Output Controller (µPD780822B Subseries) 263
11.1 Functions of Clock Output/Buzzer Output Controller 263
11.2 Configuration of Clock Output/Buzzer Output Controller 264

11.3 Register Controlling Clock Output/Buzzer Output Controller 264
11.4 Clock Output/Buzzer Output Controller Operations 267
11.4.1 Clock output operation 267
11.4.2 Operation as buzzer output 267

Chapter 12 A/D Converter (µPD780822B Subseries) 269
12.1 Functions of A/D Converter 269
12.2 Configuration of A/D Converter 270
12.3 Registers Used in A/D Converter 272
12.3.1 A/D converter mode register (ADM) 273
12.3.2 Analog input channel specification register (ADS) 275
12.3.3 A/D conversion result register (ADCR) 276
12.3.4 Power-fail comparison mode register (PFM) 277
12.3.5 Power-fail comparison threshold register (PFT) 277
12.4 A/D Converter Operations 278
12.4.1 Basic operations of A/D converter 278
12.4.2 Input voltage and conversion results 280
12.4.3 A/D converter operation mode 281
12.5 How to Read A/D Converter Characteristics Table 284
12.6 Cautions for A/D Converter 288
12.7 Cautions on Emulation performed with IE-78K0-NS-A Emulator and IE-780822-NS-EM1 Emulation Board 293

Chapter 13 Serial Interfaces CSI10 and CSI11 (µPD780822B Subseries) 295
13.1 Functions of Serial Interfaces CSI10 and CSI11 295
13.2 Configuration of Serial Interfaces CSI10 and CSI11 296
13.3 Registers Controlling Serial Interfaces CSI10 and CSI11 298
13.3.1 Serial operation mode register 1n (CSIM1n) 298
13.3.2 Serial clock selection register 1n (CSIC1n) 301
13.3.3 Port mode registers 10 and 11 (PM10, PM11) 304
13.4 Operation of Serial Interfaces CSI10 and CSI11 305
13.4.1 Operation stop mode 305
13.4.2 3-wire serial I/O mode 306

Chapter 14 Serial Interface UART6 (µPD780822B Subseries) 315
14.1 Functions of Serial Interface UART6 315
14.2 Configuration of Serial Interface UART6 319
14.3 Registers Controlling Serial Interface UART6 322
14.3.1 Asynchronous serial interface operation mode register 6 (ASIM6) 322
14.3.2 Asynchronous serial interface reception error status register 6 (ASIS6) 324
14.3.3 Asynchronous serial interface transmission status register 6 (ASIF6) 325
14.3.4 Clock selection register 6 (CKSR6) 326
14.3.5 Baud rate generator control register 6 (BRGC6) 327
14.3.6 Asynchronous serial interface control register 6 (ASICL6) 328
14.3.7 Input switch control register (ISC) 330
14.3.8 Port mode register 1 (PM1) 330
14.4 Operation of Serial Interface UART6 331
14.4.1 Operation stop mode 331
14.4.2 Asynchronous serial interface (UART) mode 332
14.4.3 Dedicated baud rate generator 347

Chapter 15 CAN Controller (µPD780822B Subseries) 355
15.1 CAN Protocol 356
15.1.1 Protocol Mode Function 356
15.1.2 Message Format 356
15.1.3 Data Frame / Remote Frame 357
15.1.4 Description of each field 358
15.1.5 Error Frame 364

15.1.6 Overload Frame 365
15.2 Function 366
15.2.1 Arbitration 366
15.2.2 Bit Stuffing 366
15.2.3 Multi Master 367
15.2.4 Multi Cast 367
15.2.5 Sleep Mode/Stop Function 367
15.2.6 Error Control Function 368
15.2.7 Baud Rate Control Function 371
15.2.8 State Shift Chart 374
15.3 Outline Description 377
15.4 Connection with Target System 378
15.5 CAN Controller Configuration. 378
15.6 Special Function Register for CAN-module 379
15.7 Message and Buffer Configuration 380
15.8 Transmit Buffer Structure 381
15.9 Transmit Message Format 381
15.10 Receive Buffer Structure. 385
15.11 Receive Message Format 386
15.12 Mask Function 392
15.13 Operation of the CAN Controller 396
15.13.1 DCAN control register (DCANC). 396
15.13.2 CAN control register (CANC) 397
15.13.3 DCAN Error Status Register (CANES) 401
15.13.4 CAN Transmit Error Counter (TEC) 404
15.13.5 CAN Receive Error Counter (REC) 404
15.13.6 Message Count Register 405
15.14 Baudrate Generation 407
15.15 Function Control 413
15.15.1 Transmit Control 413
15.15.2 Receive Control 415
15.15.3 Mask Control. 416
15.15.4 Special Functions 419
15.16 Interrupt Information 421
15.16.1 Interrupt Vectors 421
15.16.2 Transmit Interrupt 421
15.16.3 Receive Interrupt 421
15.16.4 Error Interrupt 422
15.17 Influence of the standby Function of the CAN Controller 423
15.17.1 CPU Halt Mode. 423
15.17.2 CPU Stop Mode 423
15.17.3 DCAN Sleep Mode 423
15.17.4 DCAN Stop Mode 425
15.18 Functional Description by Flowcharts. 426
15.18.1 Initialization. 426
15.18.2 Transmit Preparation 427
15.18.3 Abort Transmit 428
15.18.4 Handling by the DCAN 429
15.18.5 Receive Event Oriented 430
15.18.6 Receive Task Oriented 431
15.19 CAN Controller Precautions 432

Chapter 16 LCD Controller/Driver (µPD780822B Subseries) 433
16.1 LCD Controller/Driver Functions 433
16.2 LCD Controller/Driver Configuration 433
16.3 LCD Controller/Driver Control Registers. 435
16.4 LCD Controller/Driver Settings. 436
16.5 LCD Display Data Memory 437

16.6 Common Signals and Segment Signals 438
16.7 Supplying LCD Drive Voltage V_{LC0}, V_{LC1}, and V_{LC2} 440
16.8 Display Mode 442
16.8.1 4-time-division display example 442
16.9 Cautions on Emulation performed with IE-78K0-NS-A Emulator and IE-780822-NS-EM1 Emulation Board. 445

Chapter 17 Sound Generator (µPD780822B Subseries). 449
17.1 Sound Generator Function 449
17.2 Sound Generator Configuration 450
17.3 Sound Generator Control Registers. 450
17.4 Sound Generator Operations 455
17.4.1 Sound generator settings 455
17.4.2 To output basic cycle signal SGOF (without amplitude) 456
17.4.3 To output basic cycle signal SGO (with amplitude) 457

Chapter 18 Meter Controller / Driver (µPD780822B Subseries). 459
18.1 Meter Controller/Driver Functions 459
18.2 Meter Controller/Driver Configuration 461
18.3 Meter Controller/Driver Control Registers. 463
18.4 Meter Controller/Driver Operations 468
18.4.1 Basic operation of free-running up counter (SMCNT) 468
18.4.2 Update of PWM data 469
18.4.3 Operation of 1-bit addition circuit 470
18.4.4 PWM output operation (output with 1 clock shifted) 471

Chapter 19 Interrupt Functions (µPD780822B Subseries) 473
19.1 Interrupt Function Types. 473
19.2 Interrupt Sources and Configuration 474
19.3 Interrupt Function Control Registers 477
19.4 Interrupt Servicing Operations 483
19.4.1 Non-maskable interrupt request acknowledge operation 483
19.4.2 Maskable interrupt request acknowledge operation 485
19.4.3 Software interrupt request acknowledge operation 488
19.4.4 Multiple interrupt servicing 488
19.4.5 Interrupt request reserve 492

Chapter 20 Standby Function (µPD780822B Subseries) 495
20.1 Standby Function and Configuration. 495
20.1.1 Standby function 495
20.1.2 Registers controlling standby function 496
20.2 Standby Function Operation. 499
20.2.1 HALT mode. 499
20.2.2 STOP mode 505

Chapter 21 Reset Function (µPD780822B Subseries) 511
21.1 Register for Confirming Reset Source. 517

Chapter 22 Clock Monitor (µPD780822B Subseries) 519
22.1 Functions of Clock Monitor. 519
22.2 Configuration of Clock Monitor 519
22.3 Registers Controlling Clock Monitor 520
22.4 Operation of Clock Monitor. 521

Chapter 23 Flash Memory (µPD780822B Subseries) 527
23.1 Internal Memory Size Switching Register 528
23.2 Internal Expansion RAM Size Switching Register 529
23.3 Writing with Flash Programmer 530

23.4 Programming Environment 534
23.5 Communication Mode 534
23.6 Processing of Pins on Board 537
23.6.1 FLMD0 pin 537
23.6.2 FLMD1 pin 537
23.6.3 Serial interface pins 538
23.6.4 $\overline{\text{RESET}}$ pin 540
23.6.5 Port pins 540
23.6.6 Other signal pins 540
23.6.7 Power supply 540
23.7 Programming Method 541
23.7.1 Controlling flash memory 541
23.7.2 Flash memory programming mode 542
23.7.3 Selecting communication mode 543
23.7.4 Communication commands 544
23.8 Flash Memory Programming by Self-Programming 545
23.9 Boot Swap Function 548
23.9.1 Outline of boot swap function 548

Chapter 24 Mask Options / Option Bytes (µPD780822B Subseries) 549
24.1 Mask Options (Mask ROM Version) 549
24.2 Option Bytes (Flash Memory Version) 550

Chapter 25 Instruction Set (µPD780822B Subseries) 551
25.1 Legends Used in Operation List 551
25.1.1 Operand identifiers and description methods 551
25.1.2 Description of “operation” column 552
25.2 Operation List 553
25.3 Instructions Listed by Addressing Type 561

Chapter 26 Electrical Specifications (µPD780822B Subseries) 565
26.1 Absolute Maximum Ratings 565
26.2 Capacitance 567
26.3 Main System Clock Oscillation Circuit Characteristics 568
26.4 Internal Low-speed Oscillator Oscillation Circuit Characteristics 569
26.5 Subsystem Clock Oscillation Circuit Characteristics 570
26.6 DC Characteristics 571
26.7 AC Characteristics 576
26.7.1 Basic Operation 576
26.7.2 Serial interface 578
26.7.3 Sound Generator Characteristics 580
26.7.4 Meter Controller/Driver Characteristics 581
26.7.5 A/D Converter Characteristics 582
26.7.6 Data Memory Stop Mode Low Supply Voltage Data Retention Characteristics 583
26.8 Flash Memory Programming Characteristics 585

Chapter 27 Package Drawing (µPD780822B Subseries) 587

Chapter 28 Recommended Soldering Conditions (µPD780822B Subseries) 589

Chapter 29 Cautions For Wait (µPD780822B Subseries) 591
29.1 Cautions for Wait 591
29.2 Peripheral Hardware That Generates Wait 592
29.3 Example of Wait Occurrence 594

Appendix A Development Tools 595

Appendix B Index 599

Appendix C Revision History 603

List of Figures

Figure 1-1: Pin Configuration 31
Figure 1-2: Pin Configuration 32
Figure 1-3: 78K/0 CAN Products Expansion 34
Figure 1-4: Function Block Diagram 36
Figure 2-1: Pin Configuration 57
Figure 2-2: Terminal I/O types (1/2) 60
Figure 3-1: Memory Map of the µPD780821 63
Figure 3-2: Memory Map of the µPD780822B 64
Figure 3-3: Memory Map of the µPD78F0822 65
Figure 3-4: Program memory common area mapping 68
Figure 3-5: BANK: bank switching register format (µPD780821) 69
Figure 3-6: BANK: bank switching register format (µPD780822B, µPD78F0822) 70
Figure 3-7: Data Memory Addressing of the µPD780822B Subseries 72
Figure 3-8: Program Counter Configuration 73
Figure 3-9: Program Status Word Configuration 73
Figure 3-10: Stack Pointer Configuration 75
Figure 3-11: Data to be Saved to Stack Memory 75
Figure 3-12: Data to be Restored from Stack Memory 75
Figure 3-13: General Register Configuration 76
Figure 3-14: Relative Addressing 83
Figure 3-15: Immediate Addressing 84
Figure 3-16: Table Indirect Addressing 85
Figure 3-17: Register Addressing 86
Figure 3-18: Register Addressing 88
Figure 3-19: Direct addressing 89
Figure 3-20: Short direct addressing 90
Figure 3-21: Special-Function Register (SFR) Addressing 91
Figure 3-22: Register indirect addressing 92
Figure 3-23: Based addressing description example 93
Figure 3-24: Based indexed addressing description example 94
Figure 3-25: Stack addressing description example 95
Figure 4-1: Port Types 97
Figure 4-2: P00 to P03 Configuration 101
Figure 4-3: Port 1 Configuration (1/2) 102
Figure 4-4: P20 to P27 Configurations 104
Figure 4-5: P30 to P33 Configurations 105
Figure 4-6: P70 to P77 Configuration 106
Figure 4-7: P80 to P87 Configuration 107
Figure 4-8: Port 9 Configuration (1/2) 108
Figure 4-9: Port 10 Configuration (1/2) 110
Figure 4-10: Port 11 Configuration (1/2) 112
Figure 4-11: P120 Configuration 114
Figure 4-12: P130 Configuration 115
Figure 4-13: P140 to P147 Configuration 116
Figure 4-14: P150 to P157 Configuration 117
Figure 4-15: Port Mode Register Format (1/2) 118
Figure 4-16: Pull-Up Resistor Option Register (PU0, PU1, PU3, PU7 to PU12) Format 120
Figure 4-17: Port Function Register (PF7, PF8, PF9, PF10 and PF11) Format 121
Figure 5-1: Block Diagram of Clock Generator 126
Figure 5-2: Processor Control Register (PCC) Format (1/2) 127
Figure 5-3: Internal Low-speed Oscillator Mode Register (RCM) Format 129
Figure 5-4: Main Clock Mode Register (MCM) Format 130
Figure 5-5: Main OSC Control Register (MOC) Format 131
Figure 5-6: Oscillation Stabilization Time Counter Status Register (OSTC) Format 132
Figure 5-7: Oscillation Stabilization Time Select Register (OSTS) Format 133

Figure 5-8: Input Switch Control Register (ISC) Format (1/2) 134
Figure 5-9: External circuit of Main System Clock Oscillator 136
Figure 5-10: External circuit of Subsystem Clock Oscillator 137
Figure 5-11: Examples of oscillator with Bad Connection (1/3) 138
Figure 5-12: Subsystem Clock Feedback Resistor 141
Figure 5-13: Timing Diagram of CPU Default Start Using Internal Low-speed Oscillator 143
Figure 5-14: Status Transition Diagram (1/4) 144
Figure 5-15: Switching from Internal Low-speed Oscillator Clock to X1 Input Clock (Flowchart) .. 151
Figure 5-16: Switching from X1 Input Clock to Internal Low-speed Oscillator Clock (Flowchart) .. 152
Figure 5-17: Switching from X1 Input Clock to Subsystem Clock (Flowchart) 153
Figure 5-18: Switching from Subsystem Clock to X1 Input Clock (Flowchart) 154
Figure 6-1: Timer 2 (TM2) Block Diagram 157
Figure 6-2: 16-Bit Timer Mode Control Register (TMC2) Format 161
Figure 6-3: Capture Pulse Control Register (CRC2) Format 162
Figure 6-4: Prescaler Mode Register (PRM2) Format 163
Figure 6-5: Input Switch Control Register (ISC) Format 164
Figure 6-6: Configuration Diagram for Pulse Width Measurement by Using the Free Running Counter 166
Figure 6-7: Timing of Pulse Width Measurement Operation by Using the Free Running Counter and One Capture Register (with Both Edges Specified) 167
Figure 6-8: CR2m Capture Operation with Rising Edge Specified 168
Figure 6-9: Timing of Pulse Width Measurement Operation by Free Running Counter (with Both Edges Specified) 169
Figure 6-10: 16-Bit Timer Register Start Timing 170
Figure 6-11: Capture Register Data Retention Timing 170
Figure 7-1: Block Diagram of Timer P 175
Figure 7-2: TMPn capture/compare register 0 (TPnCCR0) Format 176
Figure 7-3: TMPn capture/compare register 1 (TPnCCR1) Format 177
Figure 7-4: TMPn counter read buffer register (TPnCNT) Format 178
Figure 7-5: TMPn control register 0 (TPnCTL0) Format) 179
Figure 7-6: TMPn timer control register 1 (TPnCTL1) Format 181
Figure 7-7: TMPn I/O control register 0 (TPnIOC0) Format 182
Figure 7-8: TMPn I/O control register 1 (TPnIOC1) Format 183
Figure 7-9: TMPn option register 0 (TPnOPT0) Format 184
Figure 7-10: Input Switch Control Register (ISC) Format 185
Figure 7-11: Flowchart of Basic Operation of Anytime Write 187
Figure 7-12: Timing Chart of Anytime Write 188
Figure 7-13: Flowchart of Basic Operation of Reload 189
Figure 7-14: Timing Chart of Reload 190
Figure 7-15: Flowchart of Basic Operation in Interval Timer Mode 191
Figure 7-16: Timing of Basic Operation in Interval Timer Mode (1/2) 192
Figure 7-17: Flowchart of Basic Operation in One-Shot Pulse Mode 195
Figure 7-18: Timing of Basic Operation in One-Shot Pulse Mode 196
Figure 7-19: Flowchart of Basic Operation in PWM Mode (1/2) 198
Figure 7-20: Timing of Basic Operation in PWM Mode (1/2) 200
Figure 7-21: Flowchart of Basic Operation in Free-Running Mode 203
Figure 7-22: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 0) .. 204
Figure 7-23: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 1, TPnCCS0 = 1) .. 205
Figure 7-24: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 1) .. 206
Figure 7-25: Flowchart of Basic Operation in Pulse Width Measurement Mode 207
Figure 7-26: Timing of Basic Operation in Pulse Width Measurement Mode 208
Figure 7-27: Tuned Operation Image (TMP0, TMP1, TMP2) 211
Figure 7-28: Basic Operation Timing of Tuned PWM Function (TMP0, TMP1, TMP2) 212
Figure 8-1: 8-Bit Timer/Event Counter 50 Block Diagram 215
Figure 8-2: 8-Bit Timer/Event Counter 51 Block Diagram 216
Figure 8-3: Block Diagram of 8-Bit Timer/Event Counters 50 and 51 Output Control Circuit 217
Figure 8-4: Timer Clock Select Register 50 (TLC50) Format 218
Figure 8-5: Timer Clock Select Register 51 (TLC51) Format 219

Figure 8-6: 8-Bit Timer Mode Control Register 50 (TMC50) Format 220
Figure 8-7: 8-Bit Timer Mode Control Register 51 (TMC51) Format 221
Figure 8-8: Port Mode Register 1 (PM1) Format 222
Figure 8-9: Port Mode Register 11 (PM11) Format 222
Figure 8-10: Port Function Register 11 (PF11) Format 223
Figure 8-11: 8-Bit Timer Mode Control Register Settings for Interval Timer Operation 224
Figure 8-12: Interval Timer Operation Timings (1/3) 225
Figure 8-13: 8-Bit Timer Mode Control Register Setting for External Event Counter Operation 229
Figure 8-14: External Event Counter Operation Timings (with Rising Edge Specified) 229
Figure 8-15: 8-Bit Timer Mode Control Register Settings for Square-Wave Output Operation 230
Figure 8-16: Square-wave Output Operation Timing 231
Figure 8-17: 8-Bit Timer Control Register Settings for PWM Output Operation 232
Figure 8-18: PWM Output Operation Timing (Active high setting) 233
Figure 8-19: PWM Output Operation Timings (CRn = 00H, active high setting) 233
Figure 8-20: PWM Output Operation Timings (CRn = FFH, active high setting) 234
Figure 8-21: PWM Output Operation Timings (CRn changing, active high setting) 234
Figure 8-22: 8-bit Timer Registers 50 and 51 Start Timings 235
Figure 8-23: External Event Counter Operation Timings 235
Figure 8-24: Timings after Compare Register Change during Timer Count Operation 236
Figure 9-1: Block Diagram of Watch Timer 237
Figure 9-2: Input Switch Control Register (ISC) Format 239
Figure 9-3: Watch Timer Mode Control Register (WTM) Format (1/2) 240
Figure 9-4: Operation Timing of Watch Timer/Interval Timer 244
Figure 9-5: Example of Generation of Watch Timer Interrupt Request (INTWT) (When Interrupt Period = 0.5 s) 245
Figure 10-1: Watchdog Timer Block Diagram 248
Figure 10-2: Watchdog Timer Enable Register (WDTE) Format 250
Figure 10-3: Watchdog Timer Mode Register (WDTM) Format (1/2) 251
Figure 10-4: Input Switch Control Register (ISC) Format 253
Figure 11-1: Block Diagram of Clock Output/Buzzer Output Controller 263
Figure 11-2: Format of Clock Output Selection Register (CKS) 265
Figure 11-3: Format of Port Mode Register 3 (PM3) 266
Figure 11-4: Remote Control Output Application Example 267
Figure 12-1: Block Diagram of A/D Converter 269
Figure 12-2: Format of A/D Converter Mode Register (ADM) 273
Figure 12-3: Timing Chart When Boost Reference Voltage Generator Is Used 274
Figure 12-4: Format of Analog Input Channel Specification Register (ADS) 275
Figure 12-5: Format of A/D Conversion Result Register (ADCR) 276
Figure 12-6: Format of Power-Fail Comparison Mode Register (PFM) 277
Figure 12-7: Format of Power-Fail Comparison Threshold Register (PFT) 277
Figure 12-8: Basic Operation of A/D Converter 279
Figure 12-9: Relationship Between Analog Input Voltage and A/D Conversion Result 280
Figure 12-10: A/D Conversion Operation 281
Figure 12-11: Power-Fail Detection (When PFEN = 1 and PFCM = 0) 282
Figure 12-12: Overall Error 284
Figure 12-13: Quantization Error 285
Figure 12-14: Zero-Scale Error 285
Figure 12-15: Full-Scale Error 286
Figure 12-16: Integral Linearity Error 286
Figure 12-17: Differential Linearity Error 287
Figure 12-18: Sampling time 287
Figure 12-19: Circuit Configuration of Series Resistor String 288
Figure 12-20: Analog Input Pin Connection 289
Figure 12-21: Timing of A/D Conversion End Interrupt Request Generation 290
Figure 12-22: Timing of A/D Converter Sampling and A/D Conversion Start Delay 291
Figure 12-23: Internal Equivalent Circuit of ANIn Pin 292
Figure 13-1: Block Diagram of Serial Interface CSI10 296
Figure 13-2: Block Diagram of Serial Interface CSI11 297

Figure 13-3: Format of Serial Operation Mode Register 10 (CSIM10) (1/2) 298
Figure 13-4: Format of Serial Operation Mode Register 11 (CSIM11) 300
Figure 13-5: Format of Serial Clock Selection Register 10 (CSIC10) (1/2) 301
Figure 13-6: Format of Serial Clock Selection Register 11 (CSIC11) (1/2) 303
Figure 13-7: Serial operation mode register 1n (CSIM1n) Format..... 305
Figure 13-8: Timing in 3-Wire Serial I/O Mode (1/2)..... 309
Figure 13-9: Timing of Clock/Data Phase..... 311
Figure 13-10: Output Operation of First Bit..... 312
Figure 13-11: Output Value of SO1n Pin (Last Bit)..... 313
Figure 14-1: LIN Transmission Operation..... 316
Figure 14-2: LIN Reception Operation..... 317
Figure 14-3: Port Configuration for LIN Reception Operation..... 318
Figure 14-4: Block Diagram of Serial Interface UART6..... 320
Figure 14-5: Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) Format(1/2) ... 322
Figure 14-6: Asynchronous Serial Interface Reception Error Status Register 6 (ASIS6) Format . 324
Figure 14-7: Asynchronous Serial Interface Transmission Status Register 6 (ASIF6) Format 325
Figure 14-8: Clock Selection Register 6 (CKSR6) Format 326
Figure 14-9: Baud Rate Generator Control Register 6 (BRGC6) Format 327
Figure 14-10: Asynchronous Serial Interface Control Register 6 (ASICL6) Format (1/2) 328
Figure 14-11: Input Switch Control Register (ISC) Format 330
Figure 14-12: Port Mode Register 1 (PM1) Format 330
Figure 14-13: Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) Format 331
Figure 14-14: Format of Normal UART Transmit/Receive Data..... 334
Figure 14-15: Example of Normal UART Transmit/Receive Data Waveform..... 335
Figure 14-16: Normal Transmission Completion Interrupt Request Timing..... 337
Figure 14-17: Example of Continuous Transmission Processing Flow..... 339
Figure 14-18: Timing of Starting Continuous Transmission..... 340
Figure 14-19: Timing of Ending Continuous Transmission..... 341
Figure 14-20: Reception Completion Interrupt Request Timing..... 342
Figure 14-21: Reception Error Interrupt..... 343
Figure 14-22: Noise Filter Circuit..... 344
Figure 14-23: SBF Transmission..... 345
Figure 14-24: SBF Reception 346
Figure 14-25: Configuration of Baud Rate Generator..... 348
Figure 14-26: Permissible Baud Rate Range During Reception..... 351
Figure 14-27: Data Frame Length During Continuous Transmission 353
Figure 15-1: Data Frame..... 357
Figure 15-2: Remote Frame..... 357
Figure 15-3: Data Frame..... 358
Figure 15-4: Arbitration Field/Standard Format Mode 358
Figure 15-5: Arbitration Field/Extended Format Mode..... 359
Figure 15-6: Control Field (Standard Format Mode)..... 360
Figure 15-7: Control Field (Extended Format Mode) 360
Figure 15-8: Data Field 361
Figure 15-9: CRC Field..... 361
Figure 15-10: ACK Field 362
Figure 15-11: End of Frame..... 362
Figure 15-12: Interframe Space/Error Active 363
Figure 15-13: Interframe Space/Error Passive 363
Figure 15-14: Error Frame 364
Figure 15-15: Overload Frame..... 365
Figure 15-16: Nominal Bit Time (8 to 25 Time Quanta)..... 371
Figure 15-17: Adjusting Synchronization of the Data Bit 372
Figure 15-18: Bit Synchronization..... 373
Figure 15-19: Transmission State Shift Chart..... 374
Figure 15-20: Reception State Shift Chart..... 375
Figure 15-21: Error State Shift Chart 376
Figure 15-22: Structural Block Diagram..... 377

Figure 15-23: Connection to the CAN Bus 378
Figure 15-24: Transmit Message Definition Bits 382
Figure 15-25: Transmit Identifier (IDTX0 to IDTX4) Registers 383
Figure 15-26: Transmit Data 384
Figure 15-27: Control bits for Receive Identifier 387
Figure 15-28: Receive Status Bits (1/2) 388
Figure 15-29: Receive Identifier (IDREC0 to IDREC4) Registers 390
Figure 15-30: Receive Data 391
Figure 15-31: Identifier Compare with Mask 393
Figure 15-32: Control Bits for Mask Identifier 394
Figure 15-33: Mask Identifier 395
Figure 15-34: DCAN Control Register (DCANC) Format 396
Figure 15-35: CAN Control Register (1/2) 397
Figure 15-36: DCAN Support 398
Figure 15-37: Time Stamp Function 400
Figure 15-38: SOFOUT Toggle Function 400
Figure 15-39: Global Time System Function 400
Figure 15-40: CAN Error Status Register (1/3) 401
Figure 15-41: Transmit Error Counter 404
Figure 15-42: Receive Error Counter 404
Figure 15-43: Message Count Register (MCNT) (1/2) 405
Figure 15-44: Bit Rate Prescaler (1/2) 407
Figure 15-45: Synchronization Control Registers 0 and 1 (1/2) 409
Figure 15-46: Transmit Control Register (1/2) 413
Figure 15-47: Receive Message Register 415
Figure 15-48: Mask Control Register (1/2) 416
Figure 15-49: Redefinition Control Register (1/2) 419
Figure 15-50: Initialization Flow Chart 426
Figure 15-51: Transmit Preparation 427
Figure 15-52: Transmit Abort 428
Figure 15-53: Handling of Semaphore Bits by DCAN-Module 429
Figure 15-54: Receive with Interrupt, Software Flow 430
Figure 15-55: Receive, Software Polling 431
Figure 16-1: LCD Controller/Driver Block Diagram 434
Figure 16-2: LCD Clock Select Circuit Block Diagram 434
Figure 16-3: LCD Display Mode Register (LCDM) Format 435
Figure 16-4: LCD Display Control Register (LCDC) Format 436
Figure 16-5: Relationship between LCD Display Data Memory Contents and Segment/Common Outputs 437
Figure 16-6: Common Signal Waveform 439
Figure 16-7: Common Signal and Segment Signal Voltages and Phases 439
Figure 16-8: Example of Connection of LCD Drive Power Supply (1/2) 440
Figure 16-9: 4-Time-Division LCD Display Pattern and Electrode Connections 442
Figure 16-10: 4-Time-Division LCD Panel Connection Example 443
Figure 16-11: 4-Time-Division LCD Drive Waveform Examples (1/3 Bias Method) 444
Figure 16-12: LCD Display Mode Register 3(LCDM3) Format 446
Figure 16-13: LCD Clock Control Register 3 (LCDC3) Format 447
Figure 17-1: Sound Generator Block Diagram 449
Figure 17-2: Concept of Each Signal 450
Figure 17-3: Sound Generator Control Register (SGCR) Format (1/2) 451
Figure 17-4: Sound Generator Buzzer Control Register (SGBR) Format 453
Figure 17-5: Sound Generator Amplitude Register (SGAM) Format 454
Figure 17-6: Sound Generator Output Operation Timing 456
Figure 17-7: Sound Generator Output Operation Timing 457
Figure 18-1: Meter Controller/Driver Block Diagram 459
Figure 18-2: 1-bit Addition Circuit Block Diagram 460
Figure 18-3: Timer Mode Control Register (MCNTC) Format 463
Figure 18-4: Compare Control Register n (MCMPCn) Format 464

Figure 18-5: Port Mode Control Register (PMC) Format (1/2) 465
Figure 18-6: Meter Controller/Driver Clock Register (SMSWI) Format 467
Figure 18-7: Restart Timing after Count Stop (Count Start Æ Count Stop Æ Count Start) 468
Figure 18-8: Update of PWM data 469
Figure 18-9: Timing in 1-bit Addition Circuit Operation 470
Figure 18-10: Timing of Output with 1 Clock Shifted 471
Figure 19-1: Basic Configuration of Interrupt Function (1/2) 475
Figure 19-2: Interrupt Request Flag Register Format 478
Figure 19-3: Interrupt Mask Flag Register Format 479
Figure 19-4: Priority Specify Flag Register Format 480
Figure 19-5: Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register 481
Figure 19-6: Program Status Word Format 482
Figure 19-7: Flowchart from Non-Maskable Interrupt Generation to Acknowledge 483
Figure 19-8: Non-Maskable Interrupt Request Acknowledge Timing 484
Figure 19-9: Non-Maskable Interrupt Request Acknowledge Operation 484
Figure 19-10: Interrupt Request Acknowledge Processing Algorithm 486
Figure 19-11: Interrupt Request Acknowledge Timing (Minimum Time) 487
Figure 19-12: Interrupt Request Acknowledge Timing (Maximum Time) 487
Figure 19-13: Multiple Interrupt Example (1/2) 490
Figure 19-14: Interrupt Request Hold 493
Figure 20-1: Format of Oscillation Stabilization Time Counter Status Register (OSTC) 497
Figure 20-2: Format of Oscillation Stabilization Time Select Register (OSTS) 498
Figure 20-3: HALT Mode Release by Interrupt Request Generation 502
Figure 20-4: HALT Mode Release by $\overline{\text{RESET}}$ Input (1/2) 503
Figure 20-5: Operation Timing When STOP Mode Is Released 507
Figure 20-6: STOP Mode Release by Interrupt Request Generation 508
Figure 20-7: STOP Mode Release by $\overline{\text{RESET}}$ Input 509
Figure 21-1: Block Diagram of Reset Function 511
Figure 21-2: Timing of Reset by $\overline{\text{RESET}}$ Input 512
Figure 21-3: Timing of Reset Due to Watchdog Timer Overflow 512
Figure 21-4: Timing of Reset in STOP Mode by $\overline{\text{RESET}}$ Input 513
Figure 21-5: Reset Control Flag Register (RESF) Format 517
Figure 22-1: Block Diagram of Clock Monitor 519
Figure 22-2: Format of Clock Monitor Mode Register (CLM) 520
Figure 22-3: Timing of Clock Monitor (1/4) 522
Figure 23-1: Internal Memory Size Switching Register (IMS) Format 528
Figure 23-2: Internal Expansion RAM Size Switching Register (IXS) Format 529
Figure 23-3: Example of wiring adapter for Flash Memory writing in 3-Wire Serial I/O (CSI10) Mode 531
Figure 23-4: Example of wiring adapter for Flash Memory writing in 3-Wire Serial I/O (CSI10 + HS) Mode 532
Figure 23-5: Example of Wiring Adapter for Flash Memory Writing in UART (UART6) Mode 533
Figure 23-6: Environment for Writing Program to Flash Memory 534
Figure 23-7: Communication with Dedicated Flash Programmer (CSI10) 534
Figure 23-8: Communication with Dedicated Flash Programmer (CSI10 + HS) 535
Figure 23-9: Communication with Dedicated Flash Programmer (UART6) 535
Figure 23-10: FLMD0 Pin Connection Example 537
Figure 23-11: FLMD1 Pin Connection Example 537
Figure 23-12: Signal Collision (Input Pin of Serial Interface) 538
Figure 23-13: Malfunction of Other Device 539
Figure 23-14: Signal Collision ($\overline{\text{RESET}}$ Pin) 540
Figure 23-15: Flash Memory Manipulation Procedure 541
Figure 23-16: Flash Memory Programming Mode 542
Figure 23-17: Communication Commands 544
Figure 23-18: Operation Mode and Memory Map for Self-Programming (µPD78F0822) 546
Figure 23-19: Self-Programming Procedure 547
Figure 23-20: Image of Boot Swap Function 548

Figure 24-1: Allocation of Option Bytes (Flash Memory Version) 550
Figure 24-2: Format of Option Bytes (Int. osc.) (Flash Memory Version) 550
Figure A-1: Development Tool Configuration 596

List of Tables

Table 1-1: The major functional differences between the subseries 35
Table 1-2: Overview of Functions 37
Table 2-1: Port Pins 39
Table 2-2: Non-port pins 42
Table 2-3: Types of Input/Output Circuits 58
Table 3-1: Internal ROM Capacities 66
Table 3-2: Vectored Interrupts 67
Table 3-3: Special Function Register List 78
Table 3-4: Implied Addressing 87
Table 3-5: Register Addressing 88
Table 3-6: Direct addressing 89
Table 3-7: Short direct addressing 90
Table 3-8: Special-Function Register (SFR) Addressing 91
Table 3-9: Register indirect addressing 92
Table 3-10: Based addressing 93
Table 3-11: Based indexed addressing 94
Table 4-1: Pin Input/Output Types 98
Table 4-2: Port Configuration 100
Table 5-1: Clock Generator Configuration 125
Table 5-2: Relationship between CPU clock and Minimum Instruction Execution Time 129
Table 5-3: Relationship Between Operation Clocks in Each Operation Status 148
Table 5-4: Oscillation Control Flags and Clock Oscillation Status 148
Table 5-5: Maximum Time Required to Switch Between Internal Low-speed Oscillator Clock and X1 Input Clock 149
Table 5-6: Maximum Time Required for CPU Clock Switchover 150
Table 5-7: Clock and Register Setting 155
Table 6-1: Timer 2 Configuration 159
Table 7-1: Configuration of TMP0 to TMP2 174
Table 7-2: Tuned Operation Mode of Timers 209
Table 7-3: Timer Modes Usable in Tuned Operation Mode 210
Table 7-4: Timer Output Functions 210
Table 8-1: 8-Bit Timer/Event Counter 50 Interval Times 213
Table 8-2: 8-Bit Timer/Event Counter 51 Interval Times 213
Table 8-3: 8-Bit Timer/Event Counter 50 Square-Wave Output Ranges 214
Table 8-4: 8-Bit Timer/Event Counter 51 Square-Wave Output Ranges 214
Table 8-5: 8-Bit Timer/Event Counters 50 and 51 Configurations 215
Table 8-6: 8-Bit Timer/Event Counters 50 Interval Times 228
Table 8-7: 8-Bit Timer/Event Counters 51 Interval Times 228
Table 8-8: 8-Bit Timer/Event Counters 50 Square-Wave Output Ranges (8-Bit Timer/Event Counter Mode) 231
Table 8-9: 8-Bit Timer/Event Counters 51 Square-Wave Output Ranges (8-Bit Timer/Event Counter Mode) 231
Table 9-1: Watch Timer Interrupt Time 238
Table 9-2: Interval Timer Interrupt Time 238
Table 9-3: Watch Timer Configuration 239
Table 9-4: Watch Timer Interrupt INTWDT Period Selection 242
Table 9-5: Interval Timer Interrupt INTWTI Period Selection 243
Table 10-1: Watchdog Timer Configuration 248
Table 10-2: Mask Option Setting and Watchdog Timer Operation Mode 249
Table 10-3: Watchdog Timer Overflow Time 255
Table 10-4: Watchdog Timer Overflow Time 257
Table 10-5: Watchdog Timer Overrun Detection Time 258
Table 11-1: Clock Output/Buzzer Output Controller Configuration 264
Table 12-1: Registers of A/D Converter Used on Software 270
Table 12-2: Settings of ADCS and ADCE 274

Table 12-3: A/D converter sampling time and A/D conversion start delay time (ADM set value) 291
Table 12-4: Resistance and Capacitance Values of Equivalent Circuit (Reference Values) 292
Table 12-5: Different SFR Setting for IE-780822-NS-EM1 Emulation Board 293
Table 13-1: Configuration of Serial Interfaces CSI10 and CSI11 296
Table 13-2: Relationship Between Register Settings and Pins for Serial interface CSI10 307
Table 13-3: Relationship Between Register Settings and Pins for Serial interface CSI11 308
Table 13-4: SO1n Output Status 314
Table 14-1: Configuration of Serial Interface UART6 319
Table 14-2: Relationship Between Register Settings and Pins 333
Table 14-3: Cause of Reception Error 343
Table 14-4: Set Data of Baud Rate Generator 350
Table 14-5: Maximum/Minimum Permissible Baud Rate Error 352
Table 15-1: Outline of the Function 355
Table 15-2: Bit Number of the Identifier 359
Table 15-3: RTR Setting 359
Table 15-4: Mode Setting 359
Table 15-5: Data Length Code Setting 360
Table 15-6: Operation in the Error State 363
Table 15-7: Definition of each Field 364
Table 15-8: Definition of each Frame 365
Table 15-9: Arbitration 366
Table 15-10: Bit Stuffing 366
Table 15-11: Error Types 368
Table 15-12: Output Timing of the Error Frame 368
Table 15-13: Types of Error 369
Table 15-14: Error Counter 370
Table 15-15: Segment Name and Segment Length 371
Table 15-16: CAN Configuration 378
Table 15-17: SFR Definitions 379
Table 15-18: SFR Bit Definitions 379
Table 15-19: Message and Buffer Configuration 380
Table 15-20: Transmit Message Format 381
Table 15-21: Receive Message Format 386
Table 15-22: Mask Function 392
Table 15-23: Possible Setup of the SOFOUT Function 399
Table 15-24: Transmission / Reception Flag 399
Table 15-25: Possible Reactions of the DCAN 404
Table 15-26: Mask Operation Buffers 417
Table 15-27: Interrupt Sources 421
Table 16-1: Maximum Number of Display Pixels 433
Table 16-2: LCD Controller/Driver Configuration 433
Table 16-3: COM Signals 438
Table 16-4: LCD Drive Voltage 439
Table 16-5: LCD Drive Voltage Supply 440
Table 16-6: Selection and Non-Selection Voltages (COM0 to COM3) 442
Table 17-1: Sound Generator Configuration 450
Table 17-2: Maximum and Minimum Values of the Buzzer Output Frequency 452
Table 18-1: Meter Controller/Driver Configuration 461
Table 19-1: Interrupt Source List 474
Table 19-2: Various Flags Corresponding to Interrupt Request Sources 477
Table 19-3: Times from Maskable Interrupt Request Generation to Interrupt Service 485
Table 19-4: Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing 489
Table 20-1: Relationship Between Operation Clocks in Each Operation Status 495
Table 20-2: Operating States in HALT Mode (1/3) 499
Table 20-3: Operation in Response to Interrupt Request in HALT Mode 504
Table 20-4: Operating States in STOP Mode 505
Table 20-5: Operation in Response to Interrupt Request in STOP Mode 510
Table 21-1: Hardware Status after Reset Acknowledgement 513

Table 21-2: RESF Status When Reset Request Is Generated 517
Table 22-1: Configuration of Clock Monitor 519
Table 22-2: Operation Status of Clock Monitor (When CLME = 1) 521
Table 23-1: Differences among µPD78F0822 and Mask ROM Version 527
Table 23-2: Internal Memory Size Switching Register Settings 528
Table 23-3: Internal Expansion RAM Size Switching Register Settings 529
Table 23-4: Wiring Between µPD78F0822 and Dedicated Flash Programmer 530
Table 23-5: Pin Connection 536
Table 23-6: Pins Used by Each Serial Interface 538
Table 23-7: Relationship of Operation Mode of FLMD0 and FLMD1 Pins 542
Table 23-8: Communication Modes 543
Table 23-9: Flash Memory Control Commands 544
Table 23-10: Response Commands 544
Table 23-11: Correspondence among Bank Numbers, CPU Addresses, and Flash Real Addresses for µPD78F0822 546
Table 25-1: Operand Identifiers and Description Methods 551
Table 25-2: Operation List 553
Table 25-3: 8-bit instructions 561
Table 25-4: 16-bit instructions 562
Table 25-5: Bit manipulation instructions 562
Table 25-6: Call/instructions/branch instructions 563
Table 29-1: Registers that Generate Wait and Number of CPU Wait Clocks 592
Table 29-2: Number of Wait Clocks and Number of Execution Clocks on Occurrence of Wait (A/D Converter) 594

Chapter 1 Outline (µPD780822B Subseries)

1.1 Features

- Internal memory

Part Number	Program Memory (ROM)	Data Memory				Package
		Internal High-Speed RAM (IHRAM)	Internal Expansion RAM (IXRAM)	LCD Display RAM	DCAN RAM	
µPD780821B	90 Kbytes	1024 bytes	4096 bytes	34 bytes	288 bytes	100-pin plastic QFP (fine pitch)
µPD780822B	120 Kbytes	1024 bytes		34 bytes	288 bytes	100-pin plastic QFP (fine pitch)
µPD78F0822B	120 Kbytes	1024 bytes		34 bytes	288 bytes	100-pin plastic QFP (fine pitch)

- Instruction execution time can be changed
- I/O ports: 76
- 10-bit resolution A/D converter: 8 channels
- Sound generator: 1 channel
- LCD Controller / Driver
- Meter Controller / Driver
- CAN interface : 1 channel
- Serial Interface : 3 channels
- 8-bit Timer : 4 channels
- 16-bit Timer : 4 channels
- Supply voltage : V_{DD} = 4.0 to 5.5 V

The CAN macro is qualified according the requirements of ISO 11898 using the test procedures defined by ISO 16845 and passed successfully the test procedures as recommended by C & S / FH Wolfenbuettel.

1.2 Application fields

The CAN ASSP3+ is ideally suited for automotive applications, like dashboard applications. It is also an excellent choice for other applications where a combination of sophisticated peripheral functions and CAN network support is required.

1.3 Ordering Information

Part Number	Package	Internal ROM
µPD780821BGC(A)-xxx-8EA	100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm)	Mask ROM
µPD780821BGF(A)-xxx-3BA	100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)	Mask ROM
µPD780822BGC(A)-xxx-8EA	100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm)	Mask ROM
µPD780822BGF(A)-xxx-3BA	100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)	Mask ROM
µPD78F0822BGC(A)-8EA	100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm)	Flash Memory
µPD78F0822BGF(A)-3BA	100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)	Flash Memory

Remark: xxx indicates ROM code suffix.

1.4 Quality Grade

Part Number	Package	Quality Grade
µPD780821BGC(A)-xxx-8EA	100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm)	Special
µPD780821BGF(A)-xxx-3BA	100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)	Special
µPD780822BGC(A)-xxx-8EA	100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm)	Special
µPD780822BGF(A)-xxx-3BA	100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)	Special
µPD78F0822BGC(A)-8EA	100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm)	Special
µPD78F0822BGF(A)-3BA	100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)	Special

Remark: xxx indicates ROM code suffix.

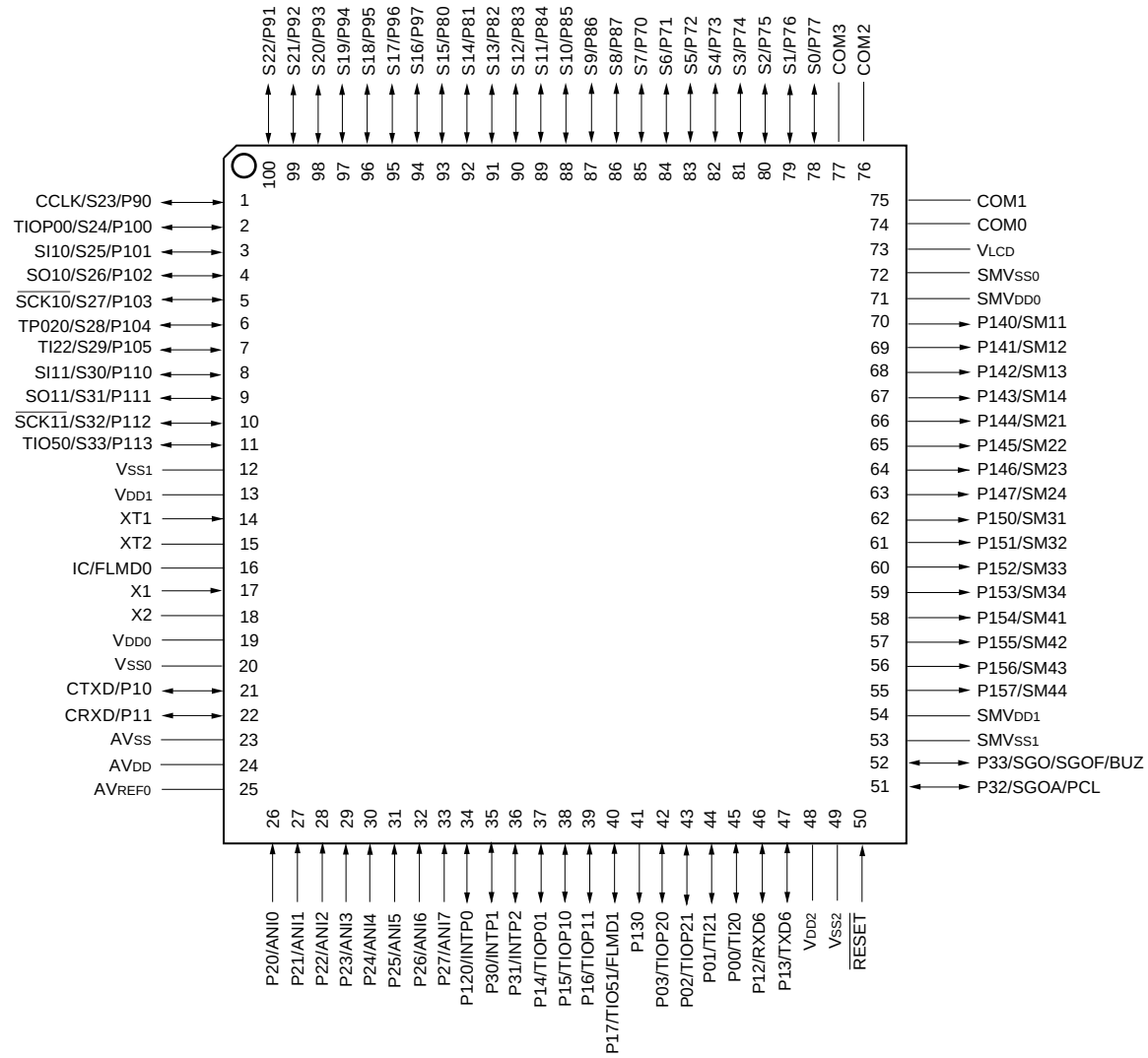
Please refer to "Quality Grades on NEC Semiconductor Device" (Document No. C11531E) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

1.5 Pin Configuration (Top View)

- 100-pin plastic QFP (14 mm × 14 mm)

  µPD780821BGC(A)-xxx-8EA, µPD780822BGC(A)-xxx-8EA, µPD78F0822BGC(A)-8EA

Figure 1-1: Pin Configuration



Cautions: 1. Connect IC (internally connected) pin directly to V_{SS}.

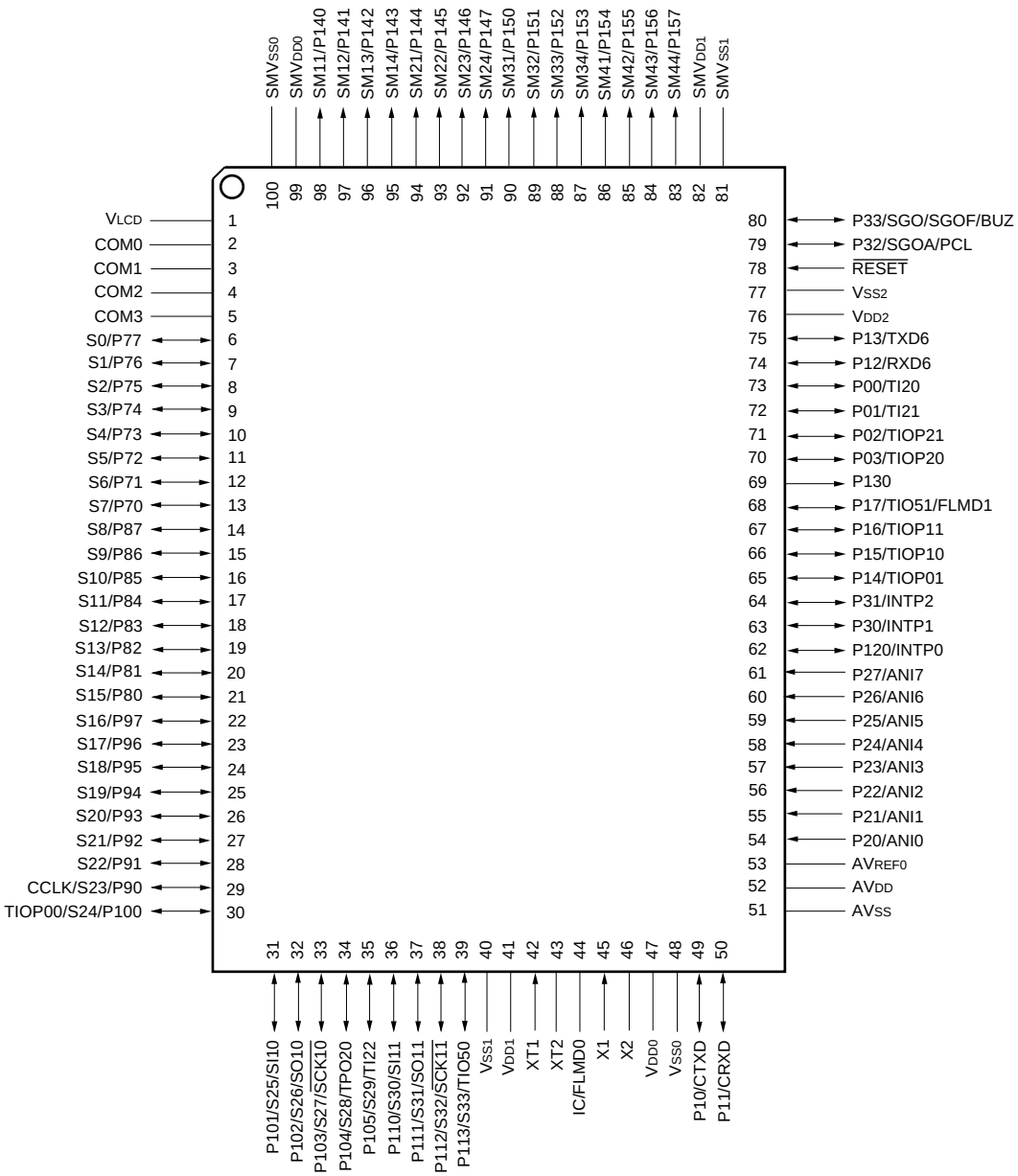
2. AV_{SS} pin should be connected to V_{SS}.

Remark: When these devices are used in applications, that require reduction of the noise, generated from inside the microcontroller, the implementation of noise reduction measures, such as connecting the V_{SS0}, V_{SS1} and V_{SS2} to different ground lines, is recommended.

- 100-pin plastic QFP (14 mm × 20 mm)

  µPD780821BGF(A)-xxx-3BA, µPD780822BGF(A)-xxx-3BA, µPD78F0822BGF(A)-3BA

Figure 1-2: Pin Configuration



Cautions: 1. Connect IC (internally connected) pin directly to V_{SS}.

2. AV_{SS} pin should be connected to V_{SS}.

Remark: When these devices are used in applications, that require reduction of the noise, generated from inside the microcontroller, the implementation of noise reduction measures, such as connecting the V_{SS0}, V_{SS1} and V_{SS2} to different ground lines, is recommended.

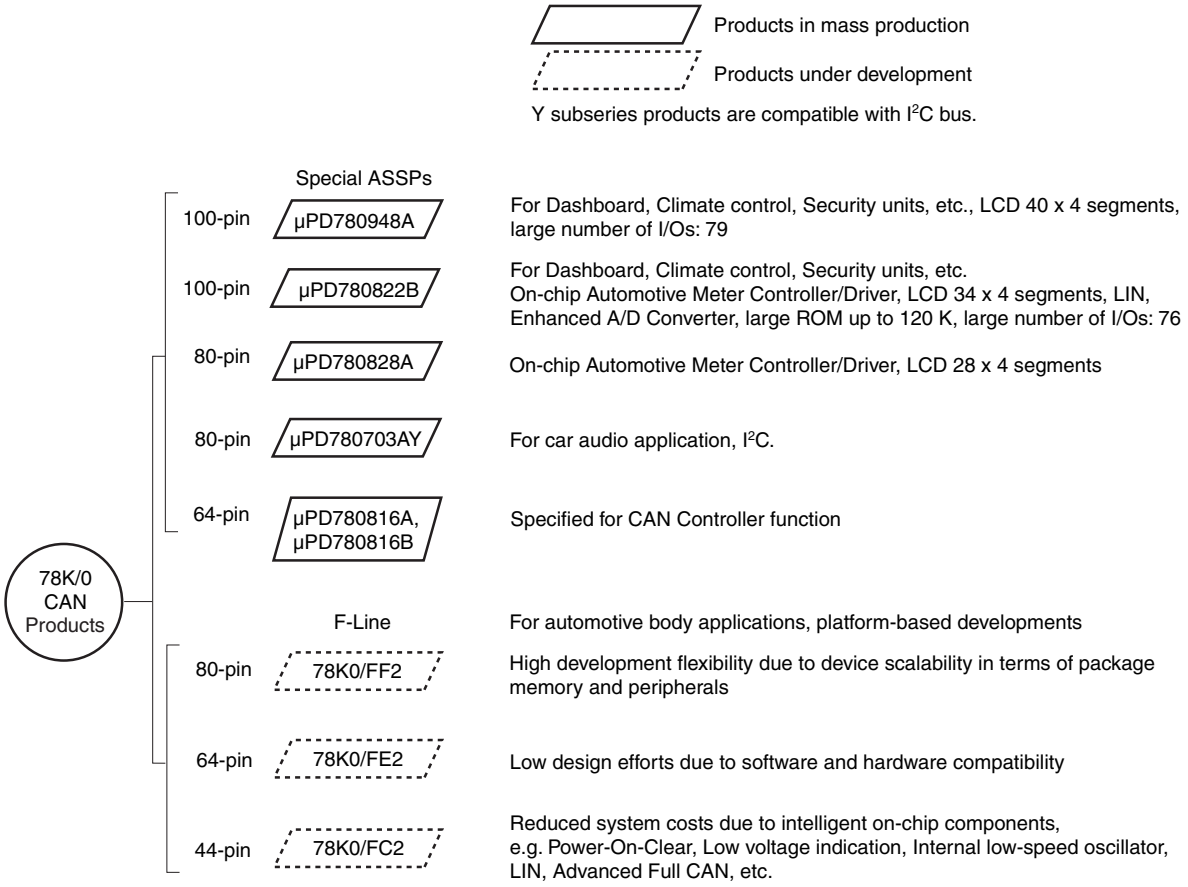
Pin Identifications

P00 to P03	Port 0	SGO	Sound Generator Output
P10 to P17	Port 1	SGOA	Sound Generator Amplitude
P20 to P27	Port 2	SGOF	Sound Generator Frequency
P30 to P33	Port 3	PCL	Programmable Clock Output
P70 to P77	Port 7	SM11 to SM14	Meter Controller/Driver
P80 to P87	Port 8	SM21 to SM24	Meter Controller/Driver
P90 to P97	Port 9	SM31 to SM34	Meter Controller/Driver
P100 to P105	Port 10	SM41 to SM44	Meter Controller/Driver
P110 to P113	Port 11	SMV_{DD0}, SMV_{DD1}	Meter Controller/Driver Power Supply
P120	Port 12	SMV_{SS0}, SMV_{SS1}	Meter Controller/Driver Ground
P130	Port 13	S0 to S33	Segment Output
P140 to P147	Port 14	COM0 to COM3	Common Output
P150 to P157	Port 15	X1	Crystal (Main-Osc)
INTP0 to INTP2	External Interrupt Inputs	X2	Crystal (Main-Osc)
TIO50, TIO51	Timer Input/Output	XT1	Crystal (Sub-Osc)
TI20 to TI22	Timer Input	XT2	Crystal (Sub-Osc)
TPO20	Timer Output	$\overline{\text{RESET}}$	Reset
TIOP00, TIOP01 TIOP10, TIOP11 TIOP20, TIOP21	Timer Input/ Output	ANI0 to ANI7	Analog Input
CRXD	CAN Receive Data	AV_{SS}	Analog Ground
CTXD	CAN Transmit Data	AV_{DD}	Analog Power Supply
CCLK	CAN Clock	AV_{REF0}	Analog Reference Voltage
SI10, SI11	Serial Input	V_{DD0}	CPU 5 V Power Supply
SO10, SO11	Serial Output	V_{SS0}	CPU Ground
$\overline{\text{SCK10}}$, $\overline{\text{SCK11}}$	Serial Clock	V_{DD1}, V_{DD2}	Port 5 V Power Supply
RXD6	UART Receive Data Input	V_{SS1}, V_{SS2}	Port Ground
TXD6	UART Transmit Data Output	V_{LCD}	LCD drive voltage
BUZ	Buzzer output	IC/FLMOD0	Internally Connected

1.6 78K/0 CAN Products Expansion

The following shows the products organized according to usage.
The names in the parallelograms are subseries.

Figure 1-3: 78K/0 CAN Products Expansion



The major functional differences between the subseries are shown below.

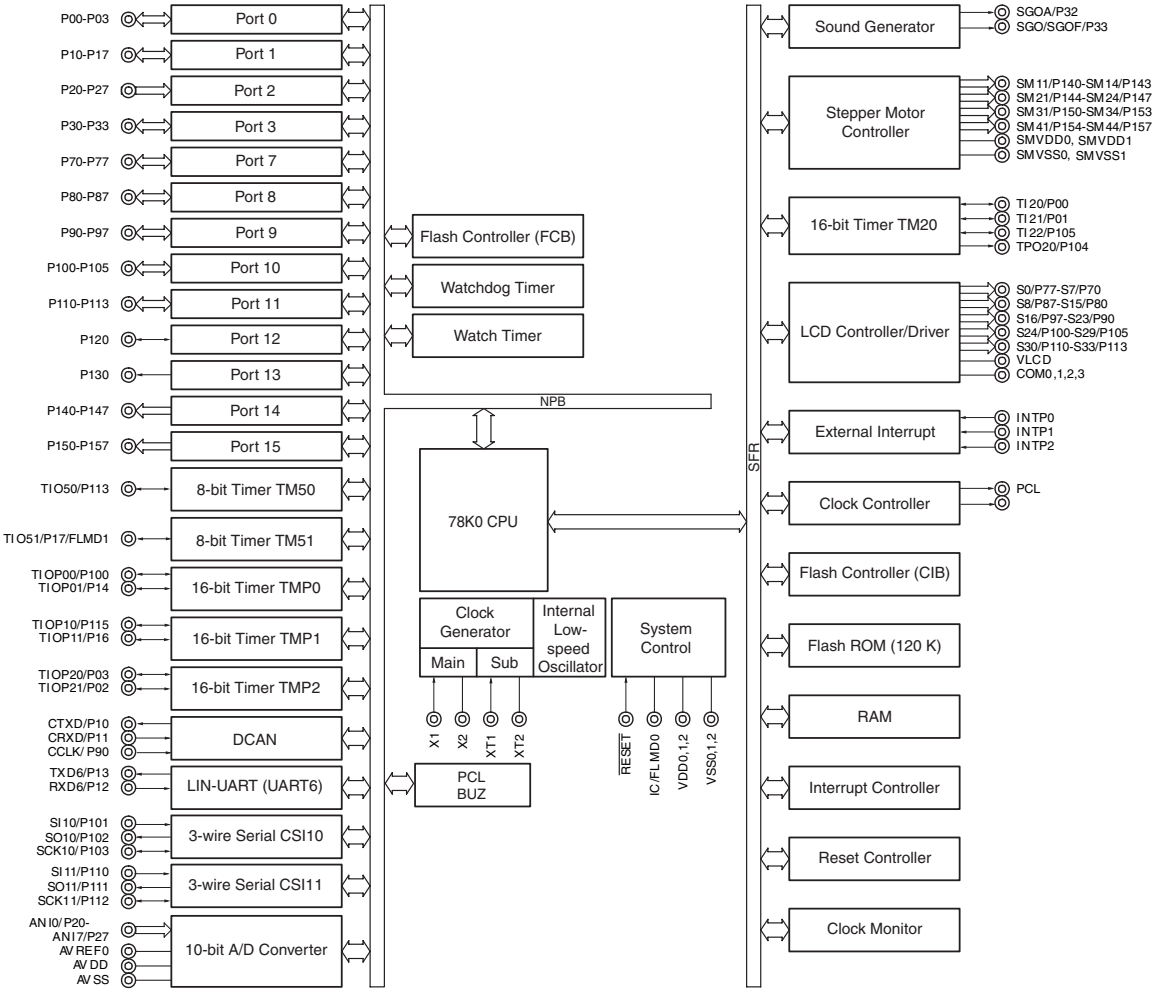
Table 1-1: The major functional differences between the subseries

Subseries Name / Function		ROM (Bytes)	Timer				A/D		LCD	Serial Interfaces	Meter C/D	I/O	V_{DD} MIN value
			8-bit	16-bit	WT	WDT	8-bit	10-bit					
Special ASSP	µPD780816A, µPD780816B	32 K to 60 K	2 ch	2 ch	1 ch	1 ch	12 ch	-	-	CSI: 2 ch UART: 1 ch DCAN: 1 ch	-	46	4.0 V
	µPD780703AY	59.5 K	3 ch				16 ch			I2C: 1 ch CSI: 2 ch UART: 1 ch DCAN: 1 ch		67	3.5 V
	µPD780822B	90 K to 120 K	2 ch	4 ch			-	8 ch	34 x 4	CSI: 2 ch UART with LIN: 1 ch DCAN: 1 ch	4 instr, 2 coils	76	4.0 V
	µPD780828A	32 K to 60 K	3 ch	1 ch			5 ch	-	28 x 4	CSI: 2 ch UART: 1 ch DCAN: 1 ch		59	
	µPD780948A	60 K	2 ch	2 ch			8 ch		40 x 4		-	79	
F-Line **Note**	78K0/FC2	32 K to 60 K	4 ch	2 ch	1 ch	1 ch	-	8 ch	-	UART with LIN/ CSI (SPI comp.): 1 ch UART with LIN: 1 ch AFCAN: 1 ch	-	37/ 40	1.8 V
	78K0/FE2	48 K to 128 K		2 ch				12 ch		CSI (SPI comp.): 1 ch UART with LIN/ CSI (SPI comp.): 1 ch UART with LIN: 1 ch AFCAN: 1 ch		55	
	78K0/FF2	60 K to 128 K		4 ch				16 ch		CSI (SPI comp.): 1 ch UART with LIN/ CSI (SPI comp.): 1 ch UART with LIN: 1 ch AFCAN: 1 ch		71	

Note: Under development (Target values)

1.7 Block Diagram

Figure 1-4: Function Block Diagram



Remark: The internal ROM and RAM capacity depends on the product.

1.8 Overview of Functions

Table 1-2: Overview of Functions

Item \ Part Number		µPD780821B	µPD780822B	µPD78F0822B
Internal memory	ROM (banked)	90 Kbytes	120 Kbytes	120 Kbytes (Flash memory)
	Internal High-speed RAM	1024 bytes		
	Internal Expansion RAM	4096 bytes		
	LCD Display RAM	34 bytes		
	DCAN RAM	288 bytes		
General registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)		
Operating frequency	X1 input clock	16 MHz (max.)		
	Subsystem clock	32.768 kHz		
	Internal low-speed oscillator clock	240 kHz (typ.)		
Instruction cycle time	X1 input clock selected	0.125 µs/0.250 µs/0.5 µs/1 µs/2 µs (at 16 MHz)		
	Subsystem clock selected	122 µs (at 32.768 kHz)		
	Internal low-speed oscillator clock selected	8.33 µs/16.67 µs/33.33 µs/66.67 µs/133.33 µs (at 240 kHz)		
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, boolean operation) • BCD adjustment, etc.		
I/O ports		Total: 76 • Input: 8 • Output: 17 • I/O: 51		
Serial Interface		• 3-wire mode: 2 channels • UART with LIN: 1 channel		
CAN		1 channel		
Timer		• 16 bit timer: 4 channels • 8 bit timer: 2 channels • Watch timer: 1 channel • Watchdog timer: 1 channel		
Timer output		5 outputs (16-bit PWM × 3, 8-bit PWM output × 2)		
Clock output		16 MHz/8 MHz/4 MHz/2 MHz/1 MHz/500 kHz/250 kHz/125 kHz (at 16 MHz)		
A/D Converter		10-bit resolution × 8 channels		
Sound Generator		1 channel		
LCD Controller/Driver		max. 34 segments × 4		
Meter Controller/Driver		4 channels		
Vectored interrupts		Maskable interrupts	Internal: 25 External: 3	
		Reset	Internal: 3	
		Software interrupts	Internal: 1	
Supply voltage		V_{DD} = 4.0 V to 5.5 V		
Package		100-pin plastic QFP (14 × 14 mm, resin thickness 1.4 mm) 100-pin plastic QFP (14 × 20 mm, resin thickness 2.7 mm)		

[MEMO]

Chapter 2 Pin Functions (μPD780822B Subseries)

2.1 List of Pin Functions

The names and functions of this product's pins are listed below. These pins can be divided into port pins and non-port pins according to their functions.

(1) Port pins

Table 2-1: Port Pins (1/3)

Pin Name	I/O	Function	Alternate Function	After Reset
P00	I/O	Port 0 4-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise	TI20	Input
P01			TI21	Input
P02			TIOP21	Input
P03			TIOP20	Input
P10	I/O	Port 1 8-bit input port 8-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise	CTXD	Input
P11			CRXD	Input
P12			RXD6	Input
P13			TXD6	Input
P14			TIOP01	Input
P15			TIOP10	Input
P16			TIOP11	Input
P17			TIO51/FLMD1	Input
P20	I	Port 2 8-bit input / output port input mode can be specified bit-wise	ANI0	Input
P21			ANI1	Input
P22			ANI2	Input
P23			ANI3	Input
P24			ANI4	Input
P25			ANI5	Input
P26			ANI6	Input
P27			ANI7	Input
P30	I/O	Port 3 4-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise	INTP1	Input
P31			INTP2	Input
P32			SGOA / PCL	Input
P33			SGO / SGOF / BUZ	Input
P70	I/O	Port 7 8-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise. This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register.	S7	Input
P71			S6	Input
P72			S5	Input
P73			S4	Input
P74			S3	Input
P75			S2	Input
P76			S1	Input
P77			S0	Input

Table 2-1: Port Pins (2/3)

Pin Name	I/O	Function	Alternate Function	After Reset
P80	I/O	Port 8 8-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise. This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register.	S15	Input
P81			S14	Input
P82			S13	Input
P83			S12	Input
P84			S11	Input
P85			S10	Input
P86			S9	Input
P87			S8	Input
P90	I/O	Port 9 8-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise. This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register.	S23 / CCLK	Input
P91			S22	Input
P92			S21	Input
P93			S20	Input
P94			S19	Input
P95			S18	Input
P96			S17	Input
P97			S16	Input
P100	I/O	Port 10 6-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise. This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register.	S24 / TIOP00	Input
P101			S25 / SI10	Input
P102			S26 / SO10	Input
P103			S27 / SCK10	Input
P104			S28 / TPO20	Input
P105			S29 / TI22	Input
P110	I/O	Port 11 4-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise. This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register.	S30 / SI11	Input
P111			S31 / SO11	Input
P112			S32 / SCK11	Input
P113			S33 / TIO50	Input
P120	I/O	Port 12 4-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise.	INTP0	Input
P130	O	Port 13 1-bit output port	-	Output

Table 2-1: Port Pins (3/3)

Pin Name	I/O	Function	Alternate Function	After Reset
P140	O	Port 14 8-bit output port	SM11	Hi-z
P141			SM12	Hi-z
P142			SM13	Hi-z
P143			SM14	Hi-z
P144			SM21	Hi-z
P145			SM22	Hi-z
P146			SM23	Hi-z
P147			SM24	Hi-z
P150	0	Port 15 8-bit output port	SM31	Hi-z
P151			SM32	Hi-z
P152			SM33	Hi-z
P153			SM34	Hi-z
P154			SM41	Hi-z
P155			SM42	Hi-z
P156			SM43	Hi-z
P157			SM44	Hi-z

2.2 Non-port Pins

Table 2-2: Non-port pins (1/2)

Pin Name	I/O	Function	After Reset	Alternate Function
ANI0 to ANI7	Input	A/D converter input pin	Input	P20 to P27
AV_{DD}	-	Power supply pin for A/D converter	-	-
AV_{REF0}	-	Reference-Voltage supply pin for A/D converter	-	-
AV_{SS}	-	Ground potential for A/D converter. Connect to V_{SS}	-	-
BUZ	Output	Buzzer output	Input	P33/SGO/SGOF
CCLK	Input	CAN external clock input	Input	P90
COM0 - COM3	Output	LCD common signal output	Output	-
CRXD	Input	CAN serial data input pin	Input	P11
CTXD	Output	CAN serial data output pin	Input	P10
INTP0	Input	External interrupts with specitable valid edges (rising edge, falling edge, both rising and falling edges)	Input	P120
INTP1				P30
INTP2				P31
PCL	Output	Processor Clock output pin	Input	P32/SGOA
$\overline{RESET}$	Input	External System reset input	-	-
RXD6	Input	Asynchronous serial interface data input	Input	P12
S0 - S7	Output	Segment signal output of LCD controller/driver	Input	P77 - P70
S8 - S15				P87 - P80
S16				P97
S17				P96
S18				P95
S19				P94
S20				P93
S21				P92
S22				P91
S23				P90/CCLK
S24				P100/TIOP00
S25				P101/SI10
S26				P102/SO10
S27				P103/SCK10
S28				P104/TPO20
S29				P105/TI22
S30				P110/SI11
S31				P111/SO11
S32				P112/SCK11
S33				P113/TIO50
$\overline{SCK10}$	I/O	Serial interface channel 0 clock input/output pin	Input	P103/S27
$\overline{SCK11}$	I/O	Serial interface channel 1 clock input/output pin	Input	P112/S32
SGO	Output	Sound Generator output	Input	P33/SGOF/BUZ
SGOA	Output	Sound Generator Amplitude output	Input	P32/PCL
SGOF	Output	Sound Generator Frequency output	Input	P33/SGO/BUZ

Table 2-2: Non-port pins (2/2)

Pin Name	I/O	Function	After Reset	Alternate Function
SI10	Input	Serial interface channel 0 data input pin	Input	P101/S25
SI11	Input	Serial interface channel 1 data input pin	Input	P110/S30
SM11 - SM14	Output	Meter controller/driver output	Hi-z	P140 - P143
SM21 - SM24				P144 - P147
SM31 - SM34				P150 - P153
SM41 - SM44				P154 - P157
SO10	Output	Serial interface channel 0 data output pin	Input	P102
SO11	Output	Serial interface channel 1 data output pin	Input	P111
TI20	Input	Capture trigger input pin	Input	P00
TI21	Input	Capture trigger input pin	Input	P01
TI22	Input	Capture trigger input pin	Input	P105/S29
TIO50	I/O	External count clock input /timer output pin	Input	P113/S33
TIO51	I/O	External count clock input /timer output pin	Input	P17/FLMD1
TIOP00	I/O	Capture input/Compare output pin	Input	P100/S24
TIOP01	I/O	Capture input/Compare output pin	Input	P14
TIOP10	I/O	Capture input/Compare output pin	Input	P15
TIOP11	I/O	Capture input/Compare output pin	Input	P16
TIOP20	I/O	Capture input/Compare output pin	Input	P03
TIOP21	I/O	Capture input/Compare output pin	Input	P02
TPO20	Output	Timer output pin	Input	P104/S28
TXD6	Output	Asynchronous serial interface data output pin	Input	P13
V_{DD0}	-	Power supply pin for CPU	-	-
V_{SS0}	-	Ground potential pin for CPU	-	-
V_{DD1}	-	Power supply pin for port buffers	-	-
V_{SS1}	-	Ground potential pin for port buffers	-	-
V_{DD2}	-	Power supply pin for port buffers	-	-
V_{SS2}	-	Ground potential pin for port buffers	-	-
SMV_{DD0}	-	Power supply pin for Meter controller/driver	-	-
SMV_{SS0}	-	Ground potential for Meter controller/driver	-	-
SMV_{DD1}	-	Power supply pin for Meter controller/driver	-	-
SMV_{SS1}	-	Ground potential for Meter controller/driver	-	-
V_{LCD}	-	Power supply pin for LCD controller/driver	-	-
X1, X2	-	Resonator connection for main system clock	-	-
XT1, XT2	-	Resonator connection for sub clock	-	-
IC	-	Internal connection. Connect to V_{SS} (only Mask ROM version)	-	-
FLMD0	-	Flash memory programming mode lead-in (only Flash ROM version)	-	
FLMD1	-		-	P17/TIO51

2.3 Description of Pin Functions

2.3.1 P00 to P03 (Port 0)

Port 0 is a 4-bit input/output port. Besides serving as an input/output port, P00 and P01 operate as input/output signals for the 16-bit timer TM2, and P02 and P03 operate as input/output signals for the 16-bit timer TMP2.

(1) Port mode

In this mode the port functions as a 4-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 0 (PM0). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 0 (PU0).

(2) Control mode

In this mode the port functions as input signals for the 16-bit timer TM2, and as input/output signals for the 16-bit timer TMP2.

(a) TI20, TI21

16-bit Timer TM2 input pin.

(b) TIOP20, TIOP21

16-bit Timer TMP2 input/output pins.

2.3.2 P10 to P17 (Port 1)

Port 1 is an 8-bit input/output port. Besides servicing as an input/output port, P10 and P11 operate as data input/output for the CAN controller, P12 and P13 operate as data input/output for the asynchronous serial interface UART6, P14 to P16 operate as input/output signals for the 16-bit timers TMP0 and TMP1, and P17 operates as input/output signal for the 8-bit timer TM51.

(1) Port mode

In this mode the port functions as an 8-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 1 (PM1). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 1 (PU1).

(2) Control mode

(a) CTXD

CAN controller serial data output pin.

(b) CRXD

CAN controller serial data input pin.

(c) RXD6

Asynchronous serial interface UART6 serial data input pin.

(d) TXD6

Asynchronous serial interface UART6 serial data output pin.

(e) TIOP01

16-bit Timer TMP0 input/output pin.

(f) TIOP10, TIOP11

16-bit Timer TMP1 input/output pins.

(g) TIO51

8-bit Timer TM51 input/output pin.

Caution: When this port is used as a serial interface, the I/O function and output latches must be set according to the function the user requires.

2.3.3 P20 to P27 (Port 2)

Port 2 is an 8-bit input port. Besides serving as an input port, P20 to P27 operate as analog input pins for the A/D converter.

(1) Port mode

In this mode the port functions as an input only port.

(2) Control mode

In this mode the port functions as analog input signal for the A/D converter.

(a) ANI0 to ANI7

A/D converter analog input pins.

2.3.4 P30 to P33 (Port 3)

Port 3 is a 4-bit input/output port. Besides serving as an input/output port, P30 and P31 operate as maskable external interrupt request pins, P32 operates as sound generator amplitude output or as processor clock output, and P33 operates as sound generator sum output, as sound generator frequency output, or as buzzer output.

(1) Port mode

In this mode the port functions as a 4-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 3 (PM3). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 3 (PU3).

(2) Control mode

In this mode the port functions as a maskable external interrupt request signal, sound generator amplitude output signal, processor clock output signal, sound generator sum output signal, sound generator frequency output signal, or as buzzer output signal.

(a) INTP0 to INTP2

External interrupt request input pins, which can be specified to valid edges (rising, falling, or both rising and falling edges).

(b) SGO, SGOA, SGOF

Pins for separated or composed output of the sound generator.

(c) PCL

Processor clock output pin.

(d) BUZ

Buzzer output pin.

2.3.5 P70 to P77 (Port 7)

Port 7 is an 8-bit input/output port. Besides serving as an input/output port, P70 to P77 operate as segment outputs of the LCD controller/driver.

(1) Port mode

In this mode the port functions as an 8-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 7 (PM7). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 7 (PU7).

(2) Control mode

In this mode the port functions as segment signal output of the LCD controller/driver. This mode can be selected by using the port function register 7 (PF7).

(a) S0 to S7

LCD controller/driver segment signal output pins.

2.3.6 P80 to P87 (Port 8)

Port 8 is an 8-bit input/output port. Besides serving as an input/output port, P80 to P87 operate as segment outputs of the LCD controller/driver.

(1) Port mode

In this mode the port functions as an 8-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 8 (PM8). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 8 (PU8).

(2) Control mode

In this mode the port functions as segment signal output of the LCD controller/driver. This mode can be selected by using the port function register 8 (PF8).

(a) S8 to S15

LCD controller/driver segment signal output pins.

2.3.7 P90 to P97 (Port 9)

Port 9 is an 8-bit input/output port. Besides serving as an input/output port, P80 to P87 operate as segment outputs of the LCD controller/driver.Additionally P90 can support the external input clock of the CAN controller.

(1) Port mode

In this mode the port functions as an 8-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 9 (PM9). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 9 (PU9).

(2) Control mode

In this mode the port functions as segment signal output of the LCD controller/driver and as external input clock of the CAN controller. These modes can be selected by using the port function register 9 (PF9).

(a) S16 to S23

LCD controller/driver segment signal output pins.

(b) CCLK

CAN controller external clock input pin.

2.3.8 P100 to P105 (Port 10)

Port 10 is a 6-bit input/output port. Besides serving as an input/output port, P100 operates as segment signal output from the LCD controller/driver, or as input/output signal for the 16-bit timer TMP0, P101 to P103 operate as clock and data input/output signals of the clock synchronous serial interface CSI10, and P104 to P105 operate as input/output signals for the 16-bit timer TM2.

(1) Port mode

In this mode the port functions as a 6-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 10 (PM10). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 10 (PU10).

(2) Control mode

In this mode the port functions as segment signal output of the LCD controller/driver, as input/output signal for the 16-bit timer TM0, as clock and data input/output signals of the clock synchronous serial interface CSI10, or as input/output signals for the 16-bit timer TM2. These modes can be selected by using the port function register 10 (PF10).

(a) S24 to S29

LCD controller/driver segment signal output pins.

(b) TIOP00

16-bit Timer TMP0 input/output pin.

(c) SI10, SO10

Clock synchronous serial interface CSI10 serial data input/output pins.

(d) $\overline{\text{SCK10}}$

Clock synchronous serial interface CSI10 serial clock input/output pin.

(e) TPO20

16-bit timer TM2 divided clock output pin.

(f) TI22

16-bit timer TM2 external capture trigger input pin.

Caution: When this port is used as a serial interface, the I/O function and output latches must be set according to the function the user requires.

2.3.9 P110 to P113 (Port 11)

Port 11 is a 4-bit input/output port. Besides serving as an input/output port, P110 to P112 operate as clock and data input/output signals of the clock synchronous serial interface CSI11, P113 operates as input/output signal for the 8-bit timer TM50. Additionally P110 to P113 operate as segment signal output of the LCD controller/driver.

(1) Port mode

In this mode the port functions as a 6-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 11 (PM11). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 11 (PU11).

(2) Control mode

In this mode the port functions as segment signal output of the LCD controller/driver, as input/output signal for the 8-bit timer TM50, as clock and data input/output signals of the clock synchronous serial interface CSI11. These modes can be selected by using the port function register 11 (PF11).

(a) S30 to S33

LCD controller/driver segment signal output pins.

(b) SI11, SO11

Clock synchronous serial interface CSI11 serial data input/output pins.

(c) $\overline{\text{SCK11}}$

Clock synchronous serial interface CSI11 serial clock input/output pin.

(d) TIO50

8-bit timer TM50 input/output pin.

2.3.10 P120 (Port 12)

Port 12 is an 1-bit input/output port. Besides serving as an input/output port, P120 operates as maskable external interrupt request signal.

(1) Port mode

In this mode the port functions as an 1-bit input/output port and can be specified to input or output bit-wise and byte-wise by using the port mode register 12 (PM12). When this port is used as input port, pull-up resistors can be connected by using the pull-up resistor option register 12 (PU12).

(2) Control mode

In this mode the port functions as maskable external interrupt request signal.

(a) INTP0

External interrupt request input pin, which can be specified to valid edges (rising, falling, or both rising and falling edges).

2.3.11 P130 (Port 13)

Port 13 is an 1-bit output port.

2.3.12 P140 to P147 (Port 14)

Port 14 is an 8-bit output port. Besides serving as an output port, P140 to P147 operate as PWM output port of the meter controller/driver.

(1) Port mode

In this mode the port functions as an 8-bit output port.

(2) Control mode

In this mode the port functions as PWM output port of the meter controller/driver.

(a) SM11 to SM14

Meter controller/driver 1 PWM output pins.

(b) SM21 to SM24

Meter controller/driver 2 PWM output pins.

2.3.13 P150 to P157 (Port 15)

Port 15 is an 8-bit output port. Besides serving as an output port, P150 to P157 operate as PWM output port of the meter controller/driver.

(1) Port mode

In this mode the port functions as an 8-bit output port.

(2) Control mode

In this mode the port functions as PWM output port of the meter controller/driver.

(a) SM31 to SM34

Meter controller/driver 3 PWM output pins.

(b) SM41 to SM44

Meter controller/driver 4 PWM output pins.

2.3.14 COM0 to COM3

There are the LCD controller/driver common signal output pins.

2.3.15 V_{LCD}

This pin is the supply voltage pin to drive the LCD.

2.3.16 $\overline{RESET}$

This is the low-level active reset input pin.

2.3.17 X1 and X2

Resonator connection pin for system clock generation.

2.3.18 XT1, XT2

Resonator connection pin for subsystem clock generation.

2.3.19 V_{DD0} to V_{DD2}

V_{DD0} is the positive power supply pin for the internal CPU.
V_{DD1} and V_{DD2} are the positive power supply pins for the peripheral interface.

2.3.20 V_{SS0} to V_{SS2}

V_{SS0} is the ground pin for the internal CPU.
V_{SS1} and V_{SS2} are the ground pins for the peripheral interface.

2.3.21 AV_{DD}

This is the analog positive power supply pin for the A/D converter.

2.3.22 AV_{SS}

This is the ground pin for the A/D converter.

2.3.23 AV_{REF}

This is the reference voltage supply pin for the A/D converter.

2.3.24 SMV_{DD0}, SMV_{DD1}

These pins are the power supply for the meter controller/driver.

2.3.25 SMV_{SS0}, SMV_{SS1}

These pins are the ground pins for the meter controller/driver.

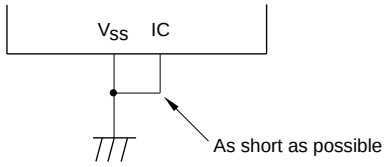
2.3.26 FLMD0, FLMD1 (Flash ROM version only)

These pins are the flash memory programming mode lead-in pins. In normal operating mode connect FLMD0 pin directly to V_{SS} with the shortest possible wiring. In case that the wiring between these pins is too long or an external noise is input to the IC pin, the user program my not run properly.

2.3.27 IC (Mask ROM version only)

The IC (internally connected) pin is provided to set the test mode to check the device at delivery. In normal operating mode connect this pin directly to V_{SS} with the shortest possible wiring.In case that the wiring between these pins is too long or an external noise is input to the IC pin, the user program my not run properly.

Figure 2-1: Pin Configuration



Caution: Connect IC pins to V_{SS} pins directly.

2.4 Pin I/O Circuits and Recommended Connection of Unused Pins

The input/output circuit type of each pin and recommended connection of unused pins are shown in the following table.
For the input/output circuit configuration of each type, please refer to the **Figure 2-2, "Terminal I/O types (1/2)," on page 60**.

Table 2-3: Types of Input/Output Circuits (1/2)

Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
P00 to P03	8-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P10	5-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P11	8-A		
P12			
P13	5-A		
P14 to P17	8-A		
P20 to P27	9-C	I	Connect to V_{DD} or V_{SS}
P30 to P33	8-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P70 to P77	17-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P80 to P87	17-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P90 to P96	17-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P97	17-B		
P100, P101	17-B	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P102	17-A		
P103	17-B		
P104	17-A		
P105	17-B		
P110	17-B	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P111	17-A		
P112, P113	17-B		
P120	8-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P130	3-C	O	Leave open
P140 to P147	4	O	Leave open
P150 to P157	4	O	Leave open
COM0 to COM3	18	O	Leave open
V_{LCD}	-	-	Connect to V_{SS}
SMV_{DD0}, SMV_{DD1}	-	-	Connect to V_{DD}
SMV_{SS0}, SMV_{SS1}	-	-	Connect to V_{SS}
AV_{DD}, AV_{REF}	-	-	Connect to V_{DD}

Table 2-3: Types of Input/Output Circuits (2/2)

Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
AV_{SS}	-	-	Connect to V_{SS}
IC	-	-	Connect to V_{SS}
FLMD0			

Figure 2-2: Terminal I/O types (1/2)

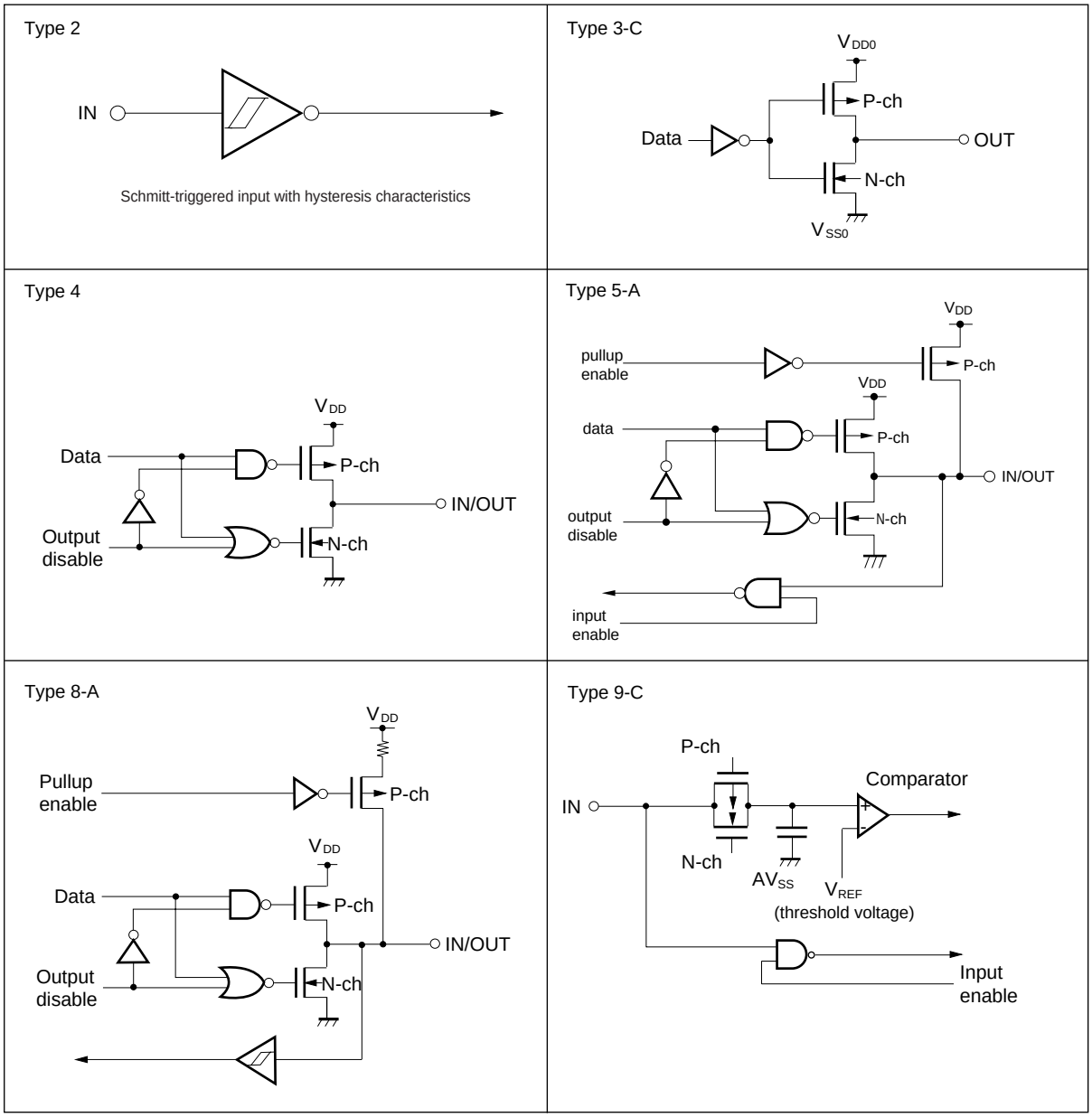
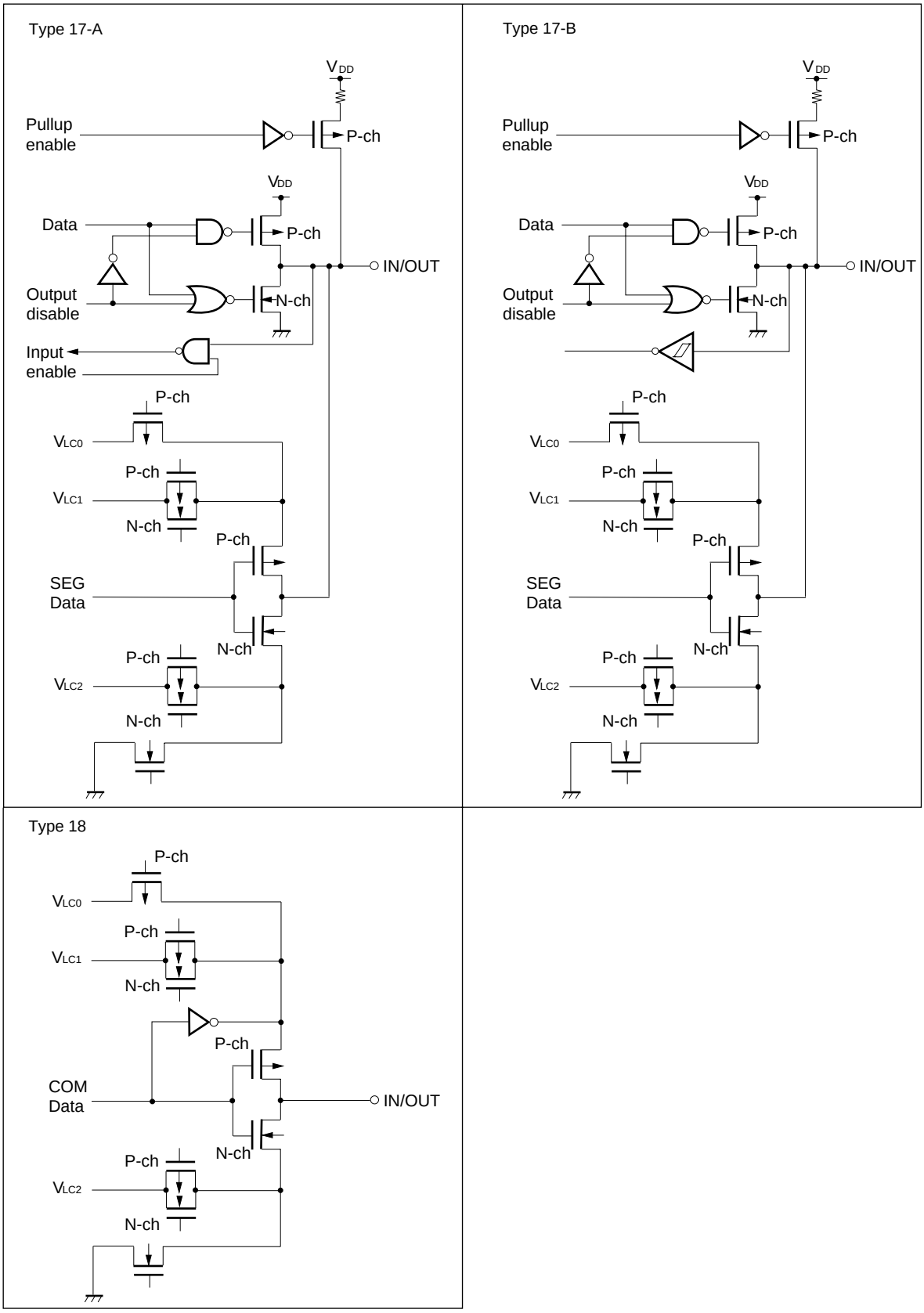


Figure 2-2: Terminal I/O types (2/2)



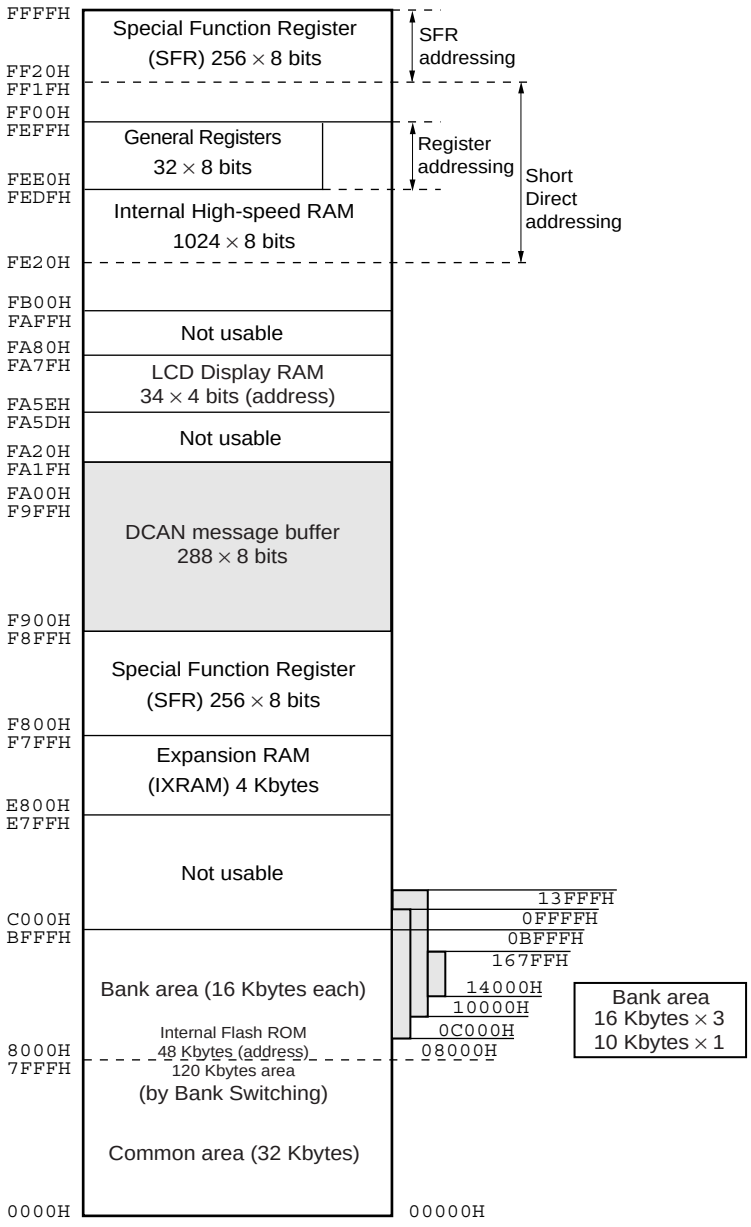
[MEMO]

Chapter 3 CPU Architecture (µPD780822B Subseries)

3.1 Memory Space

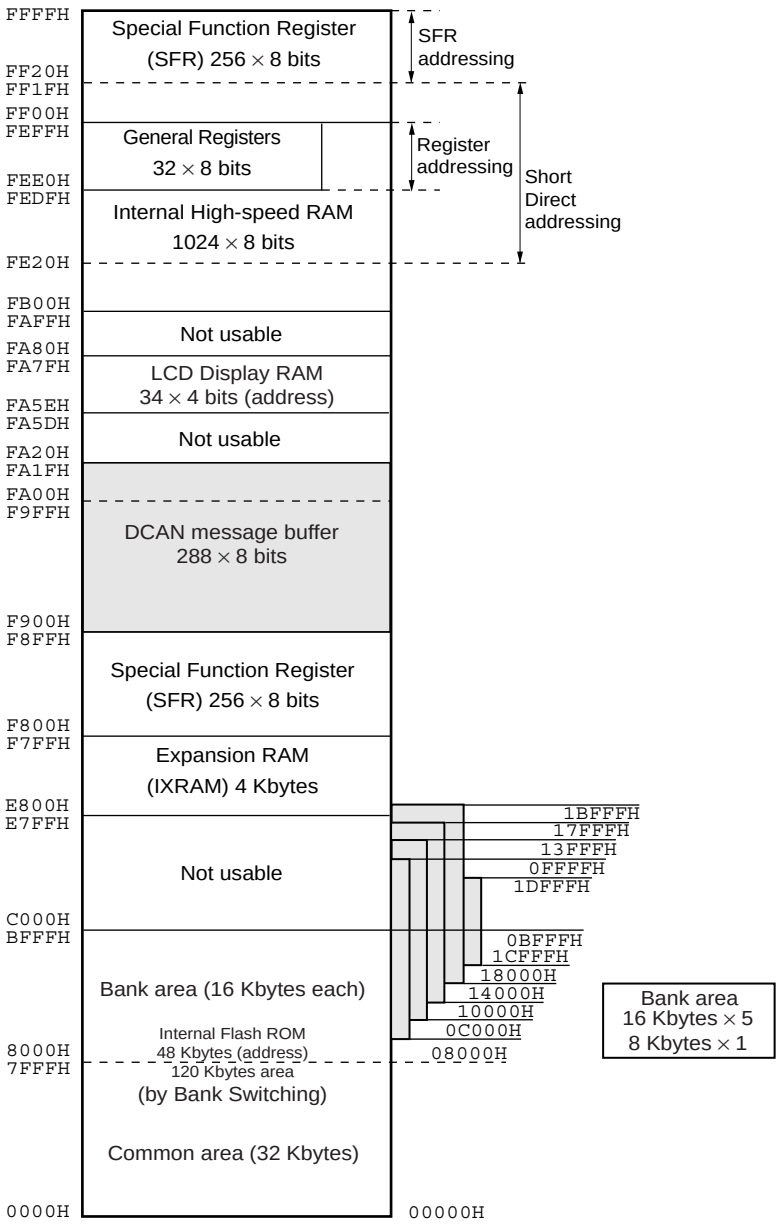
The memory map of the µPD780821 is shown in Figure 3-1.

Figure 3-1: Memory Map of the µPD780821



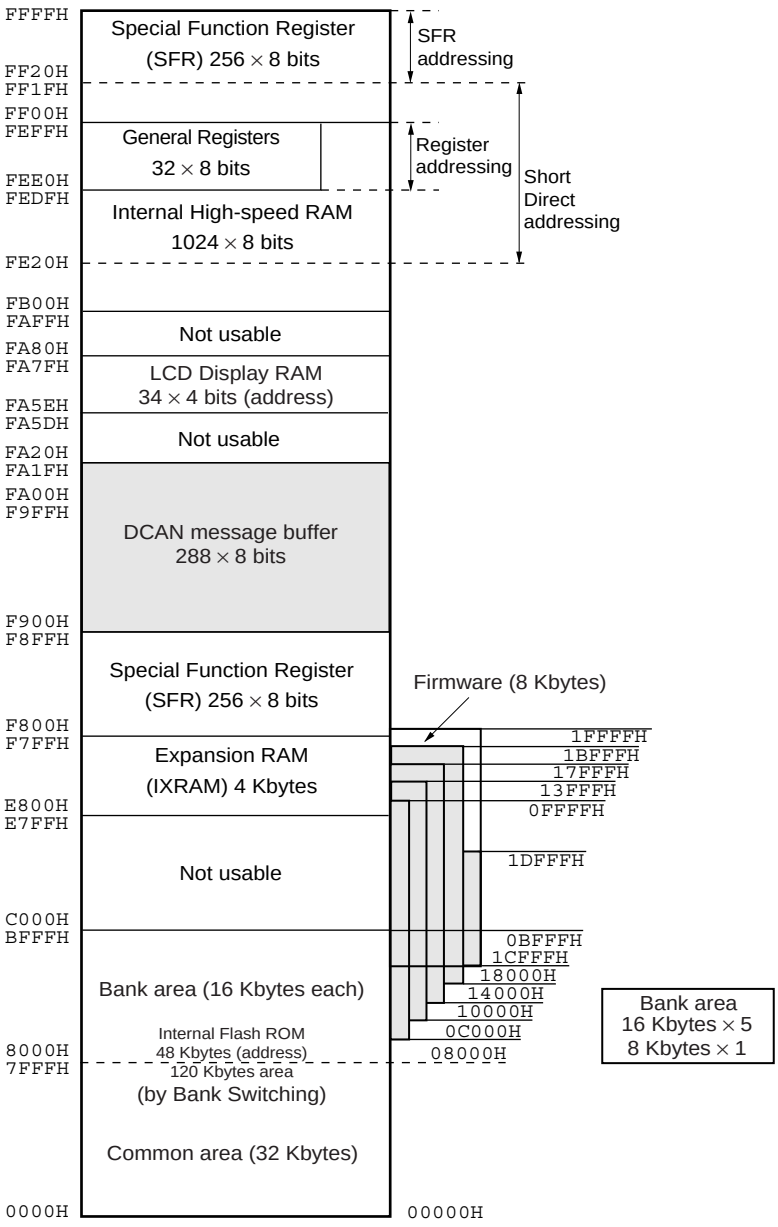
The memory map of the µPD780822B is shown in Figure 3-2.

Figure 3-2: Memory Map of the µPD780822B



The memory map of the μPD78F0822 is shown in Figure 3-3.

Figure 3-3: Memory Map of the μPD78F0822



3.1.1 Internal program memory space

The internal program memory space stores programs and table data. This is generally accessed by the program counter (PC).
The µPD780822B Subseries have sizes of internal ROMs or Flash EPROM as shown below.

Table 3-1: Internal ROM Capacities

Part Number	Internal ROM	
	Type	Capacity
µPD780821	Mask ROM	90 KB (92.160 × 8-bits)
µPD780822B	Mask ROM	120 KB (122.880 × 8-bits)
µPD78F0822	Flash EEPROM	120 KB (122.880 × 8-bits)

The internal program memory has a total capacity of maximal 120 KB. As the CPU address bus is 16 bits wide, the program memory is banked and divided into 2 areas: the first area is a common 32 KB area, and the banked area, from 8000H to BFFFH (16 KB) can be switched between 4 different banks. Banks 0 to 2 are 16 KB wide and Bank 3 is 10 KB wide. On the µPD780822B and µPD78F0822B the banked area can be switched between 6 different banks. Banks 0 to 4 are 16 KB wide and Bank 5 is 8 KB wide.

(1) Common area

The first 32 KB of the program memory are divided into three areas: vector table area, CALLT instruction table area, and CALLF instruction table area. These areas are described on the next page.

(a) Vector table area

The 64-byte area 0000H to 003FH is reserved as a vector table area. The program start addresses of the reset request and of each interrupt request handler are stored in the vector table area. Of the 16-bit address, low-order 8 bits are stored at even addresses and high-order 8 bits are stored at odd addresses.

Table 3-2: Vectored Interrupts

Vector Table Address	Interrupt Request Name
0000H	RESET
	CLM
	WDT
0004H	INTWDT
0006H	INTP0
0008H	INTP1
000AH	INTP2
000EH	INTOVF
0010H	INTTM20
0012H	INTTM21
0014H	INTTM22
0016H	INTAD
0018H	INTSRE6
001AH	INTSR6
001CH	INTST6
001EH	INTCSI10
0020H	INTCSI11
0022H	INTTM50
0024H	INTTM51
0026H	INTWTI
0028H	INTWT
002AH	INTCE
002CH	INTCR
002EH	INTCT0
0030H	INTCT1
0032H	INTTP0CC0
0034H	INTTP0CC1
0036H	INTTP1CC0
0038H	INTTP1CC1
003AH	INTTP2CC0
003CH	INTTP2CC1
003EH	BRK

(b) CALLT instruction table area

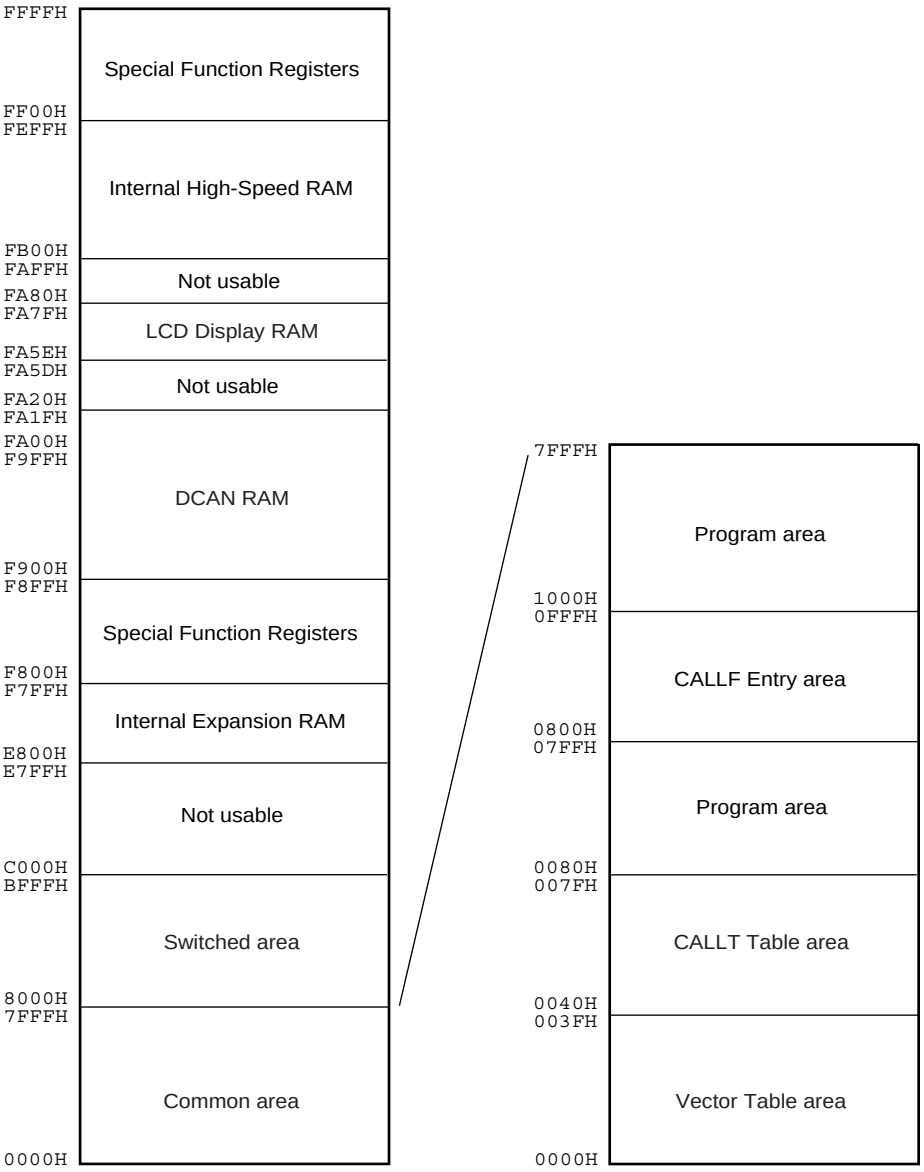
The 64-byte area 0040H to 007FH can store the subroutine entry address of an 1-byte call instruction (CALLT).

(c) CALLF instruction entry area

The area 0800H to 0FFFH can perform a direct subroutine call with a 2-byte call instruction (CALLF).

Except for the addresses 0 and 1, which hold the start address of the user application program, the areas totally or partially not used for interrupt vector addresses, CALLT table or CALLF entries can be freely used for program storage and table data.

Figure 3-4: Program memory common area mapping



(2) Banked area

The physical addresses 8000H to BFFFH are used to access the 58 K (µPD780821) or the 88 K (µPD780822B, µPD78F0822) remaining bytes of the program memory. These addresses are combined with the value of the B0 register to build a 17 bits physical address for the flash memory.

B0 is set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets B0 to 00H.

Figure 3-5: BANK: bank switching register format (µPD780821)

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
B0	0	0	0	0	0	0	BANK1	BANK0	FFF3H	R/W	00H

BANK1	BANK0	Selected program memory
0	0	Common area + BANK0 (16 K)
0	1	Common area + BANK1 (16 K)
1	0	Common area + BANK2 (16 K)
1	1	Common area + BANK3 (10 K)
Other		Setting prohibited

Figure 3-6: BANK: bank switching register format (µPD780822B, µPD78F0822)

Symbol	7	6	5	4	3	2	1	0	Address	R/W	After reset
B0	0	0	0	0	0	BANK2	BANK1	BANK0	FFF3H	R/W	00H

BANK2	BANK1	BANK0	Selected program memory
0	0	0	Common area + BANK0 (16 K)
0	0	1	Common area + BANK1 (16 K)
0	1	0	Common area + BANK2 (16 K)
0	1	1	Common area + BANK3 (16 K)
1	0	0	Common area + BANK4 (16 K)
1	0	1	Common area + BANK5 (8 K)
Other			Setting prohibited

Cautions: 1. The bank configuration of the µPD780822B Subseries uses banks of different size, because banks 0 to bank 4 (µPD780822B and µPD78F0822) / bank 0 to bank 2 (µPD780821) have a size of 16 KB and bank 5 (µPD780822B and µPD78F0822) has a size of 8 KB / bank 3 (µPD780821) has a size of 10 KB. Software development tools may not fully support banks of different sizes. Therefore, the compiler and linker may not produce error messages when the software uses more than 8 KB for bank 5 (µPD780822B and µPD78F0822) / 10 KB for bank 3 (µPD780821). But those errors will be detected later in the development flow by the debugger.

2. In case, that software packages from third parties (e.g. OS, Bootloaders, CAN Drivers, etc.) shall be used, please note, that it is mandatory to locate these software parts in the common memory area. These software parts may only be located in the banked memory area, if this is explicitly specified by the software vendor.

3.1.2 Internal data memory space

The µPD780822B Subseries units incorporate the following RAMs.

(1) Internal high-speed RAM

This is a 1024 × 8-bit configuration, allocated in the area FB00H to FEFFH. 4 banks of general registers are allocated in the 32-byte area FEE0H to FEFFH, each bank consisting of eight 8-bit registers.
The internal high-speed RAM has to be used as a stack memory.

(2) LCD-Display RAM

The LCD-Display RAM is allocated to the 34 × 4 bits area from FA5EH to FA7FH. LCD-Display RAM can also be used as normal RAM.

(3) Internal expansion RAM

Internal expansion RAM is allocated to the 4-Kbyte area from E800H to F7FFH.

(4) DCAN message RAM

The area located from F900H to FA1FH is dedicated to the DCAN interface, and holds the message buffers.

3.1.3 Special function register (SFR) area

The on-chip peripheral hardware special function registers (SFRs) are allocated in the area FF00H to FFFFH, and F800H to F8FFH. (Refer to Table 3-3, “Special Function Register List,” on page 78).

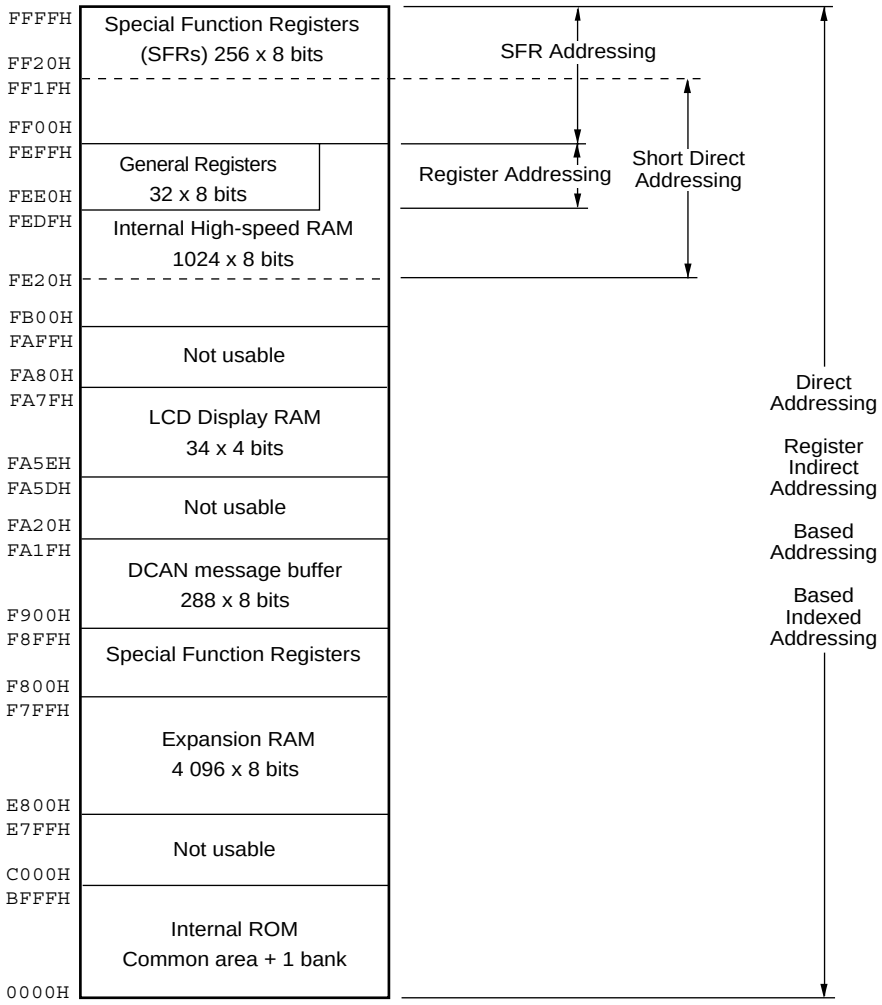
Caution: Do not access addresses where the SFR is not assigned.

3.1.4 Data memory addressing

The µPD780822B Subseries is provided with a variety of addressing modes which take account of memory manipulability, etc. Special addressing methods are possible to meet the functions of the special function registers (SFRs) and general registers. The data memory space is the entire 64-Kbyte space (0000H to FFFFH).
Figure 3-7 shows the data memory addressing modes.

For details of addressing, refer to **3.4 ”Operand Addressing” on page 87**.

Figure 3-7: Data Memory Addressing of the µPD780822B Subseries



3.2 Processor Registers

The µPD780822B Subseries units incorporate the following processor registers.

3.2.1 Control registers

The control registers control the program sequence, statuses, and stack memory. The control registers consist of a program counter, a program status word and a stack pointer.

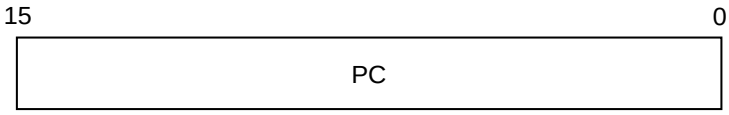
(1) Program counter (PC)

The program counter is a 16-bit register which holds the address of the next instruction to be executed.

In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set.

$\overline{\text{RESET}}$ input sets the reset vector table values at addresses 0000H and 0001H to the program counter.

Figure 3-8: Program Counter Configuration



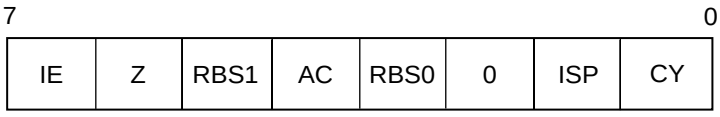
(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags to be set/reset by instruction execution.

Program status word contents are automatically saved upon interrupt request generation or PUSH PSW instruction execution and are automatically restored upon execution of the RETB, RETI and POP PSW instructions.

$\overline{\text{RESET}}$ input sets the PSW to 02H.

Figure 3-9: Program Status Word Configuration



(a) Interrupt enable flag (IE)

This flag controls the acknowledgement of an interrupt request by the CPU.
When IE=0, the CPU state is set to interrupt disabled (DI). All interrupts are disabled.
When IE=1, the CPU state is set to interrupt enabled (EI) and interrupt request acknowledgement is controlled with the in-service priority flag (ISP), an interrupt mask flag for each interrupt source, and a priority specification flag.
The IE is reset to (0) upon DI instruction execution or interrupt request acknowledgement and is set to (1) upon EI instruction execution.

(b) Zero flag (Z)

When the result of an arithmetic or logical operation is zero, this flag is set (1). It is reset (0) in all other cases.

(c) Register bank select flags (RBS0 and RBS1)

These are 2-bit flags to select one of the four register banks.
In these flags, the 2-bit information which indicates the register bank selected by SEL RBn instruction execution is stored.

(d) Auxiliary carry flag (AC)

If the operation result has a carry from bit 3 or a borrow at bit 3, this flag is set (1). It is reset (0) in all other cases. This flag is used in decimal adjustment operations.

(e) In-service priority flag (ISP)

This flag controls the priority of maskable vectored interrupts that can be acknowledged.
When this flag is 1, an interrupt request can be acknowledged regardless of its priority; actually, acknowledging or not interrupts is controlled by the interrupt enable flag (IE).
When this flag is 0, acknowledgment of the vectored interrupt with the lower priority specified by the priority specification flag registers (PR0L, PR0H, PR1L and PR1H) is disabled.

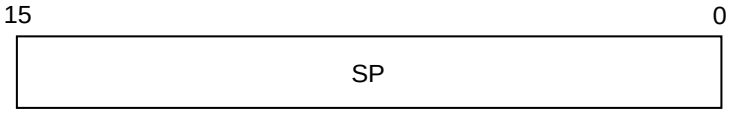
(f) Carry flag (CY)

This flag stores overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit manipulation instruction execution.

(3) Stack pointer (SP)

This is a 16-bit register that holds the first address of the memory stack area. Only the internal high-speed RAM area can be set as the stack area.

Figure 3-10: Stack Pointer Configuration



The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.
Each stack operation saves/restores data as shown in Figures 3-11 and 3-12.

Caution: Since $\overline{\text{RESET}}$ input makes SP contents indeterminate, be sure to initialize the SP before instruction execution.

Figure 3-11: Data to be Saved to Stack Memory

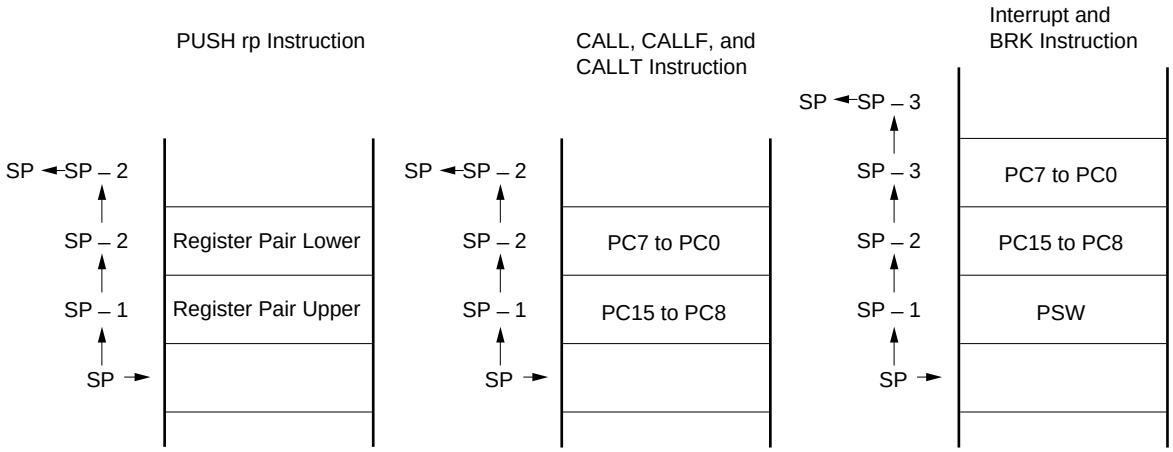
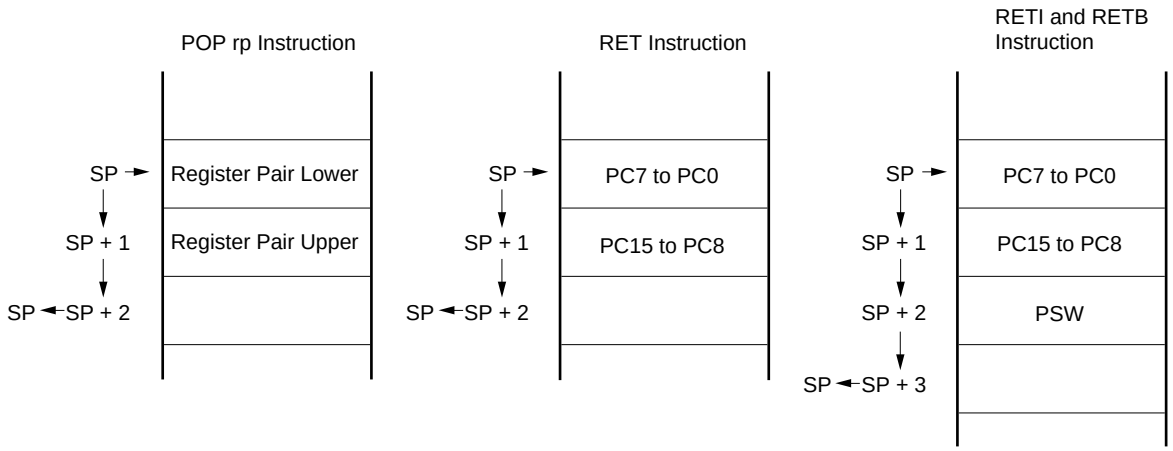


Figure 3-12: Data to be Restored from Stack Memory

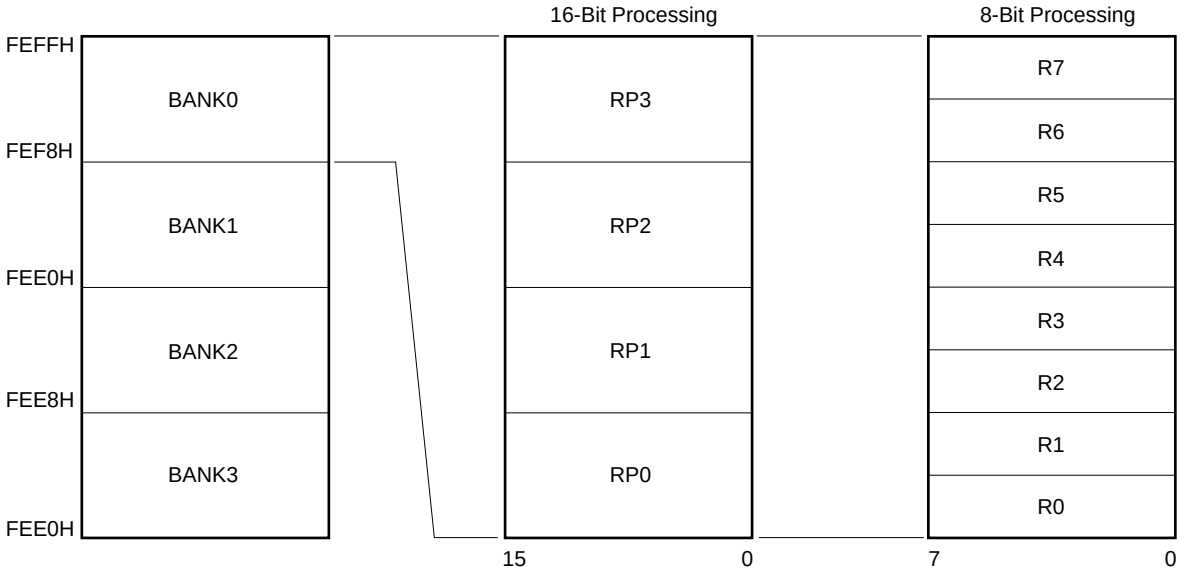


3.2.2 General registers

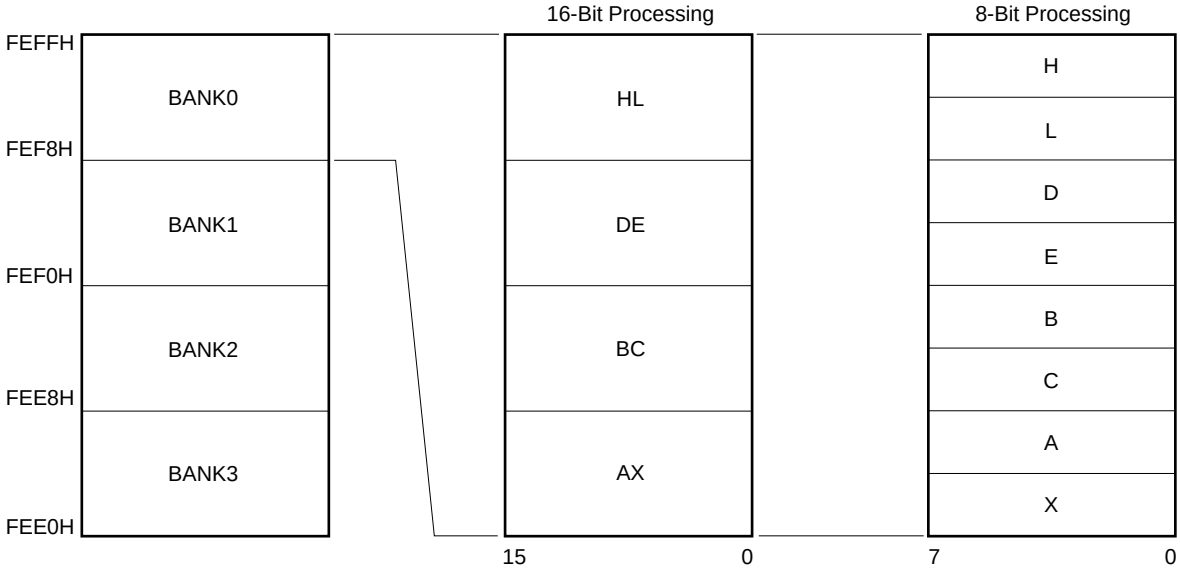
General-purpose registers are mapped to specific addresses (FEE0H to FEFFH) of the data memory. Four banks of general-purpose registers, consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H) are available.
Each register can also be used as an 8-bit register. Moreover, two 8-bit registers can be used in pairs as a 16-bit register (AX, BC, DE, and HL).
They can be described not only in function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) but also in absolute names (R0 to R7 and RP0 to RP3).
The register bank used for instruction execution is set by the CPU control instruction (SEL RBn). Because four register banks are provided, an efficient program can be developed by switching between a register bank for normal processing and a register bank for interrupt processing.

Figure 3-13: General Register Configuration

(a) Absolute Name



(b) Function Name



3.2.3 Special function register (SFR)

Unlike the general purpose registers, special function registers have their own functions and are allocated in the FF00H to FFFFH area and in the F800H to F8FFH area.
The special function registers can also be manipulated in a similar manner as the general registers, by using operation, transfer, or bit-manipulation instructions. The bit units in which one SFR is to be manipulated (1, 8, and/or 16) differ from that of another register.
The bit unit for SFR manipulation is specified as follows.

- 1-bit manipulation
  A symbol reserved by the assembler is described as the operand (sfr.bit) of an 1-bit manipulation instruction. An address can also be specified.
- 8-bit manipulation
  A symbol reserved by the assembler is described as the operand (sfr) of an 8-bit manipulation instruction. An address can also be specified.
- 16-bit manipulation
  A symbol reserved by the assembler is described as the operand (sfrp) of a 16-bit manipulation instruction. An even address can also be specified.
-

Table 3-3, “Special Function Register List,” on page 78 lists the special function registers. The meanings of symbols in this table are as follows.

- Symbol
  These symbols indicate the addresses of the internal special function registers.
  They are reserved words for the assembler and are defined in a header file for the C compiler.
  So these symbols can be used as instruction operands.
- R/W
  This column shows whether the corresponding special function register can be read or written.

  R/W : Both reading and writing are enabled.
  R : The value in the register can be read out. A write to this register is ignored.
  W : A value can be written to the register. The values read from the register are not meaningful.
- Bit units for manipulation
  Indicates the bit units (1, 8, 16) in which the SFR can be manipulated.
- After reset
  The register is set to the value immediately after the $\overline{\text{RESET}}$ signal is input.

Table 3-3: Special Function Register List (1/5)

Address	SFR Name	Symbol	R/W	Manipulation Bit Unit			After Reset
				1-bit	8-bit	16-bit	
FF00H	Port 0	P0	R/W	×	×	-	00H
FF01H	Port 1	P1	R/W	×	×	-	00H
FF02H	Port 2	P2	R	×	×	-	00H
FF03H	Port 3	P3	R/W	×	×	-	00H
FF07H	Port 7	P7	R/W	×	×	-	00H
FF08H	Port 8	P8	R/W	×	×	-	00H
FF09H	Port 9	P9	R/W	×	×	-	00H
FF0AH	Port 10	P10	R/W	×	×	-	00H
FF0BH	Port 11	P11	R/W	×	×	-	00H
FF0CH	Port 12	P12	R/W	×	×	-	00H
FF0DH	Port 13	P13	W	×	×	-	00H
FF0EH	Port 14	P14	R/W	×	×	-	00H
FF0FH	Port 15	P15	R/W	×	×	-	00H
FF10H	Clocked serial interface 10 mode register	CSIM10	R/W	×	×	-	00H
FF11H	Clocked serial interface 10 clock selection register	CSIC10	R/W	×	×	-	00H
FF12H	Clocked serial interface 10 shift register	SIO10	R	-	×	-	00H
FF14H	Clocked serial interface 10 transmit register	SOTB0	R/W	-	×	-	undefined
FF17H	Port 7 function register	PF7	R/W	×	×	-	00H
FF18H	Port 8 function register	PF8	R/W	×	×	-	00H
FF19H	Port 9 function register	PF9	R/W	×	×	-	00H
FF1AH	Port 10 function register	PF10	R/W	×	×	-	00H
FF1BH	Port 11 function register	PF11	R/W	×	×	-	00H
FF20H	Port mode register 0	PM0	R/W	×	×	-	FFH
FF21H	Port mode register 1	PM1	R/W	×	×	-	FFH
FF23H	Port mode register 3	PM3	R/W	×	×	-	FFH
FF27H	Port mode register 7	PM7	R/W	×	×	-	FFH
FF28H	Port mode register 8	PM8	R/W	×	×	-	FFH
FF29H	Port mode register 9	PM9	R/W	×	×	-	FFH
FF2AH	Port mode register 10	PM10	R/W	×	×	-	FFH
FF2BH	Port mode register 11	PM11	R/W	×	×	-	FFH
FF2CH	Port mode register 12	PM12	R/W	×	×	-	FFH
FF2EH	Port mode register 14	PM14	R/W	×	×	-	FFH
FF2FH	Port mode register 15	PM15	R/W	×	×	-	FFH
FF30H	Pull-up resistor control register 0	PU0	R/W	×	×	-	00H
FF31H	Pull-up resistor control register 1	PU1	R/W	×	×	-	00H
FF33H	Pull-up resistor control register 3	PU4	R/W	×	×	-	00H
FF37H	Pull-up resistor control register 7	PU7	R/W	×	×	-	00H
FF38H	Pull-up resistor control register 8	PU8	R/W	×	×	-	00H
FF39H	Pull-up resistor control register 9	PU9	R/W	×	×	-	00H
FF3AH	Pull-up resistor control register 10	PU10	R/W	×	×	-	00H
FF3BH	Pull-up resistor control register 11	PU11	R/W	×	×	-	00H

Table 3-3: Special Function Register List (2/5)

Address	SFR Name	Symbol		R/W	Manipulation Bit Unit			After Reset
					1-bit	8-bit	16-bit	
FF3CH	Pull-up resistor control register 12	PU12		R/W	×	×	-	00H
FF3EH	Pull-up resistor control register 14	PU14		R/W	×	×	-	00H
FF3FH	Pull-up resistor control register 15	PU15		R/W	×	×	-	00H
FF40H	Clock output select register	CKS		R/W	×	×	-	00H
FF48H	External Interrupt rising edge enable register	EGP		R/W	×	×	-	00H
FF49H	External Interrupt falling edge enable register	EGN		R/W	×	×	-	00H
FF4AH	Input switch control register	ISC		R/W	×	×	-	00H
FF4CH	Watch timer mode register	WTM		R/W	×	×	-	00H
FF50H	UART6 mode register	ASIM6		R/W	×	×	-	01H
FF52H	UART6 reception buffer register	RXB6		R	-	×	-	FFH
FF53H	UART6 reception status register	ASIS6		R	-	×	-	00H
FF54H	UART6 transmission buffer register	TXB6		R/W	-	×	-	00H
FF55H	UART6 transmission status register	ASIF6		R	-	×	-	00H
FF56H	UART6 clock selection register	CKSR6		R/W	-	×	-	00H
FF57H	UART6 baud rate selection register	BRGC6		R/W	-	×	-	FFH
FF58H	UART6 and LIN protocol control register	ASICL6		R/W	×	×	-	16H
FF5DH	UART6 transmission shift register	TXS6		R	-	×	-	FFH
FF60H	CSI11 Clocked Serial Interface mode register	CSIM11		R/W	×	×	-	00H
FF61H	CSI11 clock selection register	CSIC11		R/W	-	×	-	00H
FF62H	CSI11 shift register	SIO11		R/W	-	×	-	00H
FF64H	CSI11 transmission buffer register	SOTB11		R/W	-	×	-	undefined
FF65H	16-bit timer 2 mode control register 2	TMC2		R/W	×	×	-	00H
FF66H	16-bit timer 2 Prescaler mode register 2	PRM2		R/W	-	×	-	00H
FF67H	16-bit timer 2 Capture/Compare control register 2	CRC2		R/W	-	×	-	00H
FF68H	16-bit timer 2 timer/counter register	TM2	TM2L	R	-	-	×	00H
FF69H			TM2H	R	-	-		00H
FF6AH	16-bit timer 2 capture register 0	CR20	CR20L	R	-	-	×	00H
FF6BH			CR20H	R	-	-		00H
FF6CH	16-bit timer 2 capture register 1	CR21	CR21L	R	-	-	×	00H
FF6DH			CR21H	R	-	-		00H
FF6EH	16-bit timer 2 capture register 2	CR22	CR22L	R	-	-	×	00H
FF6FH			CR22H	R	-	-		00H
FF70H	Flash protect command register	PFCMD		W	-	×	-	undefined
FF72H	Flash status register	PFS		R/W	×	×	-	undefined
FF74H	Flash programming mode control register	FLPMC		R/W	×	×	-	08H**Note 1**
FF80H	8-bit timer 50 count register	TM50		R	-	×	-	00H
FF81H	8-bit timer 50 compare register	CR50		R/W	-	×	-	00H
FF82H	8-bit timer 50 count clock selection register	TCL50		R/W	-	×	-	00H
FF83H	8-bit timer 50 mode control register	TMC50		R/W	×	×	-	00H
FF84H	8-bit timer 51 count register	TM51		R	-	×	-	00H
FF85H	8-bit timer 51 compare register	CR51		R/W	-	×	-	00H
FF86H	8-bit timer 51 count clock selection register	TCL51		R/W	-	×	-	00H

Table 3-3: Special Function Register List (3/5)

Address	SFR Name	Symbol	R/W	Manipulation Bit Unit			After Reset
				1-bit	8-bit	16-bit	
FF87H	8-bit timer 51 mode control register	TMC51	R/W	×	×	-	00H
FF88H	AD converter mode register	ADM	R/W	×	×	-	00H
FF89H	AD converter channel select register	ADS	R/W	×	×	-	00H
FF8AH	AD converter power fail mode register	PFM	R/W	×	×	-	00H
FF8BH	A/D converter power fail threshold register	PFT	R/W	×	×	-	00H
FF8CH FF8DH	AD conversion result register	ADCR	R	-	-	×	undefined
FF90H	LCD display mode register	LCDM	R/W	×	×	-	00H
FF92H	LCD display control register	LCDC	R/W	×	×	-	00H
FF98H	Watchdog mode selection register	WDTM	R/W	×	×	-	67H
FF99H	Watchdog enable register	WDTE	R/W	-	×	-	9AH
FFA0H	Internal low-speed oscillator mode register	RCM	R/W	×	×	-	00H
FFA1H	Main Clock mode register	MCM	R/W	×	×	-	00H
FFA2H	Main oscillator control register	MOC	R/W	×	×	-	00H
FFA3H	Oscillator stabilization time count status register	OSTC	R	×	×	-	00H
FFA4H	Oscillator stabilization time selection register	OSTS	R/W	×	×	-	05H
FFA9H	Clock monitor register	CLM	R/W	×	×	-	00H
FFACH	Reset source flag register	RESF	R	-	×	-	**Note 2**
FFB0H	DCAN control register	DCANC	R/W	×	×	-	00H
FFB1H	CAN control register	CANC	R/W	×	×	-	01H
FFB2H	CAN Transmit control register	TCR	R/W	-	×	-	00H
FFB3H	CAN Received message register	RMES	R	-	×	-	00H
FFB4H	CAN Redefinition control register	REDEF	R/W	×	×	-	00H
FFB5H	CAN error status register	CANES	R/W	-	×	-	00H
FFB6H	CAN Transmit error counter	TEC	R/W	-	×	-	00H
FFB7H	CAN Receive error counter	REC	R/W	-	×	-	00H
FFB8H	CAN Message count register	MCNT	R/W	-	×	-	C0H
FFB9H	CAN Bit rate prescaler	BRPRS	R/W	-	×	-	00H
FFBAH	CAN Synchronous control register 0	SYNC0	R/W	-	×	-	18H
FFBBH	CAN Synchronous control register 1	SYNC1	R/W	-	×	-	0EH
FFBCH	CAN Mask control register	MASKC	R/W	-	×	-	00H
FFBDH	Stepper Motor controller prescaler switch register	SMSWI	R/W	×	×	-	00H
FFBFH	Stepper motor controller timer control register	MCNTC	R/W	×	×	-	00H
FFC0H	Sound generator control register	SGCR	R/W	×	×	-	00H
FFC1H	Sound generator amplitude control register	SGAM	R/W	-	×	-	00H
FFC2H	Sound generator buzzer control register	SGBR	R/W	-	×	-	00H
FFC3H	Stepper motor 1 compare register 0	MCMP10	R/W	-	×	-	00H
FFC4H	Stepper motor 1 compare register 1	MCMP11	R/W	-	×	-	00H
FFC5H	Stepper motor 2 compare register 0	MCMP20	R/W	-	×	-	00H
FFC6H	Stepper motor 2 compare register 1	MCMP21	R/W	-	×	-	00H
FFC7H	Stepper motor 3 compare register 0	MCMP30	R/W	-	×	-	00H
FFC8H	Stepper motor 3 compare register 1	MCMP31	R/W	-	×	-	00H
FFC9H	Stepper motor 4 compare register 0	MCMP40	R/W	-	×	-	00H

Table 3-3: Special Function Register List (4/5)

Address	SFR Name	Symbol		R/W	Manipulation Bit Unit			After Reset
					1-bit	8-bit	16-bit	
FFCAH	Stepper motor 4 compare register 1	MCMP41		R/W	-	×	-	00H
FFCBH	Stepper motor port mode control register	PMC		R/W	-	×	-	00H
FFCCH	Stepper motor compare 1 control register	MCMPC1		R/W	-	×	-	00H
FFCDH	Stepper motor compare 2 control register	MCMPC2		R/W	-	×	-	00H
FFCEH	Stepper motor compare 3 control register	MCMPC3		R/W	-	×	-	00H
FFCFH	Stepper motor compare 4 control register	MCMPC4		R/W	-	×	-	00H
FFE0H	Interrupt request flag register 0L	IF0	IF0L	R/W	×	×	×	00H
FFE1H	Interrupt request flag register 0H		IF0H	R/W	×	×		00H
FFE2H	Interrupt request flag register 1L	IF1	IF1L	R/W	×	×	×	00H
FFE3H	Interrupt request flag register 1H		IF1H	R/W	×	×		00H
FFE4H	Interrupt mask flag register 0L	MK0	MK0L	R/W	×	×	×	FFH
FFE5H	Interrupt mask flag register 0H		MK0H	R/W	×	×		FFH
FFE6H	Interrupt mask flag register 1L	MK1	MK1L	R/W	×	×	×	FFH
FFE7H	Interrupt mask flag register 1H		MK1H	R/W	×	×		DFH
FFE8H	Priority order specification register 0L	PR0	PR0L	R/W	×	×	×	FFH
FFE9H	Priority order specification register 0H		PR0H	R/W	×	×		FFH
FFEAH	Priority order specification register 1L	PR1	PR1L	R/W	×	×	×	FFH
FFEBH	Priority order specification register 1H		PR1H	R/W	×	×		FFH
FFF0H	Memory size switching register	IMS		R/W	-	×	-	CFH
FFF3H	Bank selection register	B0		R/W	-	×	-	00H
FFF4H	Internal expansion RAM size switching register	IXS		R/W	-	×	-	0CH
FFFBH	Processor clock control register	PCC		R/W	×	×	-	00H
F800H	Timer TMP0 control register 0	TP0CTL0		R/W	×	×	-	00H
F801H	Timer TMP0 control register 1	TP0CTL1		R/W	×	×	-	00H
F802H	Timer TMP0 input/output control register 0	TP0IOC0		R/W	×	×	-	00H
F803H	Timer TMP0 input/output control register 1	TP0IOC1		R/W	×	×	-	00H
F805H	Timer TMP0 option register	TP0OPT0		R/W	×	×	-	00H
F806H	Timer TMP0 capture/compare register 0	TP0CCR0		R/W	-	-	×	0000H
F807H								
F808H	Timer TMP0 capture/compare register 1	TP0CCR1		R/W	-	-	×	0000H
F809H								
F80AH	Timer TMP0 count register 0	TP0CNT		R	-	-	×	0000H
F80BH								
F80CH	Timer TMP0 capture/compare buffer register 0	TP0CCB0		R	-	-	×	0000H
F80DH								
F80EH	Timer TMP0 capture/compare buffer register 1	TP0CCB1		R	-	-	×	0000H
F80FH								
F810H	Timer TMP1 control register 0	TP1CTL0		R/W	×	×	-	00H
F811H	Timer TMP1 control register 1	TP1CTL1		R/W	×	×	-	00H
F812H	Timer TMP1 input/output control register 0	TP1IOC0		R/W	×	×	-	00H
F813H	Timer TMP1 input/output control register 1	TP1IOC1		R/W	×	×	-	00H
F815H	Timer TMP1 option register	TP1OPT0		R/W	×	×	-	00H

Table 3-3: Special Function Register List (5/5)

Address	SFR Name	Symbol	R/W	Manipulation Bit Unit			After Reset
				1-bit	8-bit	16-bit	
F816H F817H	Timer TMP1 capture/compare register 0	TP1CCR0	R/W	-	-	×	0000H
F818H F819H	Timer TMP1 capture/compare register 1	TP1CCR1	R/W	-	-	×	0000H
F81AH F81BH	Timer TMP1 count register 0	TP1CNT	R	-	-	×	0000H
F81CH F81DH	Timer TMP1 capture/compare buffer register 0	TP1CCB0	R	-	-	×	0000H
F81EH F81FH	Timer TMP1 capture/compare buffer register 1	TP1CCB1	R	-	-	×	0000H
F820H	Timer TMP2 control register 0	TP2CTL0	R/W	×	×	-	00H
F821H	Timer TMP2 control register 1	TP2CTL1	R/W	×	×	-	00H
F822H	Timer TMP2 input/output control register 0	TP2IOC0	R/W	×	×	-	00H
F823H	Timer TMP2 input/output control register 1	TP2IOC1	R/W	×	×	-	00H
F825H	Timer TMP2 option register	TP2OPT0	R/W	×	×	-	00H
F826H F827H	Timer TMP2 capture/compare register 0	TP2CCR0	R/W	-	-	×	0000H
F828H F829H	Timer TMP2 capture/compare register 1	TP2CCR1	R/W	-	-	×	0000H
F82AH F82BH	Timer TMP2 count register 0	TP2CNT	R	-	-	×	0000H
F82CH F82DH	Timer TMP2 capture/compare buffer register 0	TP2CCB0	R	-	-	×	0000H
F82EH F82FH	Timer TMP2 capture/compare buffer register 1	TP2CCB1	R	-	-	×	0000H

Notes: 1. The value depends on the operating mode:
User mode: 08H
On board mode: 0CH

2. This value depends on the reason of the reset source.

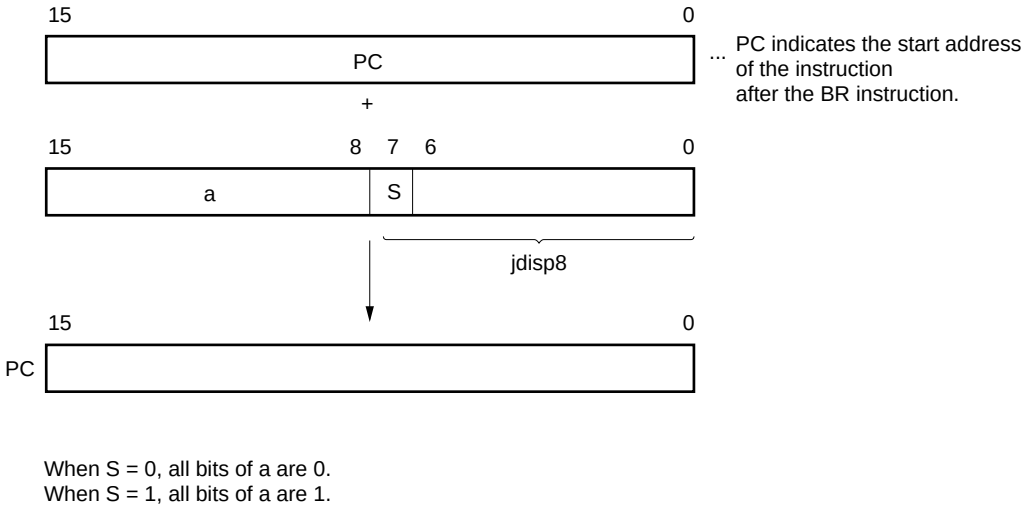
3.3 Instruction Address Addressing

An instruction address is determined by program counter (PC) contents. The PC contents are normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched every time an instruction is executed. However, when a branch instruction is executed, the branch destination information is set to the PC and branched by the following addressing.
(For details of instructions, refer to **78K/0 User's Manual - Instructions (U12326E)**.

3.3.1 Relative addressing

The value obtained by adding 8-bit immediate data (displacement value: jdisp8) of the instruction code to the address of the following instruction is transferred to the program counter (PC) and branched to. The displacement value is treated as signed two's complement data (-128 to +127) and bit 7 becomes a sign bit.
In other words, the range of branch in relative addressing is between -128 and +127 of the start address of the following instruction. This function is carried out when the BR $addr16 instruction or a conditional branch instruction is executed.

Figure 3-14: Relative Addressing



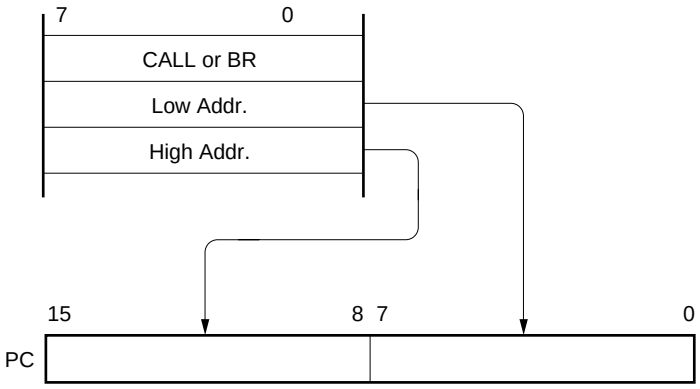
When S = 0, all bits of a are 0.
When S = 1, all bits of a are 1.

3.3.2 Immediate addressing

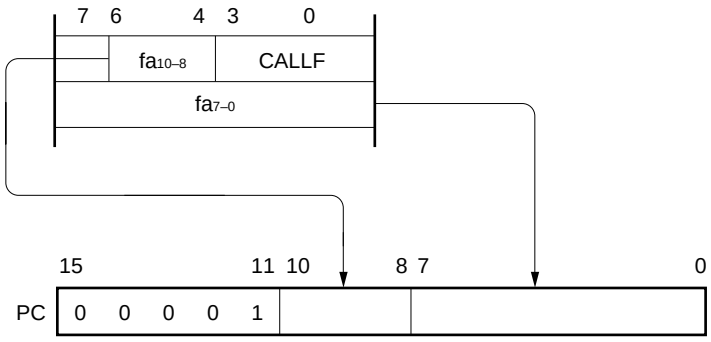
Immediate data in the instruction word is transferred to the program counter (PC) and branched. This function is carried out when the CALL!addr16 or BR!addr16 or CALLF!addr11 instruction is executed.
CALL!addr16 and BR!addr16 instructions can branch to all the memory space.
CALLF!addr11 instruction branches to the area from 0800H to 0FFFH.

Figure 3-15: Immediate Addressing

(a) In the case of CALL !addr16 and BR !addr16 instructions



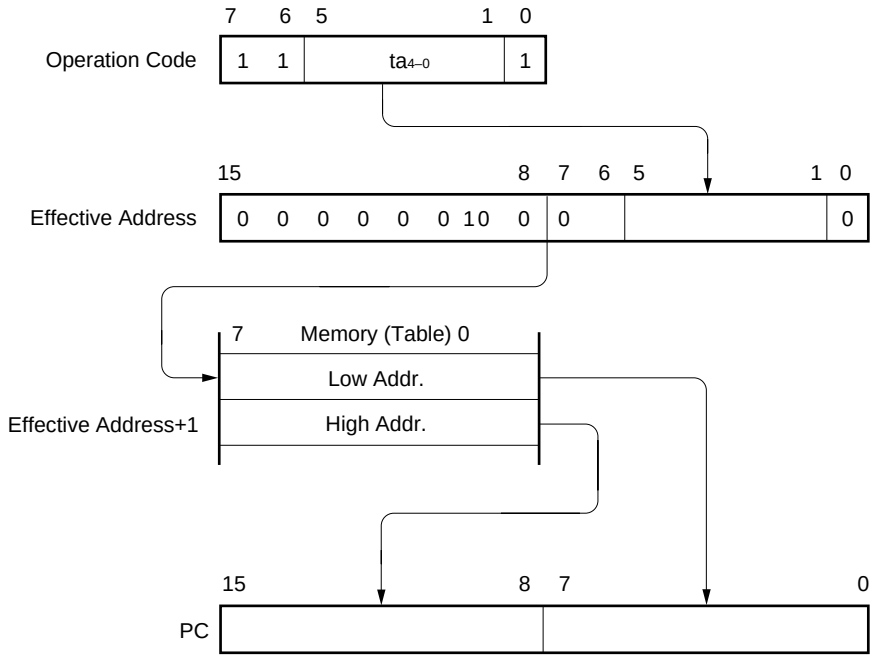
(b) In the case of CALLF !addr11 instruction



3.3.3 Table indirect addressing

Table contents (branch destination address) of the particular location to be addressed by bits 1 to 5 of the immediate data of an operation code are transferred to the program counter (PC) and branched to. Table indirect addressing is carried out when the CALLT [addr5] instruction is executed. This instruction can refer to the address stored in the memory table 40H to 7FH and branch to all the memory space.

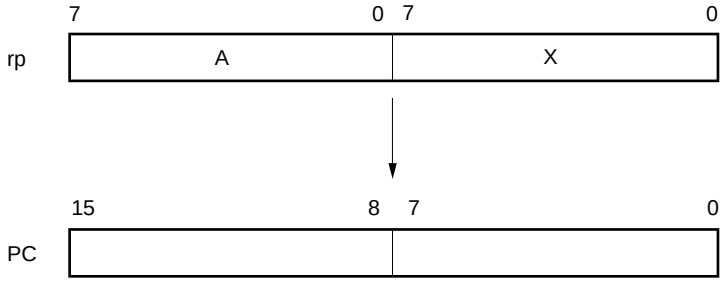
Figure 3-16: Table Indirect Addressing



3.3.4 Register addressing

Register pair (AX) contents to be specified with an instruction word are transferred to the program counter (PC) and branched.
This function is carried out when the BR AX instruction is executed.

Figure 3-17: Register Addressing



3.4 Operand Addressing

The following methods are available to specify the register and memory (addressing) which undergo manipulation during instruction execution.

3.4.1 Implied addressing

The register which functions as an accumulator (A and AX) in the general register is automatically (implicitly) addressed.
Of the µPD780822B Subseries instruction words, the following instructions employ implied addressing.

Table 3-4: Implied Addressing

Instruction	Register to be Specified by Implied Addressing
MULU	A register for multiplicant and AX register for product storage
DIVUW	AX register for dividend and quotient storage
ADJBA/ADJBS	A register for storage of numeric values which become decimal correction targets
ROR4/ROL4	A register for storage of digit data which undergoes digit rotation

Operand format
Because implied addressing can be automatically employed with an instruction, no particular operand format is necessary.

Description example
In the case of MULU X
With an 8-bit × 8-bit multiply instruction, the product of A register and X register is stored in AX. In this example, the A and AX registers are specified by implied addressing.

3.4.2 Register addressing

The general register is accessed as an operand. The general register to be accessed is specified with register bank select flags (RBS0 and RBS1) and register specify code (Rn, RPn) in the instruction code.
Register addressing is carried out when an instruction with the following operand format is executed. When an 8-bit register is specified, one of the eight registers is specified with 3 bits in the operation code.

Operand format

Table 3-5: Register Addressing

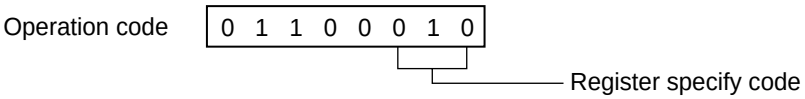
Identifier	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

'r' and 'rp' can be described with function names (X, A, C, B, E, D, L, H, AX, BC, DE and HL) as well as absolute names (R0 to R7 and RP0 to RP3).

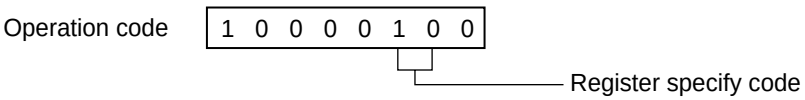
Description example

Figure 3-18: Register Addressing

(a) MOV A, C; when selecting C register as r



(b) INCW DE; when selecting DE register pair as rp



3.4.3 Direct addressing

The memory indicated by immediate data in an instruction word is directly addressed.

Operand format

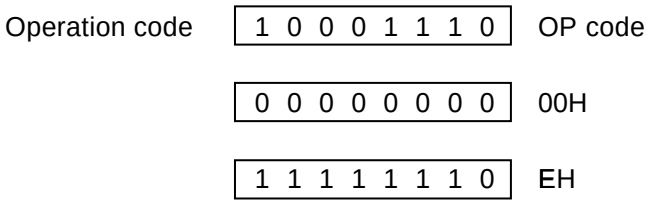
Table 3-6: Direct addressing

Identifier	Description
addr16	Label or 16-bit immediate data

Description example

MOV A, !0FE00H; when setting !addr16 to FE00H

Figure 3-19: Direct addressing



3.4.4 Short direct addressing

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word.
The fixed space to which this addressing is applied to is the 256-byte space, from FE20H to FF1FH. An internal high-speed RAM and special function registers (SFR) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.
The SFR area where short direct addressing is applied (FF00H to FF1FH) is a part of the SFR area. In this area, ports which are frequently accessed in a program, a compare register of the timer/event counter, and a capture register of the timer/event counter are mapped and these SFRs can be manipulated with a small number of bytes and clocks.
When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. Refer to Figure 3-20 below.

Operand format

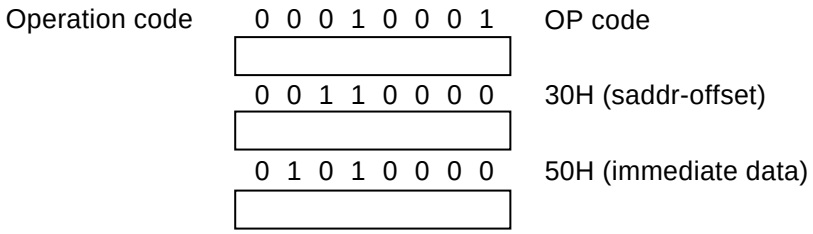
Table 3-7: Short direct addressing

Identifier	Description
saddr	Label of FE20H to FF1FH immediate data
saddrp	Label of FE20H to FF1FH immediate data (even address only)

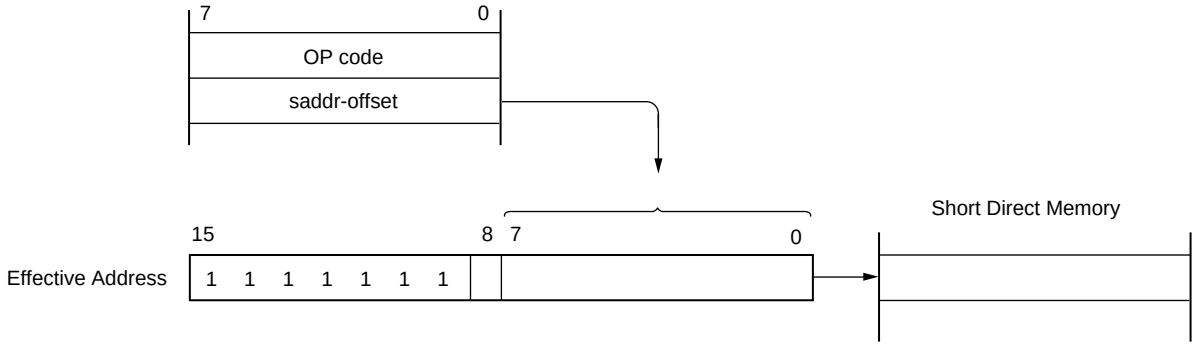
Figure 3-20: Short direct addressing

(a) Description example

MOV 0FE30H, #50H; when setting saddr to FE30H and immediate data to 50H.



(b) Illustration



Remark: When 8-bit immediate data is 20H to FFH, $\alpha = 0$
When 8-bit immediate data is 00H to 1FH, $\alpha = 1$

3.4.5 Special function register (SFR) addressing

The memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word.
This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFR mapped at FF00H to FF1FH can be accessed with short direct addressing.

Operand format

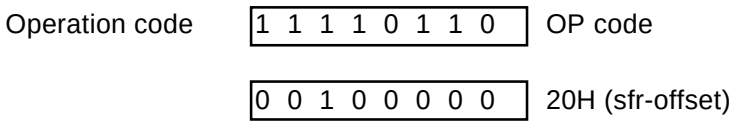
Table 3-8: Special-Function Register (SFR) Addressing

Identifier	Description
sfr	Special-function register name
sfrp	16-bit manipulatable special-function register name (even address only)

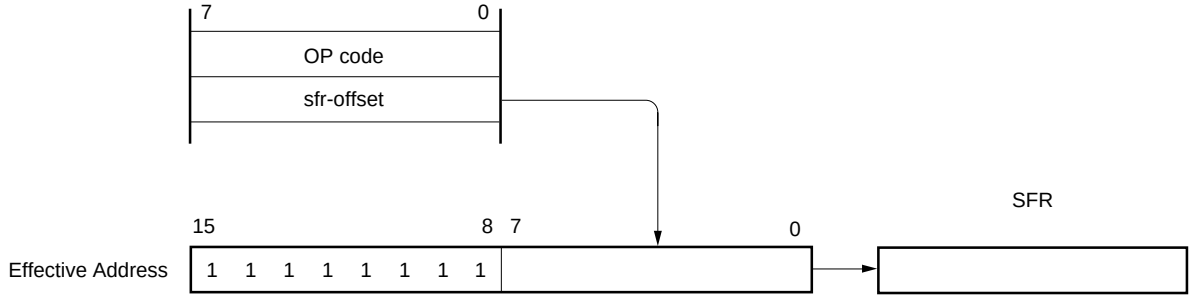
Figure 3-21: Special-Function Register (SFR) Addressing

(a) Description example

MOV PM0, A; when selecting PM0 (FE20H) as sfr



(b) Illustration



3.4.6 Register indirect addressing

The memory is addressed with the contents of the register pair specified as an operand. The register pair to be accessed is specified with the register bank select flag (RBS0 and RBS1) and the register pair specify code in the instruction code. This addressing can be carried out for all the memory spaces.

Operand format

Table 3-9: Register indirect addressing

Identifier	Description
-	[DE], [HL]

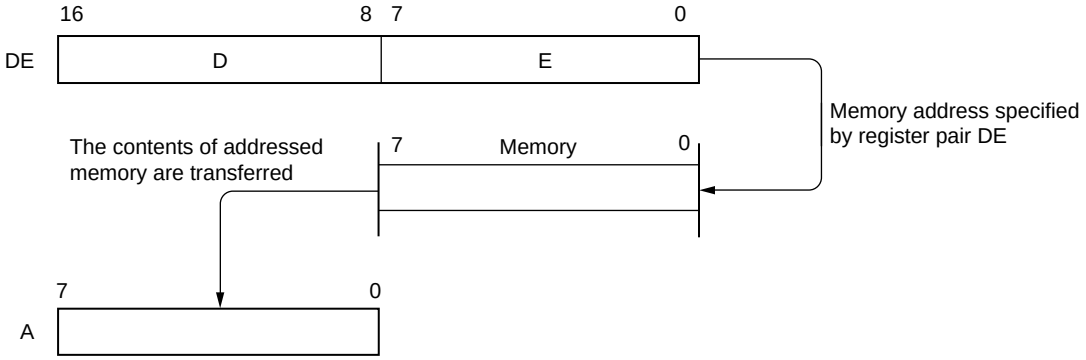
Figure 3-22: Register indirect addressing

(a) Description example

MOV A, [DE]; when selecting [DE] as register pair

Operation code	1	0	0	0	0	1	0	1

(b) Illustration



3.4.7 Based addressing

8-bit immediate data is added to the contents of the base register, that is, the HL register pair, and the sum is used to address the memory. The HL register pair to be accessed is in the register bank specified with the register bank select flags (RBS0 and RBS1). Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for all the memory spaces.

Operand format

Table 3-10: Based addressing

Identifier	Description
-	[HL + byte]

Figure 3-23: Based addressing description example

MOV A, [HL + 10H]; when setting byte to 10H

Operation code	1 0 1 0 1 1 1 0
	0 0 0 1 0 0 0 0

3.4.8 Based indexed addressing

The B or C register contents specified in an instruction are added to the contents of the base register, that is, the HL register pair, and the sum is used to address the memory. The HL, B, and C registers to be accessed are registers in the register bank specified with the register bank select flag (RBS0 and RBS1).
Addition is performed by expanding the contents of the B or C register as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for all the memory spaces.

Operand format

Table 3-11: Based indexed addressing

Identifier	Description
-	[HL + B], [HL + C]

Figure 3-24: Based indexed addressing description example

In the case of MOV A, [HL + B]

Operation code	1 0 1 0 1 0 1 1

3.4.9 Stack addressing

The stack area is indirectly addressed with the stack pointer (SP) contents.
This addressing method is automatically employed when the PUSH, POP, subroutine call and RETURN instructions are executed or the register is saved/reset upon generation of an interrupt request.
Stack addressing enables to address the internal high-speed RAM area only.

Figure 3-25: Stack addressing description example

In the case of PUSH DE

Operation code	1 0 1 1 0 1 0 1

[MEMO]

Chapter 4 Port Functions (µPD780822B Subseries)

4.1 Port Functions

The µPD780822B Subseries units incorporate eight input ports, eighty-seven output ports, and fifty-one input/output ports, for a total of seventy-six pins.
Figure 4-1 shows the port configuration. Every port is capable of 1-bit and 8-bit manipulations and can carry out considerably varied control operations. Besides port functions, the ports can also serve as on-chip hardware input/output pins.

Figure 4-1: Port Types

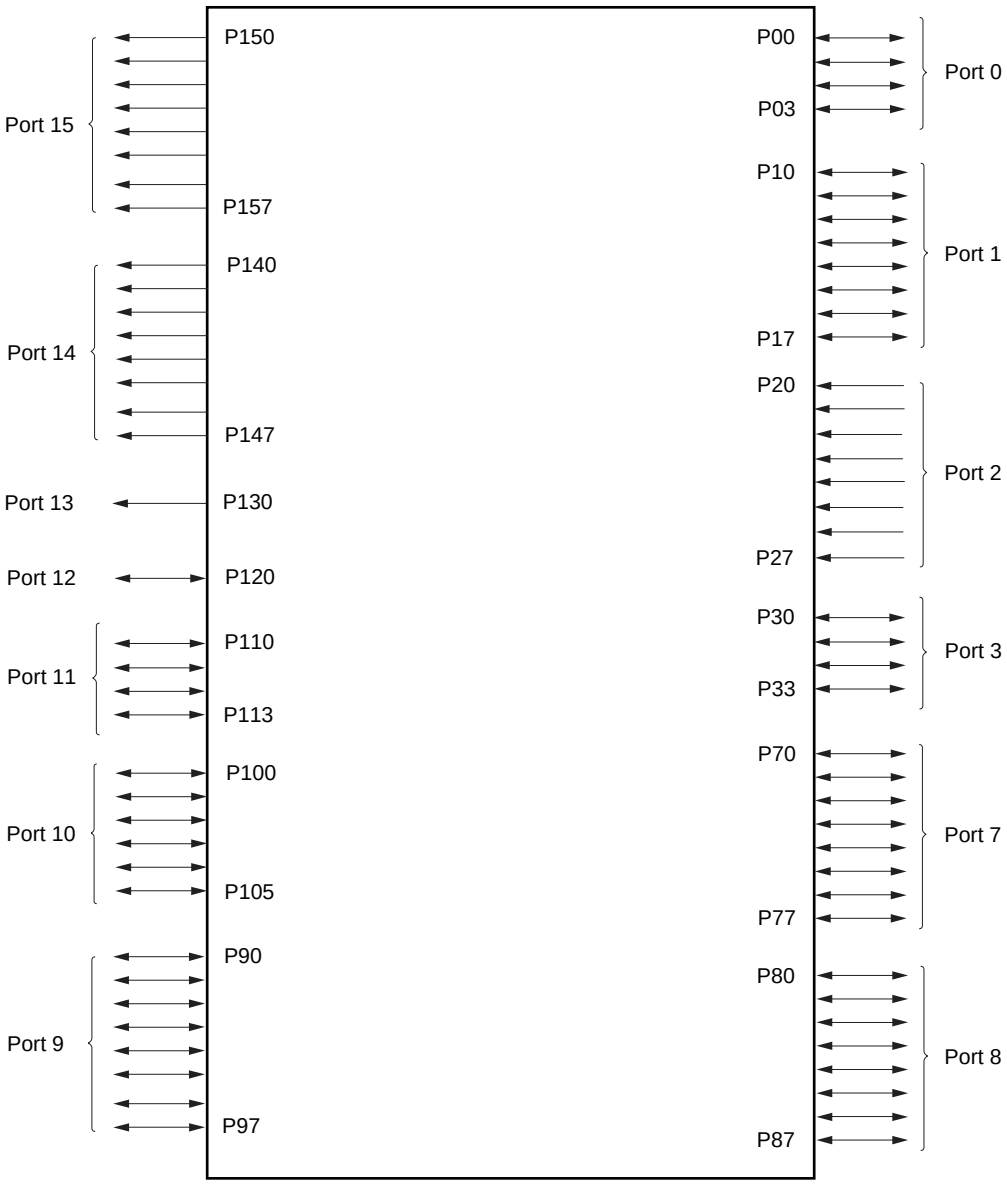


Table 4-1: Pin Input/Output Types (1/2)

Pin Name	I/O	Function	Alternate Function	After Reset
P00	I/O	Port 0 4-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	TI20	Input
P01			TI21	Input
P02			TIOP21	Input
P03			TIOP20	Input
P10	I/O	Port 1 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	CTXD	Input
P11			CRXD	Input
P12			RXD6	Input
P13			TXD6	Input
P14			TIOP01	Input
P15			TIOP10	Input
P16			TIOP11	Input
P17			TIO51/FLMD1	Input
P20-P27	I	Port 2 8-bit input only port Input mode can be specified bit-wise	ANI0 - ANI7	Input
P30	I/O	Port 3 4-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	INTP1	Input
P31			INTP2	Input
P32			PCL / SGOA	Input
P33			SGO / SGOF / BUZ	Input
P70-P77	I/O	Port 7 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register	S7 - S0	Input
P80-P87	I/O	Port 8 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register	S15 - S8	Input
P90	I/O	Port 9 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register	S23 / CCLK	Input
P91-P97			S22 - S16	Input
P100	I/O	Port 10 6-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register	S24 / TIOP00	Input
P101			S25 / SI10	Input
P102			S26 / SO10	Input
P103			S27 / $\overline{\text{SCK10}}$	Input
P104			S28 / TPO20	Input
P105			S29 / TI22	Input

Table 4-1: Pin Input/Output Types (2/2)

Pin Name	I/O	Function	Alternate Function	After Reset
P110	I/O	Port 11 4-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise This port can be used as a segment signal output port or an I/O port in 1-bit units by setting the port function register	S30 / SI11	Input
P111			S31 / SO11	Input
P112			S32 / $\overline{\text{SCK11}}$	Input
P113			S33 / TIO50	Input
P120	I/O	Port 12 1-bit input/output port If used an input port, a pull-up resistor can be connected by software bit-wise	INTP0	Input
P130	O	Port 13 1-bit output port	-	Input
P140	O	Port 14 8-bit output port	SM11	Hi-z
P141			SM12	
P142			SM13	
P143			SM14	
P144			SM21	
P145			SM22	
P146			SM23	
P147			SM24	
P150	O	Port 15 8-bit output port	SM31	Hi-z
P151			SM32	
P152			SM33	
P153			SM34	
P154			SM41	
P155			SM42	
P156			SM43	
P157			SM44	

4.2 Port Configuration

A port consists of the following hardware:

Table 4-2: Port Configuration

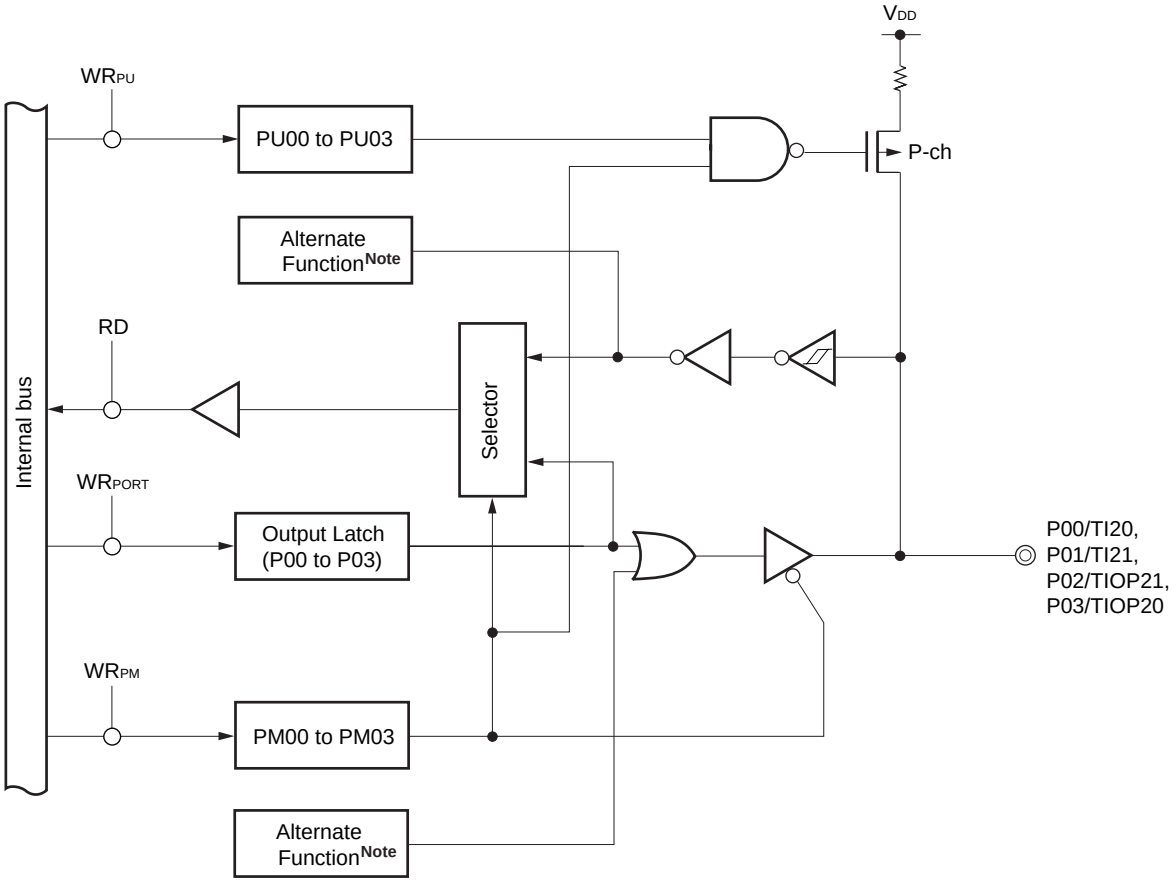
Item	Configuration
Control registers	Port mode register (PMm: m = 0, 1, 3, 7 to 12, 14, 15) Pull-up resistor option register (PUm: m = 0, 1, 3, 7 to 12) Port function register (PFm: m = 7, 8, 9, 10, 11)
Port	Total: 76 ports
Pull-up resistor	Specification by software for 67 pins

4.2.1 Port 0

Port 0 is an 4-bit input/output port with output latch. P00 to P03 pins can be specified in input or output mode in 1-bit units with the port mode register 0 (PM0). When P00 to P03 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register (PU0). Alternate functions include timer input/output.

$\overline{\text{RESET}}$ input sets port 0 to input mode.
Figure 4-2 shows block diagram of port 0.

Figure 4-2: P00 to P03 Configuration



Note: Set output latch to 0 when alternate function shall be applied to output.

Remarks: **1.** PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

4.2.2 Port 1

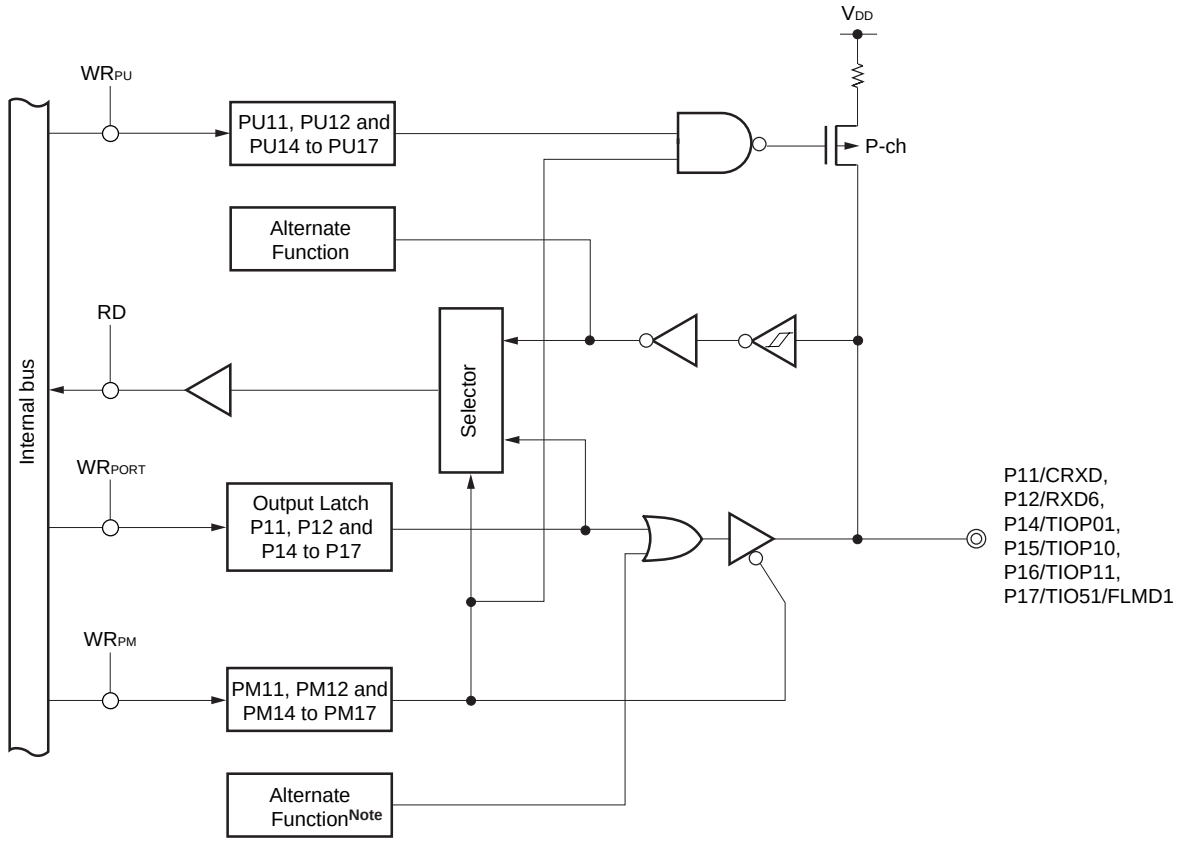
Port 1 is an 8-bit input/output port with output latch. P10 to P17 pins can be specified in input or output mode in 1-bit units with the port mode register 1 (PM1). When P10 to P17 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 1 (PU1). Alternate functions include DCAN and UART6 serial data input/output and timer input/output.

$\overline{\text{RESET}}$ input sets port 1 to input mode.

Figure 4-3 shows a block diagram of port 1.

Figure 4-3: Port 1 Configuration (1/2)

(a) P11, P12 and P14 to P17 Configuration



Note: Set output latch to 0 when alternate function shall be applied to output.

Remarks: **1.** PU : Pull-up resistor option register

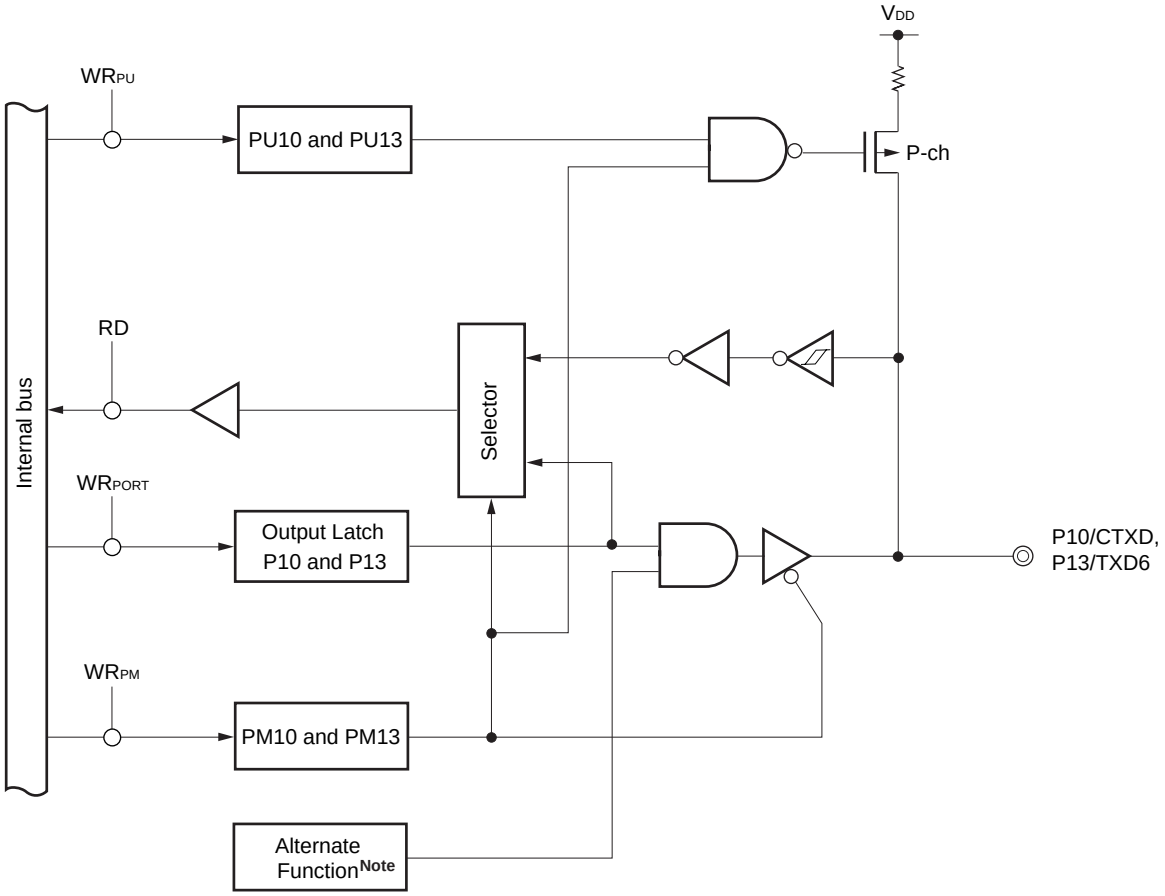
2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

Figure 4-3: P1 Configuration (2/2)

(b) P10 and P13 Configuration



Note: Set output latch to 1 when alternate function shall be applied to output.

Remarks: **1.** PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

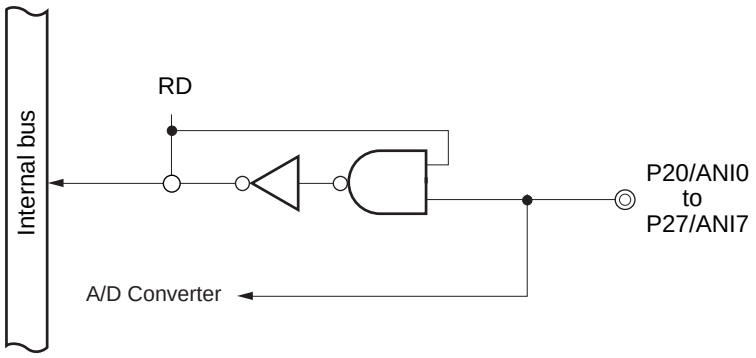
4. WR : Port write signal

4.2.3 Port 2

Port 2 is an 8-bit input port.
Alternate functions are analog input channels.

$\overline{\text{RESET}}$ input sets port 2 to input mode.
Figure 4-4 shows a block diagram of port 2.

Figure 4-4: P20 to P27 Configurations



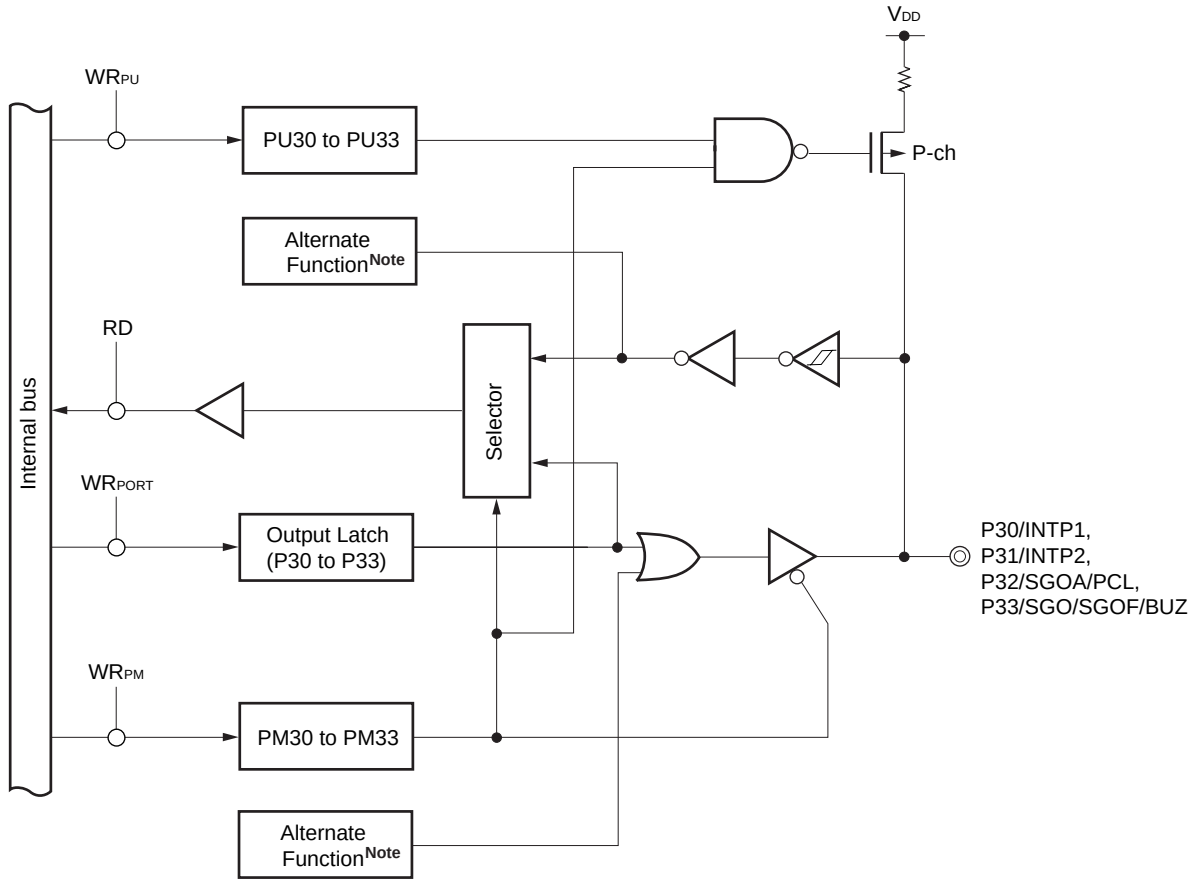
Remark: RD: Port read signal

4.2.4 Port 3

Port 3 is a 4-bit input/output port with output latch. P30 to P33 pins can be specified in input or output mode in 1-bit units with the port mode register 3 (PM3). When P30 to P33 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 3 (PU3). Alternate functions include external interrupt request inputs, buzzer output, clock output and sound generator output.

$\overline{\text{RESET}}$ input sets port 3 to input mode.
Figure 4-5 shows a block diagram of port 3.

Figure 4-5: P30 to P33 Configurations



Note: Set output latch to 0 when alternate function shall be applied to output.

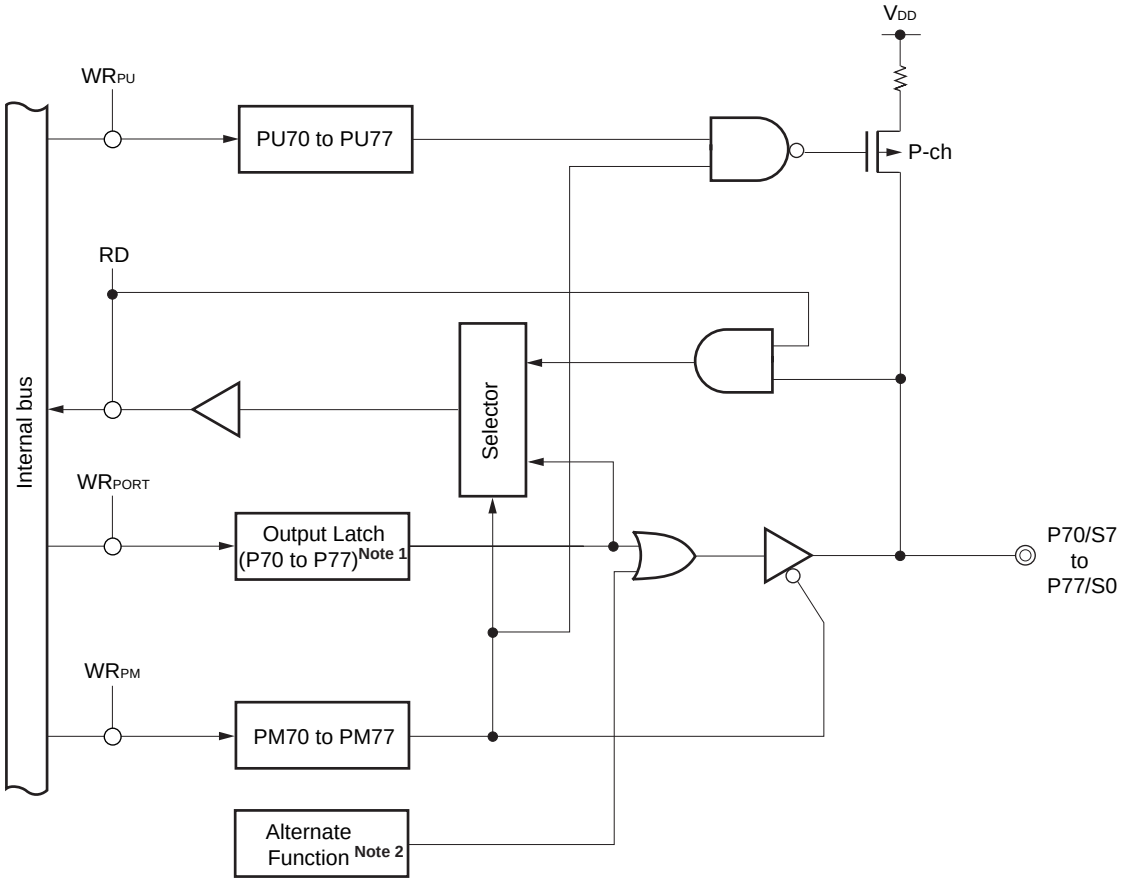
Remarks: **1.** PU : Pull-up resistor option register
2. PM : Port mode register
3. RD : Port read signal
4. WR : Port write signal

4.2.5 Port 7

Port 7 is an 8-bit output port with output latch. P70 to P77 pins can be specified in input or output mode in 1-bit units with the port mode register 7 (PM7). When P70 to P77 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 7 (PU7). Alternate functions are segment signal outputs of LCD controller/driver.

$\overline{\text{RESET}}$ input sets port 7 to input mode.
Figure 4-6 shows a block diagram of port 7.

Figure 4-6: P70 to P77 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

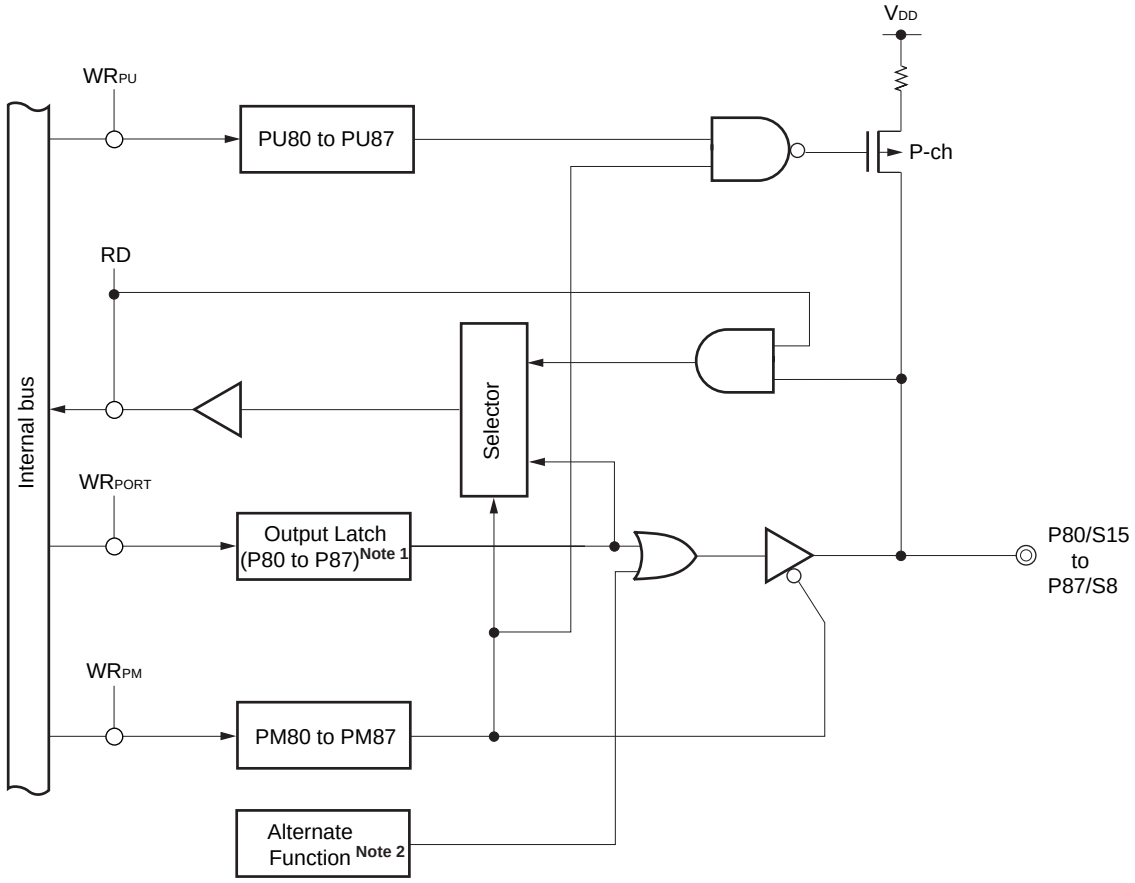
4. WR : Port write signal

4.2.6 Port 8

Port 8 is an 8-bit output port with output latch. P80 to P87 pins can be specified in input or output mode in 1-bit units with the port mode register 8 (PM8). When P80 to P87 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 8 (PU8). Alternate functions are segment signal outputs of LCD controller/driver.

$\overline{\text{RESET}}$ input sets port 8 to input mode.
Figure 4-7 shows a block diagram of port 8.

Figure 4-7: P80 to P87 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

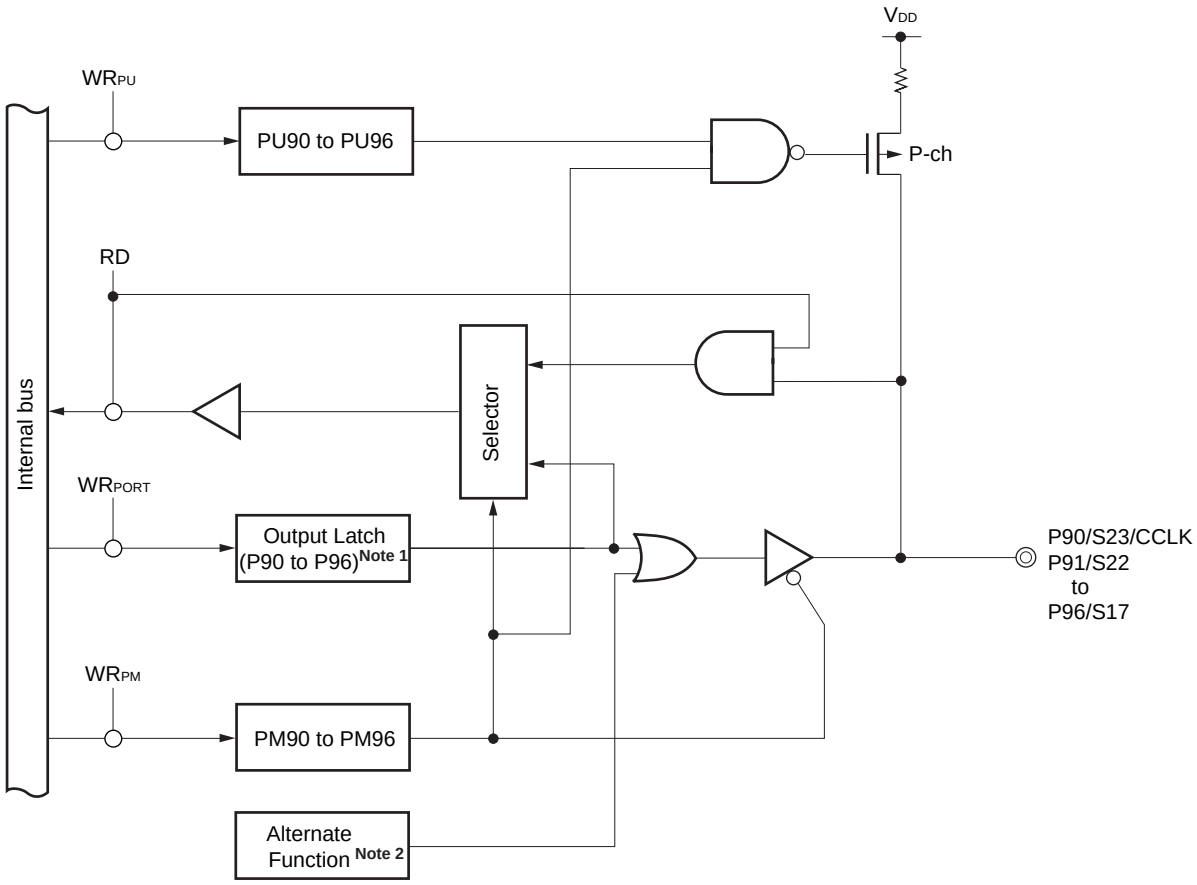
4.2.7 Port 9

Port 9 is an 8-bit output port with output latch. P90 to P97 pins can be specified in input or output mode in 1-bit units with the port mode register 9 (PM9). When P90 to P97 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 9 (PU9). Alternate functions include segment signal outputs of LCD controller/driver and DCAN external clock input.

$\overline{\text{RESET}}$ input sets port 9 to input mode.
Figure 4-8 shows a block diagram of port 9.

Figure 4-8: Port 9 Configuration (1/2)

(a) P90 to P96 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

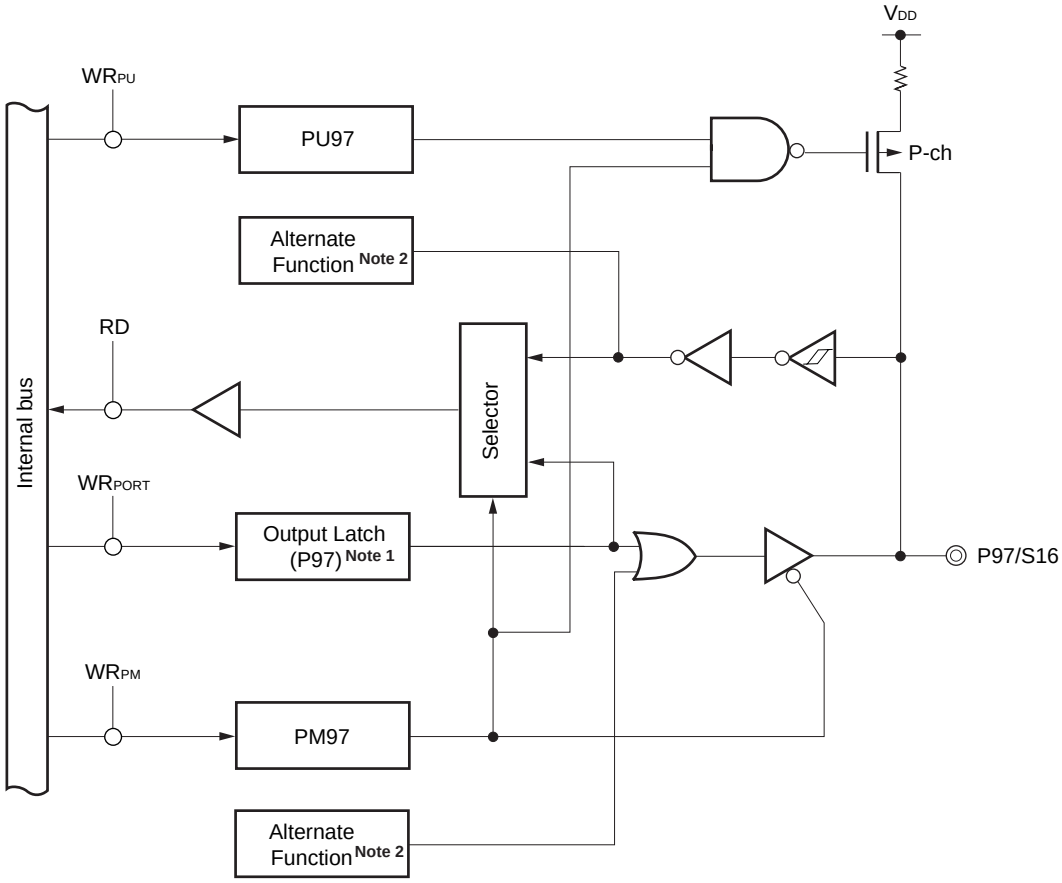
2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

Figure 4-8: Port 9 Configuration (2/2)

(b) P97 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

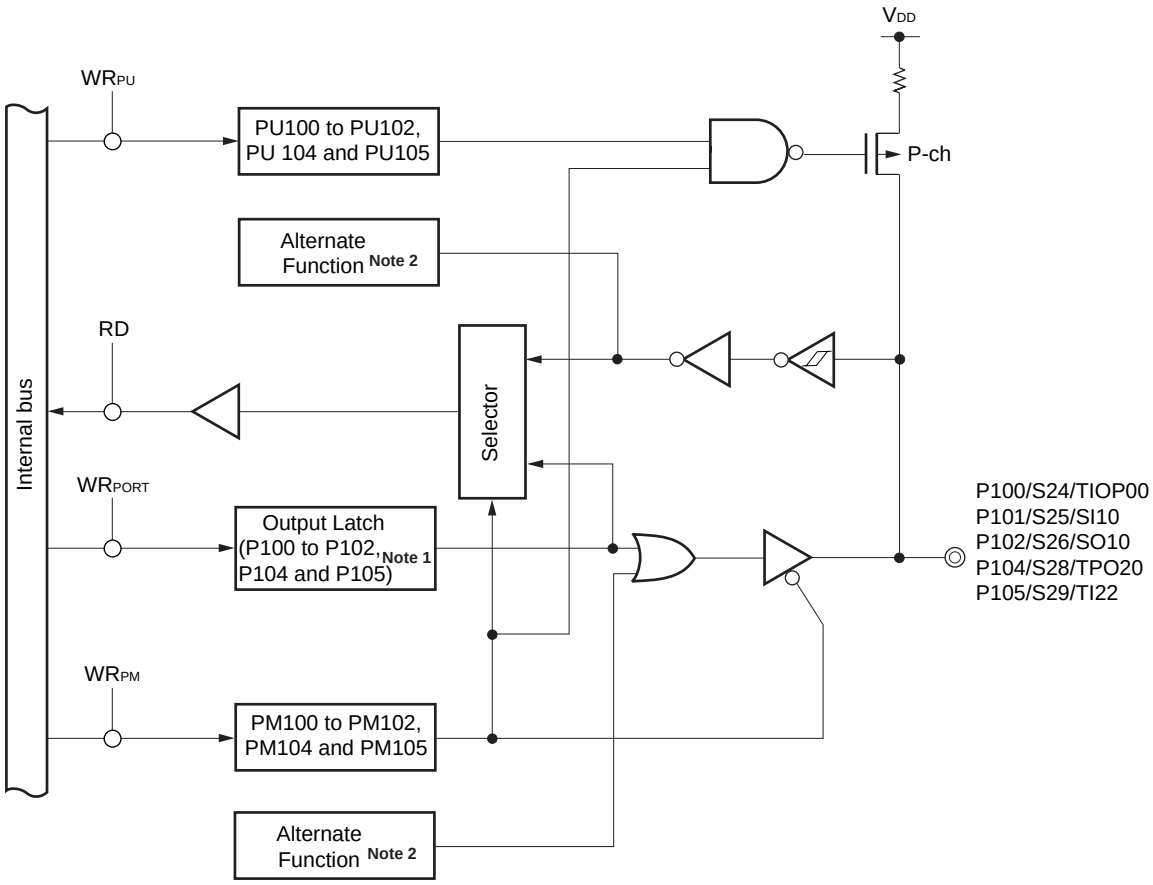
4.2.8 Port 10

Port 10 is a 6-bit input/output port with output latch. P100 to P105 pins can be specified in input or output mode in 1-bit units with the port mode register 10 (PM10). When P100 to P105 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 10 (PU10).
Alternate functions include timer input/output, synchronous serial interface input/output and segment signal outputs of LCD controller/driver.

$\overline{\text{RESET}}$ input sets port 10 to input mode.
Figure 4-9 shows block diagram of port 10.

Figure 4-9: Port 10 Configuration (1/2)

(a) P100 to P102, P104 and P105 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

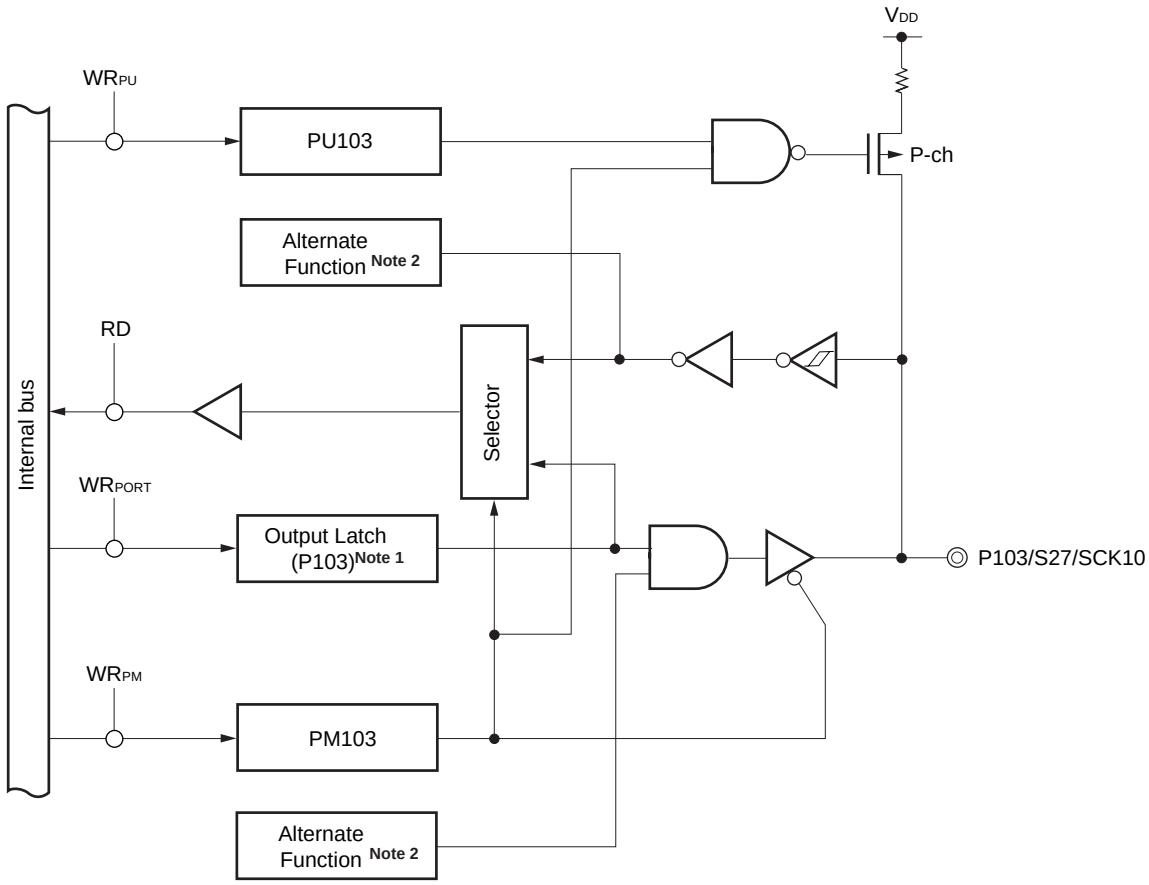
2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

Figure 4-9: Port 10 Configuration (2/2)

(b) P103 Configuration



Notes: 1. Set output latch to 1 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

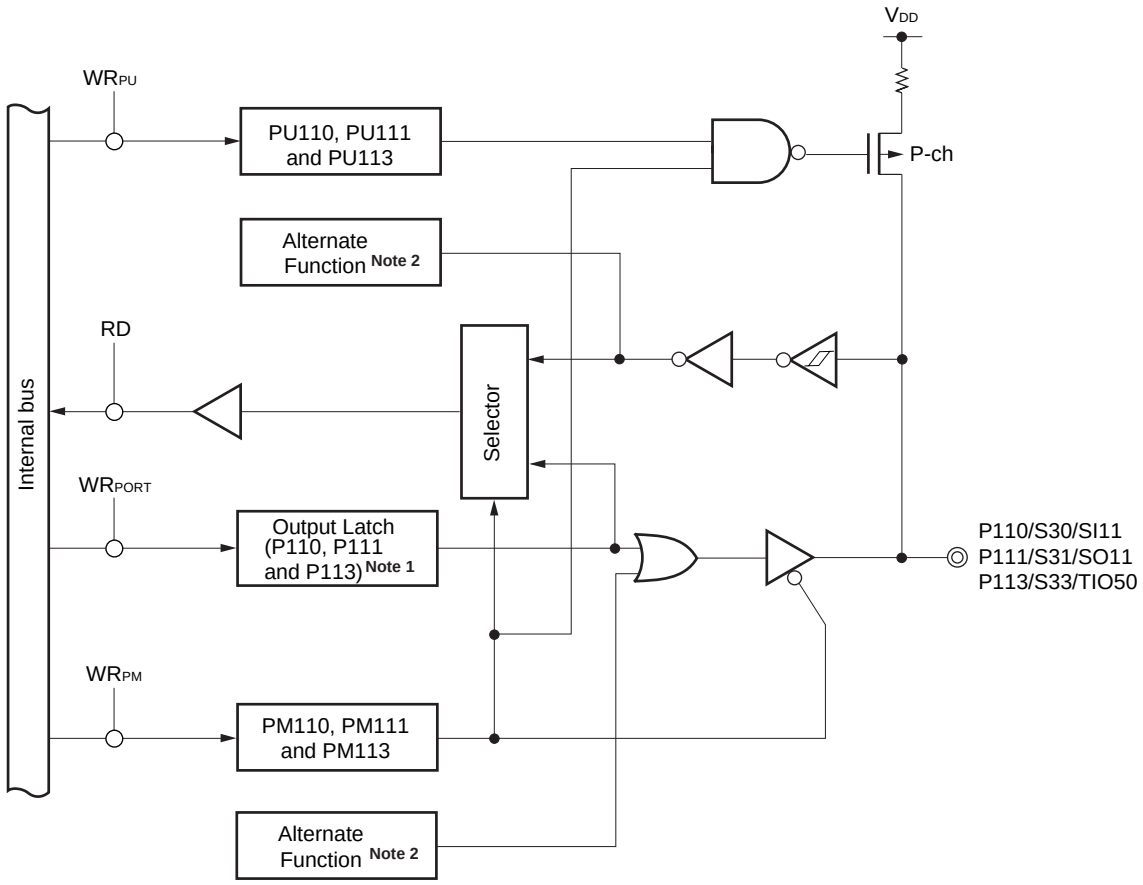
4.2.9 Port 11

Port 11 is a 4-bit input/output port with output latch. P110 to P113 pins can be specified in input or output mode in 1-bit units with the port mode register 11 (PM11). When P110 to P113 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register 11 (PU11).
Alternate functions include timer input/output, synchronous serial interface input/output and segment signal outputs of LCD controller/driver.

$\overline{\text{RESET}}$ input sets port 11 to input mode.
Figure 4-10 shows block diagram of port 11.

Figure 4-10: Port 11 Configuration (1/2)

(a) P110, P111 and P113 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

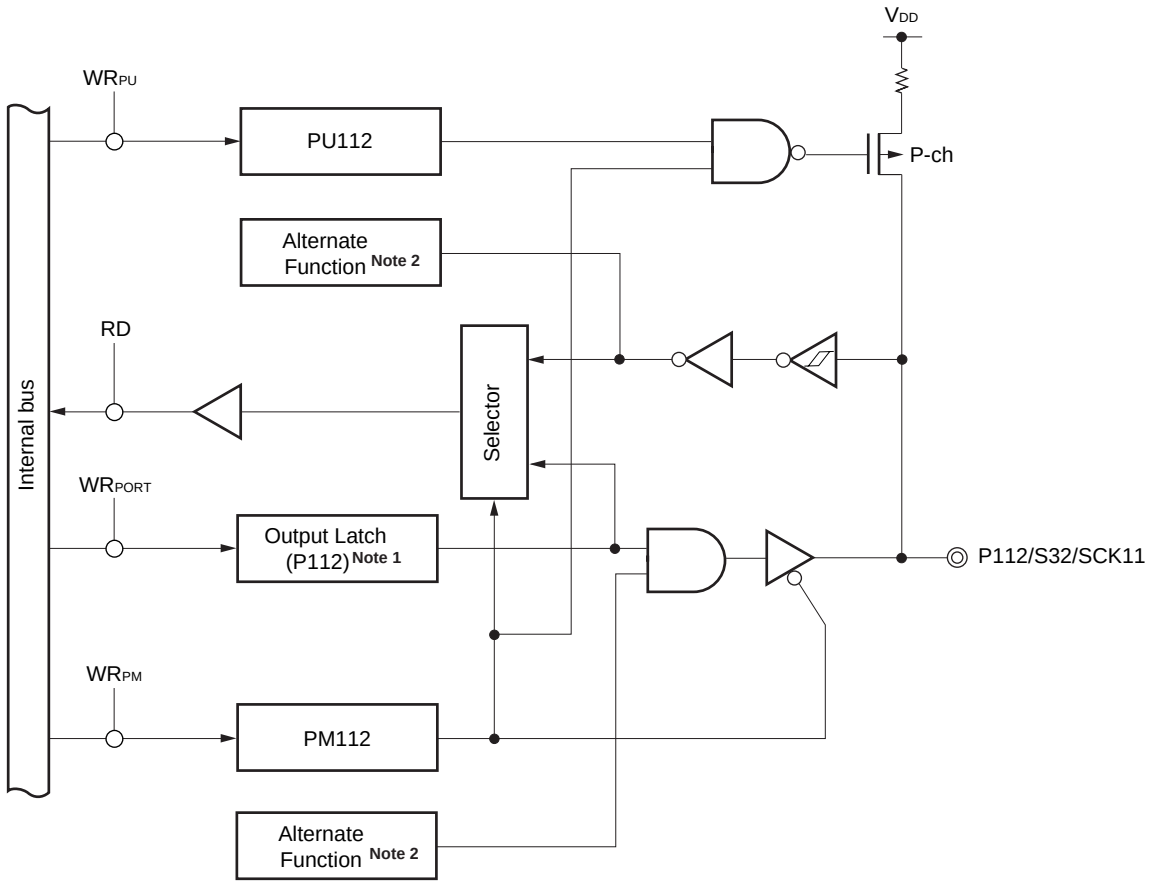
2. PM : Port mode register

3. RD : Port read signal

4. WR : Port write signal

Figure 4-10: Port 11 Configuration (2/2)

(b) P112 Configuration



Notes: 1. Set output latch to 1 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PU : Pull-up resistor option register

2. PM : Port mode register

3. RD : Port read signal

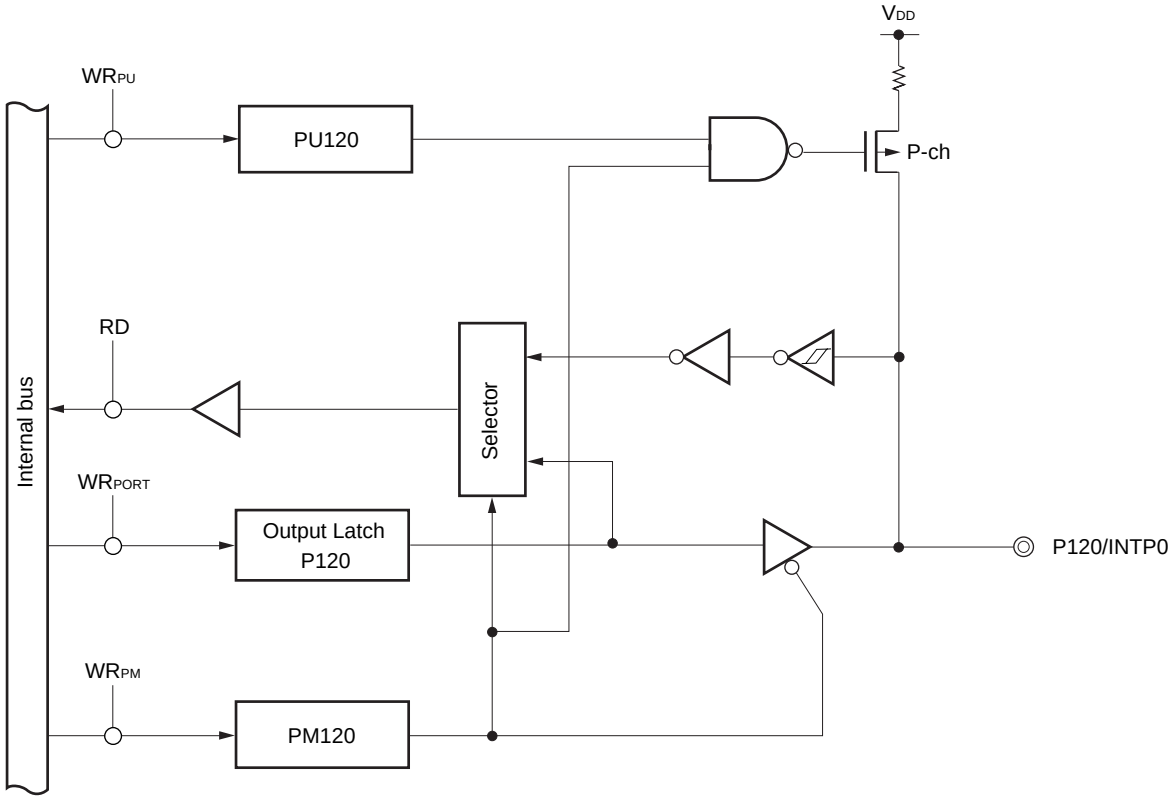
4. WR : Port write signal

4.2.10 Port 12

Port 12 is an 1-bit input/output port with output latch. The P120 pin can be specified in input or output mode with the port mode register 12 (PM12). When P120 pin is used as input pin, a pull-up resistor can be connected to it with the pull-up resistor option register 12 (PU12).
Alternate function is external interrupt request input.

$\overline{\text{RESET}}$ input sets port 12 to input mode.
Figure 4-11 shows block diagram of port 12.

Figure 4-11: P120 Configuration



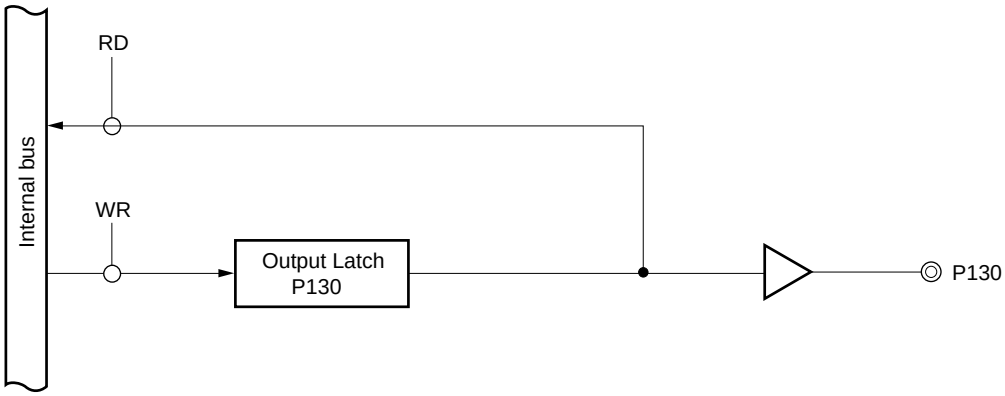
Remarks: **1.** PU : Pull-up resistor option register
2. PM : Port mode register
3. RD : Port read signal
4. WR : Port write signal

4.2.11 Port 13

Port 13 is an 1-bit output port with output latch.
Port 13 has no alternate function.

Figure 4-12 shows block diagram of port 13.

Figure 4-12: P130 Configuration



Remarks: 1. When reset is effected, P130 outputs low level.

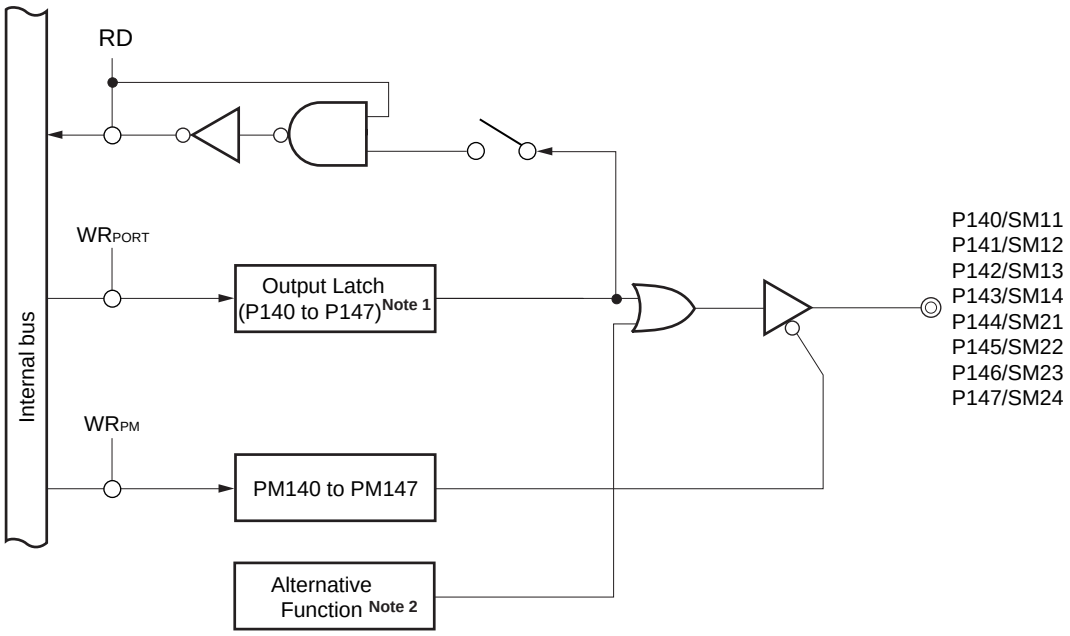
2. WR: Port 13 write signal

4.2.12 Port 14

Port 14 is an 8-bit output port with output latch. The output buffer of each pin can be separately turned off with the port mode register 14 PM14.
Alternate function is stepper motor controller outputs.

Figure 4-13 shows block diagram of port 14.

Figure 4-13: P140 to P147 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PM: Port mode register

2. RD : Port read signal

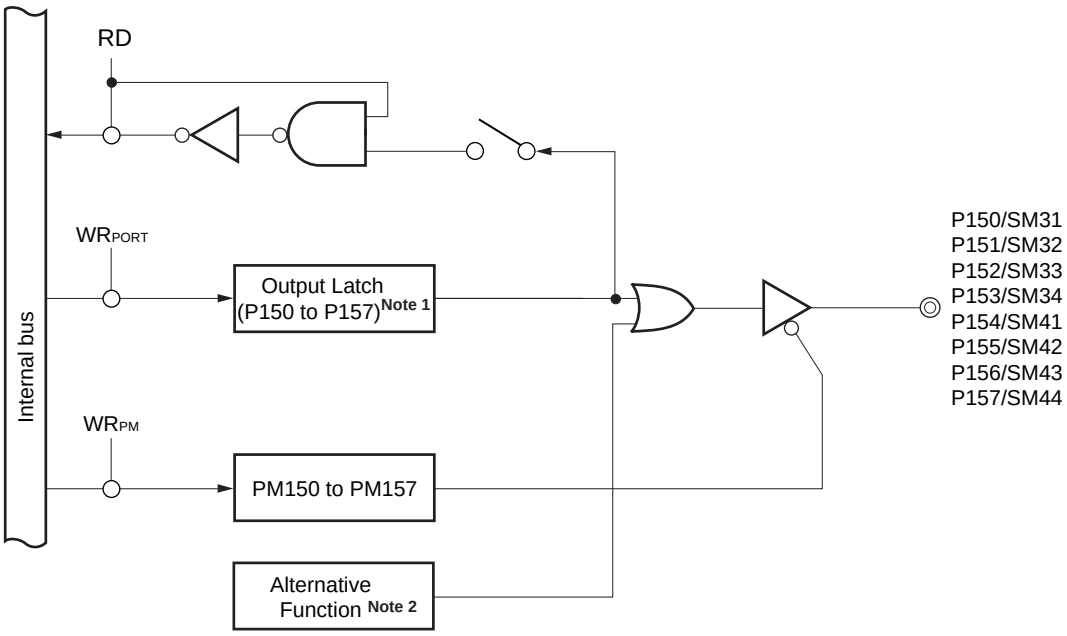
3. WR : Port write signal

4.2.13 Port 15

Port 15 is an 8-bit output port with output latch. The output buffer of each pin can be separately turned off with the port mode register 15 (PM15).
Alternate function is stepper motor controller outputs.

Figure 4-14 shows block diagram of port 15.

Figure 4-14: P150 to P157 Configuration



Notes: 1. Set output latch to 0 when alternate function shall be applied to output.

2. Disable alternate function when the content of the output latch shall be applied to output.

Remarks: 1. PM: Port mode register

2. RD : Port read signal

3. WR : Port write signal

4.3 Port Function Control Registers

The following four types of registers control the ports.

- Port mode registers (PM0, PM1, PM3, PM7 to PM12, PM14 and PM15)
- Pull-up resistor option register (PU0, PU1, PU3, and PU7 to PU12)
- Port function registers (PF7 to PF11)

(1) Port mode registers (PM0, PM1, PM3, PM7 to PM12, PM14 and PM15)

These registers are used to set the pins in input or output mode, in 1-bit units.
PM0, PM1, PM3, PM7 to PM12, PM14 and PM15 are independently set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets these registers to FFH.

When port pins are used as alternate function pins, set the port mode register and output latch according to the function.

Figure 4-15: Port Mode Register Format (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM0	1	1	1	1	PM03	PM02	PM01	PM00	R/W	FF20H	FFH
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	R/W	FF21H	FFH
PM3	1	1	1	1	PM33	PM32	PM31	PM30	R/W	FF23H	FFH
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	R/W	FF27H	FFH
PM8	PM87	PM86	PM85	PM84	PM83	PM82	PM81	PM80	R/W	FF28H	FFH
PM9	PM97	PM96	PM95	PM94	PM93	PM92	PM91	PM90	R/W	FF29H	FFH
PM10	1	1	PM105	PM104	PM103	PM102	PM101	PM100	R/W	FF2AH	FFH
PM11	1	1	1	1	PM113	PM112	PM111	PM110	R/W	FF2BH	FFH
PM12	1	1	1	1	1	1	1	PM120	R/W	FF2CH	FFH

Figure 4-15: Port Mode Register Format (2/2)

PMmn	PMmn Pin Input/Output Mode Selection (m = 0, 1, 3, 7- 12; n = 0 - 7)
0	Output mode
1	Input mode

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM14	PM147	PM146	PM145	PM144	PM143	PM142	PM141	PM140	R/W	FF2EH	FFH

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM15	PM157	PM156	PM155	PM154	PM153	PM152	PM151	PM150	R/W	FF2FH	FFH

PMmn	PMmn Pin Input/Output Mode Selection (m = 14, 15; n = 0 - 7)
0	Output mode (output buffer ON)
1	Hi-z mode or output buffer OFF

(2) Pull-up resistor option register (PU0, PU1, PU3, and PU7 to PU12)

This register is used to set whether to use an internal pull-up resistor at each port or not. The connection of the pull-up resistor is effective only when then port pin is set in input or high-impedance mode, regardless of PU0, PU1, PU3, PU7 to PU12 setting.

PU0, PU1, PU3, and PU7 to PU12 are set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets these registers to 00H.

Figure 4-16: Pull-Up Resistor Option Register (PU0, PU1, PU3, PU7 to PU12) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU0	0	0	0	0	PU03	PU02	PU01	PU00	R/W	FF30H	00H
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	R/W	FF31H	00H
PU3	0	0	0	0	PU33	PU32	PU31	PU30	R/W	FF33H	00H
PU7	PU77	PU76	PU75	PU74	PU73	PU72	PU71	PU70	R/W	FF37H	00H
PU8	PU87	PU86	PU85	PU84	PU83	PU82	PU81	PU80	R/W	FF38H	00H
PU9	PU97	PU96	PU95	PU94	PU93	PU92	PU91	PU90	R/W	FF39H	00H
PU10	0	0	PU105	PU104	PU103	PU102	PU101	PU100	R/W	FF3AH	00H
PU11	0	0	0	0	PU113	PU112	PU111	PU110	R/W	FF3BH	00H
PU12	0	0	0	0	0	0	0	PU120	R/W	FF3CH	00H

PUmn	PUmn Pin Internal Pull-up Resistor Selection (m = 0, 1, 3, 7 - 12; n = 0 - 7)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(3) Port function register (PF7 to PF11)

These registers are used to set the output buffer of the corresponding port (P7 to P11) as LCD segment driving port.

PF7 to PF11 are set with an 1-bit or an 8-bit manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to 00H.

Figure 4-17: Port Function Register (PF7, PF8, PF9, PF10 and PF11) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PF7	PF77	PF76	PF75	PF74	PF73	PF72	PF71	PF70	R/W	FF17H	00H
PF8	PF87	PF86	PF85	PF84	PF83	PF82	PF81	PF80	R/W	FF18H	00H
PF9	PF97	PF96	PF95	PF94	PF93	PF92	PF91	PF90	R/W	FF19H	00H
PF10	0	0	PF105	PF104	PF103	PF102	PF101	PF100	R/W	FF1AH	00H
PF11	0	0	0	0	PF113	PF112	PF111	PF110	R/W	FF1BH	00H

PFmn	PFmn Port Function Selection (m = 7, 8, 9, 10, 11; n = 0 - 7)
0	Port function
1	LCD segment function

4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is set, as shown below.

4.4.1 Writing to input/output port

(1) Output mode

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pins.
Once data is written to the output latch, it is retained until data is written to the output latch again.

(2) Input mode or high-impedance mode

A value is written to the output latch by a transfer instruction, but since the output buffer is OFF, the pin status does not change.
Once data is written to the output latch, it is retained until data is written to the output latch again.

Caution: In the case of 1-bit memory manipulation instruction, although a single bit is manipulated the port is accessed as an 8-bit unit, in a read-modify-write cycle. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input can change in the same time as for the manipulated bit.

4.4.2 Reading from input/output port

(1) Output mode

The output latch contents are read by a transfer instruction. The output latch contents do not change.

(2) Input mode

The pin status is read by a transfer instruction. The output latch contents do not change.

4.4.3 Operations on input/output port

(1) Output mode

An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins.
Once data is written to the output latch, it is retained until data is written to the output latch again.

(2) Input mode

The output latch contents are undefined, but since the output buffer is OFF, the pin status does not change.

Caution: In the case of 1-bit memory manipulation instruction, although a single bit is manipulated the port is accessed as an 8-bit unit. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input can change in the same time as for the manipulated bit.

[MEMO]

Chapter 5 Clock Generator (µPD780822B Subseries)

5.1 Clock Generator Functions

The clock generator generates the clock to be supplied to the CPU and peripheral hardware.
The following three system clock oscillators are available:

(1) X1 oscillator

The X1 oscillator circuit oscillates at frequencies up to f_{XP} = 16.0 MHz. Oscillation can be stopped by executing the STOP instruction or setting the main OSC control register (MOC) and processor clock control register (PCC).

(2) Internal low-speed oscillator

The internal low-speed oscillator circuit oscillates at frequencies of f_R = 240 kHz (TYP.). Oscillation can be stopped by setting the internal low-speed oscillator mode register (RCM) when “Can be stopped by software” is set by a mask option and the X1 input clock is used as the CPU clock.

(3) Subsystem clock oscillator

The subsystem clock oscillator circuit oscillates at frequencies of f_{XT} = 32.768 kHz. Oscillation cannot be stopped. When subsystem clock oscillator is not used, setting not to use the on-chip feedback resistor is possible using the processor clock control register (PCC), and the operating current can be reduced in the STOP mode.

Remarks: 1. f_{XP} : X1 input clock oscillation frequency

2. f_R : Internal low-speed oscillator clock oscillation frequency

3. f_{XT} : Subsystem clock oscillation frequency.

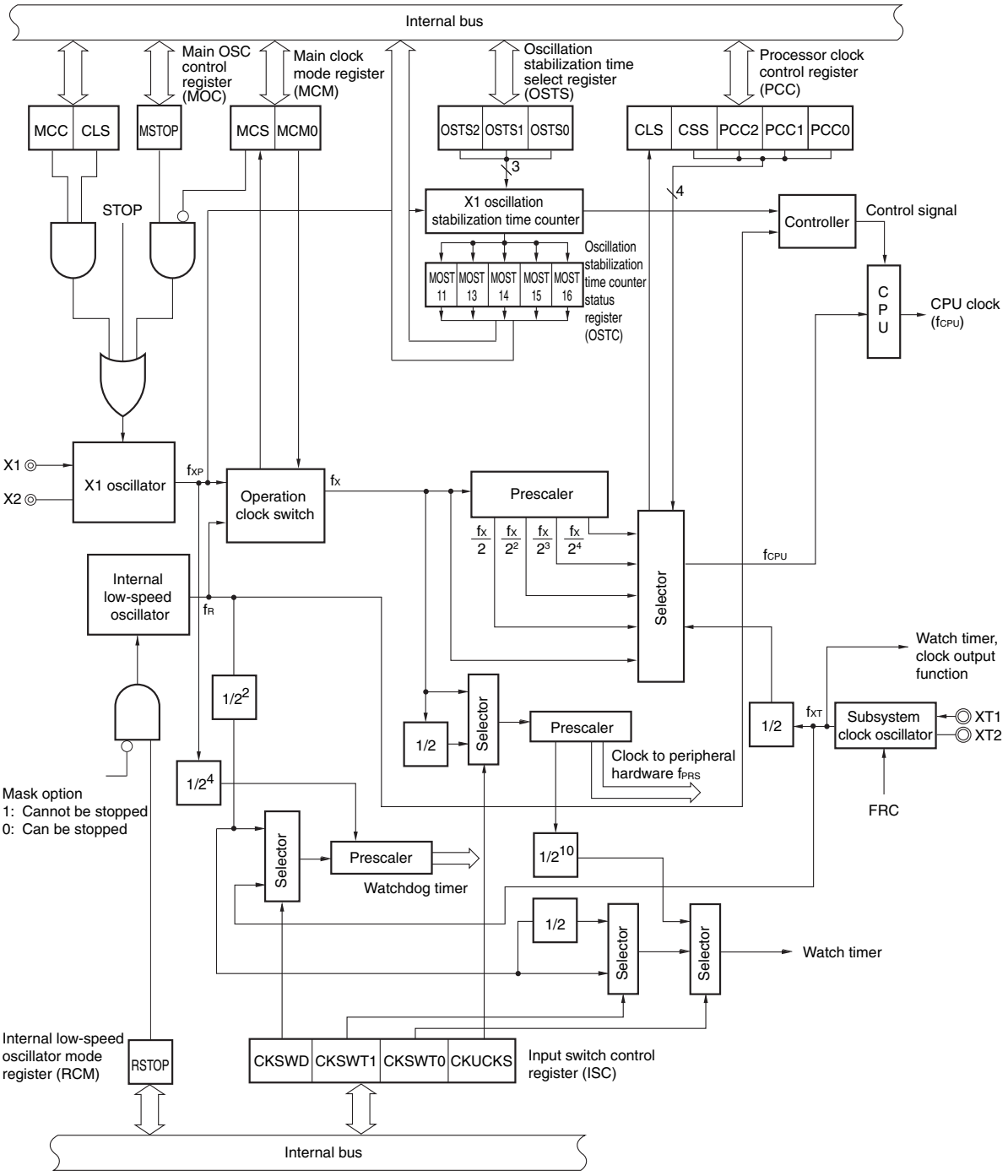
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1: Clock Generator Configuration

Item	Configuration
Control registers	Processor clock control register (PCC) Internal low-speed oscillator mode register (RCM) Main clock mode register (MCM) Main OSC control register (MOC) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time select register (OSTS) Input switch control register (ISC)
Oscillator	X1 oscillator Internal low-speed oscillator Subsystem clock oscillator

Figure 5-1: Block Diagram of Clock Generator



5.3 Clock Generator Control Register

The clock generator is controlled by the following control registers.

- Processor clock control register (PCC)
- Internal low-speed oscillator mode register (RCM)
- Main clock mode register (MCM)
- Main OSC control register (MOC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)
- Input switch control register (ISC)

(1) Processor clock control register (PCC)

The PCC register is used to select the CPU clock, the division ratio, main stem clock oscillator operation/stop and whether to use the on-chip feedback resistor[Note] of the subsystem clock oscillator.

The PCC is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears PCC to 00H.

Note: The feedback resistor is required to control the bias point of the oscillation waveform so that the bias point is in the middle of the power supply voltage. When the subsystem clock is not used, the operating current in the STOP mode can be reduced by setting bit 6 (FRC) of PCC to 1 (see **Figure 5-12, "Subsystem Clock Feedback Resistor," on page 141**).

Figure 5-2: Processor Control Register (PCC) Format (1/2)

	<7>	<6>	<5>	<4>	3	2	1	0	R/W	Address	After Reset
PCC	MCC	FRC	CLS	CSS	0	PCC2	PCC1	PCC0	R/W	FFFBH	00H[Note 1]

MCC	Control of X1 oscillator operation[Note 2]
0	Oscillation possible
1	Oscillation stopped

FRC	Subsystem clock feedback resistor selection
0	On-chip feedback resistor used
1	On-chip feedback resistor not used[Note 3]

CLS	CPU clock status
0	X1 input clock or Internal low-speed oscillator clock
1	Subsystem clock used

Figure 5-2: Processor Control Register (PCC) Format (2/2)

CSS[Note 4]	PCC2	PCC1	PCC0	CPU Clock (f_{CPU}) Selection		
					MCM0=0	MCM0=1
0	0	0	0	f_X	f_R	f_{XP}
	0	0	1	$f_X/2$	$f_R/2$	$f_{XP}/2$
	0	1	0	$f_X/2^2$	$f_R/2^2$	$f_{XP}/2^2$
	0	1	1	$f_X/2^3$	$f_R/2^3$	$f_{XP}/2^3$
	1	0	0	$f_X/2^4$	$f_R/2^4$	$f_{XP}/2^4$
1	0	0	0	$f_{XT}/2$		
	0	0	1			
	0	1	0			
	0	1	1			
	1	0	0			
Other than above				Setting prohibited		

Notes: 1. Bit 5 is read-only.

2. When the CPU is operating on the subsystem clock, MCC should be used to stop the X1 oscillator operation. When the CPU is operating on the internal low-speed oscillator clock, use bit 7 (MSTOP) of the main OSC control register (MOC) to stop the X1 oscillator operation (this cannot be set by MCC). A STOP instruction should not be used.

3. This bit can be set to 1 only when the subsystem clock is not used.

4. Be sure to switch CSS from 1 to 0 when bits 1 (MCS) and 0 (MCM0) of the main clock mode register (MCM) are 1.

Caution: Be sure to clear bit 3 to 0.

Remarks: 1. MCM0: Bit 0 of main clock mode register (MCM)

2. f_X: Main system clock oscillation frequency (X1 input clock or internal low-speed oscillator clock)

3. f_{XP}: X1 input clock oscillation frequency

4. f_R: Internal low-speed oscillator clock oscillation frequency

5. f_{XT}: Subsystem clock oscillation frequency.

In the µPD780822B Subseries the fastest instruction can be executed in 2 clocks of the CPU clock. Therefore, the relationship between the CPU clock (fCPU) and the minimum instruction execution time is as shown in Table 5-2.

Table 5-2: Relationship between CPU clock and Minimum Instruction Execution Time

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{CPU}$		
	X1 input Clock**Note** (at 16 MHz Operation)	Internal low-speed oscillator Clock**Note** (at 240 kHz (TYP.) Operation)	Subsystem Clock**Note** (at 32.768 kHz Operation)
f_X	125 ns	8.3 µs (TYP.)	-
$f_X/2$	250 ns	16.6 µs (TYP.)	-
$f_X/2^2$	500 ns	33.2 µs (TYP.)	-
$f_X/2^3$	1.0 µs	66.4 µs (TYP.)	-
$f_X/2^4$	2.0 µs	132.8 µs (TYP.)	-
$f_{XT}/2$	-	-	122.1 µs

Note: The main clock mode register (MCM) is used to set the CPU clock (X1 input clock/internal low-speed oscillator clock) (please refer to Figure 5-4, “Main Clock Mode Register (MCM) Format,” on page 130).

(2) Internal low-speed oscillator mode register (RCM)

This register sets the operation mode of internal low-speed oscillator.

This register is valid when “Can be stopped by software” is set for internal low-speed oscillator by a mask option, and the X1 input clock or subsystem clock is selected as the CPU clock. If “Cannot be stopped” is selected for internal low-speed oscillator by a mask option, settings for this register are invalid.

RCM can be set by an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 5-3: Internal Low-speed Oscillator Mode Register (RCM) Format

	7	6	5	4	3	2	1	<0>	R/W	Address	After Reset
RCM	0	0	0	0	0	0	0	RSTOP	R/W	FFA0H	00H

RSTOP	Control of internal low-speed oscillator operation
0	Internal low-speed oscillator oscillating
1	Internal low-speed oscillator stopped

Caution: Make sure that bit 1 (MCS) of the main clock mode register (MCM) is 1 before setting RSTOP.

(3) Main clock mode register (MCM)

This register sets the CPU clock (X1 input clock/internal low-speed oscillator clock).
MCM can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 5-4: Main Clock Mode Register (MCM) Format

	7	6	5	4	3	2	<1>	<0>	R/W	Address	After Reset
MCM	0	0	0	0	0	0	MCS[Note]	MCM0	R/W	FFA1H	00H

MCS	CPU clock status
0	Operates with internal low-speed oscillator clock
1	Operates with X1 input clock

MCM0	Selection of clock supplied to CPU
0	Internal low-speed oscillator clock
1	X1 input clock

Note: Bit 1 is read-only.

Cautions: 1. When internal low-speed oscillator is selected as the clock supplied to the CPU, the divided clock of the internal low-speed oscillator output (f_X) is supplied to the peripheral hardware (f_X = 240 kHz (TYP.)). Operation of the peripheral hardware with internal low-speed oscillator clock cannot be guaranteed. Therefore when internal low-speed oscillator is selected as the clock supplied to the CPU, do not use peripheral hardware. In addition, stop the peripheral hardware before switching the clock supplied to the CPU from the X1 input clock to the internal low-speed oscillator clock. Note, however, that the following peripheral hardware can be used, when the CPU operates on the internal low-speed oscillator clock.

- **Watchdog timer**
- **Watch timer**
- **Clock monitor**
- **Peripheral hardware selecting the external clock as the clock source (Except when the external count clock of TMPn (n = 0 to 2) and TM2 is selected.**

2. Set MCS = 1 and MCM0 = 1 before switching subsystem clock operation to X1 input clock operation (Bit 4 (CSS) of the processor clock control register (PCC) is changed from 1 to 0).

(4) Main OSC control register (MOC)

This register selects the operation mode of the X1 input clock.

This register is used to stop the X1 oscillator operation when the CPU is operating with the internal low-speed oscillator clock.

Therefore, this register is valid only when the CPU is operating with the internal low-speed oscillator clock.

MOC can be set by an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 5-5: Main OSC Control Register (MOC) Format

	<7>	6	5	4	3	2	<1>	<0>	R/W	Address	After Reset
MOC	MSTOP	0	0	0	0	0	0	0	R/W	FFA2H	00H

MSTOP	Control of X1 oscillator operation
0	Oscillator operating
1	Oscillator stopped

Cautions: 1. Make sure that bit 1 (MCS) of the main clock mode register (MCM) is 0 before setting MSTOP.

2. To stop X1 oscillation when the CPU is operating on the subsystem clock, set bit 7 (MCC) of the processor clock control register (PCC) to 1 (setting by MSTOP is not possible).

(5) Oscillation stabilization time counter status register (OSTC)

This is the status register of the X1 input clock oscillation stabilization time counter. If the internal low-speed oscillator clock is used as the CPU clock, the X1 input clock oscillation stabilization time can be checked.

OSTC can be read by a 1-bit or 8-bit memory manipulation instruction.

When reset is released (reset by $\overline{\text{RESET}}$ input, clock monitor, and WDT), the STOP instruction, MSTOP = 1, and MCC = 1 clear OSTC to 00H.

Figure 5-6: Oscillation Stabilization Time Counter Status Register (OSTC) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
OSTC	0	0	0	MOST11	MOST13	MOST14	MOST15	MOST16	R	FFA3H	00H

MOST11	MOST13	MOST14	MOST15	MOST16	Oscillation stabilization time status
1	0	0	0	0	$2^{11}/f_{XP}$ min. (128 μs min.)
1	1	0	0	0	$2^{13}/f_{XP}$ min. (512 μs min.)
1	1	1	0	0	$2^{14}/f_{XP}$ min. (1.024 ms min.)
1	1	1	1	0	$2^{15}/f_{XP}$ min. (2.048 ms min.)
1	1	1	1	1	$2^{16}/f_{XP}$ min. (4.096 ms min.)

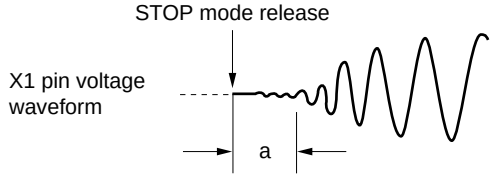
Cautions: 1. After the above time has elapsed, the bits are set to 1 in order from MOST11 and remain 1.

2. If the STOP mode is entered and then released while the internal low-speed oscillator clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- **Desired OSTC oscillation stabilization time equal or less than the Oscillation stabilization time set by OSTS**

The X1 oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

3. The wait time when STOP mode is released does not include the time after STOP mode release until clock oscillation starts ("a" below) regardless of whether STOP mode is released by $\overline{\text{RESET}}$ input or interrupt generation.



Remarks: 1. Values in parentheses are reference values for operation with f_{XP} = 16 MHz.

2. f_{XP}: X1 input clock oscillation frequency

(6) Oscillation stabilization time select register (OSTS)

This register is used to select the X1 oscillation stabilization wait time when STOP mode is released. The wait time set by OSTS is valid only after STOP mode is released with the X1 input clock selected as CPU clock. After STOP mode is released with internal low-speed oscillator selected as CPU clock, the oscillation stabilization time must be confirmed by OSTC.

OSTS can be set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets OSTS to 05H.

Figure 5-7: Oscillation Stabilization Time Select Register (OSTS) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	R/W	FFA4H	05H

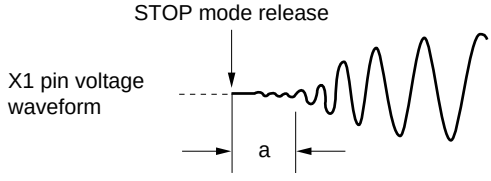
OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	$2^{11}/f_{XP}$ min. (128 µs min.)
0	0	1	$2^{13}/f_{XP}$ min. (512 µs min.)
0	1	1	$2^{14}/f_{XP}$ min. (1.024 ms min.)
1	0	0	$2^{15}/f_{XP}$ min. (2.048 ms min.)
1	0	1	$2^{16}/f_{XP}$ min. (4.096 ms min.)
Other than above			Setting prohibited

Cautions: 1. If the STOP mode is entered and then released while the internal low-speed oscillator clock is being used as the CPU clock, set the oscillation stabilization time as follows.

- **Desired OSTC oscillation stabilization time equal or less than the Oscillation stabilization time set by OSTS**

The X1 oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC after STOP mode is released.

2. The wait time when STOP mode is released does not include the time after STOP mode release until clock oscillation starts (“a” below) regardless of whether STOP mode is released by $\overline{\text{RESET}}$ input or interrupt generation.



Remarks: 1. Values in parentheses are reference values for operation with f_{XP} = 16 MHz.

2. f_{XP}: X1 input clock oscillation frequency

(7) Input switch control register (ISC)

This register is used to switch input signals, like status signals transmitted from a LIN master during LIN transmission, external interrupt INTP0 input signal selection, timer input clock selection for the 16-bit timers TMP0 and TM2, the watchdog timer and the watch timer.
ISC can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets ISC to 00H.

Figure 5-8: Input Switch Control Register (ISC) Format (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ISC	0	CKSWD	CKSWT1	CKSWT0	CKUCKS	CKCL	ISC1	ISC0	R/W	FF4AH	00H

CKSWD	Clock selection for Watchdog timer
0	$f_R/2^2$
1	f_{XT}

CKSWT1	CKSWT0	Clock selection for Watch timer
x	0	$f_X/2^{10}$
0	1	$f_R/2^2$
1	1	$f_R/2^3$

CKUCKS	Extended clock selection for Peripherals
0	$f_{PRS} = f_X$
1	$f_{PRS} = f_X/2$

CKCL	TIOP01 input source selection
0	TIOP01/P14
1	Watch timer interrupt INTWT

ISC1	TI20 input source selection
0	TI20/P00
1	RXD6/P12

Figure 5-8: Input Switch Control Register (ISC) Format (2/2)

ISC0	INTP0 input source selection
0	INTP0/P120
1	RXD6/P12

Note: Please refer to the according chapters to get more information regarding the use of the input switch control register (ISC).

Caution: When bit 3 (CKUCKS) is set to 1, the divided main system clock oscillation frequency $f_X/2$ (X1 input clock or internal low-speed oscillator clock) is supplied to the peripheral hardware.
Therefore, when internal low-speed oscillator is selected as the clock to be supplied to the CPU (main system clock oscillation frequency), the divided clock of the internal low-speed oscillator output is supplied to the peripheral hardware (f_X = 240 kHz / 2 = 120 kHz). Operation of the peripheral hardware with internal low-speed oscillator clock cannot be guaranteed. Note, however, that the following peripheral hardware can be used, when the CPU operates on the internal low-speed oscillator clock.

- **Watchdog timer**
- **Watch timer**
- **Clock monitor**
- **Peripheral hardware selecting the external clock as the clock source (Except when the external count clock of TMPn (n = 0 to 2) and TM2 is selected.**

When the X1 input clock is selected as the clock to be supplied to the CPU (main system clock oscillation frequency), the divided clock of the X1 input clock is supplied to the peripheral hardware (f_X = X1 input clock / 2).

Remarks: 1. f_X: Main system clock oscillation frequency (X1 input clock or internal low-speed oscillator clock)

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_{PRS}: Peripheral hardware clock oscillation frequency

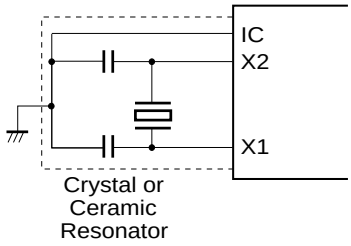
5.4 System Clock Oscillator

5.4.1 Main system clock oscillator

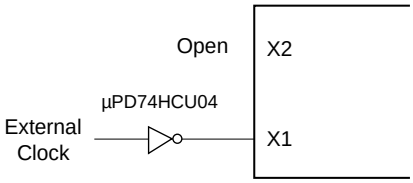
The main system clock oscillator oscillates with a crystal resonator or ceramic resonator (standard: 16 MHz) connected to the X1 and X2 pins.
An external clock can be input to the main system clock oscillator. In this case, the clock signal is supplied to the X1 pin and the X2 pin has to be left open.
Figure 5-9 shows examples of the external circuit of the X1 oscillator.

Figure 5-9: External circuit of Main System Clock Oscillator

(a) Crystal and ceramic oscillation



(b) External clock



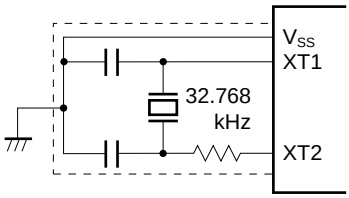
Caution: Do not execute the STOP instruction if an external clock is input. This is because when the STOP instruction is used the main system clock operation stops and the X2 pin is connected to V_{DD0} via a pull-up resistor.

5.4.2 Subsystem clock oscillator

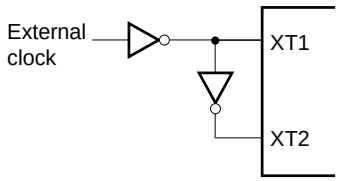
The subsystem clock oscillator oscillates with a crystal resonator (Standard: 32.768 kHz) connected to the XT1 and XT2 pins.
External clocks can be input to the subsystem clock oscillator. In this case, input the clock signal to the XT1 pin and the inverse signal to the XT2 pin.
Figure 5-10 shows examples of external circuit of the subsystem clock oscillator.

Figure 5-10: External circuit of Subsystem Clock Oscillator

(a) Crystal oscillation



(b) External clock



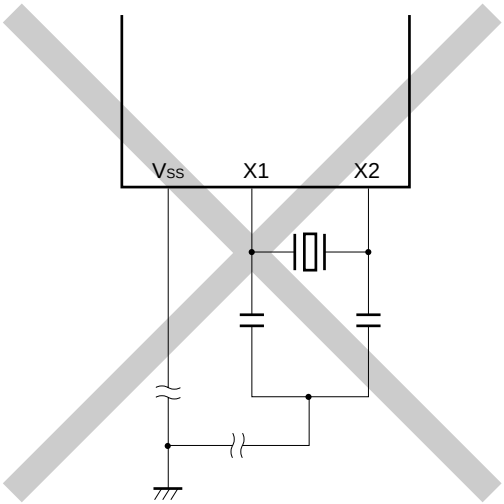
Caution: When using the X1 oscillator and subsystem clock oscillator, wire as follows in the area enclosed by the broken lines in the Figure 5-10 to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

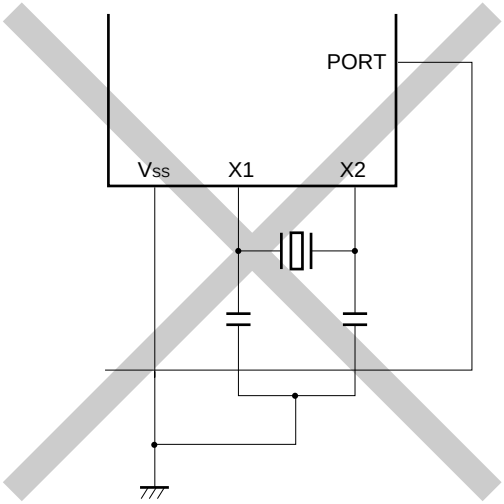
Note that the subsystem clock oscillator is designed as a low-amplitude circuit for reducing power consumption.

Figure 5-11: Examples of oscillator with Bad Connection (1/3)

(a) Wiring of connection circuits are too long



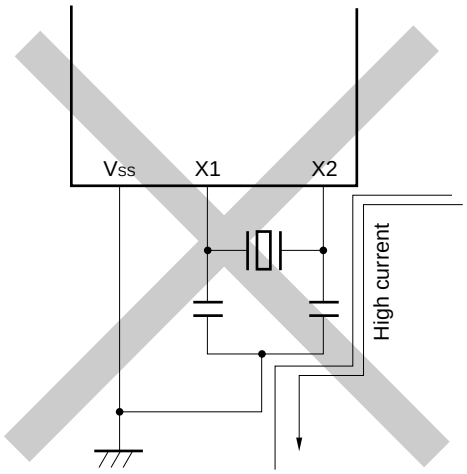
(b) A signal line crosses over oscillation circuit lines Crystal and ceramic oscillation



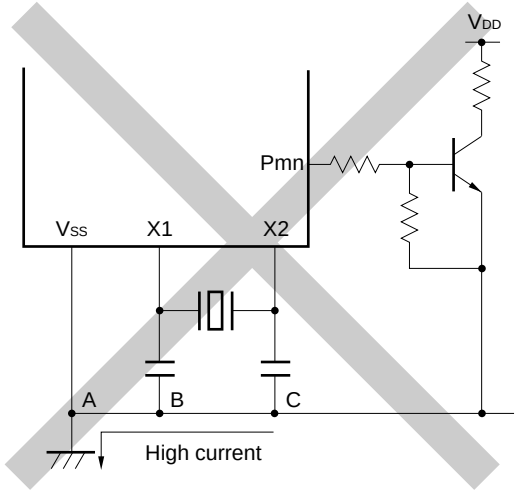
Remark: When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

Figure 5-11: Examples of oscillator with Bad Connection (2/3)

(c) Changing high current is too closed to a signal connector



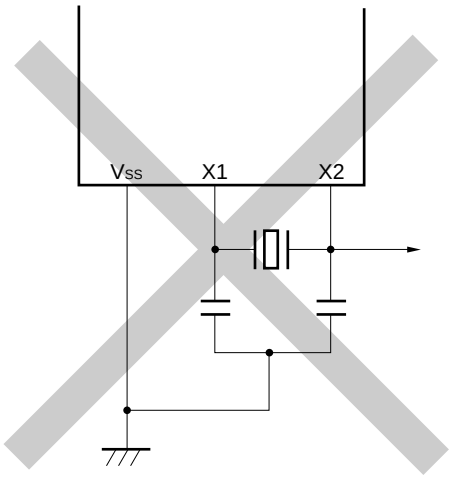
(d) (Current flows through the grounding line of the oscillator (potential at points A, B, and C fluctuate)



Remark: When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

Figure 5-11: Examples of oscillator with Bad Connection (3/3)

(e) Signals are fetched



Remark: When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

Caution: When X2 and XT1 are wired in parallel, the crosstalk noise of X2 may increase with XT1, resulting in malfunctioning.

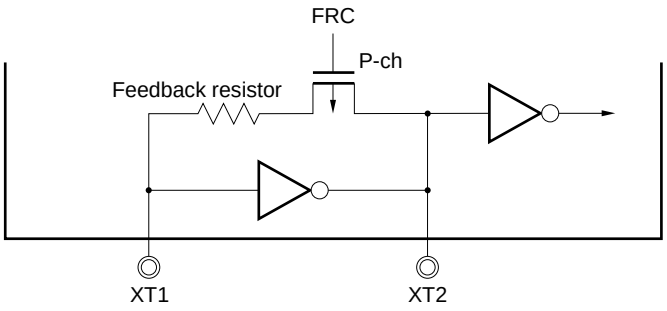
5.4.3 When subsystem clock is not used

If it is not necessary to use the subsystem clock for low power consumption operations and watch operations, connect the XT1 and XT2 pins as follows.

- XT1: Connect directly V_{DD}
- XT2: Leave open

In this state, however, some current may leak via the on-chip feedback resistor of the subsystem clock oscillator when the X1 input clock and internal low-speed oscillator clock stop. To minimize leakage current, the above on-chip feedback resistor can be set not to be used via bit 6 (FRC) of the processor clock control register (PCC). In this case also, connect the XT1 and XT2 pins as described above.

Figure 5-12: Subsystem Clock Feedback Resistor



Remark: The feedback resistor is required to control the bias point of the oscillation waveform so that the bias point is in the middle of the power supply voltage.

5.4.4 Internal low-speed oscillator

Internal low-speed oscillator is incorporated in the µPD780822B Subseries.
“Can be stopped by software” or “Cannot be stopped” can be selected by a mask option. The internal low-speed oscillator clock always oscillates after $\overline{\text{RESET}}$ release (240 kHz (TYP.)).

5.4.5 Prescaler

The prescaler generates various clocks by dividing the X1 oscillator output when the X1 input clock is selected as the clock to be supplied to the CPU.

Caution: When the internal low-speed oscillator clock is selected as the clock supplied to the CPU, the prescaler generates various clocks by dividing the internal low-speed oscillator output (f_X = 240 kHz (TYP.)).

5.5 Clock Generator Operations

The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode.

- X1 input clock f_{XP}
- Internal low-speed oscillator clock f_R
- Subsystem clock f_{XT}
- CPU clock f_{CPU}
- Clock to peripheral hardware

The CPU starts operation when the on-chip internal low-speed oscillator starts outputting after reset release in the µPD780822B Subseries, thus enabling the following.

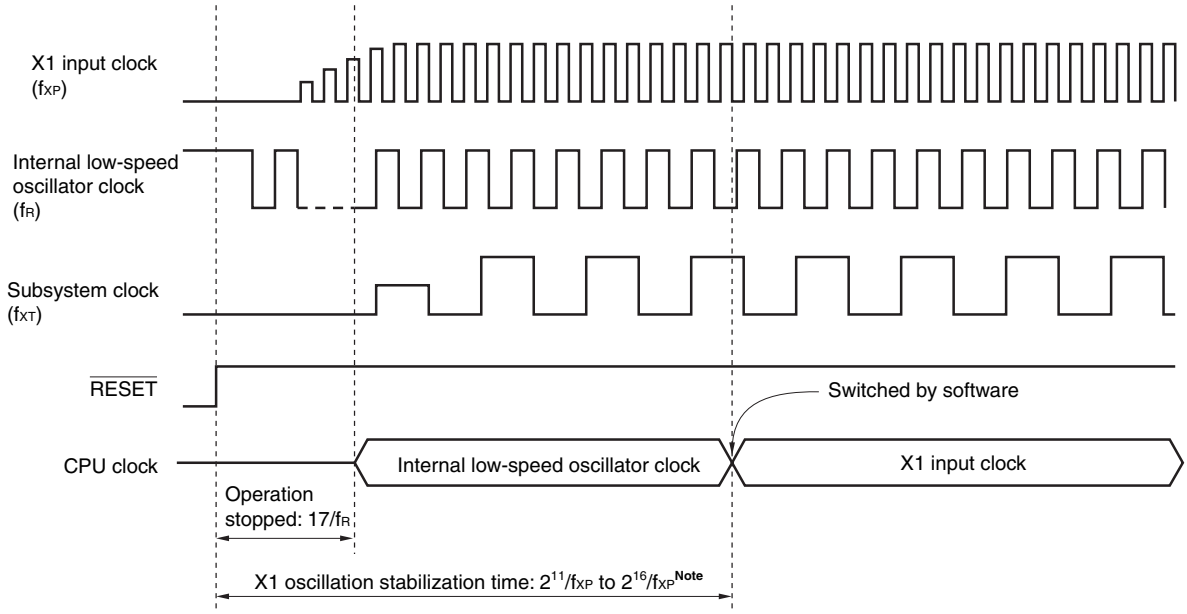
(1) Enhancement of security function

When the X1 input clock is set as the CPU clock by the default setting, the device cannot operate if the X1 input clock is damaged or badly connected and therefore does not operate after reset is released. However, the start clock of the CPU is the on-chip internal low-speed oscillator clock, so the device can be started by the internal low-speed oscillator clock after reset release by the clock monitor (detection of X1 input clock stop). Consequently, the system can be safely shut down by performing a minimum operation, such as acknowledging a reset source by software or performing safety processing when there is a malfunction.

(2) Improvement of performance

Because the CPU can be started without waiting for the X1 input clock oscillation stabilization time, the total performance can be improved.
A timing diagram of the CPU default start using internal low-speed oscillator is shown in Figure 5-13.

Figure 5-13: Timing Diagram of CPU Default Start Using Internal Low-speed Oscillator



Note: Check using the oscillation stabilization time counter status register (OSTC).

(a) When the $\overline{\text{RESET}}$ signal is generated, bit 0 of the main clock mode register (MCM) is cleared to 0 and the internal low-speed oscillator clock is set as the CPU clock. However, a clock is supplied to the CPU after 17 clocks of the internal low-speed oscillator clock have elapsed after $\overline{\text{RESET}}$ release (or clock supply to the CPU stops for 17 clocks). During the $\overline{\text{RESET}}$ period, oscillation of the X1 input clock and internal low-speed oscillator clock is stopped.

(b) After $\overline{\text{RESET}}$ release, the CPU clock can be switched from the internal low-speed oscillator clock to the X1 input clock using bit 0 (MCM0) of the main clock mode register (MCM) after the X1 input clock oscillation stabilization time has elapsed. At this time, check the oscillation stabilization time using the oscillation stabilization time counter status register (OSTC) before switching the CPU clock. The CPU clock status can be checked using bit 1 (MCS) of MCM.

(c) Internal low-speed oscillator can be set to stopped/oscillating using the internal low-speed oscillator mode register (RCM) when “Can be stopped by software” is selected for the internal low-speed oscillator by a mask option, if the X1 input or subsystem clock is used as the CPU clock. Make sure that MCS is 1 at this time.

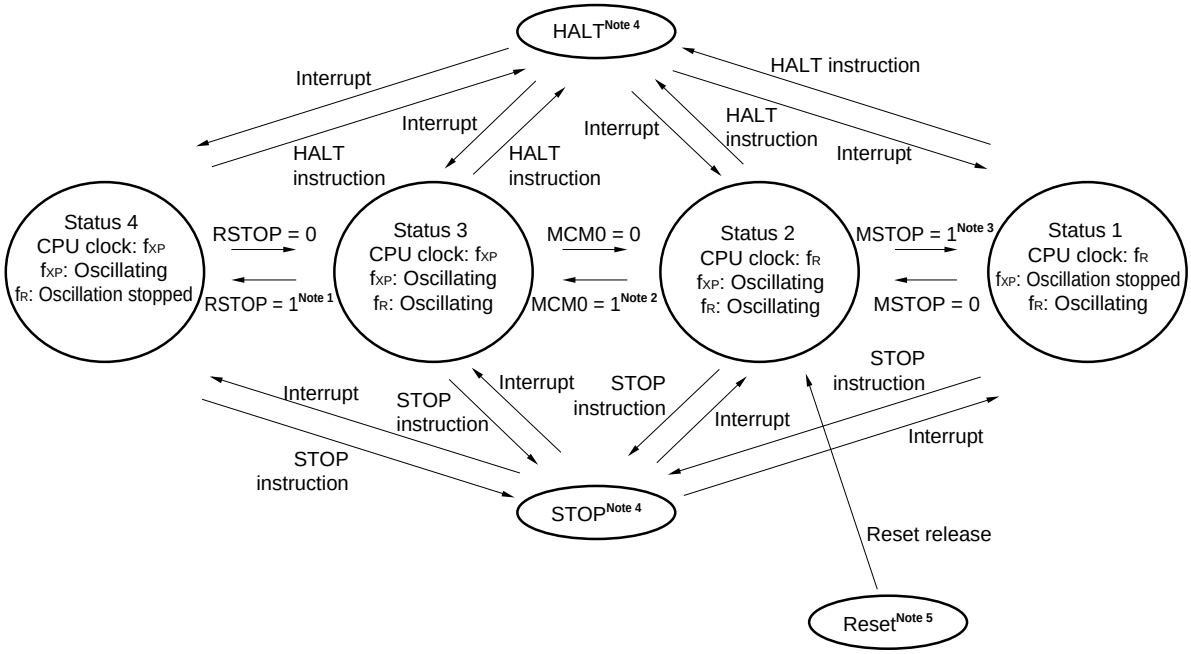
(d) When internal low-speed oscillator is used as the CPU clock, the X1 input clock can be set to stopped/oscillating using the main OSC control register (MOC). Make sure that MCS is 0 at this time. When the subsystem clock is used as the CPU clock, whether the X1 input clock stops or oscillates can be set by the processor clock control register (PCC). In addition, HALT mode can be used during operation with the subsystem clock, but STOP mode cannot be used (subsystem clock oscillation cannot be stopped by the STOP instruction).

(e) Select the X1 input clock oscillation stabilization time ($2^{11}/f_{XP}$, $2^{13}/f_{XP}$, $2^{14}/f_{XP}$, $2^{15}/f_{XP}$, $2^{16}/f_{XP}$) using the oscillation stabilization time select register (OSTS) when releasing STOP mode while X1 input clock is being used as the CPU clock. In addition, when releasing STOP mode while $\overline{\text{RESET}}$ is released and internal low-speed oscillator clock is being used as the CPU clock, check the X1 input clock oscillation stabilization time using the oscillation stabilization time counter status register (OSTC).

A status transition diagram of this product is shown in Figure 5-14, and the relationship between the operation clocks in each operation status and between the oscillation control flag and oscillation status of each clock are shown in Tables 5-3 and 5-4 on page 148, respectively.

Figure 5-14: Status Transition Diagram (1/4)

(a) When "Internal low-speed oscillator can be stopped by software" is selected by mask option (when subsystem clock is not used)



Notes: 1. When shifting from status 3 to status 4, make sure that bit 1 (MCS) of the main clock mode register (MCM) is 1.

2. Before shifting from status 2 to status 3 after reset and STOP are released, check the X1 input clock oscillation stabilization time status using the oscillation stabilization time counter status register (OSTC).

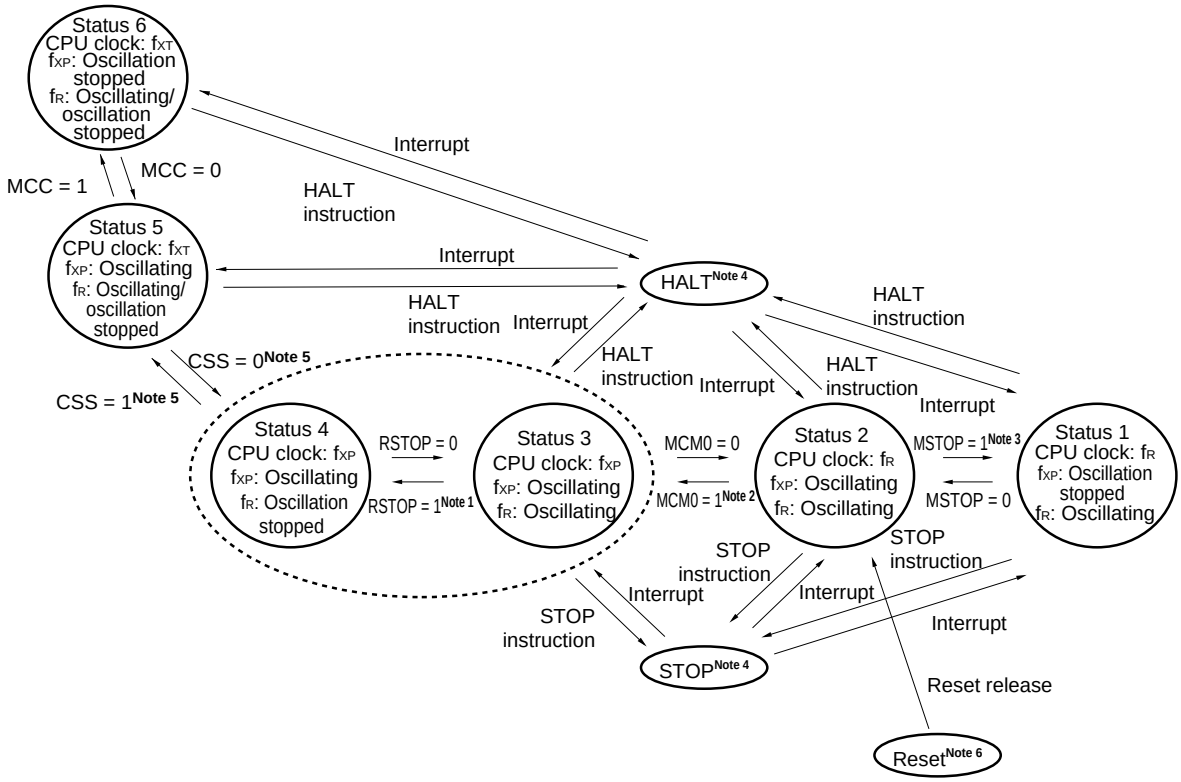
3. When shifting from status 2 to status 1, make sure that MCS is 0.

4. When "Internal low-speed oscillator can be stopped by software" is selected by a mask option, the watchdog timer stops operating in the HALT and STOP modes, regardless of the source clock of the watchdog timer. However, oscillation of internal low-speed oscillator does not stop even in the HALT and STOP modes if RSTOP = 0.

5. All reset sources ($\overline{\text{RESET}}$ input, clock monitor, and WDT).

Figure 5-14: Status Transition Diagram (2/4)

(b) When "Internal low-speed oscillator can be stopped by software" is selected by mask option (when subsystem clock is used)



Notes: 1. When shifting from status 3 to status 4, make sure that bit 1 (MCS) of the main clock mode register (MCM) is 1.

2. Before shifting from status 2 to status 3 after reset and STOP are released, check the X1 input clock oscillation stabilization time status using the oscillation stabilization time counter status register (OSTC).

3. When shifting from status 2 to status 1, make sure that MCS is 0.

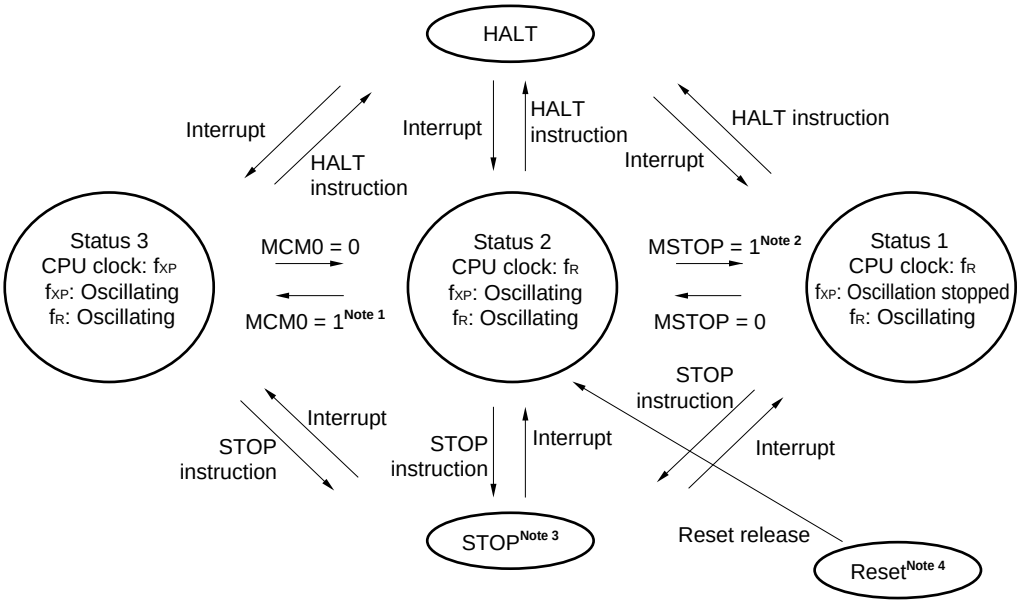
4. When "Internal low-speed oscillator can be stopped by software" is selected by a mask option, the clock supply to the watchdog timer is stopped after the HALT or STOP instruction has been executed, regardless of the setting of bit 0 (RSTOP) of the internal low-speed oscillator mode register (RCM) and bit 0 (MCM0) of the main clock mode register (MCM).

5. The operation cannot be shifted between subsystem clock operation and internal low-speed oscillator operation.

6. All reset sources ($\overline{\text{RESET}}$ input, clock monitor, and WDT)

Figure 5-14: Status Transition Diagram (3/4)

(c) When "Internal low-speed oscillator cannot be stopped" is selected by mask option (when subsystem clock is not used)



Notes: 1. Before shifting from status 2 to status 3 after reset and STOP are released, check the X1 input clock oscillation stabilization time status using the oscillation stabilization time counter status register (OSTC).

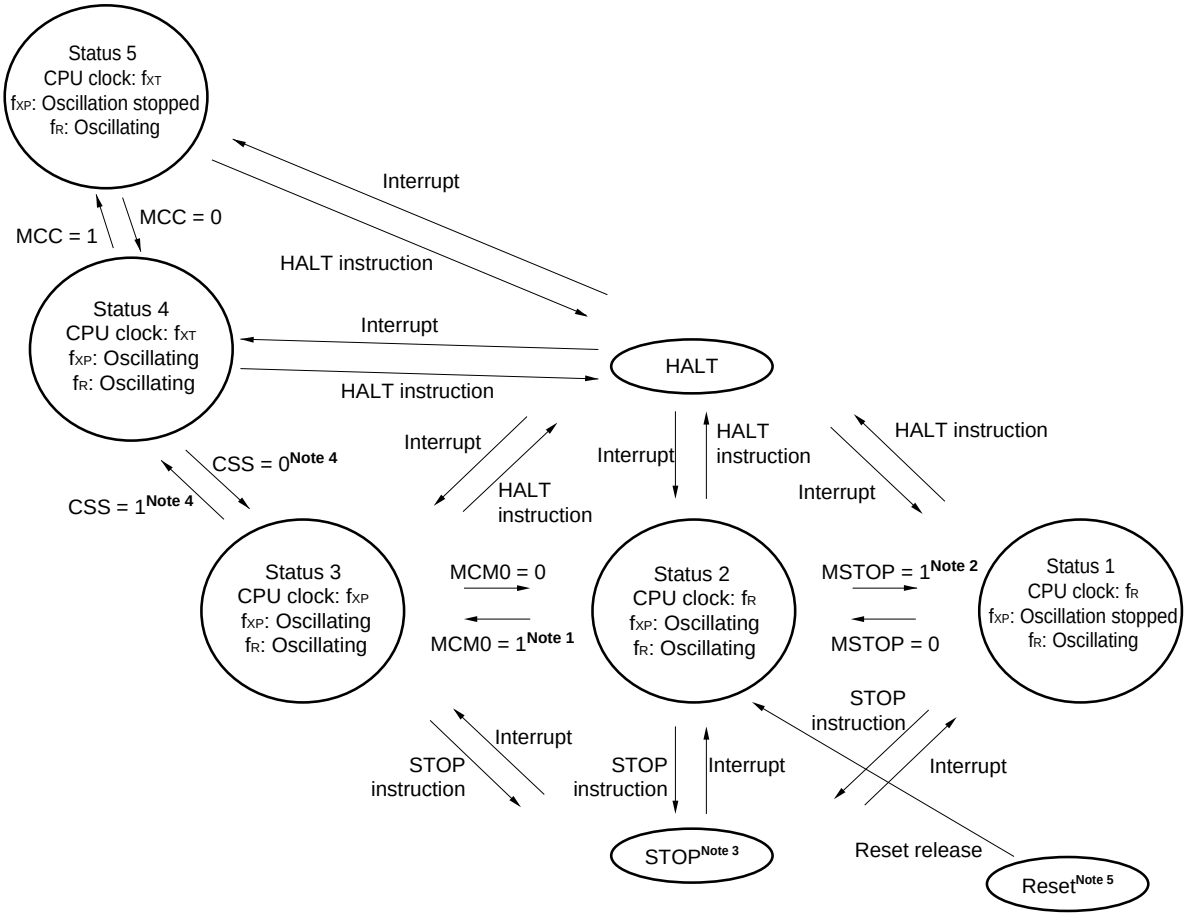
2. When shifting from status 2 to status 1, make sure that MCS is 0.

3. The watchdog timer operates using internal low-speed oscillator even in STOP mode if "Internal low-speed oscillator cannot be stopped" is selected by a mask option.

4. All reset sources ($\overline{\text{RESET}}$ input, clock monitor, and WDT)

Figure 5-14: Status Transition Diagram (4/4)

(d) When "Internal low-speed oscillator cannot be stopped" is selected by mask option (when subsystem clock is used)



Notes: 1. Before shifting from status 2 to status 3 after reset and STOP are released, check the X1 input clock oscillation stabilization time status using the oscillation stabilization time counter status register (OSTC).

2. When shifting from status 2 to status 1, make sure that MCS is 0.

3. The watchdog timer operates using internal low-speed oscillator even in STOP mode if "Internal low-speed oscillator cannot be stopped" is selected by a mask option. Internal low-speed oscillator division can be selected as the count source of 8-bit timer H1 (TMH1), so clear the watchdog timer using the TMH1 interrupt request before watchdog timer overflow. If this processing is not performed, an internal reset signal is generated at watchdog timer overflow after STOP instruction execution.

4. The operation cannot be shifted between subsystem clock operation and internal low-speed oscillator operation.

5. All reset sources ($\overline{\text{RESET}}$ input, clock monitor, and WDT)

Table 5-3: Relationship Between Operation Clocks in Each Operation Status

Status	X1 Oscillator		Internal Low-speed Oscillator			Subsystem Clock Oscillator	CPU Clock After Release	Prescaler Clock Supplied to Peripherals	
Operation Mode	MSTOP=0 MCC = 0	MSTOP=1 MCC = 1	Note 1	Note 2				MCM0=0	MCM0=1
				RSTOP=0	RSTOP=1				
RESET	Stopped		Stopped			Oscillating	Internal low-speed oscillator	Stopped	
STOP			Oscillating	Oscillating	Stopped		Note 3		
HALT	Oscillating	Stopped					Note 4	Internal low-speed oscillator	X1

Notes: 1. When “Cannot be stopped” is selected for internal low-speed oscillator by a mask option.

2. When “Can be stopped by software” is selected for internal low-speed oscillator by a mask option.

3. Operates using the CPU clock at STOP instruction execution.

4. Operates using the CPU clock at HALT instruction execution.

Caution: The RSTOP setting is valid only when “Can be stopped by software” is set for internal low-speed oscillator by a mask option.

Remarks: 1. MSTOP: Bit 7 of the main OSC control register (MOC)

2. MCC: Bit 7 of the processor clock control register (PCC)

3. RSTOP: Bit 0 of the internal low-speed oscillator mode register (RCM)

4. MCM0: Bit 0 of the main clock mode register (MCM)

Table 5-4: Oscillation Control Flags and Clock Oscillation Status

		X1 Oscillator	Internal Low-speed Oscillator
MSTOP = 1Note	RSTOP = 0	Stopped	Oscillating
	RSTOP = 1	Setting prohibited	
MSTOP = 0Note	RSTOP = 0	Oscillating	Oscillating
	RSTOP = 1		Stopped
MCC = 1Note	RSTOP = 0	Stopped	Oscillating
	RSTOP = 1		Stopped
MCC = 0Note	RSTOP = 0	Oscillating	Oscillating
	RSTOP = 1		Stopped

Note: Setting X1 oscillator oscillating/stopped differs depending on the CPU clock used.

- When the internal low-speed oscillator clock is used as the CPU clock: Set using the MSTOP bit
- When the subsystem clock is used as the CPU clock: Set using the MCC bit

Caution: The RSTOP setting is valid only when “Can be stopped by software” is set for internal low-speed oscillator by a mask option.

Remarks: 1. MSTOP: Bit 7 of the main OSC control register (MOC)

2. MCC: Bit 7 of the processor clock control register (PCC)

3. RSTOP: Bit 0 of the internal low-speed oscillator mode register (RCM)

5.6 Time Required to Switch Between Internal Low-speed Oscillator Clock and X1 Input Clock

Bit 0 (MCM0) of the main clock mode register (MCM) is used to switch between the internal low-speed oscillator clock and X1 input clock.
In the actual switching operation, switching does not occur immediately after MCM0 rewrite; several instructions are executed using the pre-switch clock after switching MCM0 (see Table 5-5).
Bit 1 (MCS) of MCM is used to judge that operation is performed using either the internal low-speed oscillator clock or X1 input clock. To stop the original clock after switching the clock, wait for the number of clocks shown in Table 5-5.

Table 5-5: Maximum Time Required to Switch Between Internal Low-speed Oscillator Clock and X1 Input Clock

PCC			Time Required for Switching	
PCC2	PCC1	PCC0	X1→internal low-speed oscillator	internal low-speed oscillator→X1
0	0	0	f_{XP}/f_R + 1 clock	2 clocks
0	0	1	$f_{XP}/2f_R$ + 1 clock	
0	1	0	$f_{XP}/4f_R$ + 1 clock	
0	1	1	$f_{XP}/8f_R$ + 1 clock	
1	0	0	$f_{XP}/16f_R$ + 1 clock	

Caution: To calculate the maximum time, set f_R = 120 kHz.

Remarks: 1. PCC: Processor clock control register

2. f_{XP}: X1 input clock oscillation frequency

3. f_R: Internal low-speed oscillator clock oscillation frequency

4. The maximum time is the number of clocks of the CPU clock before switching.

5.7 Time Required for CPU Clock Switchover

The CPU clock can be switched using bits 0 to 2 (PCC0 to PCC2) and bit 4 (CSS) of the processor clock control register (PCC).
The actual switchover operation is not performed immediately after rewriting to the PCC; operation continues on the pre-switchover clock for several instructions (see Table 5-6).
Whether the system is operating on the X1 input clock (or internal low-speed oscillator clock) or the subsystem clock can be ascertained using bit 5 (CLS) of the PCC register.

Table 5-6: Maximum Time Required for CPU Clock Switchover

Set Value Before Switchover				Set Value After Switchover																							
CSS	PCC2	PCC1	PCC0	CSS	PCC2	PCC1	PCC0	CSS	PCC2	PCC1	PCC0	CSS	PCC2	PCC1	PCC0	CSS	PCC2	PCC1	PCC0	CSS	PCC2	PCC1	PCC0	CSS	PCC2	PCC1	PCC0
				0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0	1	×	×	×
0	0	0	0					16 clocks				16 clocks				16 clocks				16 clocks				f_{XP}/f_{XT} clocks (489 clocks)			
	0	0	1	8 clocks								8 clocks				8 clocks				8 clocks				$f_{XP}/2f_{XT}$ clocks (245 clocks)			
	0	1	0	4 clocks				4 clocks								4 clocks				4 clocks				$f_{XP}/4f_{XT}$ clocks (123 clocks)			
	0	1	1	2 clocks				2 clocks				2 clocks								2 clocks				$f_{XP}/8f_{XT}$ clocks (62 clocks)			
	1	0	0	1 clock				1 clock				1 clock				1 clock								$f_{XP}/16f_{XT}$ clocks (31 clocks)			
1	×	×	×	1 clock				1 clock				1 clock				1 clock				1 clock							

Remarks: 1. The maximum time is the number of clocks of the CPU clock before switching.

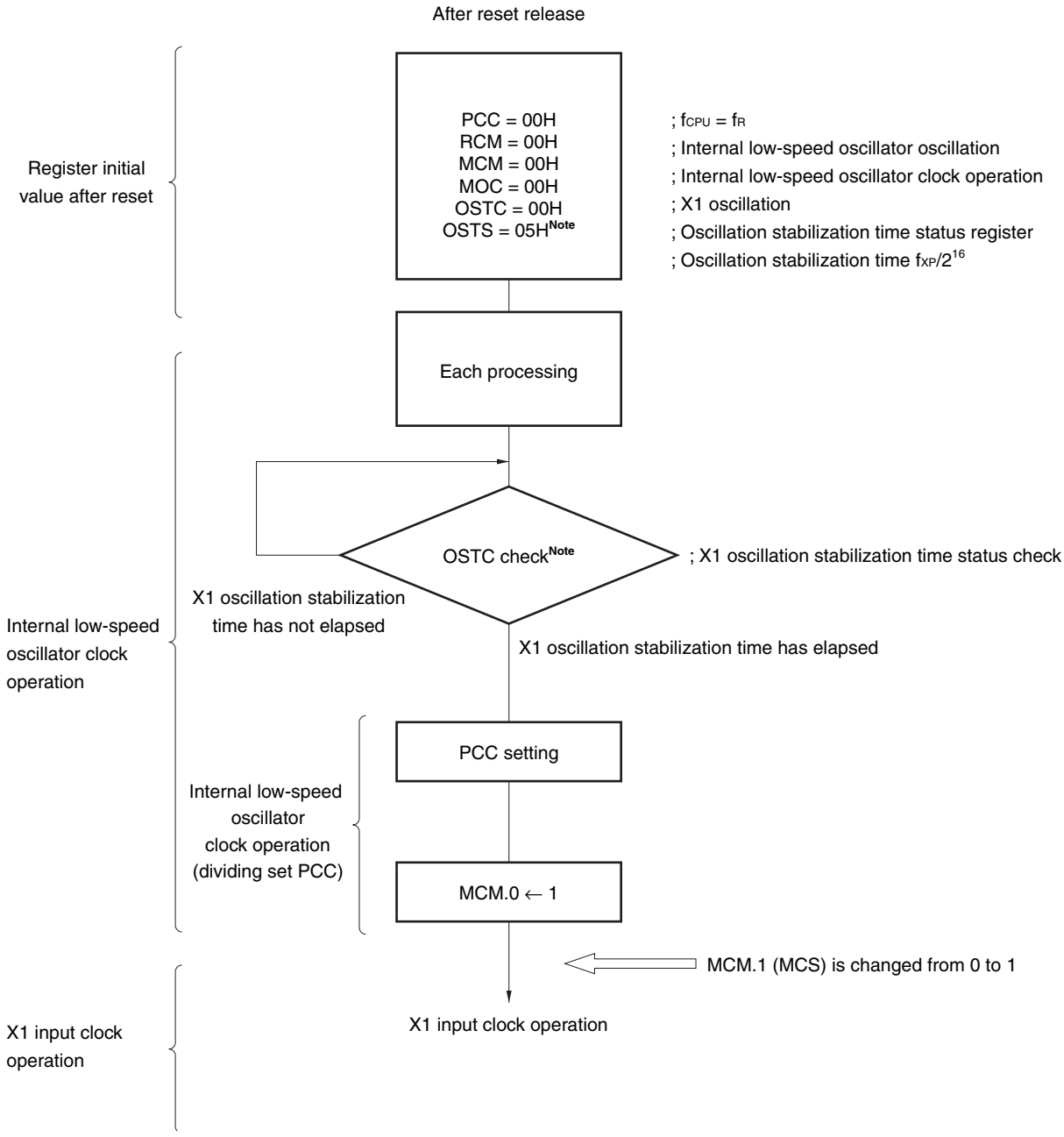
2. Figures in parentheses apply to operation with f_{XP} = 16 MHz and f_{XT} = 32.768 kHz.

Caution: Selection of the CPU clock cycle division factor (PCC0 to PCC2) and switchover from the X1 input clock to the subsystem clock (changing CSS from 0 to 1) should not be set simultaneously.
Simultaneous setting is possible, however, for selection of the CPU clock cycle division factor (PCC0 to PCC2) and switchover from the subsystem clock to the X1 input clock (changing CSS from 1 to 0).

5.8 Clock Switching Flowchart and Register Setting

5.8.1 Switching from internal low-speed oscillator clock to X1 input clock

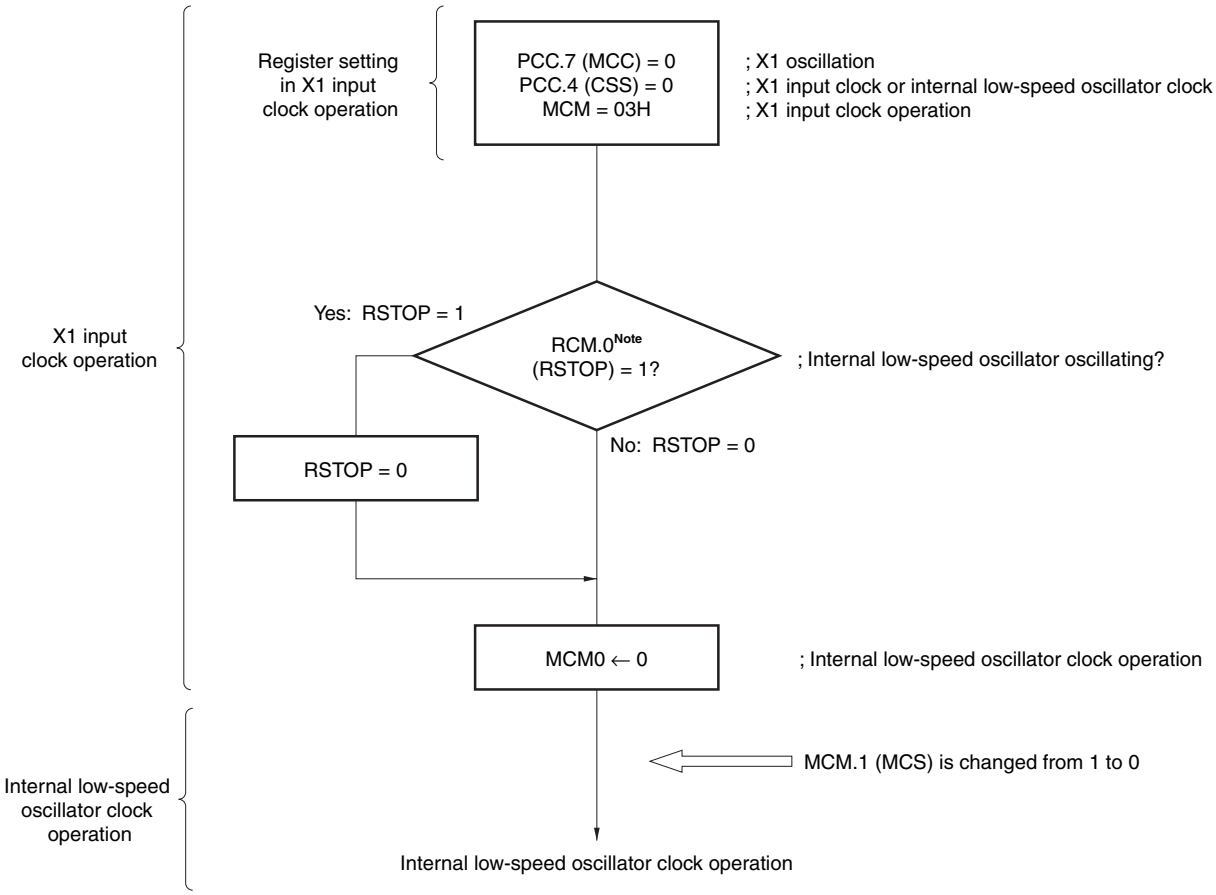
Figure 5-15: Switching from Internal Low-speed Oscillator Clock to X1 Input Clock (Flowchart)



Note: Check the oscillation stabilization wait time of the X1 oscillator after reset release using the OSTC register and then switch to the X1 input clock operation after the oscillation stabilization wait time has elapsed. The OSTS register setting is valid only after STOP mode is released by interrupt during X1 input clock operation.

5.8.2 Switching from X1 input clock to internal low-speed oscillator clock

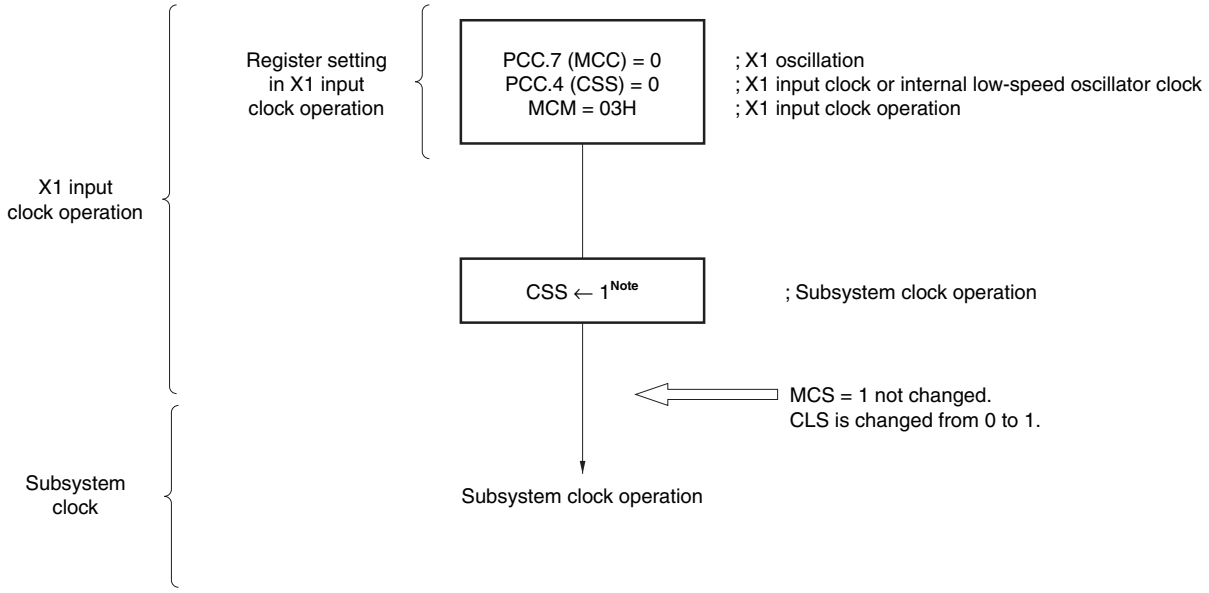
Figure 5-16: Switching from X1 Input Clock to Internal Low-speed Oscillator Clock (Flowchart)



Note: Required only when “clock can be stopped by software” is selected for internal low-speed oscillator by a mask option.

5.8.3 Switching from X1 input clock to subsystem clock

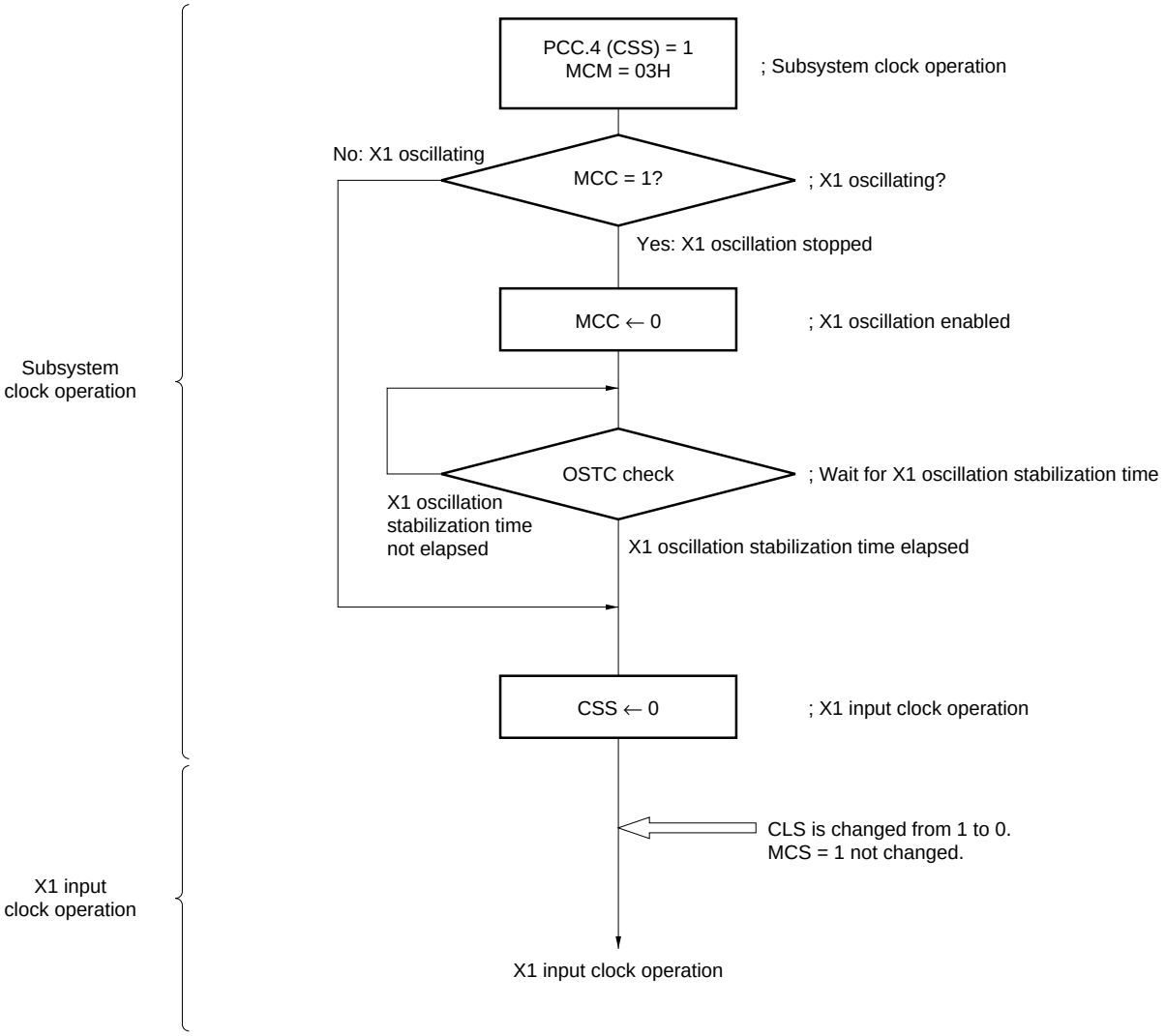
Figure 5-17: Switching from X1 Input Clock to Subsystem Clock (Flowchart)



Note: Set CSS to 1 after confirming that oscillation of the subsystem clock is stabilized.

5.8.4 Switching from subsystem clock to X1 input clock

Figure 5-18: Switching from Subsystem Clock to X1 Input Clock (Flowchart)



5.8.5 Register settings

The table below shows the states of the setting flags and status flags when each mode is set.

Table 5-7: Clock and Register Setting

f_{CPU}	Mode	Setting Flag					Status Flag	
		PCC register		MCM register	MOC register	RCM register	PCC register	MCM register
		MCC	CSS	MCM0	MSTOP	RSTOP **Note 1**	CLS	MCS
X1 input clock **Note 2**	internal low-speed oscillator oscillating	0	0	1	0	0	0	1
	internal low-speed oscillator stopped	0	0	1	0	1	0	1
Internal low-speed oscillator clock	X1 oscillating	0	0	0	0	0	0	0
	X1 stopped	0 **Note 3**	0	0	1	0	0	0
Subsystem clock **Note 4**	X1 oscillating, internal low-speed oscillator oscillating	0	1	1 **Note 5**	0 **Note 6**	0	1	1
	X1 stopped, internal low-speed oscillator oscillating	1	1	1 **Note 5**	0 **Note 6**	0	1	1
	X1 oscillating, internal low-speed oscillator stopped	0	1	1 **Note 5**	0 **Note 6**	1	1	1
	X1 stopped, internal low-speed oscillator stopped	1	1	1 **Note 5**	0 **Note 6**	1	1	1

Notes: 1. Valid only when “clock can be stopped by software” is selected for internal low-speed oscillator by a mask option.

2. Do not set MCC = 1 or MSTOP = 1 during X1 input clock operation (even if MCC = 1 or MSTOP = 1 is set, the X1 oscillation does not stop).

3. Do not set MCC = 1 during internal low-speed oscillator operation (even if MCC = 1 is set, the X1 oscillation does not stop). To stop X1 oscillation during internal low-speed oscillator operation, use MSTOP.

4. Shifting to subsystem clock operation mode must be performed from the X1 input clock operation mode. From subsystem clock operation mode, only X1 input clock operation mode can be shifted to.

5. Do not set MCM0 = 0 (shifting to internal low-speed oscillator) during subsystem clock operation.

6. Do not set MSTOP = 1 during subsystem clock operation (even if MSTOP = 1 is set, X1 oscillation does not stop). To stop X1 oscillation during subsystem clock operation, use MCC.

[MEMO]

Chapter 6 16-Bit Timer 2 (μPD780822B Subseries)

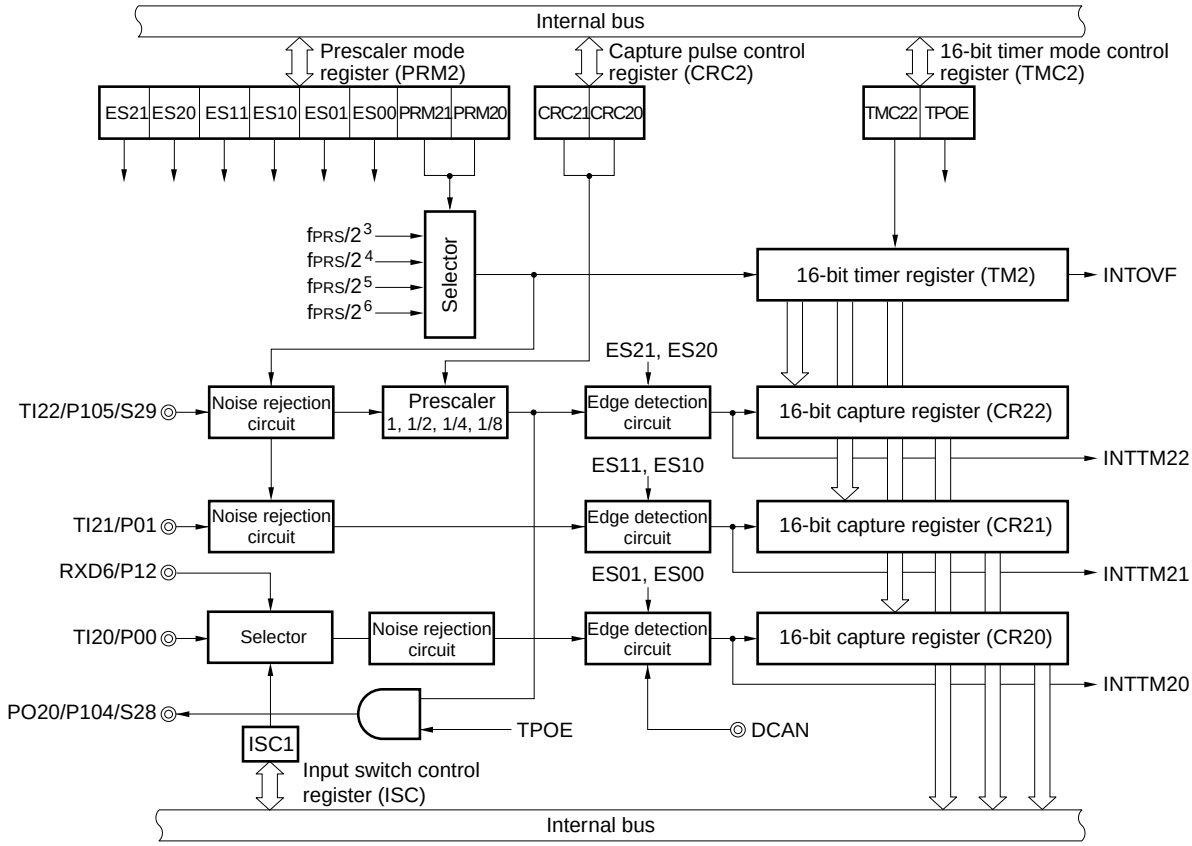
6.1 16-Bit Timer 2 Functions

The 16-bit timer 2 (TM2) has the following functions.

- Pulse width measurement
- Divided output of input pulse
- Time stamp function for the DCAN

Figure 6-1 shows 16-bit Timer 2 block diagram.

Figure 6-1: Timer 2 (TM2) Block Diagram



(1) Pulse width measurement

TM2 can measure the pulse width of an external input signal.

(2) Divided output of input pulse

The frequency of an input signal can be divided and the divided signal can be output.

(3) Timer stamp function for the DCAN

An internal signal output of the DCAN-module can be used to build a time stamp function of the system (please refer to the chapter of the DCAN-module).

(4) Capture input function for LIN

The RDX6 input signal of the UART6 can be used to calculate the baudrate error by using the Sync-Field sent by a LIN master (please refer to the **Chapter 14 ”Serial Interface UART6 (µPD780822B Subseries)” on page 315**).

6.2 16-Bit Timer 2 Configuration

Timer 2 consists of the following hardware.

Table 6-1: Timer 2 Configuration

Item	Configuration
Control register	16 bit timer mode control register (TMC2)
	Prescaler mode register (PRM2)
	Capture pulse control register (CRC2)
	Input switch control register (ISC)
Count register	16 bits × 1 (TM2)
Capture registers	16 bits × 3: CR20, CR21 and CR22

(1) 16-bit timer register (TM2)

TM2 is a 16-bit read-only register that counts count pulses.
The counter is incremented in synchronization with the rising edge of an input clock. The count value is reset to 0000H in the following case:

At $\overline{\text{RESET}}$ input

The count value is undefined in the following case:
- TMC22 is disabled.

Caution: When the timer TM2 is disabled, the value of the timer register TM2 will be undefined.

(2) Capture register 20 (CR20)

The valid edge of the TI20 pin can be selected as the capture trigger. Setting of the TI20 valid edge is performed by setting of the prescaler mode register (PRM2). When the valid edge of the TI20 is detected, an interrupt request (INTTM20) is generated.
CR20 is read by a 16-bit memory manipulation instruction.

After $\overline{\text{RESET}}$ input, the value of CR20 is undefined.

(3) Capture register 21 (CR21)

The valid edge of the TI21 pin can be selected as the capture trigger. Setting of the TI21 valid edge is performed by setting of the prescaler mode register (PRM2). When the valid edge of the TI21 is detected, an interrupt request (INTTM21) is generated.
CR21 is read by a 16-bit memory manipulation instruction.

After $\overline{\text{RESET}}$ input, the value of CR21 is undefined.

(4) Capture register 22 (CR22)

The valid edge of the TI22 pin can be selected as the capture trigger. Setting of the TI22 valid edge is performed by setting of the prescaler mode register (PRM2). When the valid edge of the TI22 is detected, an interrupt request (INTTM22) is generated.
CR22 is read by a 16-bit memory manipulation instruction.

After $\overline{\text{RESET}}$ input, the value of CR22 is undefined.

(5) Input switch control register (ISC)

The input source TI20/P00 pin or RXD6/P12 pin for the valid edge of the capture trigger can be selected by setting bit 1 (ISC1) of the input switch control register (ISC).
ISC is set by a 1-bit or 8-bit memory manipulation instruction.

After $\overline{\text{RESET}}$ input, the value of ISC is 00H.

6.3 16-Bit Timer 2 Control Registers

The following four types of registers are used to control timer 0.

- 16-bit timer mode control register (TMC2)
- Capture pulse control register (CRC2)
- Prescaler mode register (PRM2)
- Input switch control register (ISC)

(1) 16-bit timer mode control register (TMC2)

This register sets the 16-bit timer operating mode and controls the prescaler output signals. TMC0 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC2 value to 00H.

Figure 6-2: 16-Bit Timer Mode Control Register (TMC2) Format

	7	6	5	4	3	<2>	1	<0>	R/W	Address	After Reset
TMC2	0	0	0	0	0	TMC22	0	TPOE	R/W	FF65H	00H

TMC22	Timer 2 Operating Mode Selection
0	Operation stop
1	Operation enabled

TPOE	Timer 2 Prescaler Output Control
0	Prescaler signal output disabled
1	Prescaler signal output enabled

Cautions: 1. Before changing the operation mode, stop the timer operation (by setting 0 to TMC22).

2. Bit 1 and bits 3 to 7 must be set to 0.

(2) Capture pulse control register (CRC2)

This register specifies the division ratio of the capture pulse input to the 16-bit capture register (CR22) from an external source.
CRC2 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CRC2 value to 00H.

Figure 6-3: Capture Pulse Control Register (CRC2) Format

	7	6	5	4	3	2	<1>	<0>	R/W	Address	After Reset
CRC2	0	0	0	0	0	0	CRC21	CRC20	R/W	FF67H	00H

CRC21	CRC20	TI22 - Capture Pulse Selection
0	0	Does not divide capture pulse (TI22)
0	1	Divides capture pulse by 2 (TI22/2)
1	0	Divides capture pulse by 4 (TI22/4)
1	1	Divides capture pulse by 8 (TI22/8)

Cautions: 1. Timer operation must be stopped before setting CRC2.

2. Bits 2 to 7 must be set to 0.

(3) Prescaler mode register (PRM2)

This register is used to set 16-bit timer (TM2) count clock and valid edge of the capture signal TI2n (n = 0 to 2) input.
PRM2 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PRM2 value to 00H.

Figure 6-4: Prescaler Mode Register (PRM2) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PRM2	ES21	ES20	ES11	ES10	ES01	ES00	PRM21	PRM20	R/W	FF66H	00H

ES21	ES20	TI22 Valid Edge Selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

ES11	ES10	TI21 Valid Edge Selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

ES01	ES00	TI20 Valid Edge Selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

PRM21	PRM20	Count Clock Selection
0	0	$f_{PRS}/2^3$
0	1	$f_{PRS}/2^4$
1	0	$f_{PRS}/2^5$
1	1	$f_{PRS}/2^6$

Caution: Timer operation must be stopped before setting PRM2.

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

(4) Input switch control register (ISC)

This register specifies the external input source of the capture pulse input to the 16-bit capture register (CR20).
ISC is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ISC value to 00H.

Figure 6-5: Input Switch Control Register (ISC) Format

	7	6	5	4	3	2	<1>	<0>	R/W	Address	After Reset
ISC	0	CKSWD	CKSWT1	CKSWT0	CKUCKS	CKCL	ISC1	ISC0	R/W	FF4AH	00H

ISC1	TI20 Input Source Selection
0	TI20/P00
1	RXD6/P12

Remark: In this part only the ISC1 selection feature is described. The input switch control register has more selection features. Please refer to the **Figure 5-8, "Input Switch Control Register (ISC) Format (1/2)," on page 134**.

Cautions: 1. Timer operation must be stopped before setting CRC2.

2. Bit 7 must be set to 0.

(5) 16-bit timer register (TM2)

TM2 is a 16-bit read-only register that counts count pulses.
The counter TM2 is free-running and incremented in synchronization with the rising edge of the input clock. When the TM2 value overflows from FFFFH to 0000H, the INTOVF interrupt is generated. The count value is reset to 0000H when the $\overline{\text{RESET}}$ is released.
The count value is undefined when the timer is stopped (TMC22=0).

Caution: When the timer TM2 is disabled, the value of the timer register TM2 will be undefined.

(6) Capture register 20 (CR20)

When a valid edge is detected at the TI20 pin, the current value of the count register TM2 is copied into the capture register CR20, and an interrupt request (INTTM20) is generated.
The valid edge of the TI20 pin can be selected by setting the bits ES00 and ES01 of the prescaler mode register (PRM2). Sampling is performed at the count clock selected by PRM2, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short pulse width.
CR20 is read by a 16-bit memory manipulation instruction.
After $\overline{\text{RESET}}$ input, the value of CR20 is undefined.

(7) Capture register 21 (CR21)

When a valid edge is detected at the TI21 pin, the current value of the count register TM2 is copied into the capture register CR21, and an interrupt request (INTTM21) is generated.
The valid edge of the TI21 pin can be selected by setting the bits ES10 and ES11 of the prescaler mode register (PRM2). Sampling is performed at the count clock selected by PRM2, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short pulse width.
CR21 is read by a 16-bit memory manipulation instruction.
After $\overline{\text{RESET}}$ input, the value of CR21 is undefined.

(8) Capture register 22 (CR22)

The valid edges frequency at the TI22 pin is divided in a counter and the counter overflow is used as capture trigger signal: the current value of the count register TM2 is copied into the capture register CR22, and an interrupt request (INTTM22) is generated.
The valid edge of the TI22 pin can be selected by setting the bits ES21 and ES22 of the prescaler mode register (PRM2). Sampling is performed at the count clock selected by PRM2, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short pulse width.
The division ratio of the counter can be selected by setting the bits CRC21 and CRC20 of the capture control register CRC2.
CR22 is read by a 16-bit memory manipulation instruction.
After $\overline{\text{RESET}}$ input, the value of CR22 is undefined.

6.4 16-Bit Timer 2 Operations

6.4.1 Pulse width measurement operations

It is possible to measure the pulse width of the signals input to the timer input pins by using the 16-bit timer register (TM2). TM2 is used in free-running mode.

(1) Pulse width measurement with one capture register

When the edge specified by the prescaler mode register (PRM2) is input to the TI20 pin, the value of TM2 is taken into 16-bit capture register 20 (CR20) and an external interrupt request signal (INTTM20) is set.

Any of three edge specifications can be selected - rising, falling, or both edges - by means of bits 2 and 3 (ES00 and ES01) of PRM2.

For valid edge detection, sampling is performed at the count clock selected by PRM2, and a capture operation is only performed when a valid level is detected twice, thus eliminating noise with a short pulse width.

Figure 6-6: Configuration Diagram for Pulse Width Measurement by Using the Free Running Counter

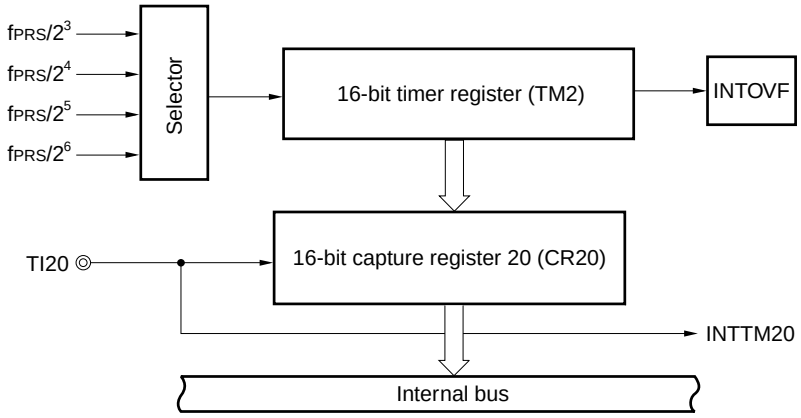
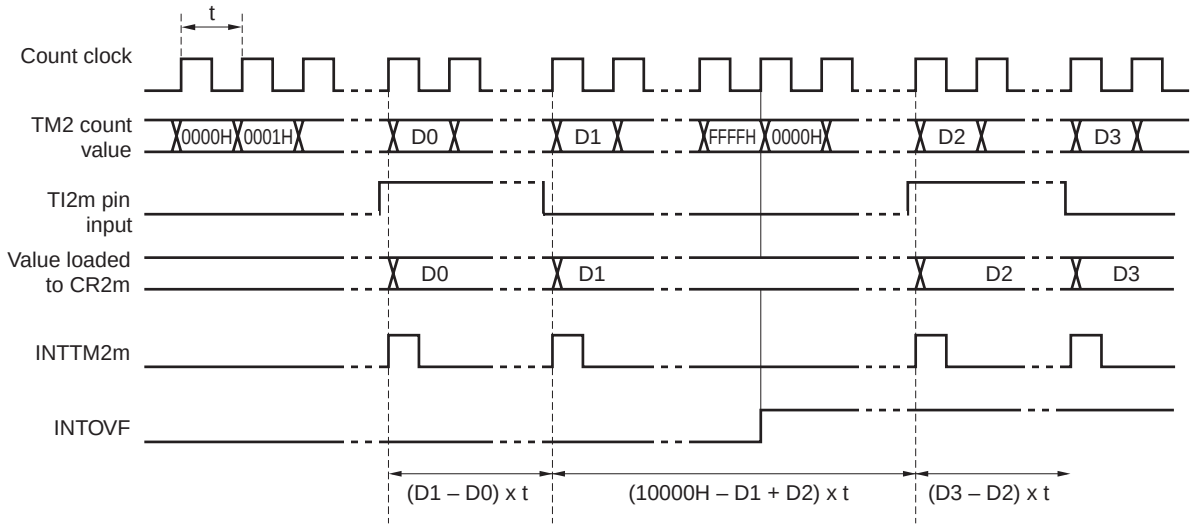


Figure 6-7: Timing of Pulse Width Measurement Operation by Using the Free Running Counter and One Capture Register (with Both Edges Specified)



Remark: m = 0 to 2

(2) Measurement of three pulse widths with the free running counter

The 16-bit timer register (TM2) allows simultaneous measurement of the pulse widths of the three signals input to the TI20 to TI22 pins.

When the edge specified by bits 2 and 3 (ES00 and ES01) of prescaler mode register (PRM2) is input to the TI20 pin, the value of TM2 is taken into 16-bit capture register 20 (CR20) and an external interrupt request signal (INTTM20) is set.

Also, when the edge specified by bits 4 and 5 (ES10 and ES11) of PRM0 is input to the TI21 pin, the value of TM2 is taken into 16-bit capture register 21 (CR21) and an external interrupt request signal (INTTM21) is set.

When the edge specified by bits 6 and 7 (ES20 and ES21) of PRM2 is input to the TI22 pin, the value of TM2 is taken into 16-bit capture register 22 (CR22) and external interrupt request signal (INTTM22) is set.

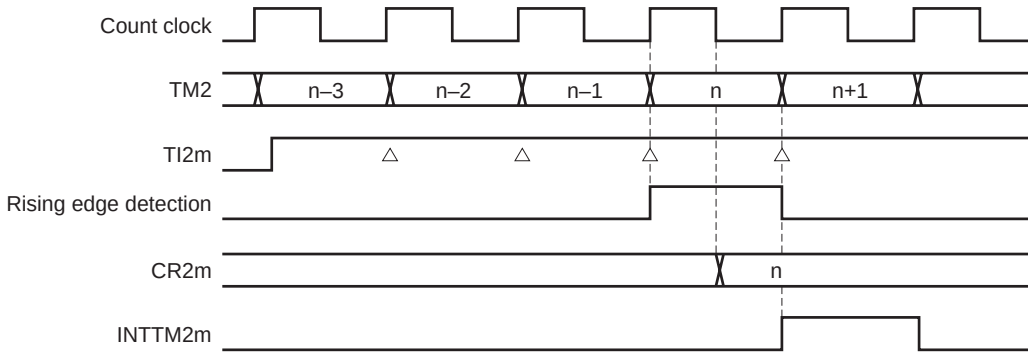
Any of three edge specifications can be selected - rising, falling, or both edges - as the valid edges for the TI20 to TI22 pins by means of bits 2 and 3 (ES00 and ES01), bits 4 and 5 (ES10 and ES11), and bits 6 and 7 (ES20 and ES21) of PRM2, respectively.

For TI20 pin valid edge detection, sampling is performed at the interval selected by the prescaler mode register (PRM2), and a capture operation is only performed when a valid level is detected twice, thus eliminates the noise of a short pulse width.

- Capture operation

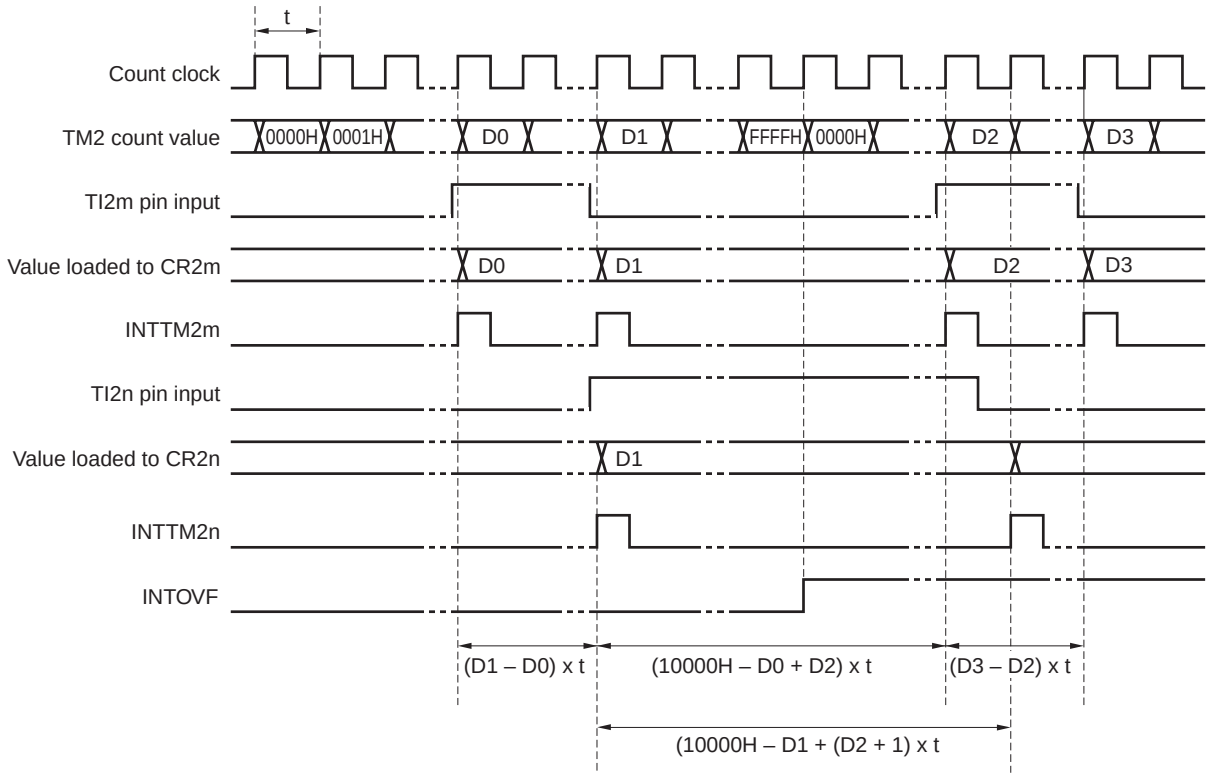
Capture register operation in capture trigger input is shown.

Figure 6-8: CR2m Capture Operation with Rising Edge Specified



Remark: m = 0 to 2

Figure 6-9: Timing of Pulse Width Measurement Operation by Free Running Counter (with Both Edges Specified)



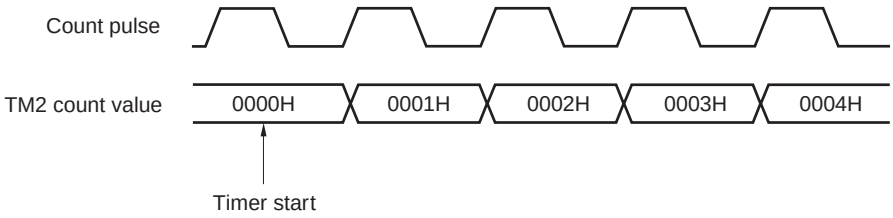
Remark: m = 0 to 2,
n = 1, 2

6.5 16-Bit Timer 2 Precautions

(1) Timer start errors

An error with a maximum of one clock may occur until counting is started after timer start, because the 16-bit timer register (TM2) can be started asynchronously with the count pulse.

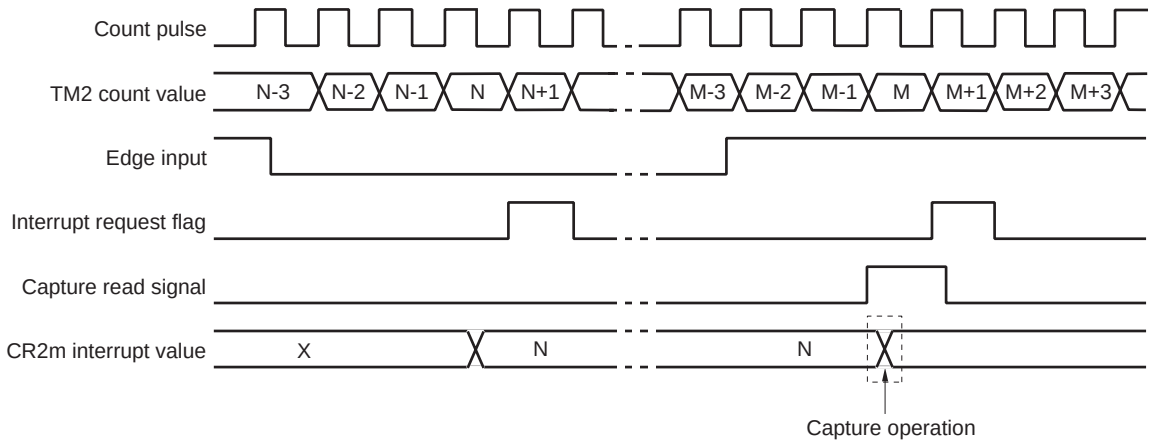
Figure 6-10: 16-Bit Timer Register Start Timing



(2) Capture register data retention timings

If the valid edge of the TI2m pin is input during the 16-bit capture register 2m (CR2m) is read, CR2m performs capture operation, but the capture value is not guaranteed. However, the interrupt request flag (INTTM2m) is set upon detection of the valid edge.

Figure 6-11: Capture Register Data Retention Timing



Remark: m = 0 to 2

(3) Valid edge setting

Set the valid edge of the TI2m pin after setting bit 2 (TMC22) of the 16-bit timer mode control register to 0, and then stopping timer operation. Valid edge setting is carried out with bits 2 to 7 (ESm0 and ESm1) of the prescaler mode register (PRM2).

Remark: m = 0 to 2

(4) Occurrence of INTTM2n

INTTM2n occurs even if no capture pulse exists, immediately after the timer operation has been started (TMC02 of TMC2 has been set to 1) with a high level applied to the input pins TI20 to TI22 of 16-bit timer 2. This occurs if the rising edge (with ESn1 and ESn0 of PRM0 set to 0, 1), or both the rising and falling edges (with ESn1 and ESn0 of PRM2 set to 1, 1) are selected. INTTM2n does not occur if a low level is applied to TI20 to TI22.

(5) Timer stop

When the timer TM2 is disabled, the value of the timer register will be undefined.

(6) Read access of the 16-bit Timer Register TM2

If the peripheral clock is divided by 2 ($f_{PRS} = f_X/2$), due to the setting of bit CKUCKS of the Input Switch control Register ISC to 1, wrong values may be read from the 16-bit Timer Register TM2.

[MEMO]

Chapter 7 16-Bit Timer/Event Counter P (µPD780822B Subseries)

The µPD780822B Subseries includes three timer P channels, TMP0 to TMP2.

7.1 Features

Timer P (TMP) is a 16-bit timer/event counter that can be used in various ways.
TMP can perform the following operations.

- 16-bit accuracy PWM output
- Interval timer
- One-shot pulse output
- Pulse width measurement function
- Timer synchronized operation function
- Free-running function

7.2 Functional Outline

- Capture trigger input signal × 2
- Clock selection × 8
- External event count input × 1
- Readable counter × 1
- Capture/compare reload register × 2
- Capture/compare match interrupt × 2
- Timer input/output (TIOPn0, TIOPn1) × 2

Remark: n = 0 to 2

7.3 Configuration

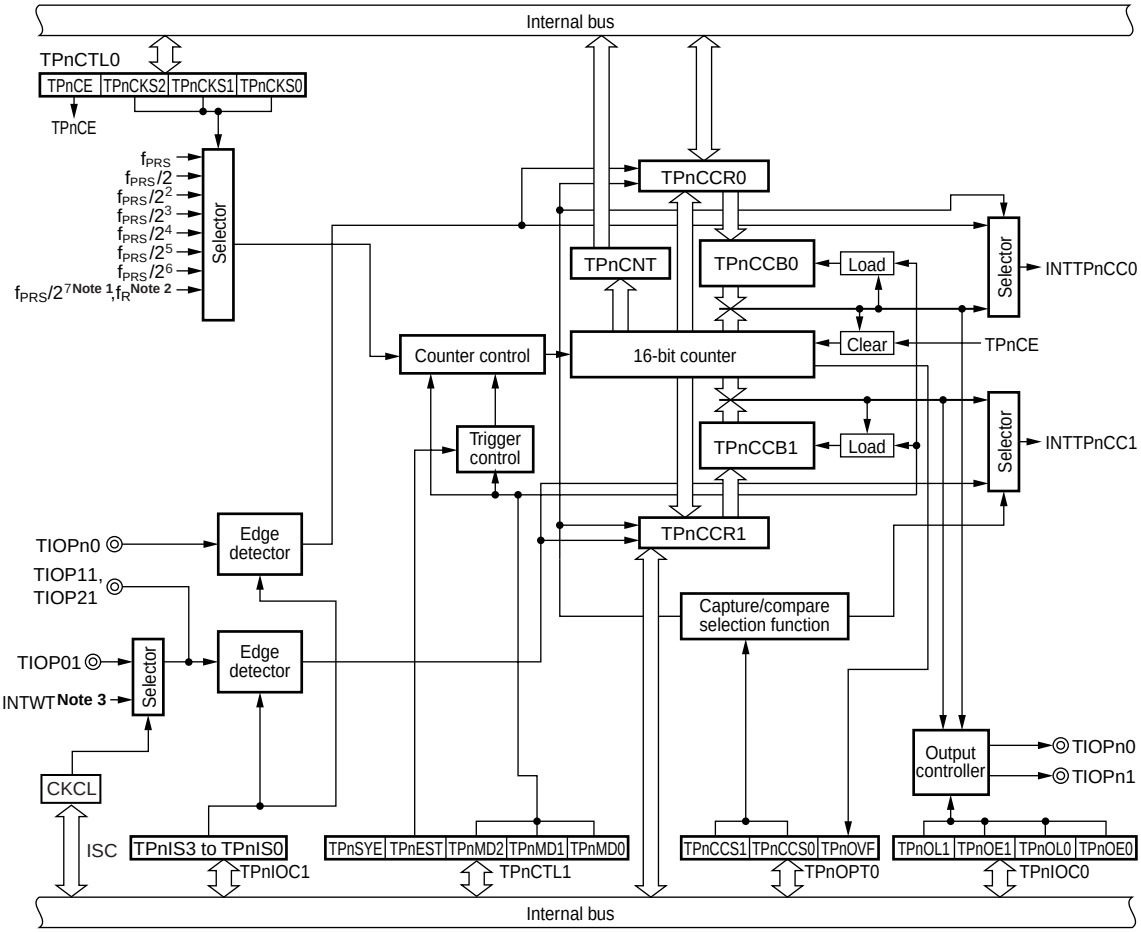
TMP consists of the following hardware.

Table 7-1: Configuration of TMP0 to TMP2

Item	Configuration
Timer register	16-bit counter
Registers	TMPn capture/compare registers 0, 1 (TPnCCR0, TPnCCR1) TMPn counter read buffer register (TPnCNT) TMPn capture/compare buffer registers 0, 1 (TPnCCB0,TPnCCB1)
Timer inputs/outputs	2 (TIOPn0, TIOPn1)
Control registers	TMPn control registers 0, 1 (TPnCTL0, TPnCTL1) TMPn I/O control registers 0, 1 (TPnIOC0, TPnIOC1) TMPn option register 0 (TPnOPT0) Input switch control register (ISC)

Remark: n = 0 to 2, m = 0, 1

Figure 7-1: Block Diagram of Timer P



Notes: 1. TMP0, TMP2

2. TMP1

3. INTWT is the watch timer interrupt signal

Remarks: 1. n=0 to 2

2. f_R: Internal low-speed oscillator clock frequency
f_{PRS}: Peripheral hardware clock oscillation frequency

(1) TMPn capture/compare register 0 (TPnCCR0)

The TPnCCR0 register is a 16-bit register that has a capture function and a compare function.
Whether this register is used as a capture register or a compare register can be specified by using the TPnCCS0 bit of the TPnOPT0 register, but only in the free-running mode.
In the pulse width measurement mode, this register can be used only as a capture register (it cannot be used as a compare register).
In all the modes other than the free-running mode and pulse width measurement mode, this register functions as a compare register.
In the default status, the TPnCCR0 register functions as a compare register.
This register can be read or written in 16-bit units.
$\overline{\text{RESET}}$ input clears this register to 0000H.

Figure 7-2: TMPn capture/compare register 0 (TPnCCR0) Format

	15	14	13	12	11	10	9	8	Address	R/W	After reset
TPnCCR0									F806H, F807H		
(n=0 to 2)	7	6	5	4	3	2	1	0	F816H, F817H	R/W	0000H
									F826H, F827H		

- When used as compare register
  - TPnCCR0 can be rewritten when TPnCE = 1.
  - TPnCCR0 is rewritten as follows when TPnCE = 1.

TMP Operation Mode	Method of Writing TPnCCR0 Register
PWM output mode	Reload
Free-running mode, one-shot pulse output mode, or interval timer mode	Any time write
Pulse width measurement mode	Cannot be used because used only as capture register

- When used as capture register
  - The count value is stored in TPnCCR0 on detection of the edge of the capture trigger (TIOPn0) input.

(2) TMPn capture/compare register 1 (TPnCCR1)

The TPnCCR1 register is a 16-bit register that has a capture function and a compare function.
Whether this register is used as a capture register or a compare register can be specified by using the TPnCCS1 bit of the TPnOPT0 register, but only in the free-running mode.
In the pulse width measurement mode, this register can be used only as a capture register (it cannot be used as a compare register).
In all the modes other than the free-running mode and pulse width measurement mode, this register functions as a compare register.
In the default status, the TPnCCR1 register functions as a reload register.
This register can be read or written in 16-bit units.
$\overline{\text{RESET}}$ input clears this register to 0000H.

Figure 7-3: TMPn capture/compare register 1 (TPnCCR1) Format

	15	14	13	12	11	10	9	8	Address	R/W	After reset
TPnCCR1									F808H, F809H		
(n=0 to 2)	7	6	5	4	3	2	1	0	F818H, F819H	R/W	0000H
									F828H, F829H		

- When used as compare register
  - TPnCCR1 can be rewritten when TPnCE = 1.
  - TPnCCR1 is rewritten as follows when TPnCE = 1.

TMP Operation Mode	Method of Writing TPnCCR1 Register
PWM output mode	Reload
Free-running mode, one-shot pulse output mode, or interval timer mode	Any time write
Pulse width measurement mode	Cannot be used because used only as capture register

- When used as capture register
  - The count value is stored in TPnCCR1 on detection of the edge of the capture trigger (TIOPn1) input.

(3) TMPn counter read buffer register (TPnCNT)

The TPnCNT register is a read buffer register that can read the value of the 16-bit counter.
This register is read-only, in 16-bit units.
$\overline{\text{RESET}}$ input sets this register to 0000H.
The value of the register can be read when TPnCE = 1.

Figure 7-4: TMPn counter read buffer register (TPnCNT) Format

	15	14	13	12	11	10	9	8	Address	R/W	After reset
TPnCNT									F80AH, F80BH		
(n=0 to 2)	7	6	5	4	3	2	1	0	F81AH, F81BH	R	0000H
									F82AH, F82BH		

7.4 Control Registers

(1) TMPn control register 0 (TPnCTL0)

The TPnCTL0 register is an 8-bit register that controls the operation of timer P.
This register can be read or written in 8-bit or 1-bit units.
$\overline{\text{RESET}}$ input clears this register to 00H.
The same value can always be written to the TPnCTL0 register by software.

Figure 7-5: TMPn control register 0 (TPnCTL0) Format)

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnCTL0	TPnCE	0	0	0	0	TPnCKS2	TPnCKS1	TPnCKS0	F800H		
(n=0 to 2)									F810H	R/W	00H
									F820H		

TPnCE	Control of operation of timer Pn
0	Disable internal operating clock operation (asynchronously reset TMPn).
1	Enable internal operating clock operation.
The TPnCE bit controls the internal operating clock and asynchronously resets TMPn. When this bit is cleared to 0, the internal operating clock of TMPn is stopped (fixed to the low level), and TMPn is asynchronously reset. When the TPnCE bit is set to 1, the internal operating clock is enabled within 2 input clocks, and TMPn counts up.	

TPnCKS2	TPnCKS1	TPnCKS0	Selection of internal count clock	
			n = 0, 2	n = 1
0	0	0	f_{PRS}	
0	0	1	$f_{PRS}/2$	
0	1	0	$f_{PRS}/2^2$	
0	1	1	$f_{PRS}/2^3$	
1	0	0	$f_{PRS}/2^4$	
1	0	1	$f_{PRS}/2^5$	
1	1	0	$f_{PRS}/2^6$	
1	1	1	$f_{PRS}/2^7$	f_R

Caution: Set the TPnCKS2 to TPnCKS0 bits when TPnCE = 0.
When the TPnCE bit setting is changed from 0 to 1, the TPnCKS2 to TPnCKS0 bits can be set at the same time.

Remark: f_{PRS} : Peripheral hardware clock oscillation frequency
f_R : Internal low-speed oscillator clock frequency

Resolution and maximum number of counts:

Internal count clock	Resolution [µs]		Maximum count time [ms]	
	f_{PRS} = 8 MHz	f_{PRS} = 16 MHz	f_{PRS} = 8 MHz	f_{PRS} = 16 MHz
f_{PRS}	0.125	0.0625	8.19	4.10
$f_{PRS}/2$	0.250	0.125	16.38	8.19
$f_{PRS}/2^2$	0.500	0.250	32.77	16.38
$f_{PRS}/2^3$	1.000	0.500	65.54	32.77
$f_{PRS}/2^4$	2.000	1.000	131.07	65.54
$f_{PRS}/2^5$	4.000	2.000	262.14	131.07
$f_{PRS}/2^6$	8.000	4.000	524.28	262.14
$f_{PRS}/2^7$	16.000	8.000	1048.56	524.28

Internal count clock	Resolution [µs]	Maximum count time [ms]
	f_R = 240 kHz	f_R = 240 kHz
f_R	4.167	273.06

(2) TMPn timer control register 1 (TPnCTL1)

The TPnCTL1 register is an 8-bit register that controls the operation of timer P.
This register can be read or written in 8-bit or 1-bit units.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 7-6: TMPn timer control register 1 (TPnCTL1) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnCTL1	TPnSYE	TPnEST	0	0	0	TPnMD2	TPnMD1	TPnMD0	F801H		
(n=0 to 2)									F811H	R/W	00H
									F821H		

TPnSYE	Tuned operation mode enable control
0	Independent operation mode (asynchronous operation mode)
1	Tuned operation mode (specification of slave operation) In this mode, timer P can operate in synchronization with a master timer

Master timer	Slave timer	
TMP0	TMP1	TMP2

Remark: For the tuned operation mode, refer to **7.6 "Timer Synchronized Operation Function" on page 209**.

TPnEST	Software trigger control
0	No operation
1	The effective signal to the trigger input is made only active in one-shot pulse mode
The TPnEST bit functions as a software trigger in the one-shot pulse mode (this bit is invalid in any other mode). By setting TPnEST to 1 when TPnCE = 1, a software trigger is issued. Therefore, be sure to set TPnEST to 1 when TPnCE = 1. The read value of the TPnEST bit is always 0.	

Remark: Be sure to clear bits 3 to 5 to 0.

TPnMD2	TPnMD1	TPnMD0	Selection of timer mode
0	0	0	Interval timer mode
0	1	1	One-shot pulse mode
1	0	0	PWM mode
1	0	1	Free-running mode
1	1	0	Pulse width measurement mode
All other settings			Are prohibited

Caution: Set the TPnMD2 to TPnMD0 bits when TPnCE = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten when TPnCE = 1, the operation cannot be guaranteed. If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

(3) TMPn I/O control register 0 (TPnIOC0)

The TPnIOC0 register is an 8-bit register that controls the timer outputs (TIOPn0 and TIOPn1).
This register can be read or written in 8-bit or 1-bit units.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 7-7: TMPn I/O control register 0 (TPnIOC0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnIOC0	0	0	0	0	TPnOL1	TPnOE1	TPnOL0	TPnOE0	F802H		
(n=0 to 2)									F812H	R/W	00H
									F822H		

TPnOLm	Setting of TIOPnm output level (m = 0, 1)
0	Normal output
1	Inverted output

TPnOEm	Setting of TIOPnm output (m = 0, 1)
0	Disable timer output (TIOPnm pin outputs low level when TPnOLm = 0, and high level when TPnOLm = 1)
1	Enable timer output (TIOPnm pin outputs pulses).

Cautions: 1. Rewrite the TPnOL1, TPnOE1, TPnOL0, and TPnOE0 bits when TPnCE = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

2. To enable the timer output, be sure to set the corresponding alternate-function pins TPnIS3 to TPnIS0 of the TPnIOC1 register to “Detect no edge” and invalidate the capture operation. Then set the corresponding alternate-function port to output mode.

(4) TMPn I/O control register 1 (TPnIOC1)

The TPnIOC1 register is an 8-bit register that controls the valid edge of the external input signals (TIOPn0, TIOP01 or INTWT[Note], TIOP11 and TIOP21).
This register can be read or written in 8-bit or 1-bit units.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 7-8: TMPn I/O control register 1 (TPnIOC1) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnIOC1	0	0	0	0	TPnIS3	TPnIS2	TPnIS1	TPnIS0	F803H		
(n=0 to 2)									F813H	R/W	00H
									F823H		

TPnIS3	TPnIS2	Setting of valid edge of capture input (TIOP01 or INTWT[Note], TIOP11, TIOP21)
0	0	Detect no edge (capture operation is invalid)
0	1	Detect rising edge
1	0	Detect falling edge
1	1	Detect both the edges

TPnIS1	TPnIS0	Setting of valid edge of capture input (TIOPn0)
0	0	Detect no edge (capture operation is invalid)
0	1	Detect rising edge
1	0	Detect falling edge
1	1	Detect both the edges

Note: INTWT is the interrupt signal from watch timer.
The selection between TIOP01 and INTWT is made with the bit CKCL of the input switch control register (ISC). Please see **7.4 Control Registers**, **(6)"Input switch control register (ISC)" on page 185**.

Cautions: 1. Rewrite the TPnIS3 to TPnIS0 bits when TPnCE = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

2. The TPnIS3 to TPnIS0 bits are valid only in the free-running mode and pulse width measurement mode. A capture operation is not performed in any other mode.

3. If used as the capture input, be sure to set the corresponding alternate-function pins TPnOE1 and TPnOE0 of the TPnIOC0 register to "Disable timer output" and set the capture input valid edge. Then set the corresponding alternate-function port to input mode.

(5) TMPn option register 0 (TPnOPT0)

The TPnOPT0 register is an 8-bit register that selects a capture or compare operation, and detects an overflow.

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

RESET input clears this register to 00H.

Figure 7-9: TMPn option register 0 (TPnOPT0) Format

	7	6	5	4	3	2	1	0	Address	R/W	After reset
TPnOPT0	0	0	TPnCCS1	TPnCCS0	0	0	0	TPnOVF	F805H		
(n=0 to 2)									F815H	R/W	00H
									F825H		

TPnCCSm	Selection of capture or compare operation of TPnCCRm register (m = 0, 1)
0	Compare register
1	Capture register
The set value of the TPnCCSm bit is valid only in the free-running mode	

TPnOVF	Detection of overflow of timer P
Set (1)	Overflow occurred
Reset (0)	0 written to TPnOVF bit or TPnCE = 0
• The TPnOVF bit is set when the 16-bit counter overflows from FFFFH to 0000H in the free-running mode and pulse width measurement mode. • The TPnOVF bit is not cleared even if the TPnOVF bit and TPnOPT0 register are read when TPnOVF = 1. • The TPnOVF bit can be read and written, but 1 cannot be written to the TPnOVF bit. Writing 1 to this bit does not affect the operation of timer P.	

Caution: Rewrite the TPnCCS1 and TPnCCS0 bits when TPnCE = 0 (the same value can be written when TPnCE = 1). If these bits are rewritten by mistake, clear TPnCE to 0 and then set them again.

(6) Input switch control register (ISC)

This register is used to select the input source of the TMP0.
ISC is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets ISC to 00H.

Figure 7-10: Input Switch Control Register (ISC) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ISC	0	CKSWD	CKSWT1	CKSWT0	CKUCKS	CKCL	ISC1	ISC0	R/W	FF4AH	00H

CKCL	TIOP01 input source selection
0	TIOP01/P14
1	Watch timer interrupt INTWT

7.5 Operation

Timer P performs the following operations.

Operation	TPnEST (Software Trigger Bit)	Capture/Compare Selection	Compare Write
Interval timer mode	Invalid	Compare only	Anytime write
One-shot pulse output mode[Note]	Valid	Compare only	Anytime write
PWM mode	Invalid	Compare only	Reload
Free-running mode	Invalid	Capture/compare selectable	Anytime write
Pulse width measurement mode[Note]	Invalid	Capture only	Not applicable

Note: To use the one-shot pulse mode, or pulse width measurement mode, select a count clock (by setting the bits TPnCKS2 to TPnCKS0 of the TPnCTL0 register).

Remark: n = 0 to 2

7.5.1 Anytime write and reload

Timer P allows rewriting of the TPnCCR0 and TPnCCR1 registers while the timer is operating (TPnCE = 1). These registers are written differently (anytime write or reload) depending on the mode.

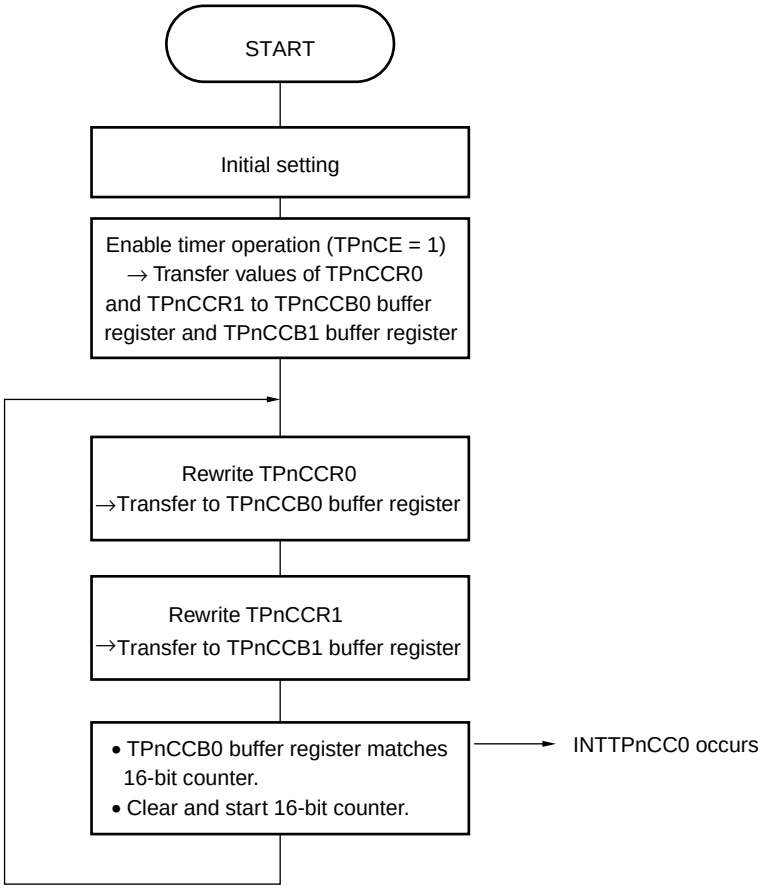
(1) Anytime write

When data is written to the TPnCCRm register during timer operation, it is transferred at any time to the TPnCCBm buffer register and is compared with the value of the 16-bit counter.

Remark: n=0 to 2

m = 0, 1

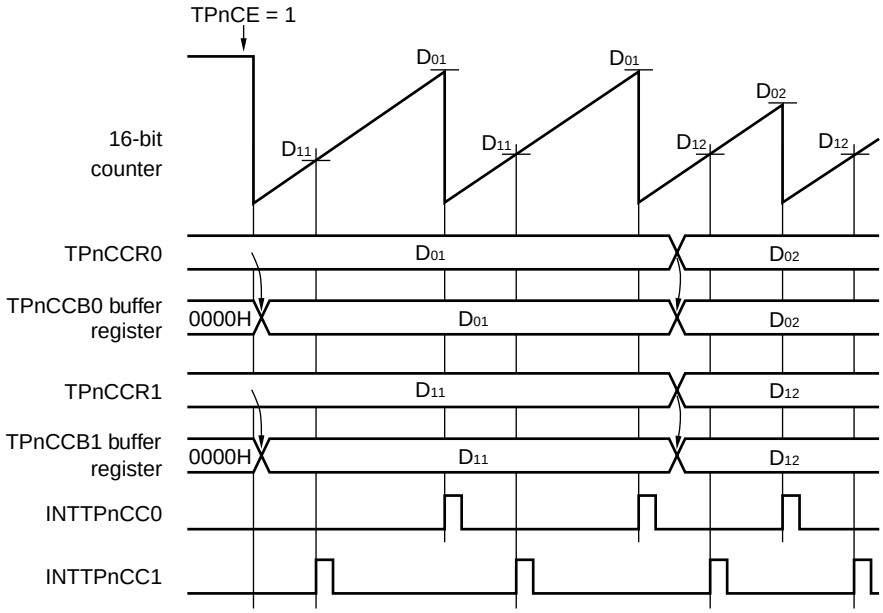
Figure 7-11: Flowchart of Basic Operation of Anytime Write



Remarks: 1. This is an example in the interval timer mode.

2. n=0 to 2

Figure 7-12: Timing Chart of Anytime Write



Remarks: 1. D_{01}, D_{02}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Set value of TPnCCR1 register (0000H to FFFFH)

2. This is an example in the interval timer mode.

3. n=0 to 2

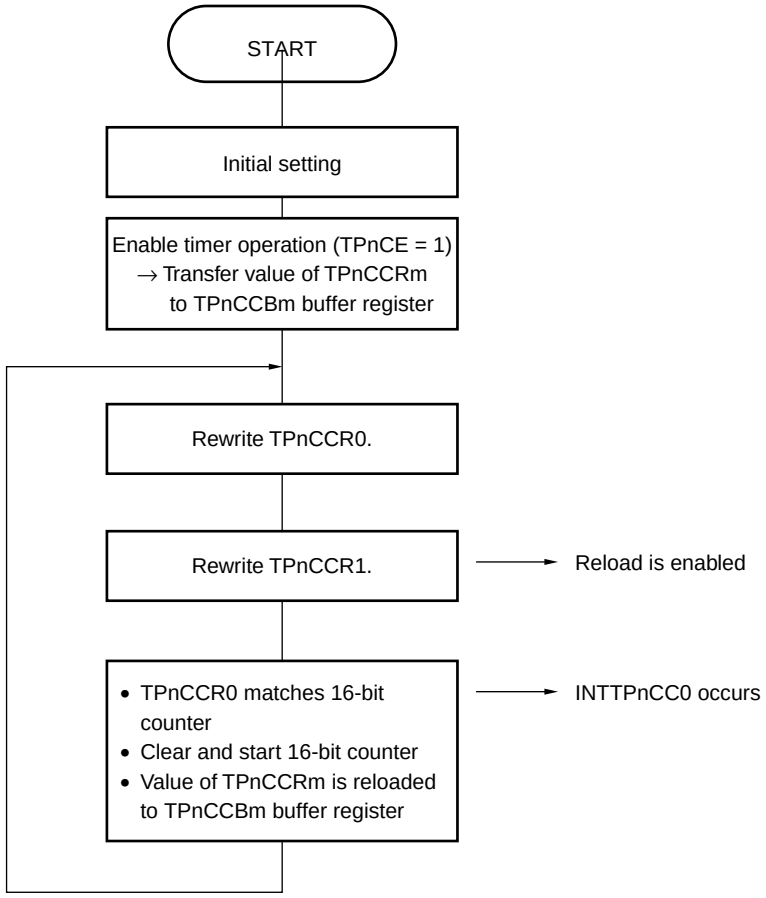
(2) Reload

When data is written to the TPnCCR0 and TPnCCR1 registers during timer operation, it is compared with the value of the 16-bit counter via the TPnCCBm buffer register. The values of the TPnCCR0 and TPnCCR1 registers can be rewritten when TPnCE = 1.

So that the set values of the TPnCCR0 and TPnCCR1 registers are compared with the value of the 16-bit counter (the set values are reloaded to the TPnCCBm buffer register), the value of the TPnCCR0 register must be rewritten and then a value must be written to the TPnCCR1 register before the value of the 16-bit counter matches the value of TPnCCR0. When the value of the TPnCCR0 register matches the value of the 16-bit counter, the values of the TPnCCR0 and TPnCCR1 registers are reloaded.

Whether the next reload timing is made valid or not is controlled by writing to the TPnCCR1 register. Therefore, write the same value to the TPnCCR1 register when it is necessary to rewrite the value of only the TPnCCR0 register.

Figure 7-13: Flowchart of Basic Operation of Reload

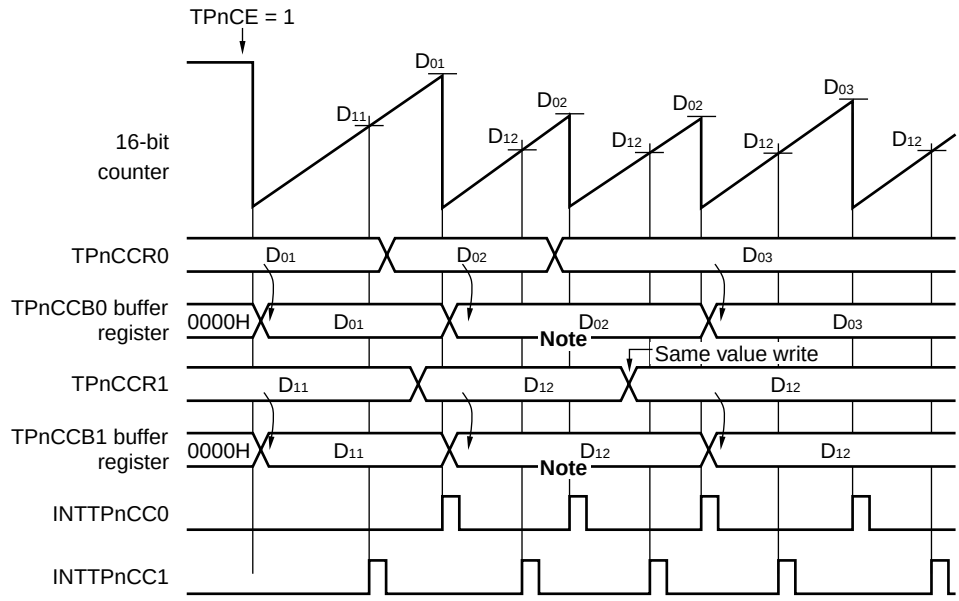


Caution: Writing the TPnCCR1 register includes an operation to enable reload. Therefore, rewrite the TPnCCR1 register after rewriting the TPnCCR0 register.

Remarks: 1. This is an example in the PWM mode.

2. n=0 to 2, m = 0, 1

Figure 7-14: Timing Chart of Reload



Note: The value is not reloaded because the TPnCCR1 register is not written.

Remarks: **1.** D_{01}, D_{02}, D_{03}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{11}, D_{12}: Set value of TPnCCR1 register (0000H to FFFFH)

2. This is an example in the PWM mode.

3. n=0 to 2

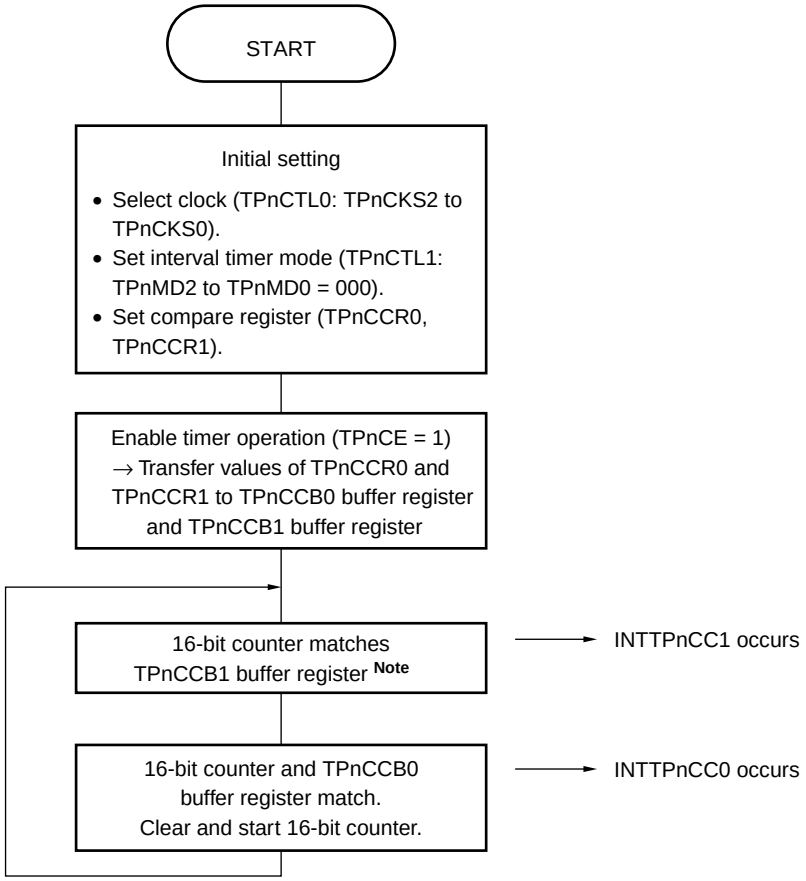
7.5.2 Interval timer mode (TPnMD2 to TPnMD0 = 000)

In the interval timer mode, an interrupt request signal (INTTPnCC0) is generated when the set value of the TPnCCR0 register matches the value of the 16-bit counter, and the 16-bit counter is cleared. Rewriting the TPnCCR0 register is enabled when TPnCE = 1. When a value is set to the TPnCCR0 register by a write instruction from the CPU, it is transferred to the TPnCCB0 buffer register by means of anytime write, and is compared with the value of the 16-bit counter.
In the interval timer mode, the 16-bit counter can be cleared only when its value matches the value of the TPnCCB0 buffer register.
The 16-bit counter is not cleared by using the TPnCCR1 register.
However, the set value of the TPnCCR1 register is transferred to the TPnCCB1 buffer register and compared with the value of the 16-bit counter. As a result, an interrupt request (INTTPnCC1) is generated.
The value can also be output from the TIOPnm pin by setting the TPnOEm bit to 1.
When the TPnCCR1 register is not used, it is recommended to set the TPnCCR1 register to FFFFH.

Remark: n=0 to 2

m = 0, 1

Figure 7-15: Flowchart of Basic Operation in Interval Timer Mode

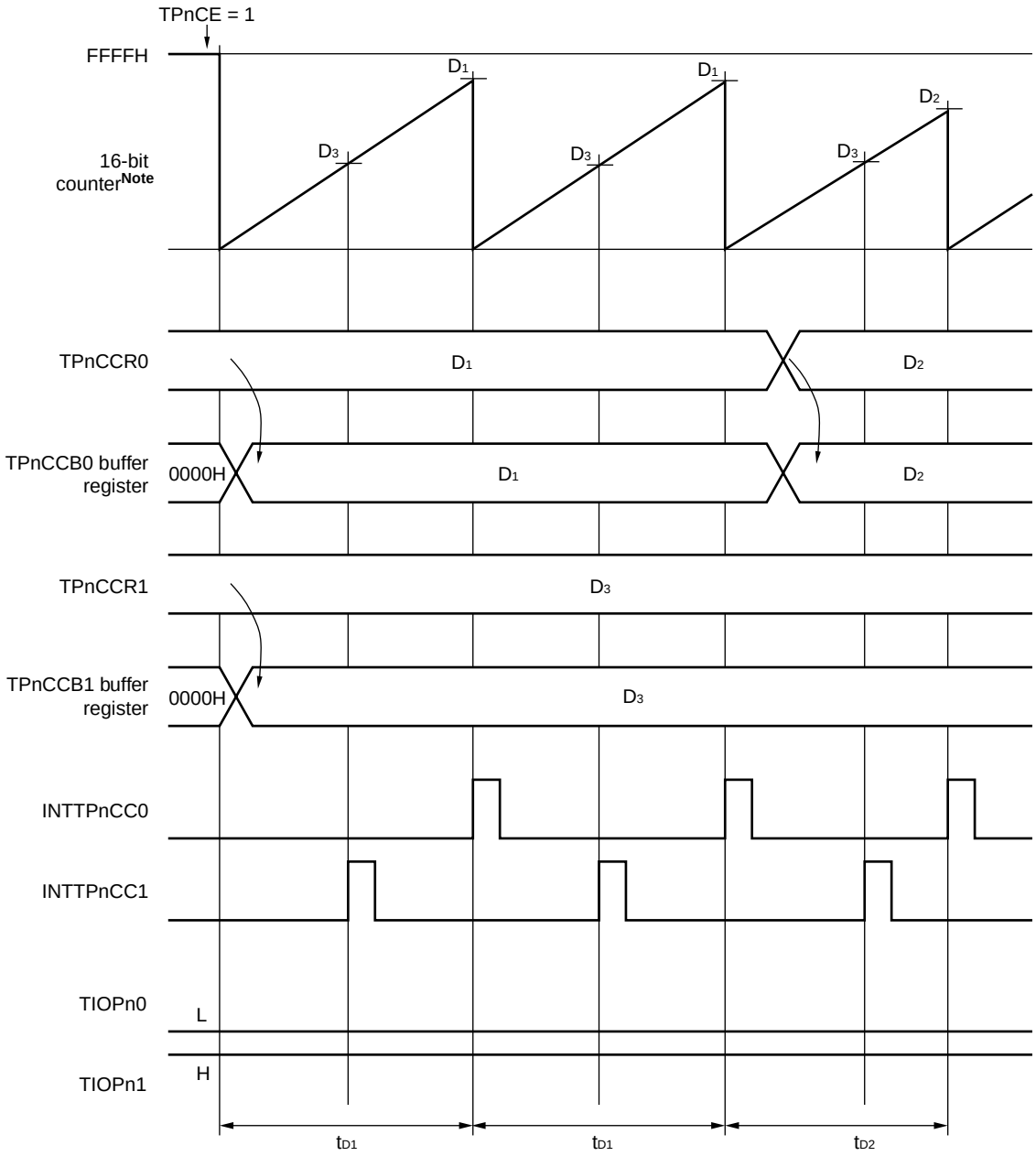


Note: The 16-bit counter is not cleared when its value matches the value of TPnCCR1.

Remark: n=0 to 2

Figure 7-16: Timing of Basic Operation in Interval Timer Mode (1/2)

(a) When $D_1 > D_2 > D_3$, only TPnCCR0 register value is written, and TIOPn0 and TIOPn1 are not output (TPnOE0 = 0, TPnOE1 = 0, TPnOL0 = 0, TPnOL1 = 1)



Note: The 16-bit counter is not cleared when its value matches the value of TPnCCR1.

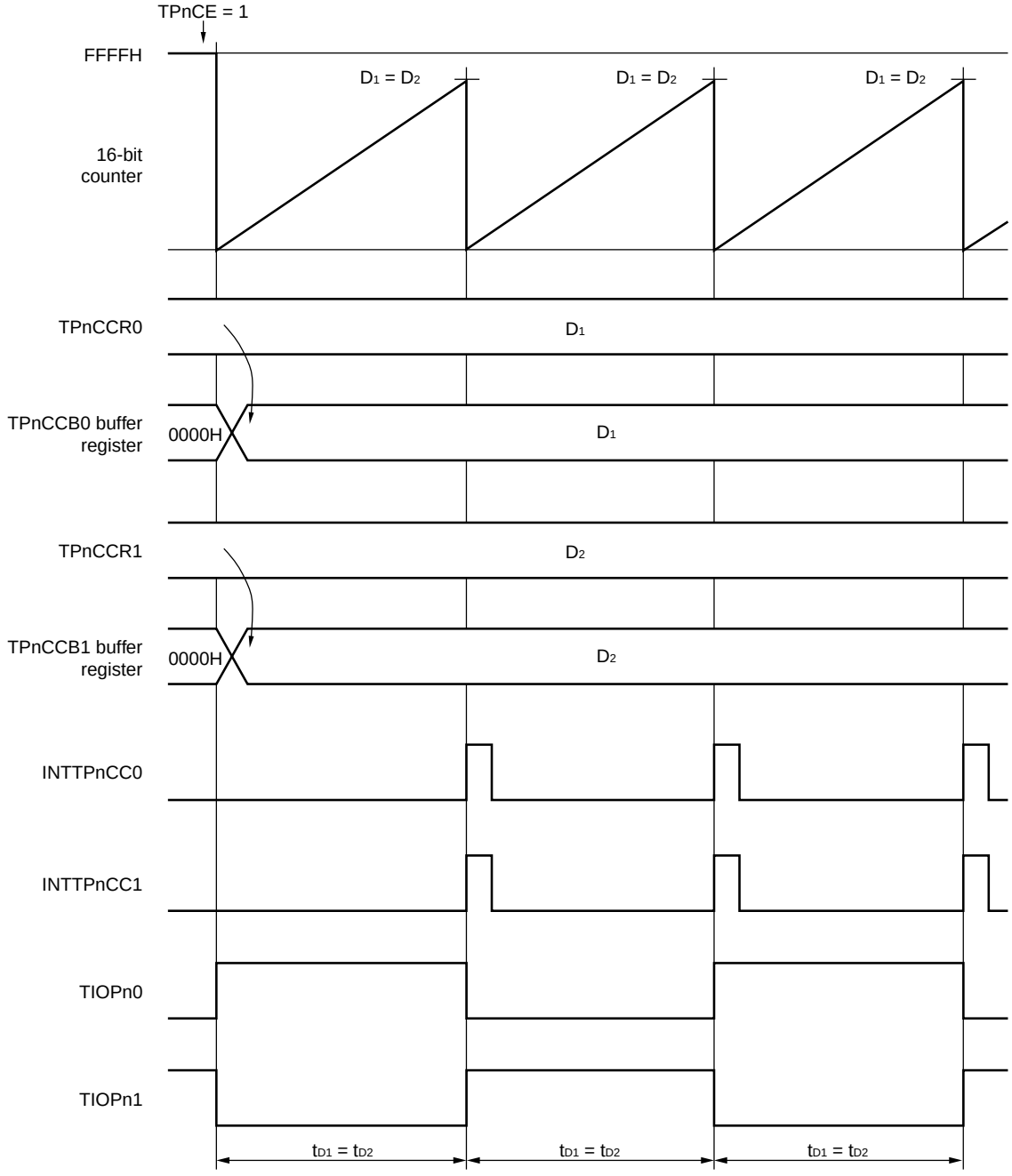
Remarks: 1. D_1, D_2: Set value of TPnCCR0 register (0000H to FFFFH)
D_3: Set value of TPnCCR1 register (0000H to FFFFH)

2. Interval time (t_{Dn}) = (Dn + 1) × (Count clock cycle)

3. n=0 to 2

Figure 7-16: Timing of Basic Operation in Interval Timer Mode (2/2)

(a) When $D_1 = D_2$, TPnCCR0 and TPnCCR1 are not rewritten, and TIOPn0 and TIOPn1 are output (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 1)



Remarks: 1. D_1: Set value of TPnCCR0 register (0000H to FFFFH)
D_2: Set value of TPnCCR1 register (0000H to FFFFH)

2. Interval time (t_{Dn}) = (Dn + 1) × (Count clock cycle)

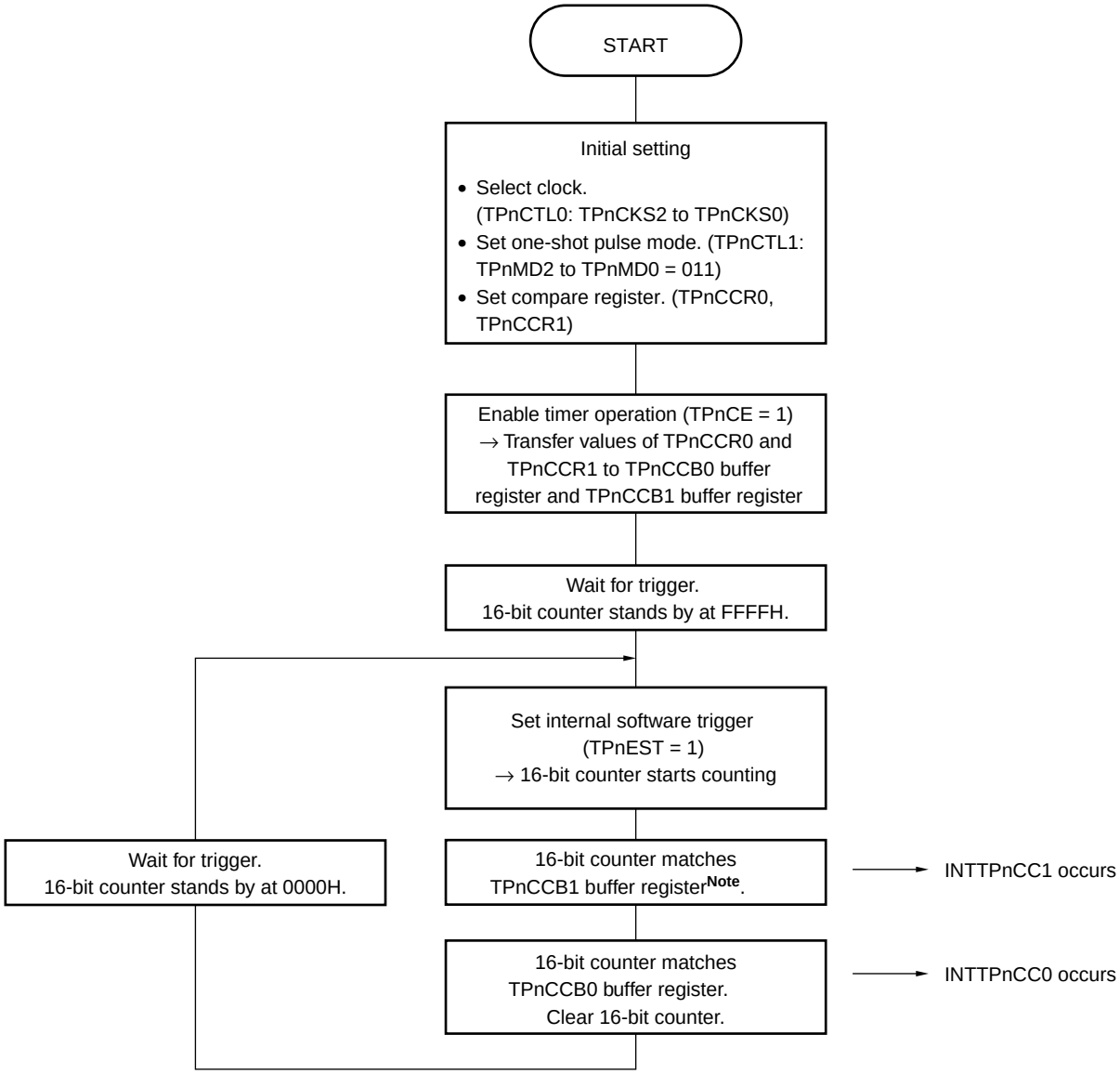
3. n=0 to 2

7.5.3 One-shot pulse mode (TPnMD2 to TPnMD0 = 011)

When TPnCE is set to 1 in the one-shot pulse mode, the 16-bit counter waits for the setting of the TPnEST bit (to 1). When the trigger is input, the 16-bit counter starts counting up. When the value of the 16-bit counter matches the value of the TPnCCB1 buffer register that has been transferred from the TPnCCR1 register, TIOPn1 goes high. When the value of the 16-bit counter matches the value of the TPnCCB0 buffer register that has been transferred from the TPnCCR0 register, TIOPn1 goes low, and the 16-bit counter is cleared to 0000H and stops input of a second software trigger (by setting 1 to bit TPnEST of the register TPnCTL1) is ignored while the 16-bit counter is operating. Be sure to input a second trigger while the 16-bit counter is stopped at 0000H. In the one-shot pulse mode, rewriting the TPnCCR0 and TPnCCR1 registers is enabled when TPnCE = 1. The set values of the TPnCCR0 and TPnCCR1 registers become valid after a write instruction from the CPU is executed. They are then transferred to the TPnCCB0 and TPnCCB1 buffer registers, and compared with the value of the 16-bit counter. The waveform of the one-shot pulse is output from the TIOPn1 pin. The TIOPn0 pin produces a toggle output when the value of the 16-bit counter matches the value of the TPnCCR0 register.
In the one-shot pulse mode, the TPnCCR0 and TPnCCR1 registers function only as compare registers. They cannot be used as capture registers.

Remark: n=0 to 2

Figure 7-17: Flowchart of Basic Operation in One-Shot Pulse Mode

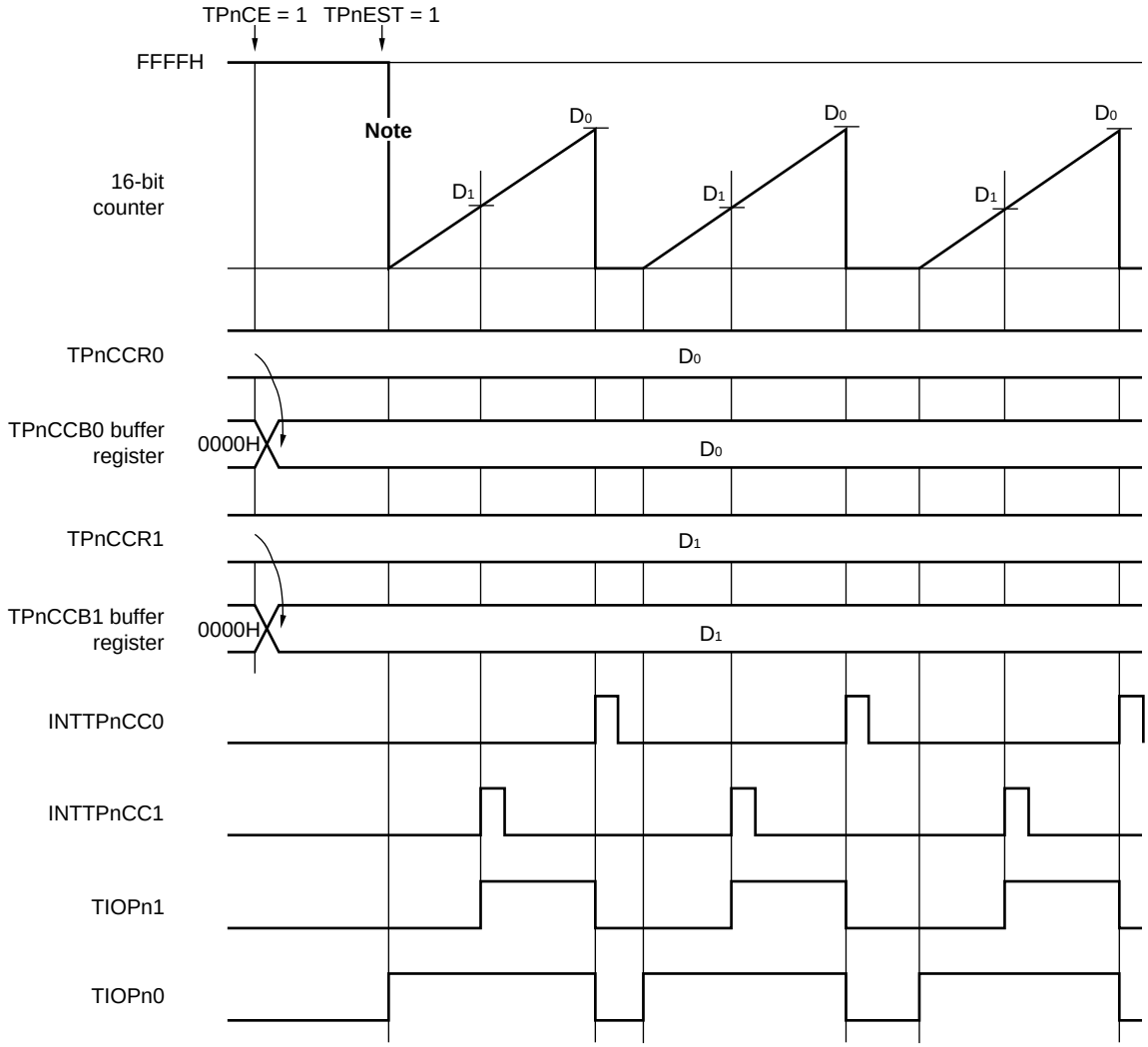


Note: The 16-bit counter is not cleared when it matches the TPnCCB1 buffer register.

Caution: The 16-bit counter is not cleared even if the software trigger is input while the counter is counting up, and the software trigger input is ignored.

Remark: n=0 to 2

Figure 7-18: Timing of Basic Operation in One-Shot Pulse Mode (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Note: The 16-bit counter starts counting up when TPnEST = 1.

Remarks: 1. D_0: Set value of TPnCCR0 register (0000H to FFFFH)
D_1: Set value of TPnCCR1 register (0000H to FFFFH)

2. n=0 to 2

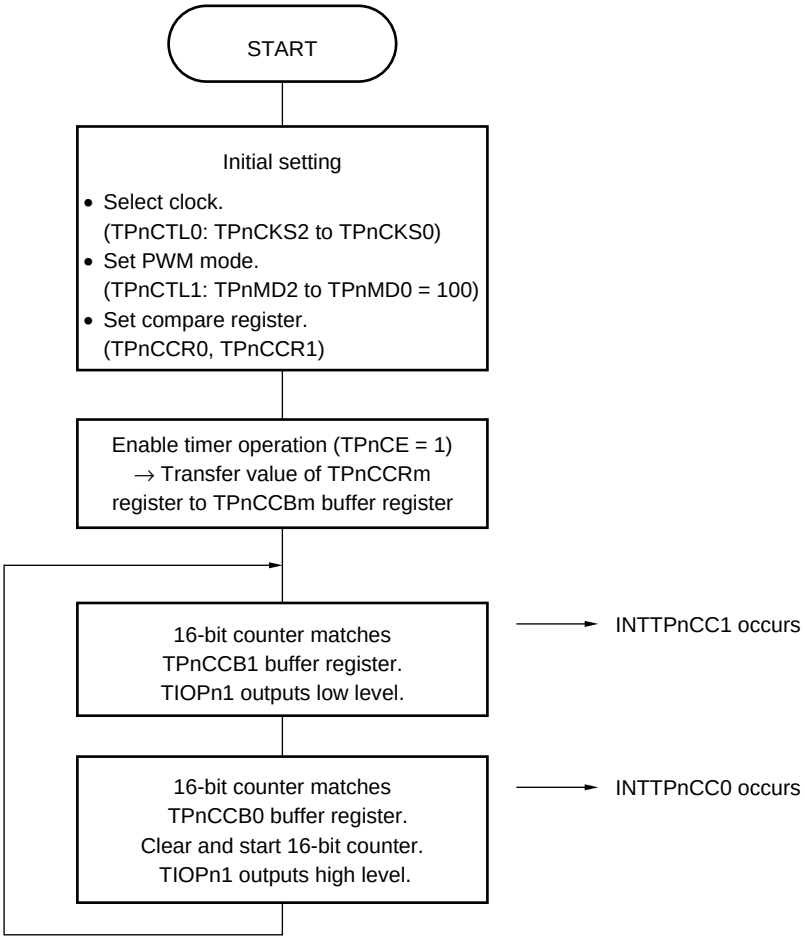
7.5.4 PWM mode (TPnMD2 to TPnMD0 = 100)

In the PWM mode, TMPn capture/compare register 1 (TPnCCR1) is used to set the duty factor and TMPn capture/compare register 0 (TPnCCR0) is used to set the cycle.
By using these two registers and operating the timer, variable-duty PWM is output.
Rewriting the TPnCCR0 and TPnCCR1 registers is enabled when TPnCE = 1.
So that the set values of the TPnCCR0 and TPnCCR1 registers are compared with the value of the 16-bit counter (reloaded to the TPnCCB0 and TPnCCB1 buffer registers), the TPnCCR0 register must be rewritten and then a value must be written to the TPnCCR1 register before the value of the 16-bit counter matches the value of the TPnCCR0 register. The values of the TPnCCR0 and TPnCCR1 registers are reloaded when the value of the TPnCCR0 register later matches the value of the 16-bit counter.
Whether the next reload timing is made valid or not is controlled by writing to the TPnCCR1 register. Therefore, write the same value to the TPnCCR1 register even when only the value of the TPnCCR0 register needs to be rewritten. Reload is invalid when only the value of the TPnCCR0 register is rewritten. To stop timer P, clear TPnCE to 0. The waveform of PWM is output from the TIOPn1 pin. The TIOPn0 pin produces a toggle output when the 16-bit counter matches the TPnCCR0 register.
In the PWM mode, the TPnCCR0 and TPnCCR1 registers are used only as compare registers. They cannot be used as capture registers.

Remark: n=0 to 2

Figure 7-19: Flowchart of Basic Operation in PWM Mode (1/2)

(a) When values of TPnCCR0 and TPnCCR1 registers are not rewritten during timer operation

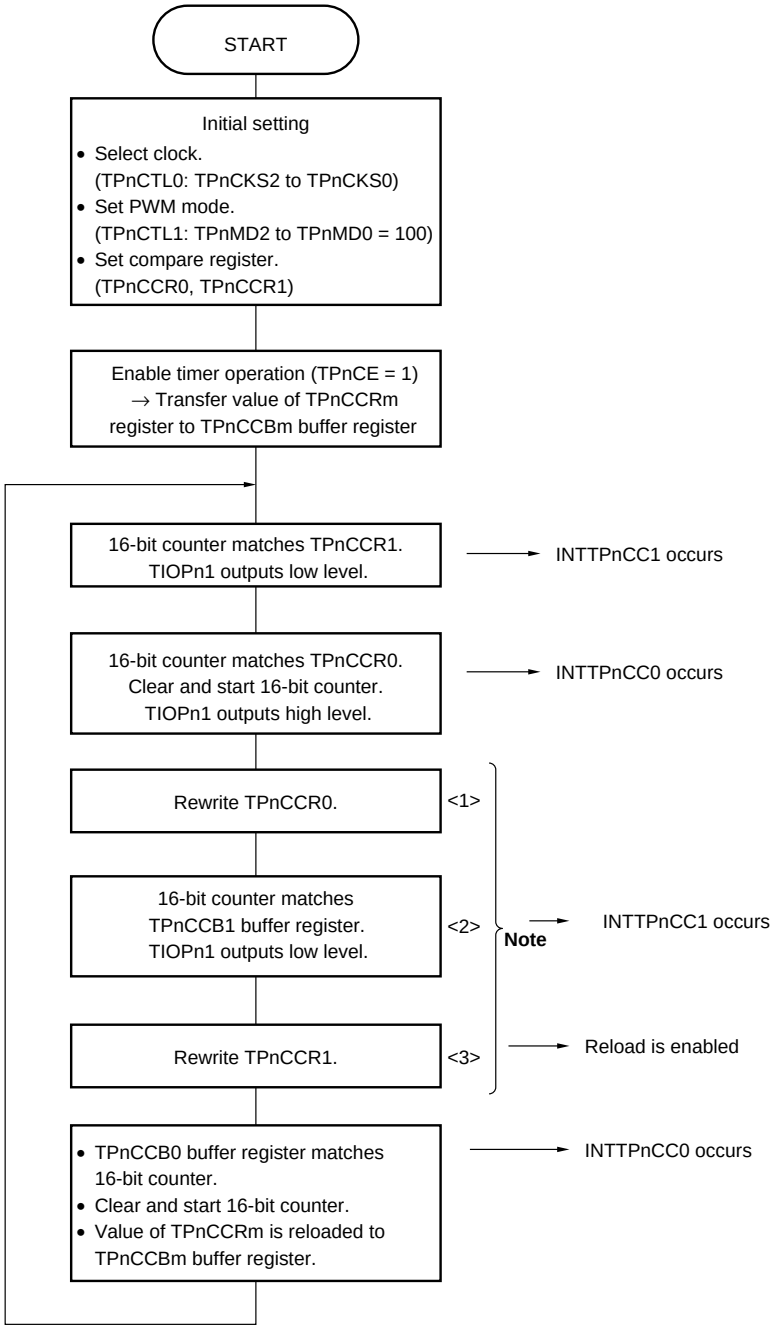


Remark: n=0 to 2

m = 0, 1

Figure 7-19: Flowchart of Basic Operation in PWM Mode (2/2)

(b) When values of TPnCCR0 and TPnCCR1 registers are rewritten during timer operation



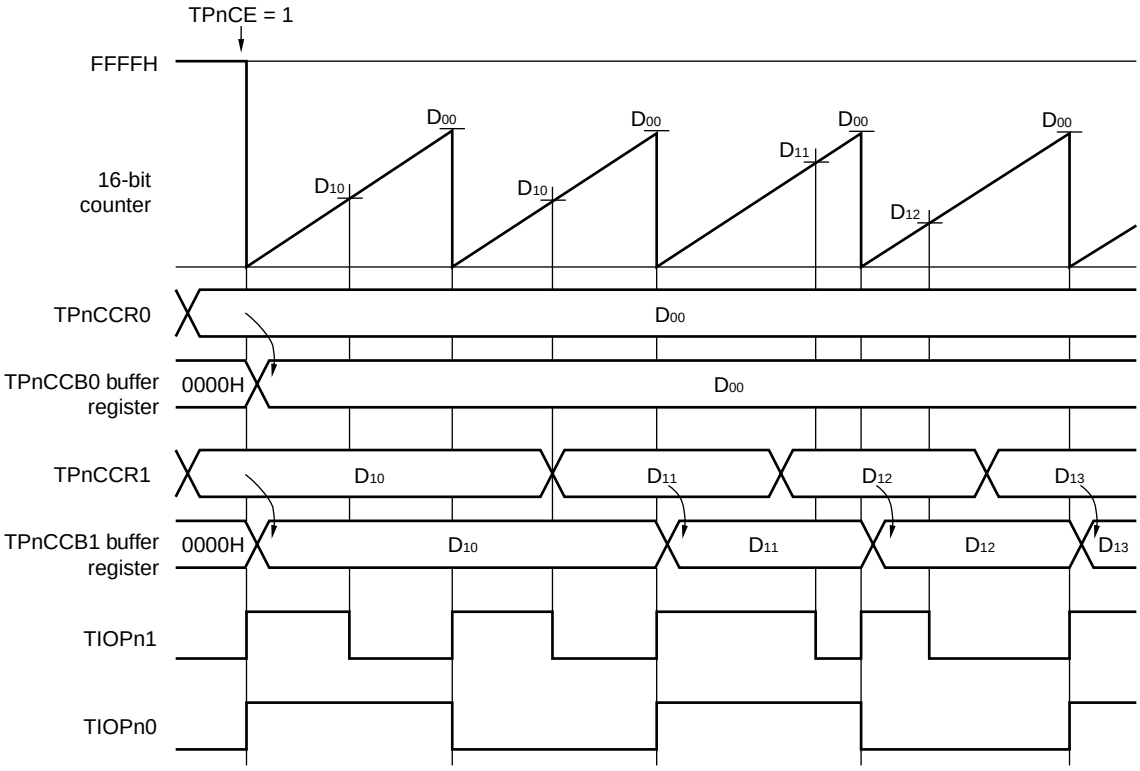
Note: The timing of <2> may differ depending on the rewrite timing of <1> and <3> and the value of TPnCCR1, but make sure that <3> comes after <1>.

Remark: n=0 to 2

m = 0, 1

Figure 7-20: Timing of Basic Operation in PWM Mode (1/2)

(a) When rewriting value of TPnCCR1 (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



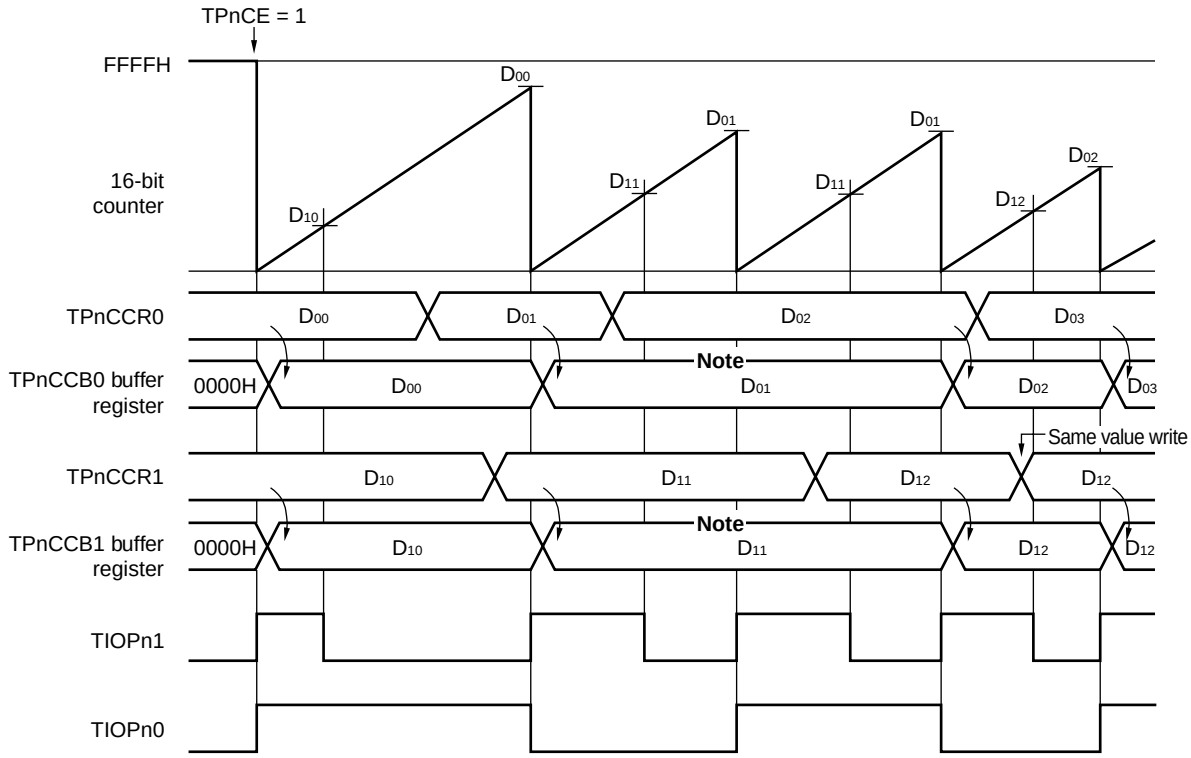
Remarks: 1. D_{00}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}, D_{13}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Duty factor of TIOPn1 output = (Set value of TPnCCR1 register) / (Set value of TP0CCR0 register + 1)
Active level width of TIOPn1 output = (Set value of TPnCCR0 register) × (Count clock cycle)
Cycle of TIOPn1 output = (Set value of TPnCCR0 register + 1) × (Count clock period)

3. n=0 to 2

Figure 7-20: Timing of Basic Operation in PWM Mode (2/2)

(b) When rewriting values of TPnCCR0 and TPnCCR1 (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Note: No value is reloaded because the TPnCCR1 register is not rewritten.

Remarks: 1. D_{00}, D_{01}, D_{02}, D_{03}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}, D_{13}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Duty factor of TIOPn1 output = (Set value of TPnCCR1 register) / (Set value of TP0CCR0 register + 1)
Active level width of TIOPn1 output = (Set value of TPnCCR0 register) × (Count clock cycle)
Cycle of TIOPn1 output = (Set value of TPnCCR0 register + 1) × (Count clock period)

3. n=0 to 2

7.5.5 Free-running mode (TPnMD2 to TPnMD0 = 101)

In the free-running mode, the 16-bit counter free-runs, and the bit that selects the capture or compare register function is made valid by the setting of the TPnCCS1 and TPnCCS0 bits, so that an interval function and a capture function can be realized.
Setting of the TPnCCS1 and TPnCCS0 bits of the TPnOPT0 register is valid only in the free-running mode.

TPnCCS1	Operation
0	TPnCCR1 register is used as compare register.
1	TPnCCR1 register is used as capture register.

TPnCCS0	Operation
0	TPnCCR0 register is used as compare register.
1	TPnCCR0 register is used as capture register.

- When TPnCCR1 register is used as compare register

When the value of the 16-bit counter matches the value of the TPnCCB1 buffer register in the free-running mode, an interrupt is generated (interval function).
Rewriting the value of the compare register is enabled during timer operation, and a value can be written to the register at any time (when the value to be compared is written to the register, it is synchronized with the internal clock and compared with the value of the 16-bit counter).
If timer output (TIOPn1) is enabled, TIOPn1 produces a toggle output when the value of the 16-bit counter matches the value of the TPnCCB1 buffer register.

- When TPnCCR1 register is used as capture register

The value of the 16-bit counter is stored in the TPnCCR1 register when the edge of the TIOPn1 pin is detected.

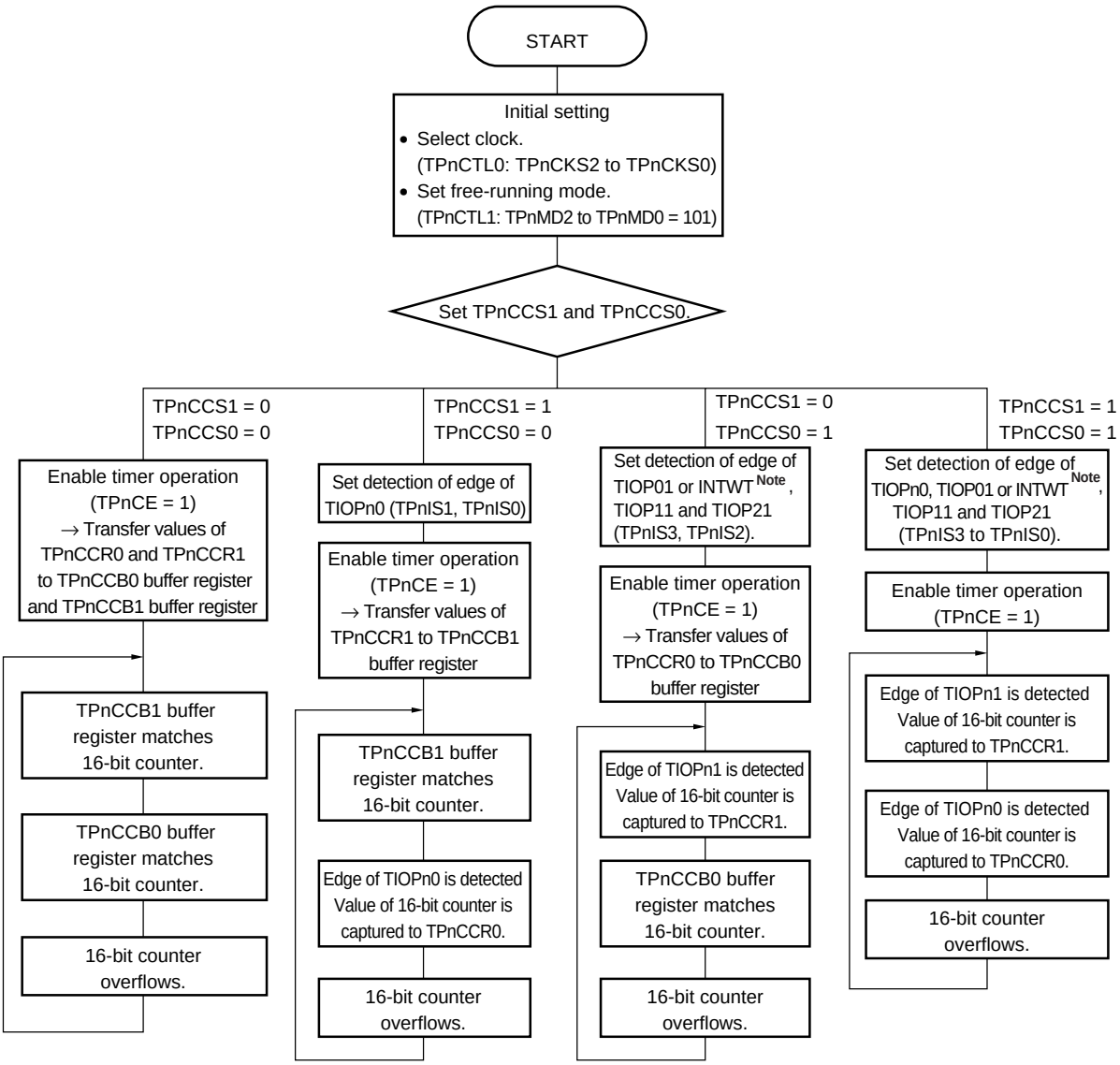
- When TPnCCR0 register is used as compare register

When the value of the 16-bit counter matches the value of the TPnCCB0 buffer register in the free-running mode, an interrupt is generated (interval function).
Rewriting the value of the compare register is enabled during timer operation, and a value can be written to the register at any time (when the value to be compared is written to the register, it is synchronized with the internal clock and compared with the value of the 16-bit counter).
If timer output (TIOPn0) is enabled, TIOPn0 produces a toggle output when the value of the 16-bit counter matches the value of the TPnCCB0 buffer register.

- When TPnCCR0 register is used as capture register

The value of the 16-bit counter is stored in the TPnCCR0 register when the edge of the TIOPn0 pin is detected.

Figure 7-21: Flowchart of Basic Operation in Free-Running Mode



Note: INTWT is the interrupt signal from watch timer.
The selection between TIOP01 and INTWT is made with the bit CKCL of the input switch control register (ISC). Please see **7.4 Control Registers**, **(6)"Input switch control register (ISC)" on page 185**.

Remark: n=0 to 2

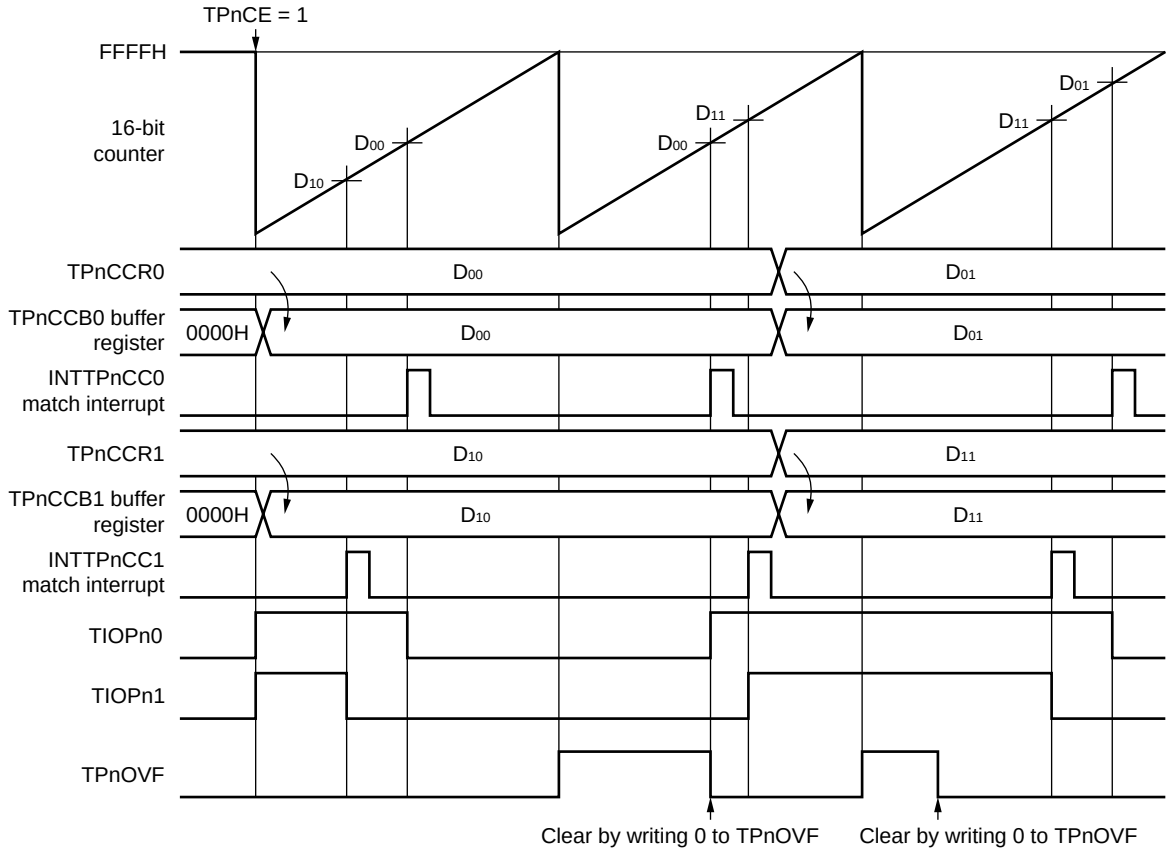
(1) When TPnCCS1 = 0 and TPnCCS0 = 0 (compare function)

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH, and continues counting up in the free-running mode until TPnCE is cleared to 0. If a value is written to the TPnCCR0 and TPnCCR1 registers in this mode, it is transferred to the TPnCCB0 and TPnCCB1 buffer registers (anytime write). If TPnOEm is set to 1, TIOPnm produces a toggle output when the value of the 16-bit counter matches the value of the TPnCCBm buffer register.

Remark: n=0 to 2

m = 0, 1

Figure 7-22: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 0) (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}: Set value of TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}: Set value of TPnCCR1 register (0000H to FFFFH)

2. Toggle width of TIOPn0 output = (Set value of TPnCCR0 register) × (Count clock cycle)
Toggle width of TIOPn1 output = (Set value of TPnCCR1 register)

3. TIOPnm output goes high when counting is started.

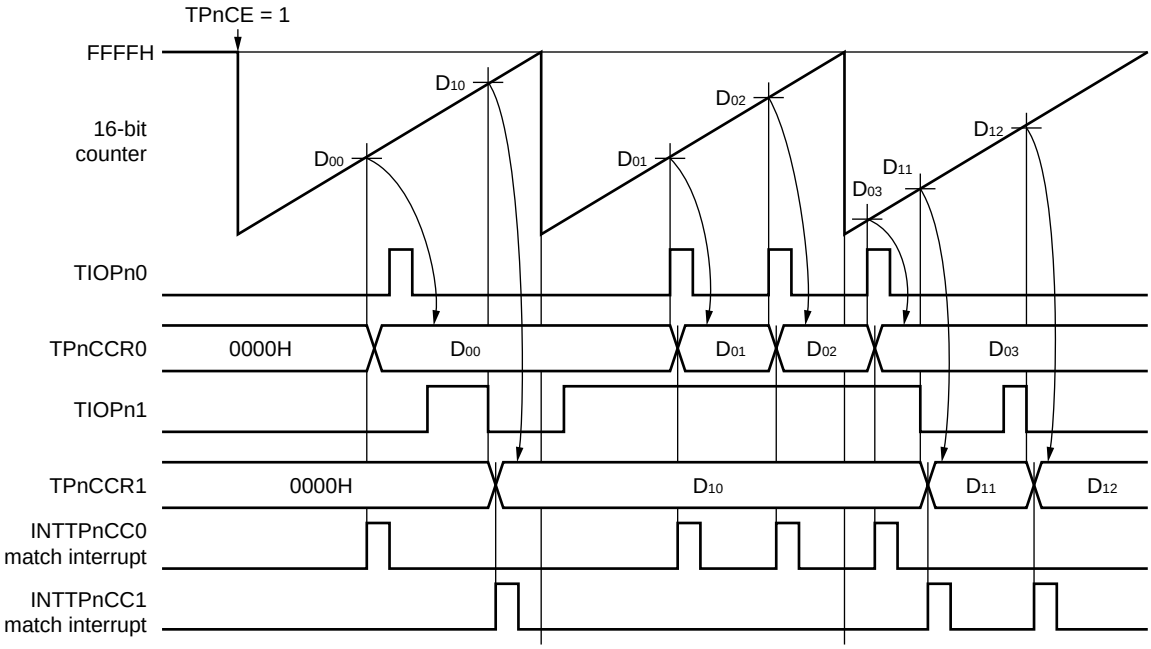
4. n=0 to 2

m = 0, 1

(2) When TPnCCS1 = 1 and TPnCCS0 = 1 (capture function)

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH, and continues counting up in the free-running mode until TPnCE is cleared to 0. The value captured by the capture trigger is written to the TPnCCR0 and TPnCCR1 registers.
Capturing close to an overflow (FFFFH) is judged using the overflow flag (TPnOVF).
However, if the interval of the capture trigger is such that the overflow occurs twice (two or more cycles of free-running), the TPnOVF flag cannot be used for judgment.

Figure 7-23: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 1, TPnCCS0 = 1) (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}, D_{02}, D_{03}: Value captured to TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Value captured to TPnCCR1 register (0000H to FFFFH)

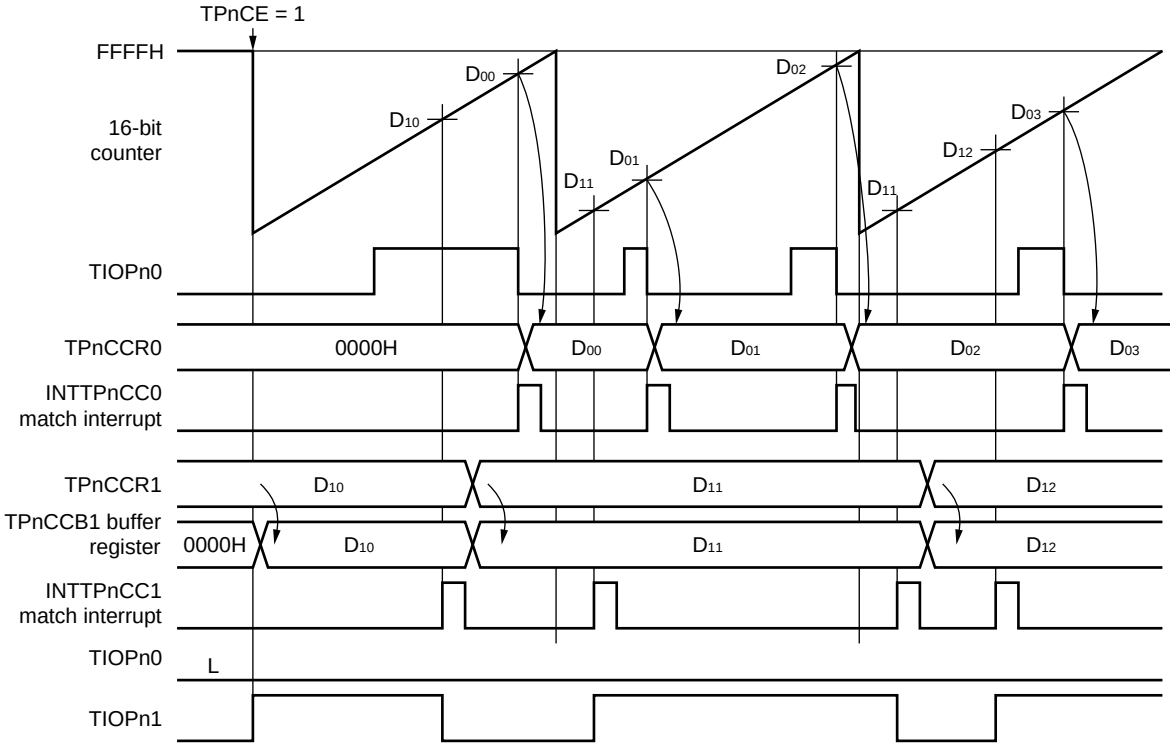
2. TIOPn0: Rising edge is detected (TPnIS1, TPnIS0 = 01).
TIOPn1: Falling edge is detected (TPnIS3, TPnIS2 = 10).

3. n=0 to 2

(3) When TPnCCS1 = 0 and TPnCCS0 = 1

When TPnCE is set to 1, the 16-bit counter counts from 0000H to FFFFH, and continues counting up in the free-running mode until TPnCE is cleared to 0. The TPnCCR1 register is used as a compare register. If TPnOE1 is set to 1, TIOPn1 produces a toggle output when the value of the 16-bit counter matches the set value of the TPnCCR1 register.

Figure 7-24: Timing of Basic Operation in Free-Running Mode (TPnCCS1 = 0, TPnCCS0 = 1) (TPnOE0 = 1, TPnOE1 = 1, TPnOL0 = 0, TPnOL1 = 0)



Remarks: 1. D_{00}, D_{01}, D_{02}, D_{03}: Value captured to TPnCCR0 register (0000H to FFFFH)
D_{10}, D_{11}, D_{12}: Value captured to TPnCCR1 register (0000H to FFFFH)

2. TIOPn0: Falling edge is detected (TPnIS1, TPnIS0 = 10).

3. n=0 to 2

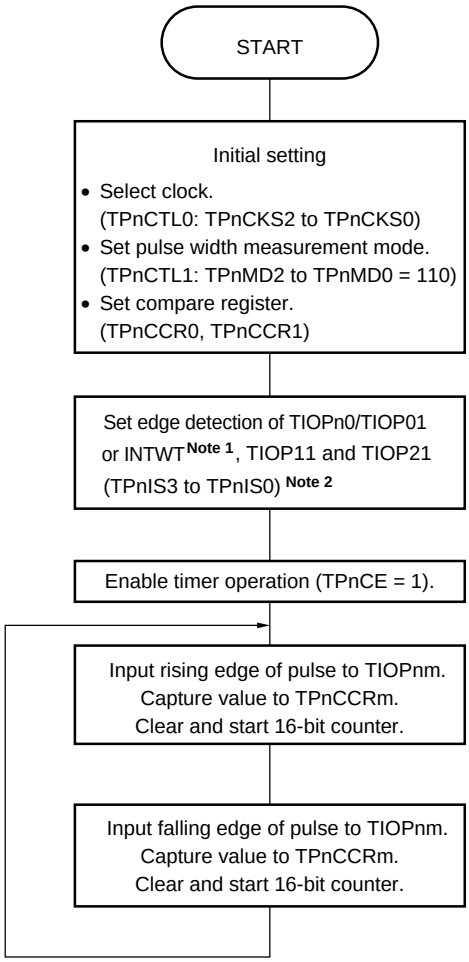
(4) Overflow flag

When the counter overflows from FFFFH to 0000H in the free-running mode, the overflow flag (TPnOVF) is set to 1.
The overflow flag is cleared by the CPU by writing 0 to it.

7.5.6 Pulse width measurement mode (TPnMD2 to TPnMD0 = 110)

In the pulse width measurement mode, free-running counting is performed. The value of the 16-bit counter is captured to capture register 0 (TPnCCR0) when both the rising and falling edges of the TIOPn0 pin are detected, and the 16-bit counter is cleared to 0000H. In this way, the external input pulse width can be measured.
To measure a long pulse width that exceeds the overflow of the 16-bit counter, use the overflow flag for detection. A pulse width that causes overflow to occur twice or more cannot be measured. Adjust the operating frequency of the 16-bit counter. When the edge of the TIOPn1 pin is detected, the value of the 16-bit counter is stored in capture register 1 (TPnCCR1), and the 16-bit counter is cleared.

Figure 7-25: Flowchart of Basic Operation in Pulse Width Measurement Mode

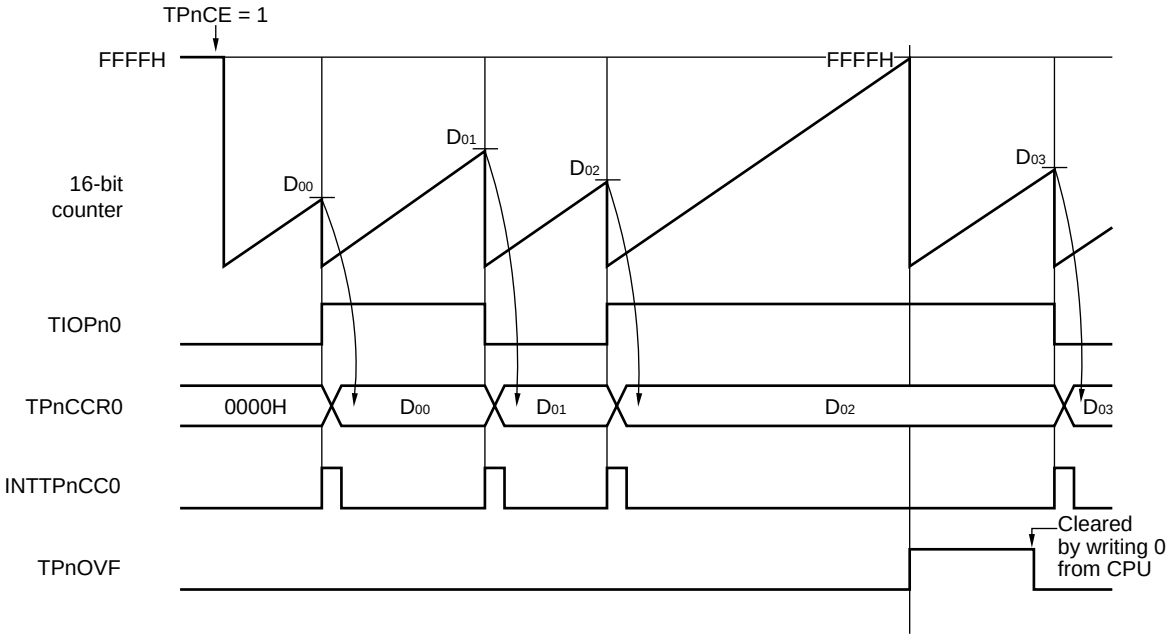


Notes: 1. INTWT is the interrupt signal from watch timer.
The selection between TIOP01 and INTWT is made with the bit CKCL of the input switch control register (ISC). Please see **7.4 Control Registers**, **(6)"Input switch control register (ISC)" on page 185**.

2. An external pulse can be input from either TIOPn0 or TIOP01 or INTWT, TIOP11 or TIOP21. Only one of them can be used. Specify that both the rising and falling edges are detected. Specify that the input edge of an external pulse input that is not used is not detected.

Remark: n=0 to 2
m = 0, 1

Figure 7-26: Timing of Basic Operation in Pulse Width Measurement Mode (TPnOE0 = 0, TPnOE1 = 0, TPnOL0 = 0, TPnOL1 = 0)



Remarks: **1.** D_{00}, D_{01}, D_{02}, D_{03}: Value captured to TPnCCR0 register (0000H to FFFFH)

2. TIOPn0: Both the rising and falling edges are detected (TPnIS1, TPnIS0 = 11).

3. n=0 to 2

7.6 Timer Synchronized Operation Function

Timer P has a timer synchronized operation function (tuned operation mode).
The three timers P that can be synchronized are listed in Table 7-2.

Table 7-2: Tuned Operation Mode of Timers

Master Timer	Slave Timer	
TMP0	TMP1	TMP2

Cautions: 1. The tuned operation mode is enabled or disabled by the TPnSYE bit of the TPnCTL1 register.

2. Set the tuned operation mode using the following procedure.
<1> Set the TPnSYE bit of the TPnCTL1 register of the slave timer to enable the tuned operation.
Set the TPnMD2 to TPnMD0 bits of the TPnCTL1 register of the slave timer to the free-running mode.
<2> Set the master timer mode by using the TPnMD2 to TPnMD0 bits of the TPnCTL1 register to a mode, which is usable in the tuned operation mode. Please see Table 7-3, “Timer Modes Usable in Tuned Operation Mode,” on page 204.
At this time, do not set the TP0SYE bit of the TP0CTL1of the master timer.
<3> Set the compare register value of the master and slave timers.
<4> Set the TPnCE bit of the TPnCTL0 register of the slave timer to enable operation on the internal operating clock.
<5> Set the TPnCE bit of the TPnCTL0 register of the master timer to enable operation on the internal operating clock.

Tables 7-3 and 7-4 show the timer modes that can be used in the tuned operation mode.

Table 7-3: Timer Modes Usable in Tuned Operation Mode

Master Timer	Free-Running Mode	PWM Mode
TMP0	√	√
TMP1	√	√
TMP2	√	√

Table 7-4: Timer Output Functions

Timer	Pin	Free-Running Mode		PWM Mode[Note]	
		Tuning OFF	Tuning ON	Tuning OFF	Tuning ON
TMP0 (master)	TIOP00	PPG	←	Toggle	Toggle
	TIOP01	PPG	←	PWM	PWM
TMP1 (slave)	TIOP10	PPG	←	Toggle	PWM
	TIOP11	PPG	←	PWM	PWM
TMP2 (slave)	TIOP20	PPG	←	Toggle	PWM
	TIOP21	PPG	←	PWM	PWM

Remarks: 1. The timing of transmitting data from the compare register of the master timer to the compare register of the slave timer is as follows:
PPG: CPU write timing
Toggle, PWM: Timing at which timer counter and compare register match TOPn0

2. n = 0 to 2

Note: The compare values of the slave timers will become valid after rewriting of the compare values of the master timer.

Figure 7-27: Tuned Operation Image (TMP0, TMP1, TMP2)

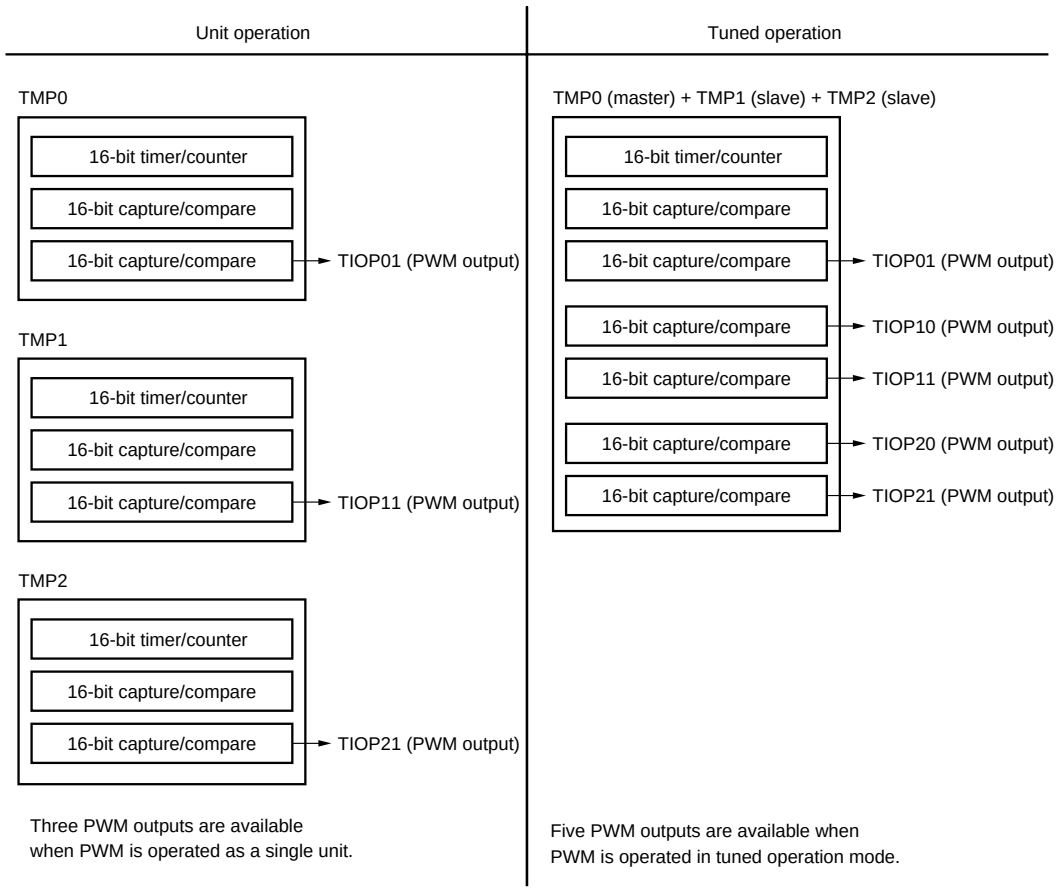
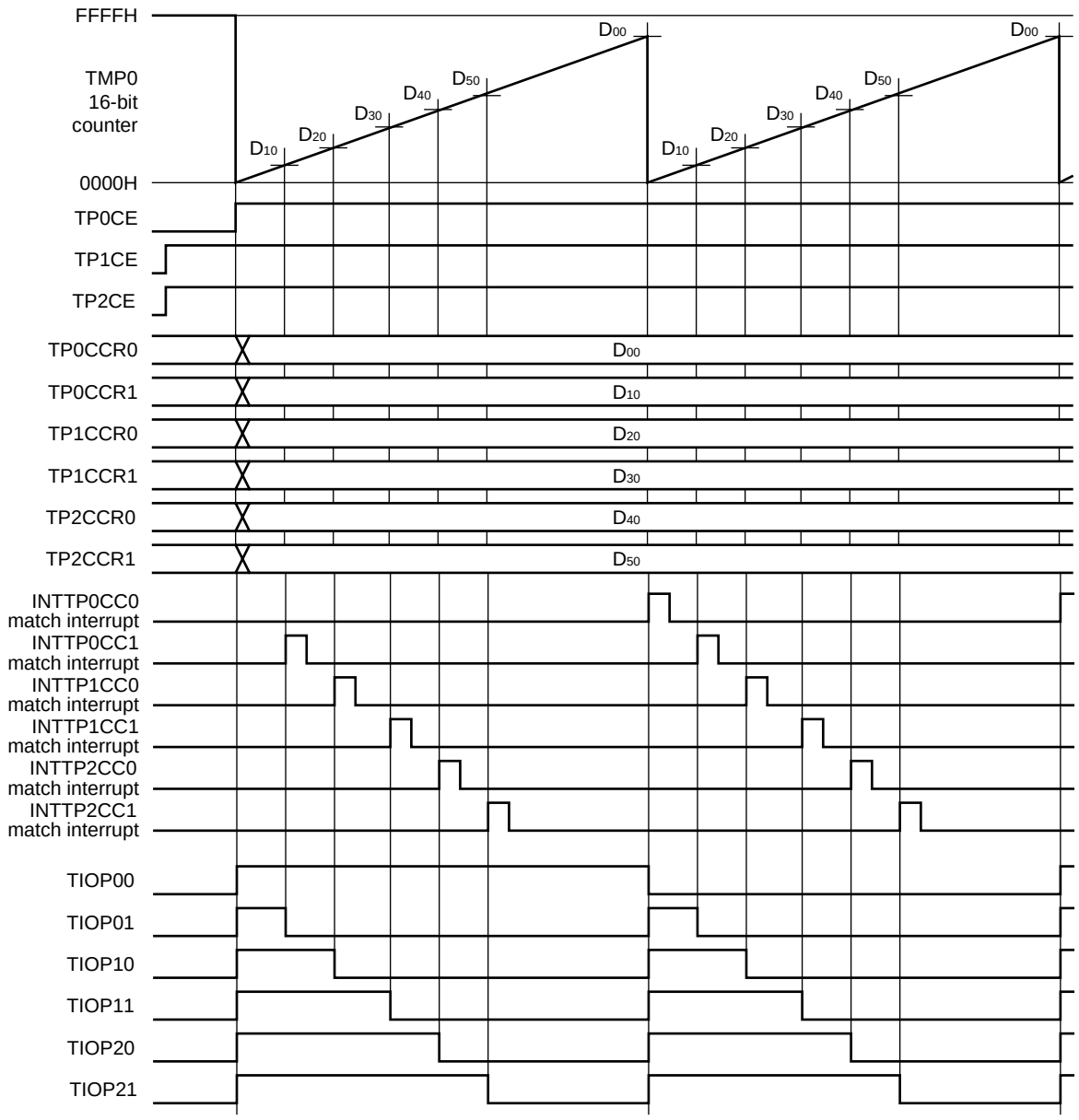


Figure 7-28: Basic Operation Timing of Tuned PWM Function (TMP0, TMP1, TMP2)



Chapter 8 8-Bit Timer/Event Counters 50 and 51 (µPD780822B Subseries)

8.1 8-Bit Timer/Event Counters 50 and 51 Functions

The timer 50 and 51 have the following functions:

- Interval timer
- External event counter
- Square-wave output
- PWM output.

(1) 8-bit interval timer

Interrupts are generated at the present time intervals.

Table 8-1: 8-Bit Timer/Event Counter 50 Interval Times

Minimum Interval Width	Maximum Interval Width	Resolution
$2^3 \times 1/f_{PRS}$ (500 ns)	$2^{11} \times 1/f_{PRS}$ (128 µs)	$2^3 \times 1/f_{PRS}$ (500 ns)
$2^5 \times 1/f_{PRS}$ (2 µs)	$2^{13} \times 1/f_{PRS}$ (512 µs)	$2^5 \times 1/f_{PRS}$ (2 µs)
$2^7 \times 1/f_{PRS}$ (8 µs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 µs)
$2^8 \times 1/f_{PRS}$ (16 µs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 µs)
$2^9 \times 1/f_{PRS}$ (32 µs)	$2^{17} \times 1/f_{PRS}$ (8 ms)	$2^9 \times 1/f_{PRS}$ (32 µs)
$2^{11} \times 1/f_{PRS}$ (128 µs)	$2^{19} \times 1/f_{PRS}$ (32 ms)	$2^{11} \times 1/f_{PRS}$ (128 µs)

Table 8-2: 8-Bit Timer/Event Counter 51 Interval Times

Minimum Interval Width	Maximum Interval Width	Resolution
$2^4 \times 1/f_{PRS}$ (1 µs)	$2^{12} \times 1/f_{PRS}$ (256 µs)	$2^4 \times 1/f_{PRS}$ (1 µs)
$2^6 \times 1/f_{PRS}$ (4 µs)	$2^{14} \times 1/f_{PRS}$ (1 ms)	$2^6 \times 1/f_{PRS}$ (4 µs)
$2^7 \times 1/f_{PRS}$ (8 µs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 µs)
$2^8 \times 1/f_{PRS}$ (16 µs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 µs)
$2^{10} \times 1/f_{PRS}$ (64 µs)	$2^{18} \times 1/f_{PRS}$ (16 ms)	$2^{10} \times 1/f_{PRS}$ (64 µs)
$2^{12} \times 1/f_{PRS}$ (256 µs)	$2^{20} \times 1/f_{PRS}$ (65 ms)	$2^{12} \times 1/f_{PRS}$ (256 µs)

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. Values in parentheses when operated at $f_{PRS} = f_X = 16.0$ MHz.

(2) External event counter

The number of pulses of an externally input signal can be measured.

(3) Square-wave output

A square wave with any selected frequency can be output.

Table 8-3: 8-Bit Timer/Event Counter 50 Square-Wave Output Ranges

Minimum Interval Width	Maximum Interval Width	Resolution
$2^3 \times 1/f_{PRS}$ (500 ns)	$2^{11} \times 1/f_{PRS}$ (128 μs)	$2^3 \times 1/f_{PRS}$ (500 ns)
$2^5 \times 1/f_{PRS}$ (2 μs)	$2^{13} \times 1/f_{PRS}$ (512 μs)	$2^5 \times 1/f_{PRS}$ (2 μs)
$2^7 \times 1/f_{PRS}$ (8 μs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 μs)
$2^8 \times 1/f_{PRS}$ (16 μs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 μs)
$2^9 \times 1/f_{PRS}$ (32 μs)	$2^{17} \times 1/f_{PRS}$ (8 ms)	$2^9 \times 1/f_{PRS}$ (32 μs)
$2^{11} \times 1/f_{PRS}$ (128 μs)	$2^{19} \times 1/f_{PRS}$ (32 ms)	$2^{11} \times 1/f_{PRS}$ (128 μs)

Table 8-4: 8-Bit Timer/Event Counter 51 Square-Wave Output Ranges

Minimum Interval Width	Maximum Interval Width	Resolution
$2^4 \times 1/f_{PRS}$ (1 μs)	$2^{12} \times 1/f_{PRS}$ (256 μs)	$2^4 \times 1/f_{PRS}$ (1 μs)
$2^6 \times 1/f_{PRS}$ (4 μs)	$2^{14} \times 1/f_{PRS}$ (1 ms)	$2^6 \times 1/f_{PRS}$ (4 μs)
$2^7 \times 1/f_{PRS}$ (8 μs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 μs)
$2^8 \times 1/f_{PRS}$ (16 μs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 μs)
$2^{10} \times 1/f_{PRS}$ (64 μs)	$2^{18} \times 1/f_{PRS}$ (16 ms)	$2^{10} \times 1/f_{PRS}$ (64 μs)
$2^{12} \times 1/f_{PRS}$ (256 μs)	$2^{20} \times 1/f_{PRS}$ (65 ms)	$2^{12} \times 1/f_{PRS}$ (256 μs)

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. Values in parentheses when operated at $f_{PRS} = f_X = 16.0$ MHz.

(4) PWM output

TM50 and TM51 can generate an 8-bit resolution PWM output.

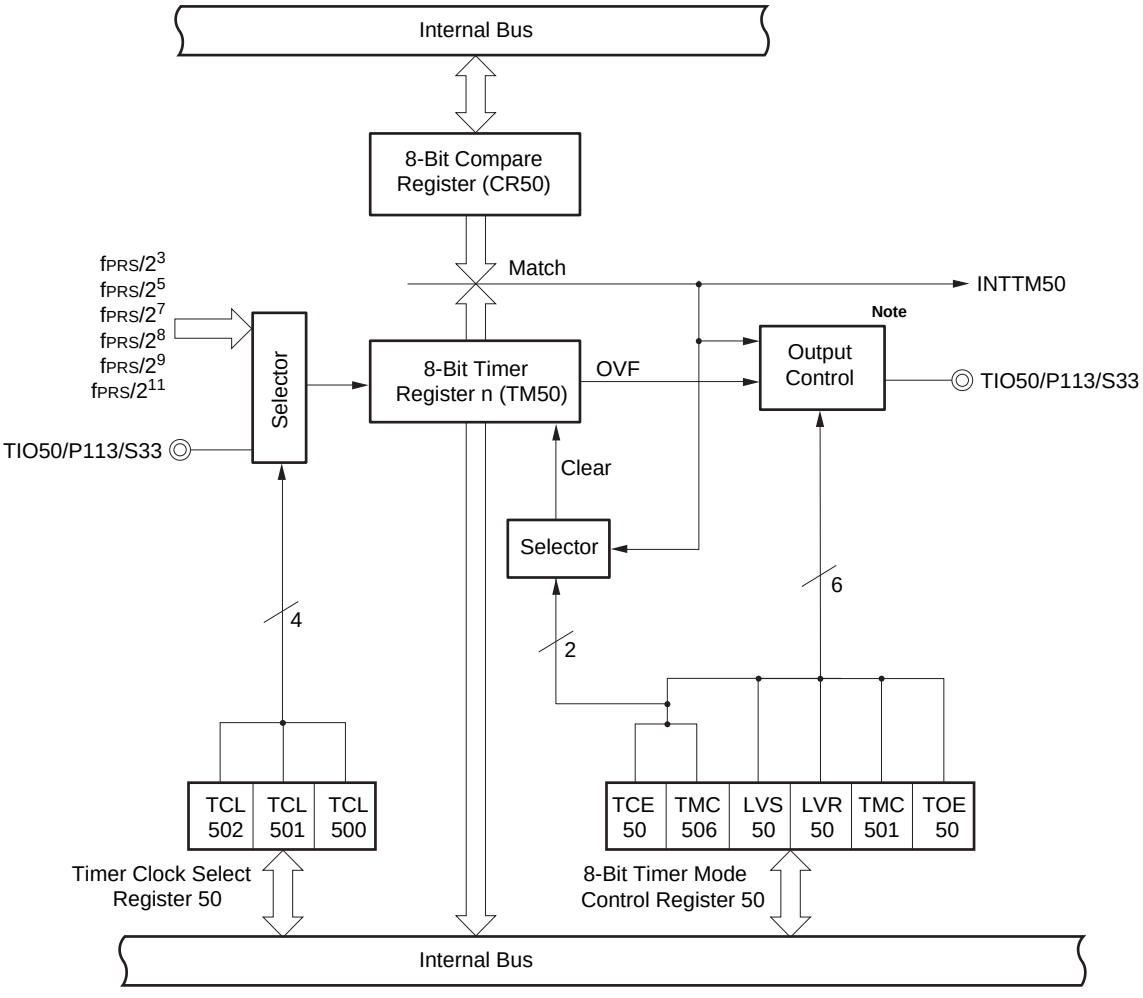
8.2 8-Bit Timer/Event Counters 50 and 51 Configurations

The 8-bit timer/event counters 50 and 51 consist of the following hardware.

Table 8-5: 8-Bit Timer/Event Counters 50 and 51 Configurations

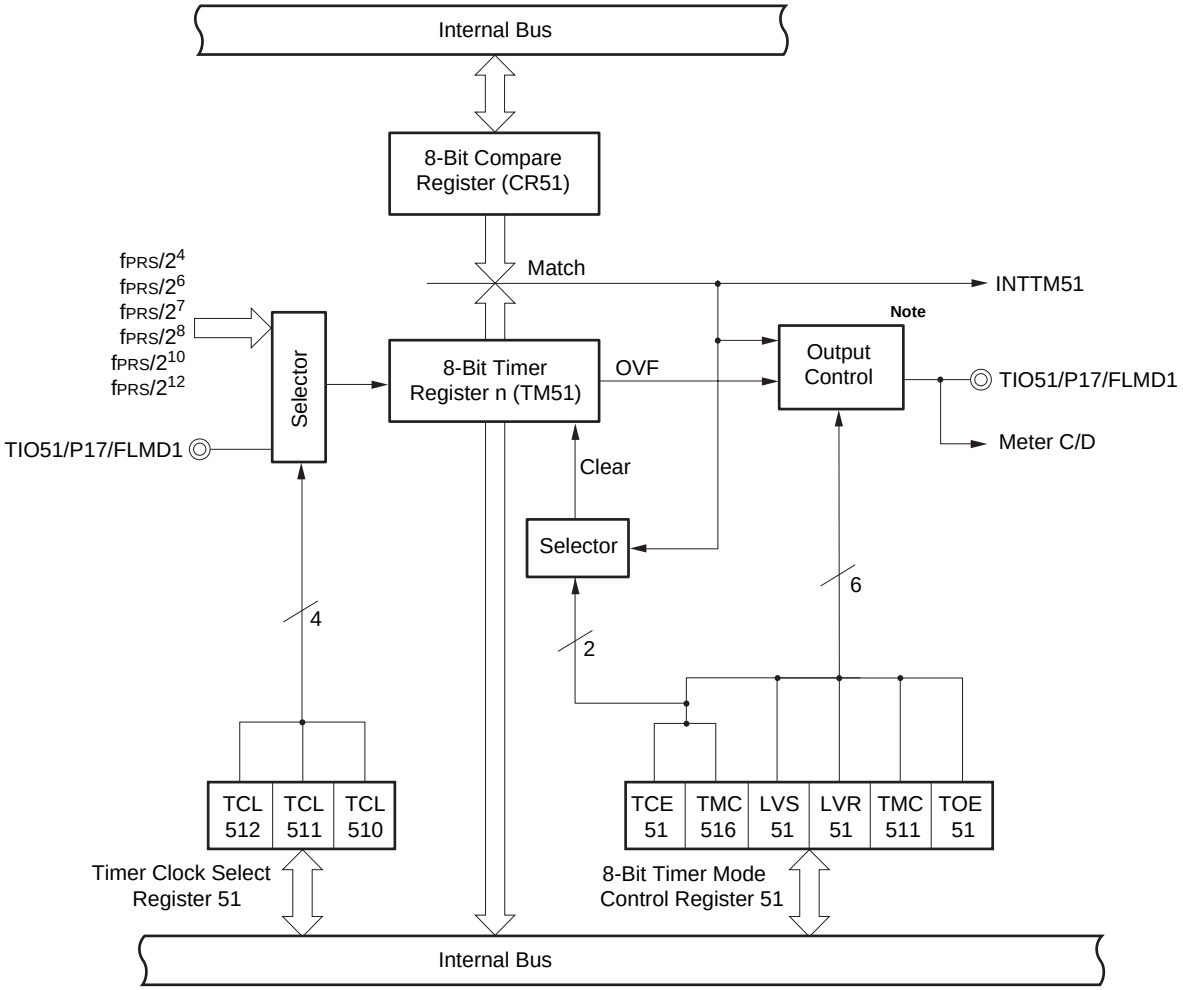
Item	Configuration
Timer register	8 bits x 2 (TM50, TM51)
Register	Compare register 8 bits x 2 (CR50, CR51)
Timer output	2 (TO50, TO51)
Control register	Timer clock select register 50 and 51 (TCL50, TCL51)
	8-bit timer mode control registers 50 and 51 (TMC50, TMC51)
	Port mode registers 1 and 11 (PM1, PM11)
	Port function register 11 (PF11)

Figure 8-1: 8-Bit Timer/Event Counter 50 Block Diagram



Note: Refer to Figure 8-3 for details of configurations of 8-bit timer/event counters 50 and 51 output control circuits.

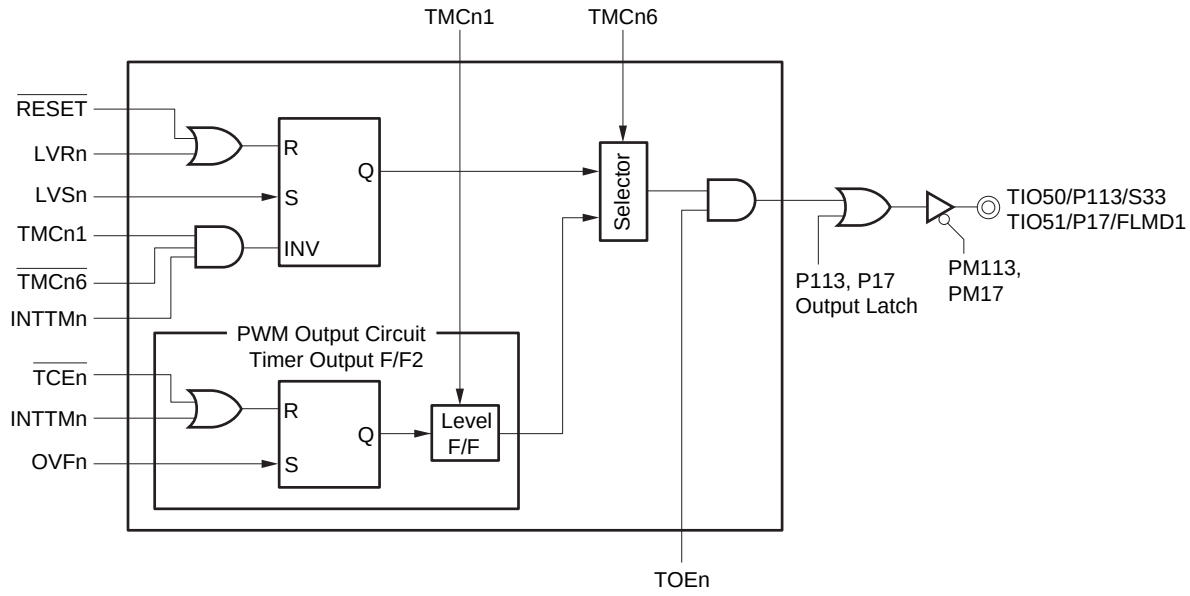
Figure 8-2: 8-Bit Timer/Event Counter 51 Block Diagram



Notes: 1. Refer to Figure 8-3 for details of configurations of 8-bit timer/event counters 50 and 51 output control circuits.

2. The output signal of the timer TM51 can be used as the clock input of the Meter controller/driver.

Figure 8-3: Block Diagram of 8-Bit Timer/Event Counters 50 and 51 Output Control Circuit



Remarks: 1. The section in the broken line is an output control circuit.

2. n = 50, 51

(1) Compare register 50 and 51 (CR50, CR51)

These 8-bit registers compare the value set to CR50 to 8-bit timer register 5 (TM50) count value, and the value set to CR51 to the 8-bit timer register 51 (TM51) count value, and, if they match, generate interrupts request (INTTM50 and INTTM51, respectively).
CR50 and CR51 are set with an 8-bit memory manipulation instruction. They cannot be set with a 16-bit memory manipulation instruction. The 00H to FFH values can be set.

$\overline{\text{RESET}}$ input sets CR50 and CR51 values to 00H.

Caution: To use PWM mode, set CRn value before setting TMCn (n = 50, 51) to PWM mode.

(2) 8-bit timer registers 50 and 51 (TM50, TM51)

These 8-bit registers count pulses.
TM50 and TM51 are read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TM50 and TM51 to 00H.

8.3 8-Bit Timer/Event Counters 50 and 51 Control Registers

The following four types of registers are used to control the 8-bit timer/event counters 50 and 51.

- Timer clock select register 50 and 51 (TCL50, TCL51)
- 8-bit timer mode control registers 50 and 51 (TMC50, TMC51)
- Port mode registers 1 and 11 (PM1, PM11)
- Port function register 11 (PF11)

(1) Timer clock select register 50 (TCL50)

This register sets count clocks of 8-bit timer register 50.
TCL50 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TCL50 to 00H.

Figure 8-4: Timer Clock Select Register 50 (TLC50) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	R/W	FF82H	00H

TCL502	TCL501	TCL500	8-bit Timer Register 50 Count Clock Selection
0	0	0	TI50 falling edge **Note**
0	0	1	TI50 rising edge **Note**
0	1	0	$f_{PRS}/2^3$ (2.0 MHz)
0	1	1	$f_{PRS}/2^5$ (500 kHz)
1	0	0	$f_{PRS}/2^7$ (125 kHz)
1	0	1	$f_{PRS}/2^8$ (62.5 kHz)
1	1	0	$f_{PRS}/2^9$ (31.25 kHz)
1	1	1	$f_{PRS}/2^{11}$ (7.8 kHz)
Other than above			Setting prohibited

Note: When clock is input from the external, timer output (PWM output) cannot be used.

Cautions: 1. When rewriting TCL50 to other data, stop the timer operation beforehand.

2. Set always bits 3 to 7 to "0".

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. TI50: 8-bit timer register 50 input pin

3. Values in parentheses apply to operation with $f_{PRS} = f_X = 16.0$ MHz

(2) Timer clock select register 51 (TCL51)

This register sets count clocks of 8-bit timer register 51.
TCL51 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TCL51 to 00H.

Figure 8-5: Timer Clock Select Register 51 (TLC51) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	R/W	FF86H	00H

TCL512	TCL511	TCL510	8-bit Timer Register 51 Count Clock Selection
0	0	0	TI51 falling edge **Note**
0	0	1	TI51 rising edge **Note**
0	1	0	$f_{PRS}/2^4$ (1.0 MHz)
0	1	1	$f_{PRS}/2^6$ (250 kHz)
1	0	0	$f_{PRS}/2^7$ (125 kHz)
1	0	1	$f_{PRS}/2^8$ (62.5 kHz)
1	1	0	$f_{PRS}/2^{10}$ (15.6 kHz)
1	1	1	$f_{PRS}/2^{12}$ (3.9 kHz)
Other than above			Setting prohibited

Note: When clock is input from the external, timer output (PWM output) cannot be used.

Cautions: 1. When rewriting TCL51 to other data, stop the timer operation beforehand.

2. Set always bits 3 to 7 to "0".

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. TI51: 8-bit timer register 51 input pin

3. Values in parentheses apply to operation with $f_{PRS} = f_X = 16.0$ MHz

(3) 8-bit timer mode control register 50 (TMC50)

This register enables/stops operation of 8-bit timer register 50, sets the operating mode of 8-bit timer register 50 and controls operation of 8-bit timer/event counter 50 output control circuit.
It selects the R-S flip-flop (timer output F/F 1, 2) setting/resetting, the active level in PWM mode, inversion enabling/disabling in modes other than PWM mode and 8-bit timer/event counter 5 timer output enabling/disabling.
TMC50 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC50 to 00H.

Figure 8-6: 8-Bit Timer Mode Control Register 50 (TMC50) Format

	<7>	6	5	4	<3>	<2>	1	<0>	R/W	Address	After Reset
TMC50	TCE50	TMC506	0	0	LVS50	LVR50	TMC501	TOE50	R/W	FF83H	00H

TOE50	8-Bit Timer/Event Counter 50 Output Control
0	Output disabled (Port mode)
1	Output enabled

TMC501	In PWM Mode	In Other Mode
	Active level selection	Timer output F/F1 control
0	Active high	Inversion operation disabled
1	Active low	Inversion operation enabled

LVS50	LVR50	8-Bit Timer/Event Counter 50 Timer Output F/F1 Status Setting
0	0	No change
0	1	Timer output F/F1 reset (0)
1	0	Timer output F/F1 set (1)
1	1	Setting prohibited

TMC506	8-Bit Timer/Event Counter 50 Operating Mode Selection
0	Clear & start mode on match of TM50 and CR50
1	PWM mode (free-running)

TCE50	8-Bit Timer Register 50 Operation Control
0	Operation Stop (TM50 clear to 0)
1	Operation Enable

Cautions: 1. Timer operation must be stopped before setting TMC50.

2. If LVS50 and LVR50 are read after data are set, they will be 0.

3. Be sure to set bit 4 and bit 5 to 0.

(4) 8-bit timer mode control register 51 (TMC51)

This register enables/stops operation of 8-bit timer register 51, sets the operating mode of 8-bit timer register 51 and controls operation of 8-bit timer/event counter 51 output control circuit.
It selects the R-S flip-flop (timer output F/F 1, 2) setting/resetting, active level in PWM mode, inversion enabling/disabling in modes other than PWM mode and 8-bit timer/event counter 51 timer output enabling/disabling.
TMC51 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC51 to 00H.

Figure 8-7: 8-Bit Timer Mode Control Register 51 (TMC51) Format

	<7>	6	5	4	<3>	<2>	1	<0>	R/W	Address	After Reset
TMC51	TCE51	TMC516	0	0	LVS51	LVR51	TMC511	TOE51	R/W	FF87H	00H

TOE51	8-Bit Timer/Event Counter 51 Output Control
0	Output disabled (Port mode)
1	Output enabled

TMC511	In PWM Mode	In Other Mode
	Active level selection	Timer output F/F1 control
0	Active high	Inversion operation disabled
1	Active low	Inversion operation enabled

LVS51	LVR51	8-Bit Timer/Event Counter 51 Timer Output F/F1 Status Setting
0	0	No change
0	1	Timer output F/F1 reset (0)
1	0	Timer output F/F1 set (1)
1	1	Setting prohibited

TMC516	8-Bit Timer/Event Counter 51 Operating Mode Selection
0	Clear & start mode on match of TM51 and CR51
1	PWM mode (free-running)

TCE51	8-Bit Timer Register 51 Operation Control
0	Operation Stop (TM51 clear to 0)
1	Operation Enable

Cautions: 1. Timer operation must be stopped before setting TMC51.

2. If LVS51 and LVR51 are read after data are set, they will be 0.

3. Be sure to set bit 5 to 0.

(5) Port mode register 1 (PM1)

This register sets port 1 input/output in 1-bit units.
When using the TIO51/P17 pin for timer output, set PM17 and the output latches of P17 to 0.
PM1 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM1 to FFH.

Figure 8-8: Port Mode Register 1 (PM1) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	R/W	FF21H	FFH

PM1n	PM1n Input/Output mode Selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(6) Port mode register 11 (PM11)

This register sets port 11 input/output in 1-bit units.
When using the TIO50/P113/S33 pin for timer output, set PM113 and the output latches of P113 to 0.
PM11 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM11 to FFH.

Figure 8-9: Port Mode Register 11 (PM11) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM11	1	1	1	1	PM113	PM112	PM111	PM110	R/W	FF2BH	FFH

PM11n	PM11n Input/Output mode Selection (n = 0 to 3)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(7) Port function register 11 (PF11)

This register sets the output buffer function of port 11 to LCD segment/port function in 1-bit units.
When using the TIO50/P113/S33 pin for timer output, set PM113 to 0.
PF11 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PF11 to 00H.

Figure 8-10: Port Function Register 11 (PF11) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PF11	0	0	0	0	PF113	PF112	PF111	PF110	R/W	FF1BH	00H

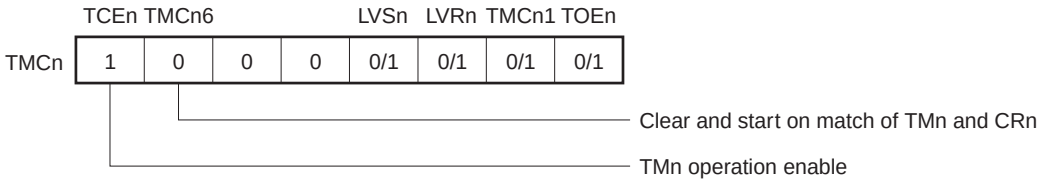
PF11n	PF11n Input/Output mode Selection (n = 0 to 3)
0	Port function
1	LCD segment function

8.4 8-Bit Timer/Event Counters 50 and 51 Operations

8.4.1 Interval timer operations (8-bit timer/event counter mode)

Setting the 8-bit timer mode control registers (TMC50 and TMC51) as shown in Figure 8-11 allows operation as an interval timer. Interrupts are generated repeatedly using the count value preset in 8-bit compare registers (CR50 and CR51) as the interval.
When the count value of the 8-bit timer register 50 or 51 (TM50, TM51) matches the value set to CR50 or CR51, counting continues with the TM50 or TM51 value cleared to 0 and the interrupt request signal (INTTM50, INTTM51) is generated.
Count clock of the 8-bit timer register 50 (TM50) can be selected with the timer clock select register 50 (TCL50) and count clock of the 8 bit timer register 51 (TM51) can be selected with the timer clock select register 51 (TCL51).

Figure 8-11: 8-Bit Timer Mode Control Register Settings for Interval Timer Operation



Remark: n = 50, 51

Setting Method

(1) Set each register
- TCL5n : Selects the count clock
- CR5n : Compare value
- TMC5n : Selects the clear and start mode when TM5n and CR5n match. (TMC5n = 0000xxx0B, x = don't care).

(2) When TCE5n = 1 is set, counting starts.

(3) When the values of TM5n and CR5n match, INTTM5n is generated (TM5n is cleared to 00H).

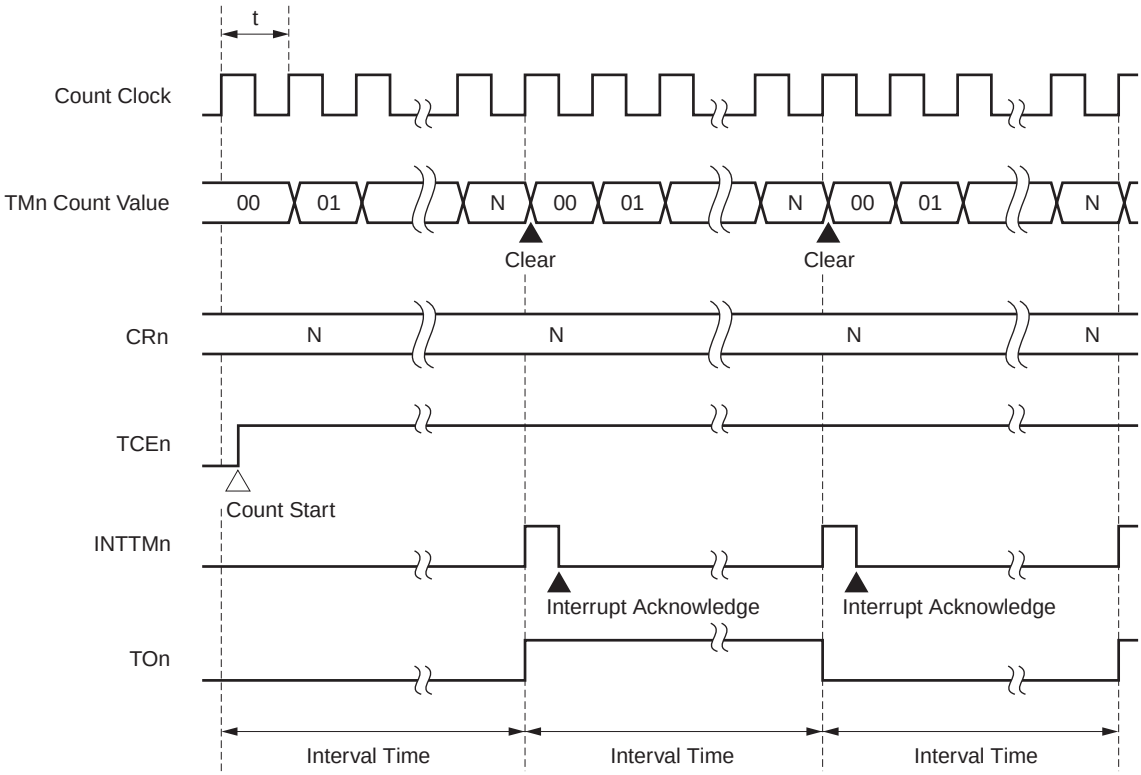
(4) Then, INTTM5n is repeatedly generated during the same interval. When counting stops, set TCE5n = 0.

Remarks: 1. 0/1: Setting 0 or 1 allows another function to be used simultaneously with the interval timer. See 8.3 (3)"8-bit timer mode control register 50 (TMC50)" on page 220 and (4)"8-bit timer mode control register 51 (TMC51)" on page 221 for details.

2. n = 0, 1

Figure 8-12: Interval Timer Operation Timings (1/3)

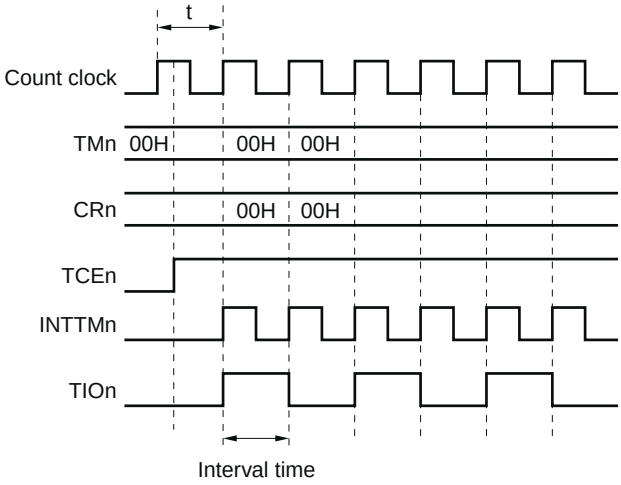
(a) When N = 00H to FFH



Remarks: 1. Interval time = (N + 1) x t: N = 00H to FFH

2. n = 50, 51

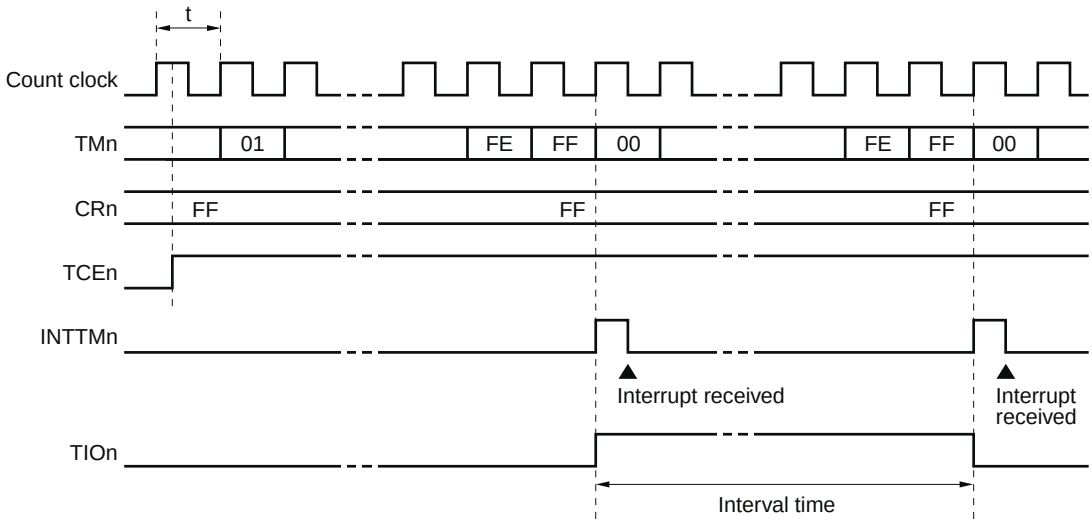
(b) When CRn = 00H



Remark: n = 50, 51

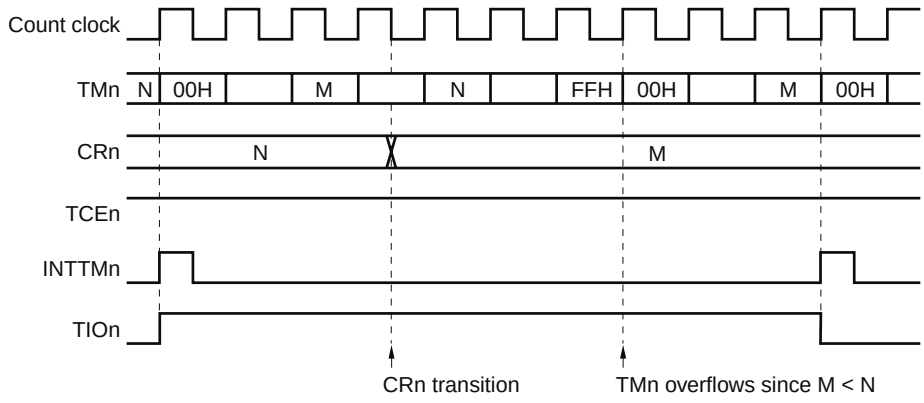
Figure 8-12: Interval Timer Operation Timings (2/3)

(c) When CRn = FFH



Remark: n = 50, 51

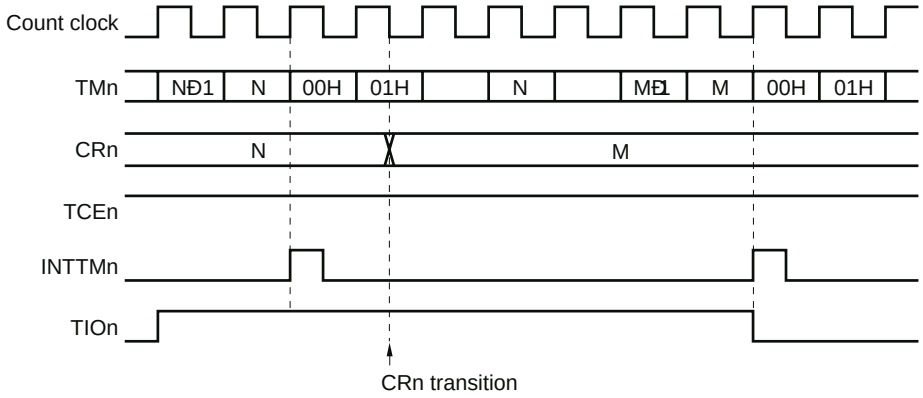
(d) Operated by CRn transition (M < N)



Remark: n = 50, 51

Figure 8-12: Interval Timer Operation Timings (3/3)

(e) Operated by CRn transition (M > N)



Remark: n = 50, 51

Table 8-6: 8-Bit Timer/Event Counters 50 Interval Times

TCLn2	TCLn1	TCLn0	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	TIn input cycle	$2^8 \times$ TIn input cycle	TIn input edge input cycle
0	0	1	TIn input cycle	$2^8 \times$ TIn input cycle	TIn input edge input cycle
0	1	0	$2^3 \times 1/f_{PRS}$ (500 ns)	$2^{11} \times 1/f_{PRS}$ (128 μs)	$2^3 \times 1/f_{PRS}$ (500 ns)
0	1	1	$2^5 \times 1/f_{PRS}$ (2 μs)	$2^{13} \times 1/f_{PRS}$ (512 μs)	$2^5 \times 1/f_{PRS}$ (2 μs)
1	0	0	$2^7 \times 1/f_{PRS}$ (8 μs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 μs)
1	0	1	$2^8 \times 1/f_{PRS}$ (16 μs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 μs)
1	1	0	$2^9 \times 1/f_{PRS}$ (32 μs)	$2^{17} \times 1/f_{PRS}$ (8 ms)	$2^9 \times 1/f_{PRS}$ (32 μs)
1	1	1	$2^{11} \times 1/f_{PRS}$ (128 μs)	$2^{19} \times 1/f_{PRS}$ (32 ms)	$2^{11} \times 1/f_{PRS}$ (128 μs)
Other than above			Setting prohibited		

Table 8-7: 8-Bit Timer/Event Counters 51 Interval Times

TCLn2	TCLn1	TCLn0	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	TIn input cycle	$2^8 \times$ TIn input cycle	TIn input edge input cycle
0	0	1	TIn input cycle	$2^8 \times$ TIn input cycle	TIn input edge input cycle
0	1	0	$2^4 \times 1/f_{PRS}$ (1 μs)	$2^{12} \times 1/f_{PRS}$ (256 μs)	$2^4 \times 1/f_{PRS}$ (1 μs)
0	1	1	$2^6 \times 1/f_{PRS}$ (4 μs)	$2^{14} \times 1/f_{PRS}$ (1 ms)	$2^6 \times 1/f_{PRS}$ (4 μs)
1	0	0	$2^7 \times 1/f_{PRS}$ (8 μs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 μs)
1	0	1	$2^8 \times 1/f_{PRS}$ (16 μs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 μs)
1	1	0	$2^{10} \times 1/f_{PRS}$ (64 μs)	$2^{18} \times 1/f_{PRS}$ (16 ms)	$2^{10} \times 1/f_{PRS}$ (64 μs)
1	1	1	$2^{12} \times 1/f_{PRS}$ (256 μs)	$2^{20} \times 1/f_{PRS}$ (65 ms)	$2^{12} \times 1/f_{PRS}$ (256 μs)
Other than above			Setting prohibited		

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

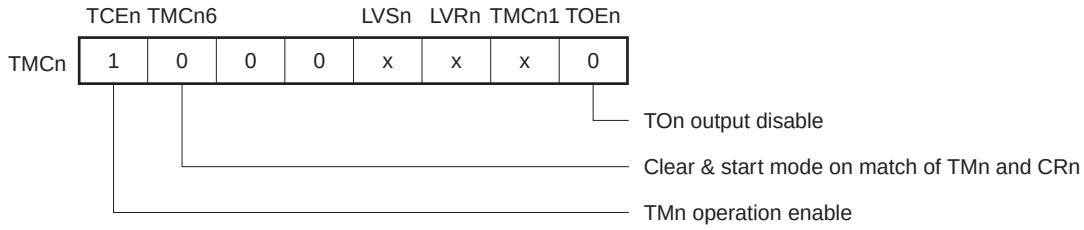
2. Values in parentheses apply to operation with $f_{PRS} = f_X = 16.0$ MHz

3. n = 50, 51

8.4.2 External event counter operation

The external event counter counts the number of external clock pulses to be input to the TIO50/P113/S33 and TIO51/P17 pins with 8-bit timer registers 50 and 51 (TM50 and TM51).
TM50 and TM51 are incremented each time the valid edge specified with timer clock select registers 50 and 51 (TCL50 and TCL51) is input. Either rising or falling edge can be selected.
When the TM50 and TM51 counted values match the values of 8-bit compare registers (CR50 and CR51), TM50 and TM51 are cleared to 0 and the interrupt request signals (INTTM50 and INTTM51) are generated.

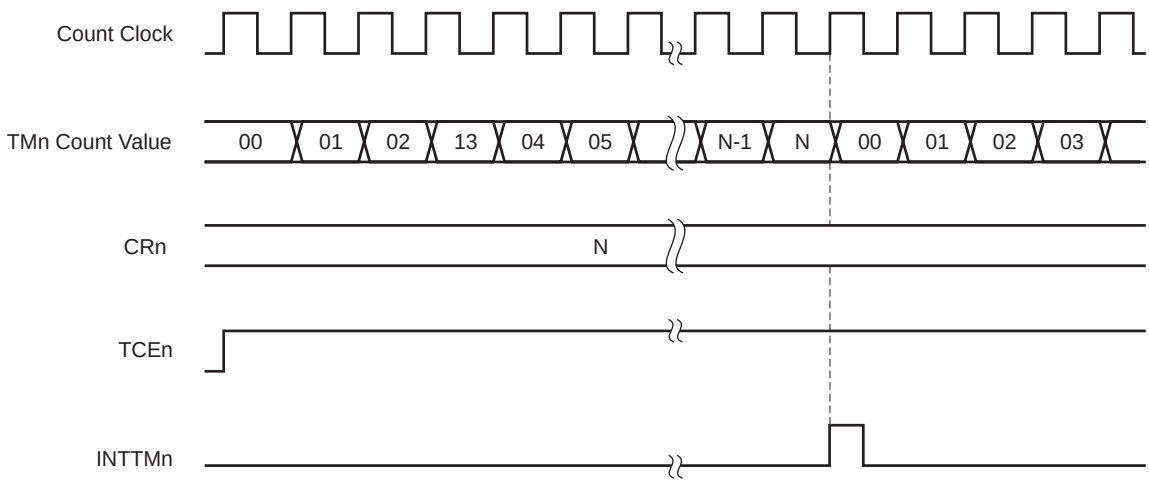
Figure 8-13: 8-Bit Timer Mode Control Register Setting for External Event Counter Operation



Remarks: 1. n = 50, 51

2. x: don't care

Figure 8-14: External Event Counter Operation Timings (with Rising Edge Specified)



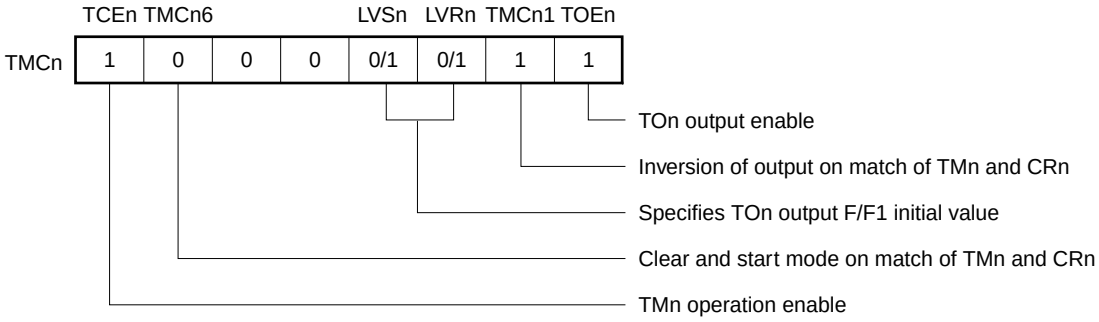
Remarks: 1. N = 00H to FFH

2. n = 50, 51

8.4.3 Square-wave output

A square wave with any selected frequency is output at intervals of the value preset to 8-bit compare registers (CR50 and CR51).
The TIO50/P113/S33 or TIO51/P17 pin output status is reversed at intervals of the count value preset to CR50 or CR51 by setting bit 1 (TMC501) and bit 0 (TOE50) of the 8-bit timer output control register 5 (TMC50), or bit 1 (TMC511) and bit 0 (TOE51) of the 8-bit timer mode control register 6 (TMC51) to 1. This enables a square wave of a selected frequency to be output.

Figure 8-15: 8-Bit Timer Mode Control Register Settings for Square-Wave Output Operation



Remark: n= 50, 51

Setting Method

(1) Set the registers
Set the port latch and port mode register to 0.
TCL5n : Selects the count clock
CR5n : Compare value
TMC5n : Selects the clear and start mode when TM5n and CR5n match.

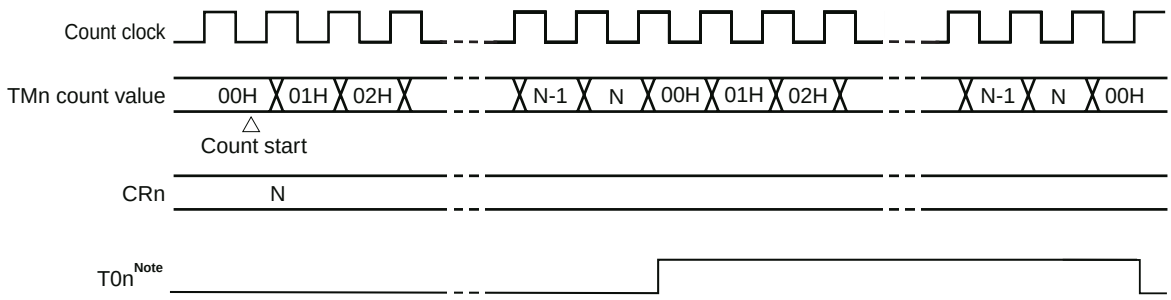
LVS5n	LVR5n	Setting State of Timer Output flip-flop
1	0	High level output
0	1	Low level output

Inversion of timer output flip-flop enabled
Timer output enabled → TOE5n = 1

(2) When TCE5n = 1 is set, the counter starts operating.
(3) When the values of TM5n and CR5n match, the timer output flip-flop inverts. Also, INTTM5n is generated and TM5n is cleared to 00H.
(4) Then, the timer output flip-flop is inverted for the same interval to output a square wave from TO5n.

Remark: n = 0, 1

Figure 8-16: Square-wave Output Operation Timing



Note: TOn output initial value can be set by bits 2 and 3 (LVRn, LVSn) of the 8-bit timer mode control register TCMn.

Remark: n = 50, 51

Table 8-8: 8-Bit Timer/Event Counters 50 Square-Wave Output Ranges (8-Bit Timer/Event Counter Mode)

TCL502	TCL501	TCL500	Minimum Pulse Time	Maximum Pulse Time	Resolution
0	1	0	$2^3 \times 1/f_{PRS}$ (500 ns)	$2^{11} \times 1/f_{PRS}$ (128 μs)	$2^3 \times 1/f_{PRS}$ (500 ns)
0	1	1	$2^5 \times 1/f_{PRS}$ (2 μs)	$2^{13} \times 1/f_{PRS}$ (512 μs)	$2^5 \times 1/f_{PRS}$ (2 μs)
1	0	0	$2^7 \times 1/f_{PRS}$ (8 μs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 μs)
1	0	1	$2^8 \times 1/f_{PRS}$ (16 μs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 μs)
1	1	0	$2^9 \times 1/f_{PRS}$ (32 μs)	$2^{17} \times 1/f_{PRS}$ (8 ms)	$2^9 \times 1/f_{PRS}$ (32 μs)
1	1	1	$2^{11} \times 1/f_{PRS}$ (128 μs)	$2^{19} \times 1/f_{PRS}$ (32 ms)	$2^{11} \times 1/f_{PRS}$ (128 μs)

Table 8-9: 8-Bit Timer/Event Counters 51 Square-Wave Output Ranges (8-Bit Timer/Event Counter Mode)

TCL502	TCL501	TCL500	Minimum Pulse Time	Maximum Pulse Time	Resolution
0	1	0	$2^4 \times 1/f_{PRS}$ (1 μs)	$2^{12} \times 1/f_{PRS}$ (256 μs)	$2^4 \times 1/f_{PRS}$ (1 μs)
0	1	1	$2^6 \times 1/f_{PRS}$ (4 μs)	$2^{14} \times 1/f_{PRS}$ (1 ms)	$2^6 \times 1/f_{PRS}$ (4 μs)
1	0	0	$2^7 \times 1/f_{PRS}$ (8 μs)	$2^{15} \times 1/f_{PRS}$ (2 ms)	$2^7 \times 1/f_{PRS}$ (8 μs)
1	0	1	$2^8 \times 1/f_{PRS}$ (16 μs)	$2^{16} \times 1/f_{PRS}$ (4 ms)	$2^8 \times 1/f_{PRS}$ (16 μs)
1	1	0	$2^{10} \times 1/f_{PRS}$ (64 μs)	$2^{18} \times 1/f_{PRS}$ (16 ms)	$2^{10} \times 1/f_{PRS}$ (64 μs)
1	1	1	$2^{12} \times 1/f_{PRS}$ (256 μs)	$2^{20} \times 1/f_{PRS}$ (65 ms)	$2^{12} \times 1/f_{PRS}$ (256 μs)

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

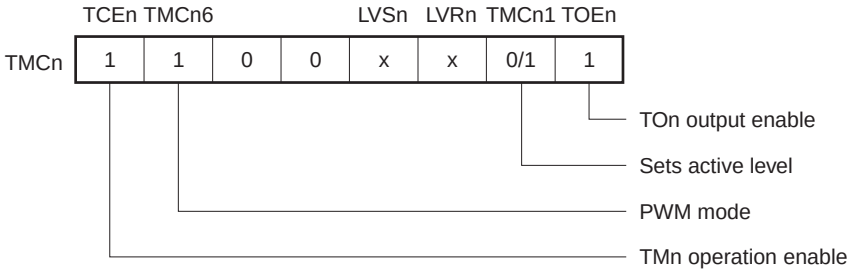
2. Values in parentheses apply to operation with $f_{PRS} = f_X = 16.0$ MHz

3. n = 50, 51

8.4.4 PWM output operations

Setting the 8-bit timer mode control registers (TMC50 and TMC51) as shown in Figure 8-17 allows operation as PWM output. Pulses with the duty rate determined by the values preset in 8-bit compare registers (CR50 and CR51) output from the TIO50/P113/S33 or TIO51/P17 pin.
Select the active level of PWM pulse with bit 1 of the 8-bit timer mode control register 50 (TMC50) or bit 1 of the 8-bit timer mode control register 51 (TMC51).
This PWM pulse has an 8-bit resolution. The pulse can be converted into an analog voltage by integrating it with an external low-pass filter (LPF). Count clock of the 8-bit timer register 50 (TM50) can be selected with the timer clock select register 50 (TCL50) and count clock of the 8-bit timer register 51 (TM51) can be selected with the timer clock select register 51 (TCL51).
PWM output enable/disable can be selected with bit 0 (TOE50) of TMC50 or bit 0 (TOE51) of TMC51.

Figure 8-17: 8-Bit Timer Control Register Settings for PWM Output Operation



Remark: n= 50, 51

Setting Method

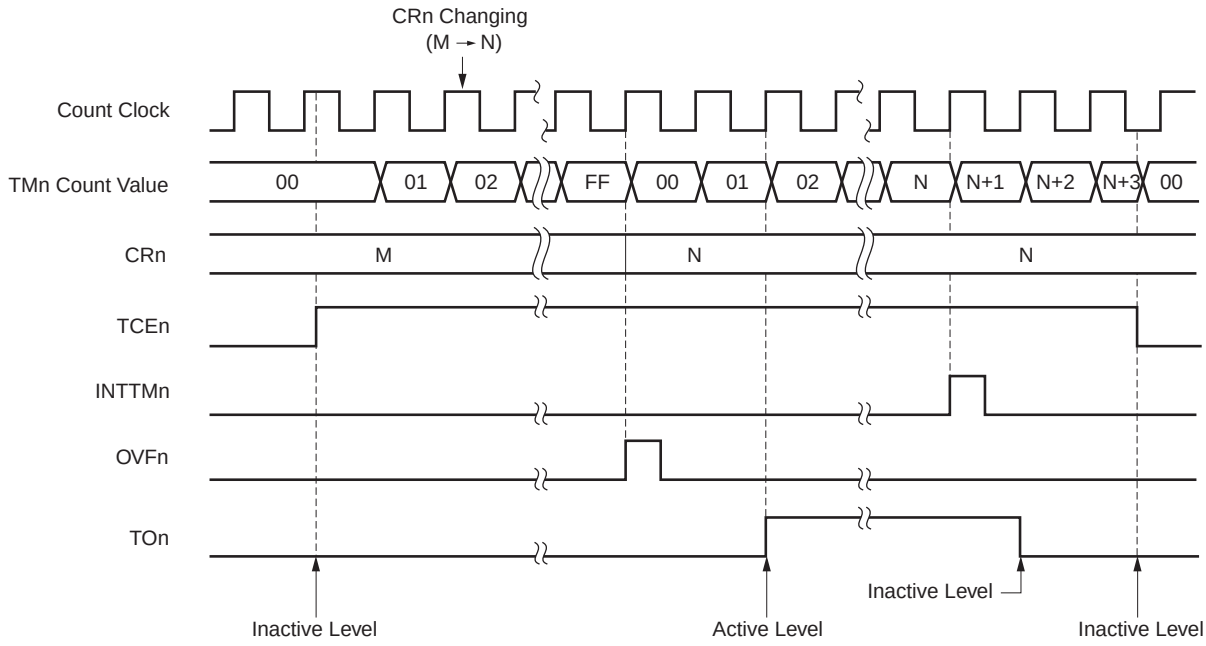
(1) Set the port latch and port mode register to "0".
(2) Set the active level width in the 8-bit compare register n (CR5n).
(3) Select the count clock in the timer clock selection register n (TCL5n).
(4) Set the active level in bit 1 (TMCn1) of TMCn.
(5) Count operation starts when bit 7 (TCEn) of TMCn is set to “1”.
Set TCEn to “0” to stop count operation.

PWM Output Operation

(1) When counting starts, the PWM output (output from TO5n) outputs the inactive level until an overflow occurs.
(2) When the overflow occurs, the active level specified in step (1) in the setting method is output. The active level is output until CRn and the count of the 8-bit counter n (TMn) match.
(3) The PWM output after CR5n and the count match is the inactive level until an overflow occurs again.
(4) Steps (2) and (3) repeat until counting stops.
(5) If counting is stopped by TCEn = 0, the PWM output goes to the inactive level.

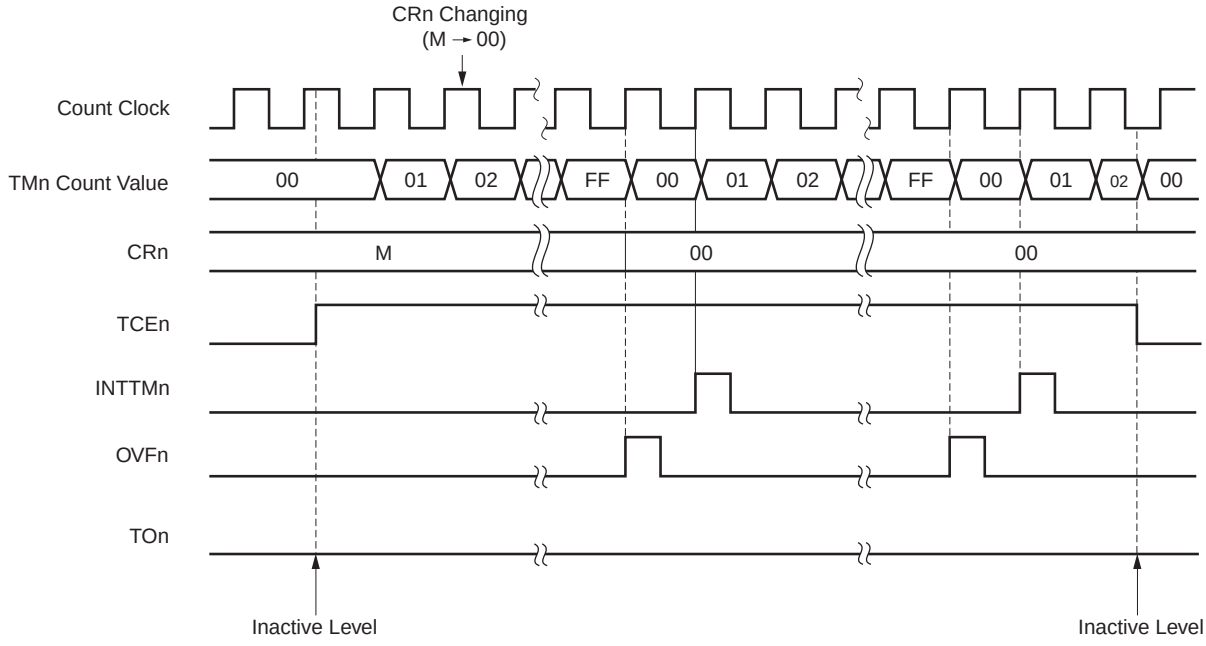
Remark: n = 0, 1

Figure 8-18: PWM Output Operation Timing (Active high setting)



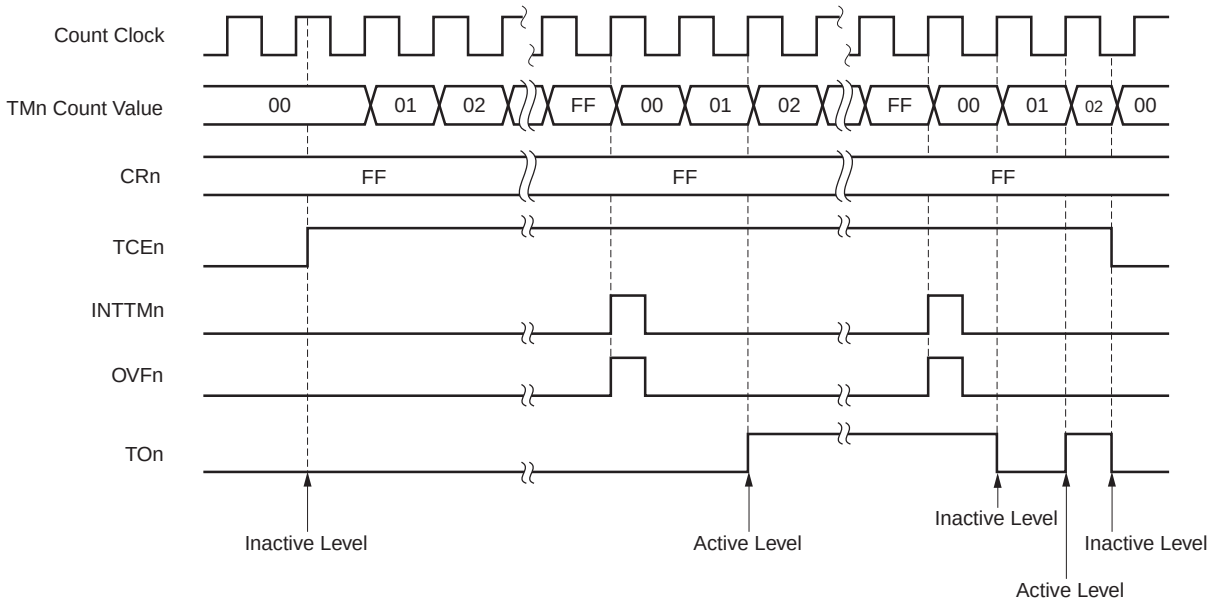
Remark: n = 50, 51

Figure 8-19: PWM Output Operation Timings (CRn = 00H, active high setting)



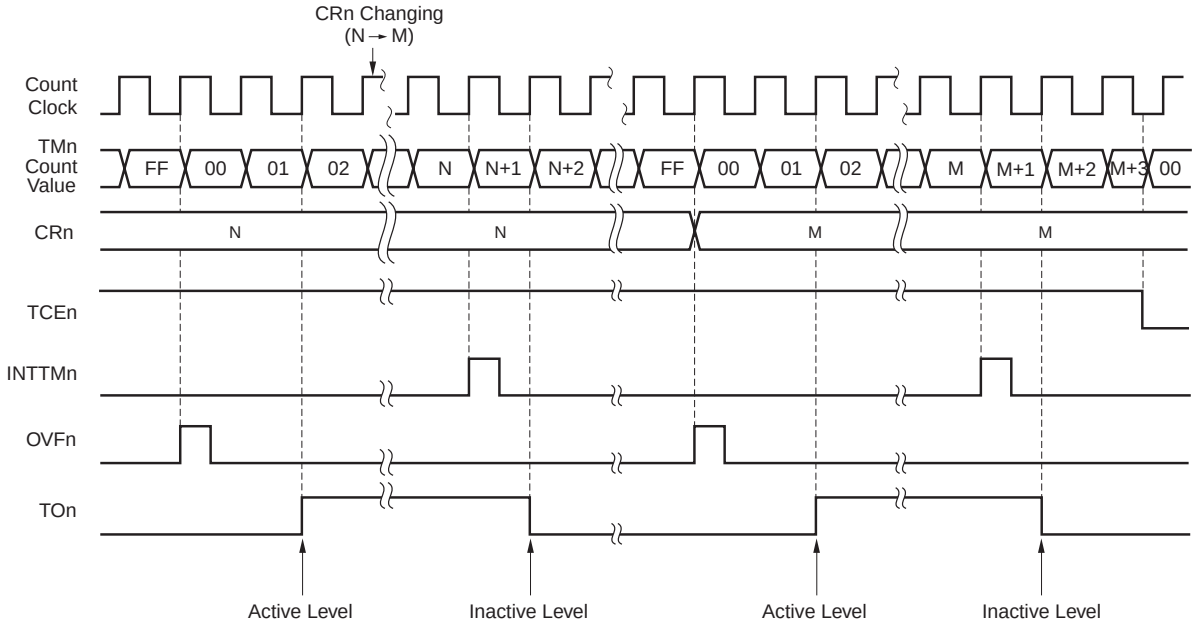
Remark: n = 50, 51

Figure 8-20: PWM Output Operation Timings (CRn = FFH, active high setting)



Remark: n = 50, 51

Figure 8-21: PWM Output Operation Timings (CRn changing, active high setting)



Remark: n = 50, 51

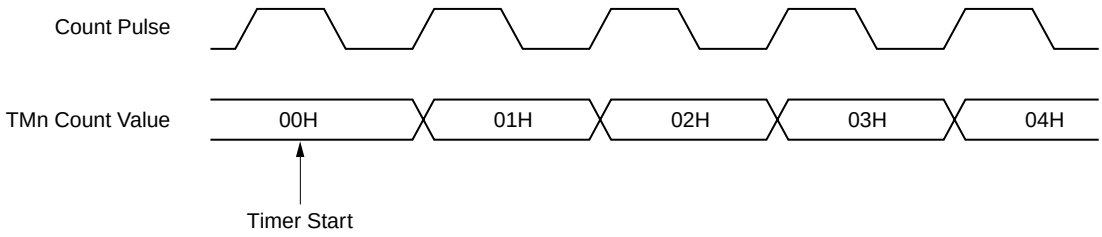
Caution: If CRn is changed during TMn operation, the value changed is not reflected until TMn overflows.

8.5 Cautions on 8-Bit Timer/Event Counters 50 and 51

(1) Timer start errors

An error with a maximum of one clock might occur concerning the time required for a match signal to be generated after the timer starts. This is because 8-bit timer registers 50 and 51 are started asynchronously with the count pulse.

Figure 8-22: 8-bit Timer Registers 50 and 51 Start Timings

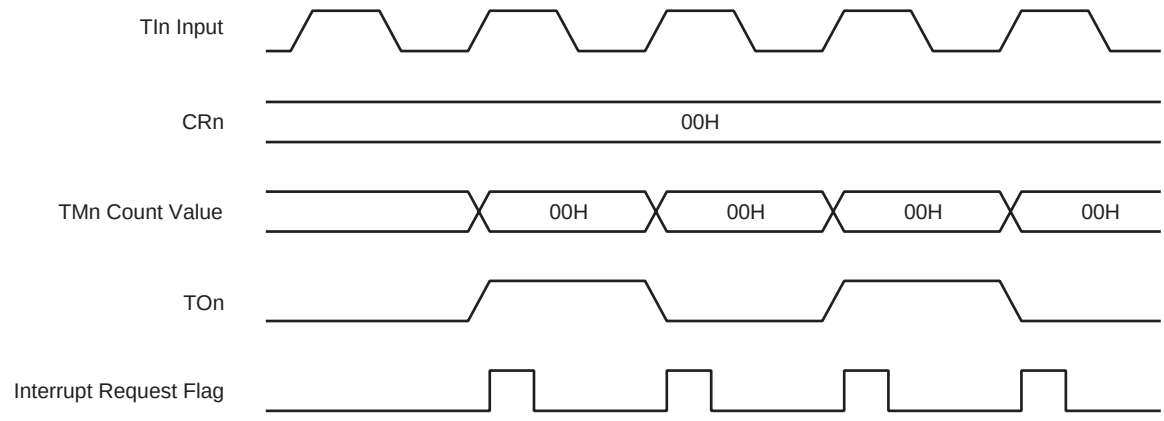


Remark: n = 50, 51

(2) Compare registers 50 and 51 sets

The 8-bit compare registers (CR50 and CR51) can be set to 00H.
Thus, when an 8-bit compare register is used as an event counter, one-pulse count operation can be carried out.

Figure 8-23: External Event Counter Operation Timings

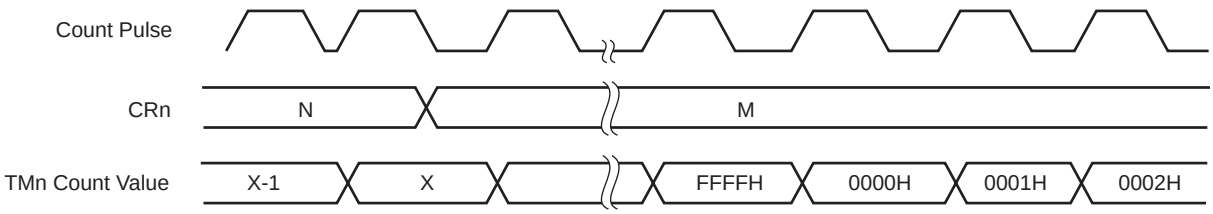


Remark: n = 50, 51

(3) Operation after compare register change during timer count operation

If the values after the 8-bit compare registers (CR50 and CR51) are changed are smaller than those of 8-bit timer registers (TM50 and TM51), TM50 and TM51 continue counting, overflow and then restarts counting from 0. Thus, if the value (M) after CR50 and CR51 change is smaller than that (N) before change it is necessary to restart the timer after changing CR50 and CR51.

Figure 8-24: Timings after Compare Register Change during Timer Count Operation



Remark: n = 50, 51

(4) TM50 and TM51 read during timer operation

When TM50 and TM51 are read during operation, choose a select clock which has a longer high/low level wave because the select clock is stopped temporarily.

Chapter 9 Watch Timer (µPD780822B Subseries)

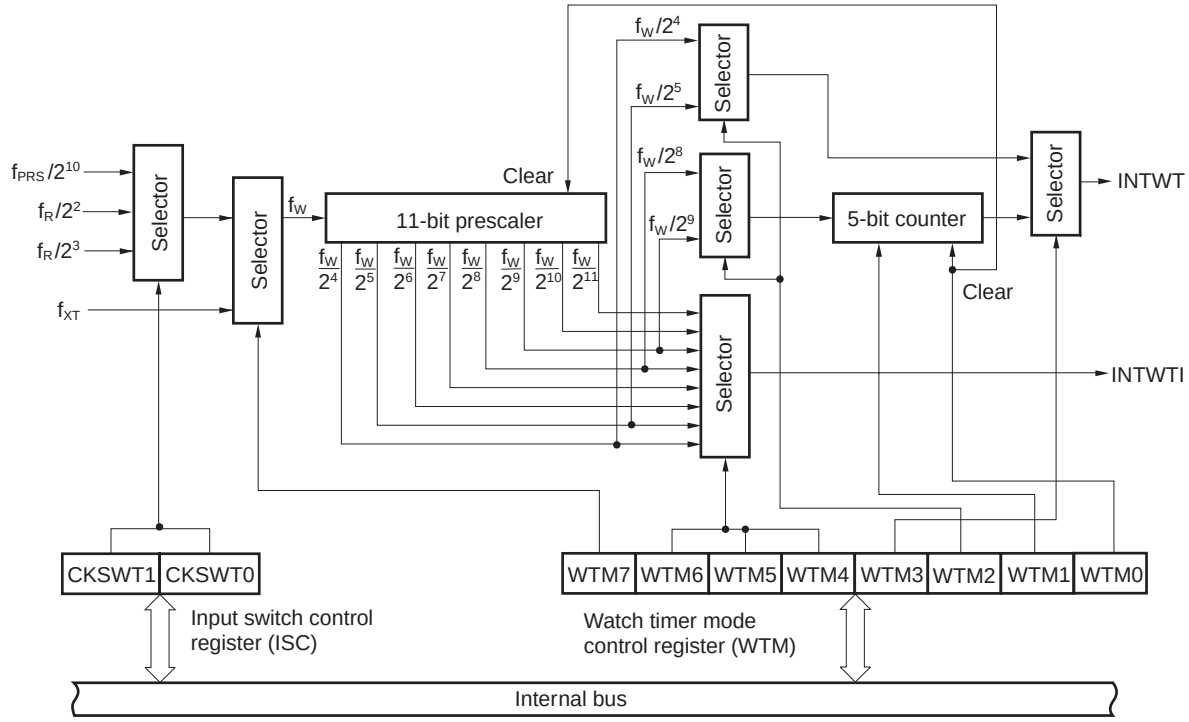
9.1 Watch Timer Functions

The watch timer has the following functions:

- Watch timer
- Interval timer

The watch timer and the interval timer can be used simultaneously.
The Figure 9-1 shows Watch Timer Block Diagram.

Figure 9-1: Block Diagram of Watch Timer



Remarks: **1.** f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_W: Watch timer clock oscillation frequency

(1) Watch timer

When X1 input clock, internal low-speed oscillator clock or subsystem clock is used, interrupt requests (INTWT) are
generated at preset intervals.

Table 9-1: Watch Timer Interrupt Time

Interrupt Time	$f_{PRS}/2^{10}$ (f_{PRS}=16 MHz)	$f_R/2^2$ (f_R=240 kHz)	$f_R/2^3$ (f_R=240 kHz)	f_{XT} (f_{XT}=32.768 kHz)
$2^4/f_W$	1 ms	267 μs	533 μs	488 μs
$2^5/f_W$	2 ms	533 μs	1 ms	977 μs
$2^{13}/f_W$	524 ms	137 ms	273 ms	250 ms
$2^{14}/f_W$	1.05 s	273 ms	546 ms	500 ms

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_W: Watch timer clock oscillation frequency

(2) Interval timer

Interrupt requests (INTWTI) are generated at the preset time interval.

Table 9-2: Interval Timer Interrupt Time

Interrupt Time	$f_{PRS}/2^{10}$ (f_{PRS}=16 MHz)	$f_R/2^2$ (f_R=240 kHz)	$f_R/2^3$ (f_R=240 kHz)	f_{XT} (f_{XT}=32.768 kHz)
$2^4/f_W$	1 ms	267 μs	533 μs	488 μs
$2^5/f_W$	2 ms	533 μs	1 ms	977 μs
$2^6/f_W$	4.1 ms	1 ms	2.1 ms	2 ms
$2^7/f_W$	8.2 ms	2.1 ms	4.3 ms	3.9 ms
$2^8/f_W$	16.4 ms	4.3 ms	8.5 ms	5.8 ms
$2^9/f_W$	32.8 ms	8.5 ms	17.1 ms	15.6 ms
$2^{10}/f_W$	65.5 ms	17.1 ms	34.1 ms	31.2 ms
$2^{11}/f_W$	131.1 ms	34.1 ms	68.3 ms	62.5 ms

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_W: Watch timer clock oscillation frequency

9.2 Watch Timer Configuration

The watch timer consists of the following hardware.

Table 9-3: Watch Timer Configuration

Item	Configuration
Counter	5 bits × 1
Prescaler	9 bits × 1
Control register	Input switch control register (ISC) Watch timer mode control register (WTM)

9.2.1 Input Switch Control Register (ISC)

This register specifies the input clock source for the watch timer.
ISC is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets WTM to 00H.

Figure 9-2: Input Switch Control Register (ISC) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ISC	0	CKSWD	CKSWT1	CKSWT0	CKUCKS	CKCL	ISC1	ISC0	R/W	FF4AH	00H

CKSWT1	CKSWT0	Watch Timer Count Clock Selection
0	0	$f_{PRS}/2^{10}$
0	1	$f_R/2^2$
1	1	$f_R/2^3$
Others		Setting prohibited

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. In this part only the CKWT1 and CKWT0 selection feature is described. The input switch control register has more selection features. Please refer to the **Figure 5-8, "Input Switch Control Register (ISC) Format (1/2)," on page 134**.

9.2.2 Watch Timer Mode Register (WTM)

This register sets the watch timer count clock, the watch timer operating mode, and prescaler interval time and enables/disables prescaler and 5-bit counter operations. WTM is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets WTM to 00H.

Figure 9-3: Watch Timer Mode Control Register (WTM) Format (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
WTM	WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0	R/W	FF4CH	00H

WTM7	Watch Timer Count Clock Selection		
	ISC.CKSWT0=0 ISC.CKSWT1=0	ISC.CKSWT0=1 ISC.CKSWT1=0	ISC.CKSWT0=1 ISC.CKSWT1=1
0	$f_{PRS}/2^{10}$	$f_R/2^2$	$f_R/2^3$
1	f_{XT}		

WTM6	WTM5	WTM4	Interval Timer Time Selection (INTWTI period)
0	0	0	$2^4/f_W$
0	0	1	$2^5/f_W$
0	1	0	$2^6/f_W$
0	1	1	$2^7/f_W$
1	0	0	$2^8/f_W$
1	0	1	$2^9/f_W$
1	1	0	$2^{10}/f_W$
1	1	1	$2^{11}/f_W$

WTM3	WTM2	Watch Timer Time Selection (INTWT period)
0	0	$2^{14}/f_W$
0	1	$2^{13}/f_W$
1	0	$2^5/f_W$
1	1	$2^4/f_W$

Figure 9-3: Watch Timer Mode Control Register (WTM) Format (2/2)

WTM1	5-Bit Counter Operation Control
0	Clear after operation stop
1	Operation enable

WTM0	Prescaler Operation Control
0	Clear after operation stop
1	Operation enable

Caution: When the watch timer is used, the prescaler should not be cleared frequently. When rewriting WTM4 to WTM6 to other data, stop the timer operation beforehand.

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_W: Watch timer clock oscillation frequency

9.3 Watch Timer Operations

9.3.1 Watch timer operation

The watch timer generates an interrupt request (INTWT) at the specific time interval by using the X1 input clock of subsystem clock.
When bit 0 (WTM0) and bit 1 (WTM1) of the watch timer mode control register (WTM) are set to 1, the count operation starts. When set to 0, the 5-bit counter is cleared and the count operation stops.
When the interval timer is simultaneously operated, zero-second start can be only the watch timer by clearing WTM1 to 0. In this case, however, the 11-bit prescaler is not cleared. Therefore, an error up to $2^9 \times 1/f_W$ seconds occurs in the first overflow (INTWT) after zero-second start.
The interval request is generated at the following time intervals.

Table 9-4: Watch Timer Interrupt INTWDT Period Selection

WTM3	WTM2		Watch Timer Time Selection (INTWT period)			
			$f_{PRS}/2^{10}$ (f_{PRS}=16 MHz)	$f_R/2^2$ (f_R=240 kHz)	$f_R/2^3$ (f_R=240 kHz)	f_{XT} (f_{XT}=32.768 kHz)
0	0	$2^{14}/f_W$	1.05 s	273 ms	546 ms	500 ms
0	1	$2^{13}/f_W$	524 ms	137 ms	273 ms	250 ms
1	0	$2^5/f_W$	2 ms	533 μs	1 ms	977 μs
1	1	$2^4/f_W$	1 ms	267 μs	533 μs	488 μs

Remarks: **1.** f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_W: Watch timer clock frequency

9.3.2 Interval timer operation

The watch timer operates also as interval timer which generates interrupt request repeatedly at an interval of the preset count value.
The interval time can be selected with bits 4 to 6 (WTM4 to WTM6) of the watch timer mode control register (WTM).

Table 9-5: Interval Timer Interrupt INTWTI Period Selection

WTM6	WTM5	WTM4		Interval Timer Time Selection (INTWTI period)			
				$f_{PRS}/2^{10}$ (f_{PRS}=16 MHz)	$f_R/2^2$ (f_R=240 kHz)	$f_R/2^3$ (f_R=240 kHz)	f_{XT} (f_{XT}=32.768 kHz)
0	0	0	$2^4/f_W$	1 ms	267 μs	533 μs	488 μs
0	0	1	$2^5/f_W$	2 ms	533 μs	1 ms	977 μs
0	1	0	$2^6/f_W$	4.1 ms	1 ms	2.1 ms	2 ms
0	1	1	$2^7/f_W$	8.2 ms	2.1 ms	4.3 ms	3.9 ms
1	0	0	$2^8/f_W$	16.4 ms	4.3 ms	8.5 ms	5.8 ms
1	0	1	$2^9/f_W$	32.8 ms	8.5 ms	17.1 ms	15.6 ms
1	1	0	$2^{10}/f_W$	65.5 ms	17.1 ms	34.1 ms	31.2 ms
1	1	1	$2^{11}/f_W$	131.1 ms	34.1 ms	68.3 ms	62.5 ms

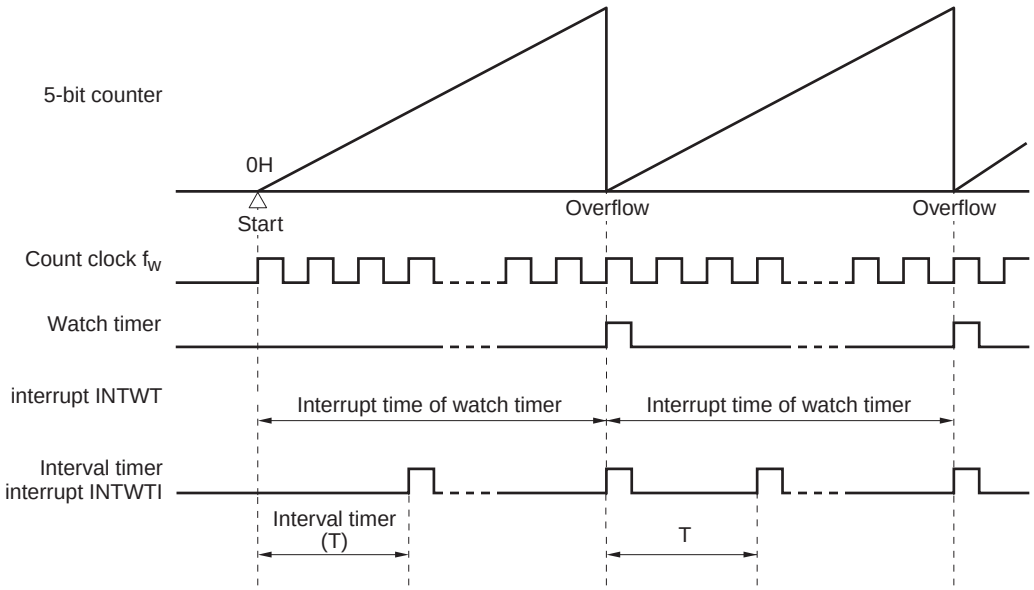
Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. f_W: Watch timer clock frequency

Figure 9-4: Operation Timing of Watch Timer/Interval Timer



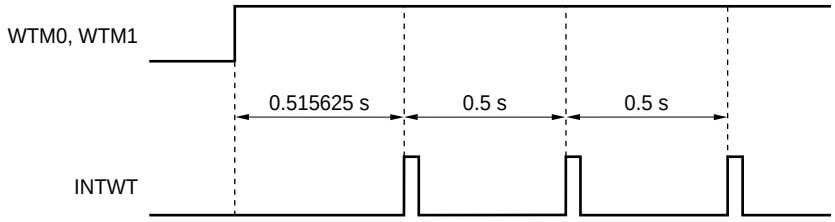
Remark: f_W : Watch timer clock frequency

Caution: If the watch timer and 5-bit counter are enabled by the watch timer operation mode control register 0 (WTM), the time from this setting to the occurrence of the first interrupt request (INTWT) is not exactly the value set by the WTM value. This is because the 5-bit counter is late by one output cycle of the 11-bit prescaler in starting to count. The second INTWT signal and those that follow are generated exactly at the set time.

9.4 Cautions for Watch Timer

When operation of the watch timer and 5-bit counter is enabled by the watch timer mode control register (WTM) (by setting bits 0 (WTM0) and 1 (WTM1) of WTM to 1), the interval until the first interrupt request (INTWT) is generated after the register is set does not exactly match the specification made with bits 2 and 3 (WTM2 and WTM3) of WTM.
Subsequently, however, the INTWT signal is generated at the specified intervals.

Figure 9-5: Example of Generation of Watch Timer Interrupt Request (INTWT) (When Interrupt Period = 0.5 s)



It takes 0.515625 second (max.) for the first INTWT to be generated ($2^9 \times 1/32768 = 0.015625$ s longer). INTWT is then generated every 0.5 seconds.

Chapter 10 Watchdog Timer (μPD780822B Subseries)

10.1 Watchdog Timer Functions

The watchdog timer has the following functions:

- Watchdog timer
- Interval timer

Caution: Select the watchdog timer mode or the interval timer mode with the watchdog timer mode register (WDTM).

(1) Watchdog timer mode

The watchdog timer detects an inadvertent program loop. Upon detection of an inadvertent program loop, a non-maskable interrupt request or $\overline{\text{RESET}}$ can be generated.

(2) Interval timer mode

Interrupts are generated at the preset time intervals.

Caution: Select the watchdog timer mode or the interval timer mode using the watchdog timer mode register (WDTM). The watchdog timer and the interval timer cannot be used simultaneously.

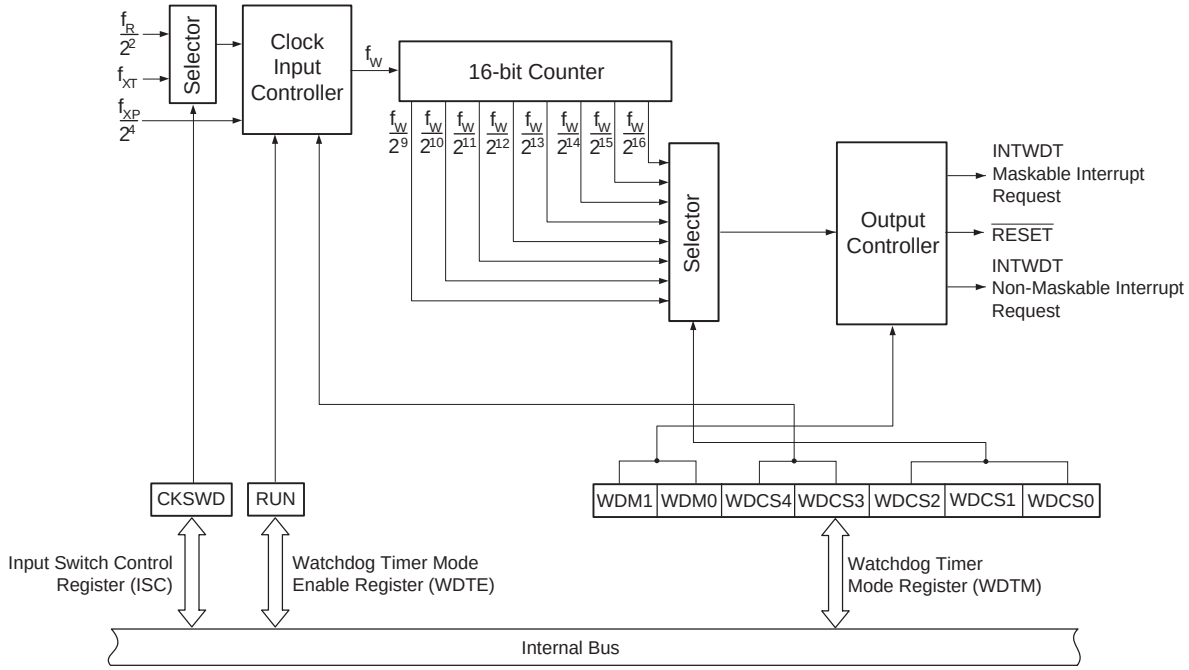
10.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 10-1: Watchdog Timer Configuration

Item	Configuration
Control register	Watchdog timer mode register (WDTM)
	Watchdog timer enable register (WDTE)
	Input switch control register (ISC)

Figure 10-1: Watchdog Timer Block Diagram



Remarks: 1. f_{XP} : X1 input clock oscillation frequency

2. f_R : Internal low-speed oscillator clock oscillation frequency

3. f_{XT} : Subsystem clock oscillation frequency.

The operation mode of the watchdog timer (WDT) is switched according to the mask option (option byte if using a flash memory version) setting to the on-chip internal low-speed oscillator clock as shown in Table 10-2.

Table 10-2: Mask Option Setting and Watchdog Timer Operation Mode

	Mask Option	
	Internal low-speed oscillator cannot be stopped	Internal low-speed oscillator can be stopped by Software
Watchdog timer clock source	Fixed to f_R **Note 1**	• Selectable by software (f_{XP}, f_R, f_{XT} or stopped) • When reset is released: f_R
Operation after reset	Operation starts with the maximum interval ($f_R/2^{18}$)	
Operation mode selection	The interval can be changed only once	The clock selection/interval can be changed only once
Features	The watchdog timer cannot be stopped	The watchdog timer can be stopped in standby mode **Note 2**

Notes: 1. As long as power is being supplied, internal low-speed oscillator oscillation cannot be stopped (except in the reset period).

2. The conditions under which clock supply to the watchdog timer is stopped differ depending on the clock source of the watchdog timer.

<1> If the clock source is f_{XP}, clock supply is stopped under the following conditions:

- When f_{XP} is stopped.
- In HALT/STOP mode
- During oscillation stabilization time

<2> If the clock source is f_R, clock supply is stopped under the following conditions:

- If the CPU clock is f_{XP} and f_R is stopped by software before execution of the STOP instruction.
- In HALT/STOP mode

Remarks: 1. f_{XP} : X1 input clock oscillation frequency

2. f_R : Internal low-speed oscillator clock oscillation frequency

3. f_{XT} : Subsystem clock oscillation frequency.

10.3 Watchdog Timer Control Registers

The following three types of registers are used to control the watchdog timer.

- Watchdog timer enable register (WDTE)
- Watchdog timer mode register (WDTM)
- Input switch control register (ISC)

(1) Watchdog timer enable register (WDTE)

Writing ACH to WDTE clears the watchdog timer counter and starts counting again.
This register can be set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to 9AH.

Figure 10-2: Watchdog Timer Enable Register (WDTE) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
WDTE									R/W	FF99H	9AH

Cautions: 1. If a value other than ACH is written to WDTE, an internal reset signal is generated. If the source clock of the watchdog timer is stopped, however, an internal reset signal is generated when the source clock of the watchdog timer starts operating again.

2. If an 1-bit memory manipulation instruction is executed for WDTE, an internal reset signal is generated. If the source clock of the watchdog timer is stopped, however, an internal reset signal is generated when the source clock of the watchdog timer starts operating again.

3. The value read from WDTE is 9AH (this differs from the written value (ACH)).

(2) Watchdog timer mode register (WDTM)

This register sets the overflow time and operation clock of the watchdog timer.
This register can be set by an 8-bit memory manipulation instruction and can be read many times, but can be written only once after reset is released.

$\overline{\text{RESET}}$ input sets WDTM to 67H.

Figure 10-3: Watchdog Timer Mode Register (WDTM) Format (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
WDTM	0	WDM1	WDM0	WDCS4	WDCS3	WDCS2	WDCS1	WDCS0	R/W	FF98H	67H

WDM1	WDM0	Watchdog Timer Operation Mode Selection **Note 1**
0	0	Interval timer mode (Maskable interrupt occurs upon generation of an overflow)
0	1	Watchdog timer mode 1 (Non-maskable interrupt occurs upon generation of an overflow)
1	x	Watchdog timer mode 2 (Reset operation is activated upon generation of an overflow)

WDCS4	WDCS3	Watchdog Timer count clock selection**Note 2**	
		ISC.CKSWD = 0	ISC.CKSWD = 1
0	0	$f_R/2^2$	f_{XT}
0	1	$f_{XP}/2^4$	
1	x	Watchdog timer operation stopped	

Notes: 1. Once set to 1, WDM1 and WDM0 cannot be cleared to 0 by software.

2. If “internal low-speed oscillator clock cannot be stopped” is specified by a mask option, this cannot be set. The internal low-speed oscillator clock will be selected no matter what value is written.

Figure 10-3: Watchdog Timer Mode Register (WDTM) Format (2/2)

WDCS2 **Note 3**	WDCS1 **Note 3**	WDCS0 **Note 3**	Watchdog Timer Overflow Time		
			During internal low-speed oscillator operation	During X1 input clock operation	During subsystem clock operation
0	0	0	$f_R/2^{11}$ (8.53 ms)	$f_{XP}/2^{13}$ (512 µs)	$f_{XT}/2^9$ (15.625 ms)
0	0	1	$f_R/2^{12}$ (17.07 ms)	$f_{XP}/2^{14}$ (1.02 ms)	$f_{XT}/2^{10}$ (31.25 ms)
0	1	0	$f_R/2^{13}$ (34.13 ms)	$f_{XP}/2^{15}$ (2.05 ms)	$f_{XT}/2^{11}$ (62.5 ms)
0	1	1	$f_R/2^{14}$ (68.27 ms)	$f_{XP}/2^{16}$ (4.10 ms)	$f_{XT}/2^{12}$ (0.125 s)
1	0	0	$f_R/2^{15}$ (136.53 ms)	$f_{XP}/2^{17}$ (8.19 ms)	$f_{XT}/2^{13}$ (0.25 s)
1	0	1	$f_R/2^{16}$ (273.07 ms)	$f_{XP}/2^{18}$ (16.38 ms)	$f_{XT}/2^{14}$ (0.5 s)
1	1	0	$f_R/2^{17}$ (546.13 ms)	$f_{XP}/2^{19}$ (32.77 ms)	$f_{XT}/2^{15}$ (1 s)
1	1	1	$f_R/2^{18}$ (1.09 s)	$f_{XP}/2^{20}$ (65.54 ms)	$f_{XT}/2^{16}$ (2 s)

Notes: 1. Once set to 1, WDM1 and WDM0 cannot be cleared to 0 by software.

2. If “Internal low-speed oscillator clock cannot be stopped” is specified by a mask option, this cannot be set. The internal low-speed oscillator clock will be selected no matter what value is written.

3. Reset is released at the maximum cycle (WDCS2, 1, 0 = 1, 1, 1).

Cautions: 1. If data is written to WDTM, a wait cycle is generated. For details, refer to Chapter 29 ”Cautions For Wait (µPD780822B Subseries)” on page 591.

2. Set bits 7, 6, and 5 to 0, 1, and 1, respectively (when “Internal low-speed oscillator clock cannot be stopped” is selected by a mask option, other values are ignored).

3. After reset is released, WDTM can be written only once by an 8-bit memory manipulation instruction. If writing attempted a second time, an internal reset signal is generated. If the source clock of the watchdog timer is stopped, however, an internal reset signal is generated when the source clock of the watchdog timer starts operating again.

4. WDTM cannot be set by an 1-bit memory manipulation instruction.

5. When “Internal low-speed oscillator clock can be stopped by software” is selected by a mask option and the watchdog timer is stopped by setting WDCS4 to 1, the watchdog timer does not operate even if WDCS4 is cleared to 0 again. An internal reset signal is not generated.

Remarks: 1. f_{XP} :X1 input clock oscillation frequency

2. f_R : Internal low-speed oscillator clock oscillation frequency

3. f_{XT} : Subsystem clock oscillation frequency.

4. Overflow times in parentheses during X1 input clock selection for f_{XP} = 16 MHz operation.

5. x = don't care.

(3) Input switch control register (ISC)

This register specifies the input clock source for the watchdog timer.
ISC is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ISC to 00H.

Figure 10-4: Input Switch Control Register (ISC) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ISC	0	CKSWD	CKSWT1	CKSWT0	CKUCKS	CKCL	ISC1	ISC0	R/W	FF4AH	00H

CKSWD	Clock selection for Watchdog timer
0	$f_R/2^2$
1	f_{XT}

Remarks: 1. f_R : Internal low-speed oscillator clock oscillation frequency

2. f_{XT} : Subsystem clock oscillation frequency.

3. In this part only the CKSWDT selection feature is described. The input switch control register has more selection features. Please refer to the **Figure 5-8, “Input Switch Control Register (ISC) Format (1/2),” on page 134**.

10.4 Watchdog Timer Operations

10.4.1 Interval timer operation

This mode can be selected only when “Internal low-speed oscillator can be stopped by software” is selected by mask option by setting bits 6 and 5 (WDM1 and WDM0) of the watchdog timer mode register (WDTM) to 0.
The operation clock of the watchdog timer can be selected as either the internal low-speed oscillator clock, the subsystem clock or the X1 input clock. After reset is released, operation is started at the maximum cycle
(bits 2, 1, and 0 (WDCS2, WDCS1, WDCS0) of the watchdog timer mode register (WDTM) = 1, 1, 1).

The following shows the watchdog timer operation after reset release.

1. The status after reset release is as follows.
   - Operation clock: Internal low-speed oscillator clock
   - Cycle: $f_R/2^{18}$ (1.09 seconds: At operation with f_R = 240 kHz (TYP.))
   - Counting starts
2. The following should be set in the watchdog timer mode register (WDTM) by an 8-bit memory manipulation instruction**Notes 1, 2** and the input switch control register (ISC) by an 1-bit or an 8-bit memory manipulation instruction.
   - Operation clock: Any of the following can be selected using bits 3 and 4 (WDCS3 and WDCS4) of the watchdog timer mode register (WDTM) and bit 6 (CKSWD) of the input switch control register (ISC).
     - Internal low-speed oscillator clock (f_R)
     - Subsystem clock (f_{XT})
     - X1 input clock (f_{XP})
     - Watchdog timer operation stopped
   - Operation mode: Select interval timer mode by clearing bits 6 and 5 (WDM1 and WDM0) of the watchdog timer mode register (WDTM) to 0.
   - Interval time: Set using bits 2 to 0 (WDCS2 to WDCS0)
   - Interrupt servicing: The interrupt servicing and the interrupt priority can be specified by using bit 0 of the interrupt mask flag register (MK0L) and the priority specify flag register (PR0L).
3. After the above procedures are executed, writing ACH to WDTE clears the count to 0, enabling recounting.

Notes: 1. As soon as WDTM is written, the counter of the watchdog timer is cleared.

2. If the watchdog timer is stopped by setting WDCS4 and WDCS3 to 1 and, respectively, an internal reset signal is not generated even if the following processing is performed.
- WDTM is written a second time.
- An 1-bit memory manipulation instruction is executed to WDTE.
- A value other than ACH is written to WDTE.

Caution: In this mode, if the X1 input clock (f_{XP}) is selected as the watchdog timer count clock, the interval timer continues operating in the HALT mode, but it stops in STOP mode. Thus, set bit 7 (RUN) of the watchdog timer enable register (WDTE) to 1 before the STOP mode is set, clear the interval timer and then execute the STOP instruction.

Table 10-3: Watchdog Timer Overflow Time

WDCS2	WDCS1	WDCS0	Watchdog Timer Overflow Time[Note]		
			During internal low-speed oscillator operation	During X1 input clock operation	During subsystem clock operation
0	0	0	$f_R/2^{11}$ (8.53 ms)	$f_{XP}/2^{13}$ (512 μs)	$f_{XT}/2^9$ (15.625 ms)
0	0	1	$f_R/2^{12}$ (17.07 ms)	$f_{XP}/2^{14}$ (1.02 ms)	$f_{XT}/2^{10}$ (31.25 ms)
0	1	0	$f_R/2^{13}$ (34.13 ms)	$f_{XP}/2^{15}$ (2.05 ms)	$f_{XT}/2^{11}$ (62.5 ms)
0	1	1	$f_R/2^{14}$ (68.27 ms)	$f_{XP}/2^{16}$ (4.10 ms)	$f_{XT}/2^{12}$ (0.125 s)
1	0	0	$f_R/2^{15}$ (136.53 ms)	$f_{XP}/2^{17}$ (8.19 ms)	$f_{XT}/2^{13}$ (0.25 s)
1	0	1	$f_R/2^{16}$ (273.07 ms)	$f_{XP}/2^{18}$ (16.38 ms)	$f_{XT}/2^{14}$ (0.5 s)
1	1	0	$f_R/2^{17}$ (546.13 ms)	$f_{XP}/2^{19}$ (32.77 ms)	$f_{XT}/2^{15}$ (1 s)
1	1	1	$f_R/2^{18}$ (1.09 s)	$f_{XP}/2^{20}$ (65.54 s)	$f_{XT}/2^{16}$ (2 s)

Note: The repeated interval time in interval timer mode takes 2^7 clocks x watchdog timer input clock longer than the watchdog timer overflow time.

Example:

$f_R/2^2$ selected for watchdog timer clock source

$$t_{INT} = \text{Overflow time} + [2^7 \times 1 / (f_R/2^2)]$$

Remark: Overflow times in parentheses during X1 input clock selection for f_{XP} = 16 MHz operation.

10.4.2 Watchdog timer mode 1 operation

This mode can be selected only when “Internal low-speed oscillator can be stopped by software” is selected by mask option by setting bits 6 and 5 (WDM1 and WDM0) of the watchdog timer mode register (WDTM) to 0, 1. The operation clock of the watchdog timer can be selected as either the internal low-speed oscillator clock, the subsystem clock or the X1 input clock. After reset is released, operation is started at the maximum cycle
(bits 2, 1, and 0 (WDCS2, WDCS1, WDCS0) of the watchdog timer mode register (WDTM) = 1, 1, 1).

The following shows the watchdog timer operation after reset release.

1. The status after reset release is as follows.
   - Operation clock: Internal low-speed oscillator clock
   - Cycle: $f_R/2^{18}$ (1.09 seconds: At operation with f_R = 240 kHz (TYP.))
   - Counting starts
2. The following should be set in the watchdog timer mode register (WDTM) by an 8-bit memory manipulation instruction**Note 1, 2** and the input switch control register (ISC) by an 1-bit or an 8-bit memory manipulation instruction.
   - Operation clock: Any of the following can be selected using bits 3 and 4 (WDCS3 and WDCS4) of the watchdog timer mode register (WDTM) and bit 6 (CKSWD) of the input switch control register (ISC).
     - Internal low-speed oscillator clock (f_R)
     - Subsystem clock (f_{XT})
     - X1 input clock (f_{XP})
     - Watchdog timer operation stopped
   - Operation mode: Select Watchdog timer mode 1 by setting bits 6 and 5 (WDM1 and WDM0) of the watchdog timer mode register (WDTM) to 0, 1.
   - Interval time: Set using bits 2 to 0 (WDCS2 to WDCS0)
   - Interrupt servicing: The interrupt servicing and the interrupt priority can be specified by using bit 0 of the interrupt mask flag register (MK0L) and the priority specify flag register (PR0L).
3. After the above procedures are executed, writing ACH to WDTE clears the count to 0, enabling recounting.

Notes: 1. As soon as WDTM is written, the counter of the watchdog timer is cleared.

2. If the watchdog timer is stopped by setting WDCS4 and WDCS3 to 1 and, respectively, an internal reset signal is not generated even if the following processing is performed.
- WDTM is written a second time.
- An 1-bit memory manipulation instruction is executed to WDTE.
- A value other than ACH is written to WDTE.

Caution: In this mode, if the X1 input clock (f_{XP}) is selected as the watchdog timer count clock, the interval timer continues operating in the HALT mode, but it stops in STOP mode. Thus, set bit 7 (RUN) of the watchdog timer enable register (WDTE) to 1 before the STOP mode is set, clear the interval timer and then execute the STOP instruction.

Table 10-4: Watchdog Timer Overflow Time

WDCS2	WDCS1	WDCS0	Watchdog Timer Overflow Time[Note]		
			During internal low-speed oscillator operation	During X1 input clock operation	During subsystem clock operation
0	0	0	$f_R/2^{11}$ (8.53 ms)	$f_{XP}/2^{13}$ (512 µs)	$f_{XT}/2^9$ (15.625 ms)
0	0	1	$f_R/2^{12}$ (17.07 ms)	$f_{XP}/2^{14}$ (1.02 ms)	$f_{XT}/2^{10}$ (31.25 ms)
0	1	0	$f_R/2^{13}$ (34.13 ms)	$f_{XP}/2^{15}$ (2.05 ms)	$f_{XT}/2^{11}$ (62.5 ms)
0	1	1	$f_R/2^{14}$ (68.27 ms)	$f_{XP}/2^{16}$ (4.10 ms)	$f_{XT}/2^{12}$ (0.125 s)
1	0	0	$f_R/2^{15}$ (136.53 ms)	$f_{XP}/2^{17}$ (8.19 ms)	$f_{XT}/2^{13}$ (0.25 s)
1	0	1	$f_R/2^{16}$ (273.07 ms)	$f_{XP}/2^{18}$ (16.38 ms)	$f_{XT}/2^{14}$ (0.5 s)
1	1	0	$f_R/2^{17}$ (546.13 ms)	$f_{XP}/2^{19}$ (32.77 ms)	$f_{XT}/2^{15}$ (1 s)
1	1	1	$f_R/2^{18}$ (1.09 s)	$f_{XP}/2^{20}$ (65.54 s)	$f_{XT}/2^{16}$ (2 s)

Note: The repeated interval time in interval timer mode takes 2^7 clocks x watchdog timer input clock longer than the watchdog timer overflow time.

Example:

$f_R/2^2$ selected for watchdog timer clock source

$$t_{INT} = \text{Overflow time} + [2^7 \times 1 / (f_R/2^2)]$$

Remark: Overflow times in parentheses during X1 input clock selection for f_{XP} = 16 MHz operation.

10.4.3 Watchdog timer mode 2 operation when "Internal low-speed oscillator cannot be stopped" is selected by mask option

When "Internal low-speed oscillator cannot be stopped" is selected by mask option, only the watchdog timer mode 2 can be selected by setting bits 6 and 5 (WDM1, WDM0) of the watchdog timer mode register (WDTM) to 1. The operation clock of watchdog timer is fixed to the internal low-speed oscillator. After reset is released, operation is started at the maximum cycle (bits 2, 1, and 0 (WDCS2, WDCS1, WDCS0) of the watchdog timer mode register (WDTM) = 1, 1, 1). The watchdog timer operation cannot be stopped.

The following shows the watchdog timer operation after reset release.

1. The status after reset release is as follows.
   - Operation clock: Internal low-speed oscillator clock
   - Cycle: $f_R/2^{18}$ (1.09 seconds: At operation with f_R = 240 kHz (TYP.))
   - Counting starts
2. The following should be set in the watchdog timer mode register (WDTM) by an 8-bit memory manipulation instruction**Notes 1, 2**.
   - Cycle: Set using bits 2 to 0 (WDCS2 to WDCS0)
3. After the above procedures are executed, writing ACH to WDTE clears the count to 0, enabling recounting.

Notes: 1. The operation clock (internal low-speed oscillator clock) cannot be changed. If any value is written to bits 3 and 4 (WDCS3, WDCS4) of WDTM, it is ignored.

2. As soon as WDTM is written, the counter of the watchdog timer is cleared.

Caution: In this mode, operation of the watchdog timer absolutely cannot be stopped even during STOP instruction execution. An internal reset signal is generated when the watchdog timer overflows after STOP instruction execution.

Table 10-5: Watchdog Timer Overrun Detection Time

WDCS2	WDCS1	WDCS0	Runaway Detection Time
0	0	0	$f_R/2^{11}$ (8.53 ms)
0	0	1	$f_R/2^{12}$ (17.07 ms)
0	1	0	$f_R/2^{13}$ (34.13 ms)
0	1	1	$f_R/2^{14}$ (68.27 ms)
1	0	0	$f_R/2^{15}$ (136.53 ms)
1	0	1	$f_R/2^{16}$ (273.07 ms)
1	1	0	$f_R/2^{17}$ (546.13 ms)
1	1	1	$f_R/2^{18}$ (1.09 s)

Remark: f_R: Internal low-speed oscillator clock oscillation frequency

10.4.4 Watchdog timer mode 2 operation when "Internal low-speed oscillator can be stopped by software" is selected by mask option

When "Internal low-speed oscillator can be stopped" is selected by mask option, the watchdog timer modes 1 and the watchdog timer mode 2 can be selected by setting bits 6 and 5 (WDM1, WDM0) of the watchdog timer mode register (WDTM) according to the operating modes. The operation clock of the watchdog timer can be selected as either the internal low-speed oscillator clock, the subsystem clock or the X1 input clock. After reset is released, operation is started at the maximum cycle (bits 2, 1, and 0 (WDCS2, WDCS1, WDCS0) of the watchdog timer mode register (WDTM) = 1, 1, 1).

The following shows the watchdog timer operation after reset release.

1. The status after reset release is as follows.
   - Operation clock: Internal low-speed oscillator clock oscillation frequency (f_R)
   - Cycle: $f_R/2^{18}$ (1.09 seconds: At operation with f_R = 240 kHz (TYP.))
   - Counting starts
2. The following should be set in the watchdog timer mode register (WDTM) by an 8-bit memory manipulation instruction**Notes 1, 2** and the input switch control register (ISC) by an 1-bit or an 8-bit memory manipulation instruction.
   - Operation clock: Any of the following can be selected using bits 3 and 4 (WDCS3 and WDCS4) of the watchdog timer mode register (WDTM) and bit 6 (CKSWD) of the input switch control
     - Internal low-speed oscillator clock (f_R)
     - Subsystem clock (f_{XT})
     - X1 input clock (f_{XP})
     - Watchdog timer operation stopped
   - Operation mode: Any of the following can be selected using bits 6 and 5 (WDM1 and WDM0) of the watchdog timer mode register (WDTM)
     - Interval timer mode
     - Watchdog timer mode 1
     - Watchdog timer mode 2
   - Cycle: Set using bits 2 to 0 (WDCS2 to WDCS0)
3. After the above procedures are executed, writing ACH to WDTE clears the count to 0, enabling recounting.

Notes: 1. As soon as WDTM is written, the counter of the watchdog timer is cleared.

2. If the watchdog timer is stopped by setting WDCS4 and WDCS3 to 1 and, respectively, an internal reset signal is not generated even if the following processing is performed.
- WDTM is written a second time.
- 1-bit memory manipulation instruction is executed to WDTE.
- A value other than ACH is written to WDTE.

Caution: In this mode, watchdog timer operation is stopped during HALT/STOP instruction execution. After HALT/STOP mode is released, counting is started again using the operation clock of the watchdog timer set before HALT/STOP instruction execution by WDTM. At this time, the counter is not cleared to 0 but holds its value.

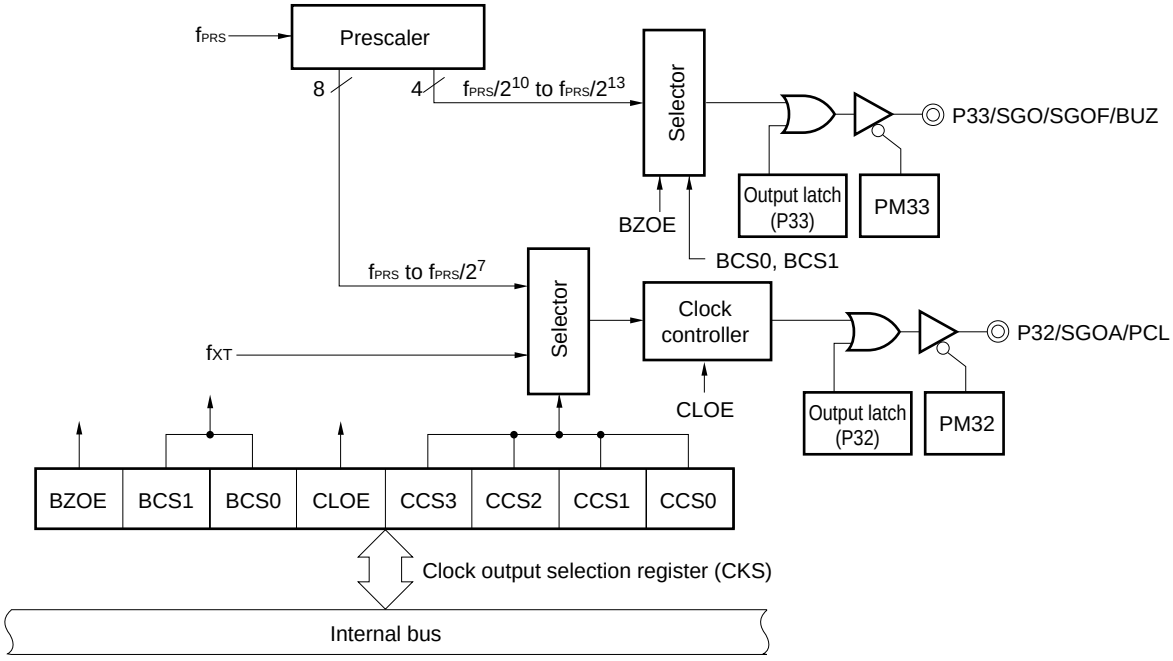
[MEMO]

Chapter 11 Clock Output / Buzzer Output Controller (μPD780822B Subseries)

11.1 Functions of Clock Output/Buzzer Output Controller

The clock output controller is intended for carrier output during remote controlled transmission and clock output for supply to peripheral LSIs. The clock selected with the clock output selection register (CKS) is output.
In addition, the buzzer output is intended for square-wave output of buzzer frequency selected with CKS.
Figure 12-1 shows the block diagram of clock output/buzzer output controller.

Figure 11-1: Block Diagram of Clock Output/Buzzer Output Controller



11.2 Configuration of Clock Output/Buzzer Output Controller

The clock output/buzzer output controller consists of the following hardware.

Table 11-1: Clock Output/Buzzer Output Controller Configuration

Item	Configuration
Control registers	Clock output selection register (CKS) Port mode register 3 (PM3) Port register 3 (P3)

11.3 Register Controlling Clock Output/Buzzer Output Controller

The following two registers are used to control the clock output/buzzer output controller.

- Clock output selection register (CKS)
- Port mode register 3 (PM3)

(1) Clock output selection register (CKS)

This register sets output enable/disable for clock output (PCL) and for the buzzer frequency output (BUZ), and sets the output clock.
CKS is set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears CKS to 00H.

Figure 11-2: Format of Clock Output Selection Register (CKS)

	<7>	6	5	<4>	3	2	1	0	R/W	Address	After Reset
CKS	BZOE	BCS1	BCS0	CLOE	CCS3	CCS2	CCS1	CCS0	R/W	FF40H	00H

BZOE	BUZ output enable/disable specification
0	Clock division circuit operation stopped. BUZ fixed to low level.
1	Clock division circuit operation enabled. BUZ output enabled.

BCS1	BCS0	BUZ output clock selection
0	0	$f_{PRS}/2^{10}$ (15.625 kHz)
0	1	$f_{PRS}/2^{11}$ (7.81 kHz)
1	0	$f_{PRS}/2^{12}$ (3.91 kHz)
1	1	$f_{PRS}/2^{13}$ (1.95 kHz)

CLOE	PCL output enable/disable specification
0	Clock division circuit operation stopped. PCL fixed to low level.
1	Clock division circuit operation enabled. PCL output enabled.

CCS3	CCS2	CCS1	CCS0	PCL output clock selection
0	0	0	0	f_{PRS} (16 MHz)
0	0	0	1	$f_{PRS}/2$ (8 MHz)
0	0	1	0	$f_{PRS}/2^2$ (4 MHz)
0	0	1	1	$f_{PRS}/2^3$ (2 MHz)
0	1	0	0	$f_{PRS}/2^4$ (1 MHz)
0	1	0	1	$f_{PRS}/2^5$ (500 kHz)
0	1	1	0	$f_{PRS}/2^6$ (250 kHz)
0	1	1	1	$f_{PRS}/2^7$ (125 kHz)
1	0	0	0	f_{XT} (32.768 kHz)
Other than above				Setting prohibited

Remarks: 1. f_{PRS}: Peripheral hardware clock oscillation frequency

2. f_{XT}: Subsystem clock oscillation frequency

3. Figures in parentheses are for operation with $f_{PRS} = f_X = 16$ MHz, or $f_{XT} = 32.768$ kHz.

(2) Port mode register 3 (PM3)

This register sets port 3 input/output in 1-bit units.
When using the P32/SGOA/PCL pin for clock output and the P33/SGO/SGOF/BUZ pin for buzzer output, clear PM32, PM33 and the output latch of P32, P33 to 0.
PM3 is set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets PM3 to FFH.

Figure 11-3: Format of Port Mode Register 3 (PM3)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM3	1	1	1	1	PM33	PM32	PM31	PM30	R/W	FF23H	FFH

PM3n	P3n pin I/O mode selection (n = 0 to 3)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

11.4 Clock Output/Buzzer Output Controller Operations

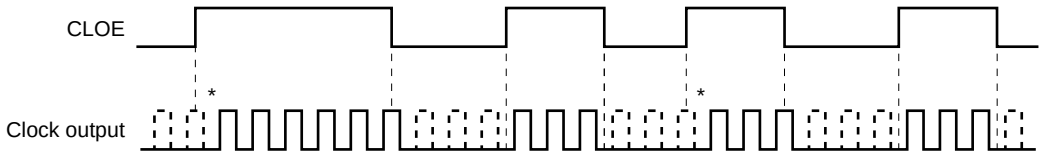
11.4.1 Clock output operation

The clock pulse is output as the following procedure.

<1> Select the clock pulse output frequency with bits 0 to 3 (CCS0 to CCS3) of the clock output selection register (CKS) (clock pulse output in disabled status).
<2> Set bit 4 (CLOE) of CKS to 1 to enable clock output.

Remark: The clock output controller is designed not to output pulses with a small width during output enable/disable switching of the clock output. As shown in Figure 11-4, be sure to start output from the low period of the clock (marked with * in the figure). When stopping output, do so after securing high level of the clock.

Figure 11-4: Remote Control Output Application Example



11.4.2 Operation as buzzer output

The buzzer frequency is output as the following procedure.

<1> Select the buzzer output frequency with bits 5 and 6 (BCS0, BCS1) of the clock output selection register (CKS) (buzzer output in disabled status).
<2> Set bit 7 (BZOE) of CKS to 1 to enable buzzer output.

[MEMO]

Chapter 12 A/D Converter (μPD780822B Subseries)

12.1 Functions of A/D Converter

The A/D converter converts an analog input signal into a digital value, and consists of up to eight channels (ANI0 to ANI7) with a resolution of 10 bits.
The A/D converter has the following two functions.

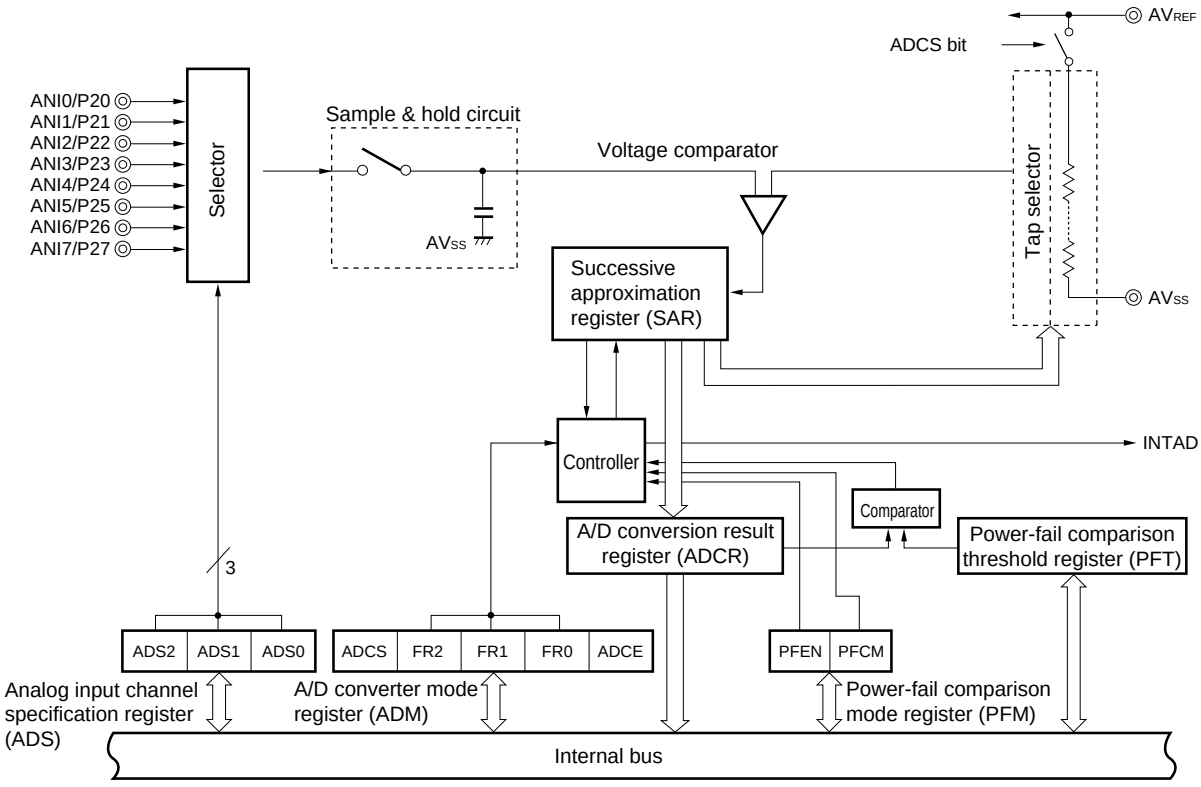
(1) 10-bit resolution A/D conversion

10-bit resolution A/D conversion is carried out repeatedly for one channel selected from analog inputs ANI0 to ANI7. Each time an A/D conversion operation ends, an interrupt request (INTAD) is generated.

(2) Power-fail detection function

This function is used to detect a voltage drop in a battery. The A/D conversion result (ADCR register value) and power-fail comparison threshold register (PFT) value are compared. INTAD is generated only when a comparative condition has been matched.

Figure 12-1: Block Diagram of A/D Converter



12.2 Configuration of A/D Converter

The A/D converter consists of the following hardware.

Table 12-1: Registers of A/D Converter Used on Software

Item	Configuration
Analog Input	8 channels (ANI0 to ANI7)
Registers	Successive approximation register (SAR) A/D conversion result register (ADCR)
Control registers	A/D converter mode register (ADM) Analog input channel specification register (ADS) Power-fail comparison mode register (PFM) Power-fail comparison threshold register (PFT)

(1) ANI0 to ANI7 pins

These are the analog input pins of the 8-channel A/D converter. They input analog signals to be converted into digital signals. Pins other than the one selected as the analog input pin by the analog input channel specification register (ADS) can be used as input port pins.

Caution: Use ANI0 to ANI7 input voltage within the specified range. If a voltage higher than AV_{DD} or lower than AV_{SS} is applied (even if within the absolute maximum rating range), the conversion value of that channel will be undefined and the conversion values of other channels may also be affected.

(2) Sample & hold circuit

The sample & hold circuit samples the input signal of the analog input pin selected by the selector when A/D conversion is started, and holds the sampled analog input voltage value during A/D conversion.

(3) Series resistor string

The series resistor string is connected between AV_{REF} and AV_{SS}, and generates a voltage to be compared with the analog input signal.

(4) Voltage comparator

The voltage comparator compares the sampled analog input voltage and the output voltage of the series resistor string.

(5) Successive approximation register (SAR)

This register compares the sampled analog voltage and the voltage of the series resistor string, and converts the result, starting from the most significant bit (MSB).
When the voltage value is converted into a digital value down to the least significant bit (LSB) (end of A/D conversion), the contents of the SAR register are transferred to the A/D conversion result register (ADCR).

(6) A/D conversion result register (ADCR)

The result of A/D conversion is loaded from the successive approximation register (SAR) to this register each time A/D conversion is completed, and the ADCR register holds the result of A/D conversion in its higher 10 bits (the lower 6 bits are fixed to 0).

(7) Controller

When A/D conversion has been completed or when the power-fail detection function is used, this controller compares the result of A/D conversion (value of the ADCR register) and the value of the power-fail comparison threshold register (PFT). It generates the interrupt INTAD only if a specified comparison condition is satisfied as a result.

(8) AV_{REF} pin

This pin inputs an analog power/reference voltage to the A/D converter. Always use this pin at the same potential as that of the V_{DD} pin even when the A/D converter is not used.

The signal input to ANI0 to ANI7 is converted into a digital signal, based on the voltage applied across AV_{REF} and AV_{SS}.

In the standby mode, the current flowing through the series resistor string can be reduced by lowering the voltage input to the AV_{REF} pin to the AV_{SS} level.

(9) AV_{SS} pin

This is the ground potential pin of the A/D converter. Always use this pin at the same potential as that of the V_{SS} pin even when the A/D converter is not used.

(10) A/D converter mode register (ADM)

This register is used to set the conversion time of the analog input signal to be converted, and to start or stop the conversion operation.

(11) Analog input channel specification register (ADS)

This register is used to specify the port that inputs the analog voltage to be converted into a digital signal.

(12) Power-fail comparison mode register (PFM)

This register is used to set the power-fail monitor mode.

(13) Power-fail comparison threshold register (PFT)

This register is used to set the threshold value that is to be compared with the value of the A/D conversion result register (ADCR).

12.3 Registers Used in A/D Converter

The A/D converter uses the following five registers.

- A/D converter mode register (ADM)
- Analog input channel specification register (ADS)
- A/D conversion result register (ADCR)
- Power-fail comparison mode register (PFM)
- Power-fail comparison threshold register (PFT)

12.3.1 A/D converter mode register (ADM)

This register sets the conversion time for analog input to be A/D converted, and starts/stops conversion.
ADM can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 12-2: Format of A/D Converter Mode Register (ADM)

	<7>	6	5	4	3	2	1	<0>	R/W	Address	After Reset
ADM	ADCS	0	FR2	FR1	FR0	0	0	ADCE	R/W	FF88H	00H

ADCS	A/D conversion operation Control
0	Stops conversion operation
1	Enables conversion operation

FR2	FR1	FR0		Conversion time selection[Note 1]		
				f_{PRS} = 8 MHz	f_{PRS} = 10 MHz	f_{PRS} = 16 MHz
0	0	0	$288/f_{PRS}$	36 µs	28.8 µs	18 µs
0	0	1	$240/f_{PRS}$	30 µs	24 µs	15 µs
0	1	0	$192/f_{PRS}$	24 µs	19.2 µs	12 µs
1	0	0	$144/f_{PRS}$	18 µs	14.4 µs	9 µs
1	0	1	$120/f_{PRS}$	15 µs	12 µs	7.5 µs
1	1	0	$96/f_{PRS}$	12 µs	9.6 µs	6 µs
Other than above			Setting prohibited			

ADCE	Boost reference voltage generator operation control[Note 2]
0	Stops operation of reference voltage generator
1	Enables operation of reference voltage generator ADCE

Caution: Always write 0 to bit 6, 2 and 1.

Notes: 1. Set so that the A/D conversion time is as follows:

- Standard products, (A) grade products: 14 µs or longer but less than 100 µs

2. A booster circuit is incorporated to realize low-voltage operation. The operation of the circuit that generates the reference voltage for boosting is controlled by ADCE, and it takes 14 µs from operation start to operation stabilization. Therefore, when ADCS is set to 1 after 14 µs or more has elapsed from the time ADCE is set to 1, the conversion result at that time has priority over the first conversion result.

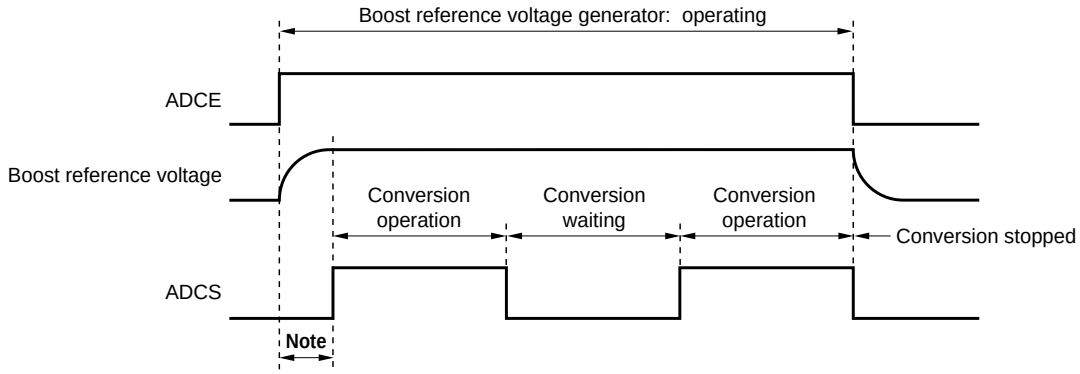
Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

Table 12-2: Settings of ADCS and ADCE

ADCS	ADCE	A/D Conversion Operation
0	0	Stop status (DC power consumption path does not exist)
0	1	Conversion waiting mode (only reference voltage generator consumes power)
1	0	Conversion mode (reference voltage generator operation stopped[Note])
1	1	Conversion mode (reference voltage generator operates)

Note: Data of first conversion cannot be used.

Figure 12-3: Timing Chart When Boost Reference Voltage Generator Is Used



Note: The time from the rising of the ADCE bit to the falling of the ADCS bit must be 14 µs or longer to stabilize the reference voltage.

Cautions: 1. A/D conversion must be stopped before rewriting bits FR0 to FR2 to values other than the identical data.

2. For the sampling time of the A/D converter and the A/D conversion start delay time, see (11) in 13.6 Cautions for A/D Converter.

3. If data is written to ADM, a wait cycle is generated. Do not write data to ADM when the CPU is operating on the subsystem clock and the X1 input clock is stopped. For details, see Chapter 29 "Cautions For Wait (µPD780822B Subseries)" on page 591.

12.3.2 Analog input channel specification register (ADS)

This register specifies the input port of the analog voltage to be A/D converted.
ADS can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 12-4: Format of Analog Input Channel Specification Register (ADS)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ADS	0	0	0	0	0	ADS2	ADS1	ADS0	R/W	FF89H	00H

ADS2	ADS1	ADS0	Analog input channel specification
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

Cautions: 1. Be sure to clear bits 3 to 7 of ADS to 0.

2. If data is written to ADS, a wait cycle is generated. Do not write data to ADS when the CPU is operating on the subsystem clock and the X1 input clock is stopped. For details, see Chapter 29 ”Cautions For Wait (µPD780822B Subseries)” on page 591.

12.3.3 A/D conversion result register (ADCR)

This register is a 16-bit register that stores the A/D conversion result. The lower six bits are fixed to 0. Each time A/D conversion ends, the conversion result is loaded from the successive approximation register, and is stored in ADCR in order starting from the most significant bit (MSB). FF8DH indicates the higher 8 bits of the conversion result, and FF8CH indicates the lower 2 bits of the conversion result.
ADCR can be read by a 16-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input makes ADCR undefined.

Figure 12-5: Format of A/D Conversion Result Register (ADCR)

Address: FF8DH, FF8CH After reset: Undefined R

Symbol	FF8DH								FF8CH							
ADCR											0	0	0	0	0	0

Cautions: 1. When writing to the A/D converter mode register (ADM) and analog input channel specification register (ADS), the contents of ADCR may become undefined. Read the conversion result following conversion completion before writing to ADM and ADS. Using timing other than the above may cause an incorrect conversion result to be read.

2. If data is read from ADCR, a wait cycle is generated. Do not read data from ADCR when the CPU is operating on the subsystem clock and the X1 input clock is stopped. For details, see Chapter 29 ”Cautions For Wait (µPD780822B Subseries)” on page 591.

12.3.4 Power-fail comparison mode register (PFM)

The power-fail comparison mode register (PFM) is used to compare the A/D conversion result (value of the ADCR register) and the value of the power-fail comparison threshold register (PFT).
PFM can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 12-6: Format of Power-Fail Comparison Mode Register (PFM)

	<7>	<6>	5	4	3	2	1	0	R/W	Address	After Reset
PFM	PFEN	PFCM	0	0	0	0	0	0	R/W	FF8AH	00H

PFEN	Power-fail comparison enable
0	Stops power-fail comparison (used as a normal A/D converter)
1	Enables power-fail comparison (used for power-fail detection) PFEN

PFEN		Power-fail comparison mode selection
0	Higher 8 bits of ADCR ≥ PFT	Interrupt request signal (INTAD) generation
	Higher 8 bits of ADCR < PFT	No INTAD generation
1	Higher 8 bits of ADCR ≥ PFT	No INTAD generation
	Higher 8 bits of ADCR < PFT	INTAD generation

Caution: If data is written to PFM, a wait cycle is generated. Do not write data to PFM when the CPU is operating on the subsystem clock and the X1 input clock is stopped. For details, see Chapter 29 "Cautions For Wait (µPD780822B Subseries)" on page 591.

12.3.5 Power-fail comparison threshold register (PFT)

The power-fail comparison threshold register (PFT) is a register that sets the threshold value when comparing the values with the A/D conversion result.
8-bit data in PFT is compared to the higher 8 bits (FF8DH) of the 10-bit A/D conversion result.
PFT can be set by an 1-bit and an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 12-7: Format of Power-Fail Comparison Threshold Register (PFT)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PFT	PFT7	PFT6	PFT5	PFT4	PFT3	PFT2	PFT1	PFT0	R/W	FF8BH	00H

Caution: If data is written to PFT, a wait cycle is generated. Do not write data to PFT when the CPU is operating on the subsystem clock and the X1 input clock is stopped. For details, see Chapter 29 "Cautions For Wait (µPD780822B Subseries)" on page 591.

12.4 A/D Converter Operations

12.4.1 Basic operations of A/D converter

<1> Select one channel for A/D conversion using the analog input channel specification register (ADS).
<2> Set ADCE to 1 and wait for 14 µs or longer.
<3> Set ADCS to 1 and start the conversion operation.
(<4> to <10> are operations performed by hardware.)
<4> The voltage input to the selected analog input channel is sampled by the sample & hold circuit.
<5> When sampling has been done for a certain time, the sample & hold circuit is placed in the hold state and the input analog voltage is held until the A/D conversion operation has ended.
<6> Bit 9 of the successive approximation register (SAR) is set. The series resistor string voltage tap is set to (1/2) AV_{REF} by the tap selector.
<7> The voltage difference between the series resistor string voltage tap and analog input is compared by the voltage comparator. If the analog input is greater than (1/2) AV_{REF}, the MSB of SAR remains set to 1. If the analog input is smaller than (1/2) AVREF, the MSB is reset to 0.
<8> Next, bit 8 of SAR is automatically set to 1, and the operation proceeds to the next comparison. The series resistor string voltage tap is selected according to the preset value of bit 9, as described below.

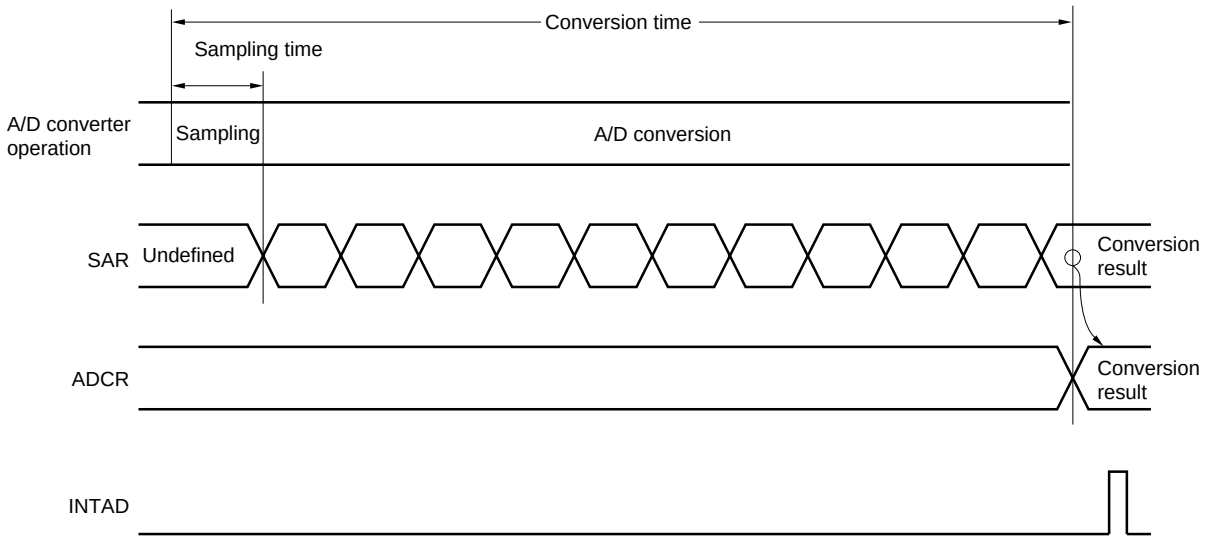
- Bit 9 = 1: (3/4) AV_{REF}
- Bit 9 = 0: (1/4) AV_{REF}

The voltage tap and analog input voltage are compared and bit 8 of SAR is manipulated as follows.

- Analog input voltage ≥ Voltage tap: Bit 8 = 1
- Analog input voltage < Voltage tap: Bit 8 = 0

<9> Comparison is continued in this way up to bit 0 of SAR.
<10>Upon completion of the comparison of 10 bits, an effective digital result value remains in SAR, and the result value is transferred to the A/D conversion result register (ADCR) and then latched.
At the same time, the A/D conversion end interrupt request (INTAD) can also be generated.
<11>Repeat steps <4> to <10>, until ADCS is cleared to 0.
To stop the A/D converter, clear ADCS to 0.
To restart A/D conversion from the status of ADCE = 1, start from <3>. To restart A/D conversion from the status of ADCE = 0, however, start from <2>.

Figure 12-8: Basic Operation of A/D Converter



A/D conversion operations are performed continuously until bit 7 (ADCS) of the A/D converter mode register (ADM) is reset (0) by software.

If a write operation is performed to one of the ADM, analog input channel specification register (ADS), power-fail comparison mode register (PFM), or power-fail comparison threshold register (PFT) during an A/D conversion operation, the conversion operation is initialized, and if the ADCS bit is set (1), conversion starts again from the beginning.

$\overline{\text{RESET}}$ input makes the A/D conversion result register (ADCR) undefined.

12.4.2 Input voltage and conversion results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI7) and the theoretical A/D conversion result (stored in the A/D conversion result register (ADCR)) is shown by the following expression.

$$SAR = INT\left(\frac{V_{AIN}}{AV_{REF}} \times 1024 + 0.5\right)$$

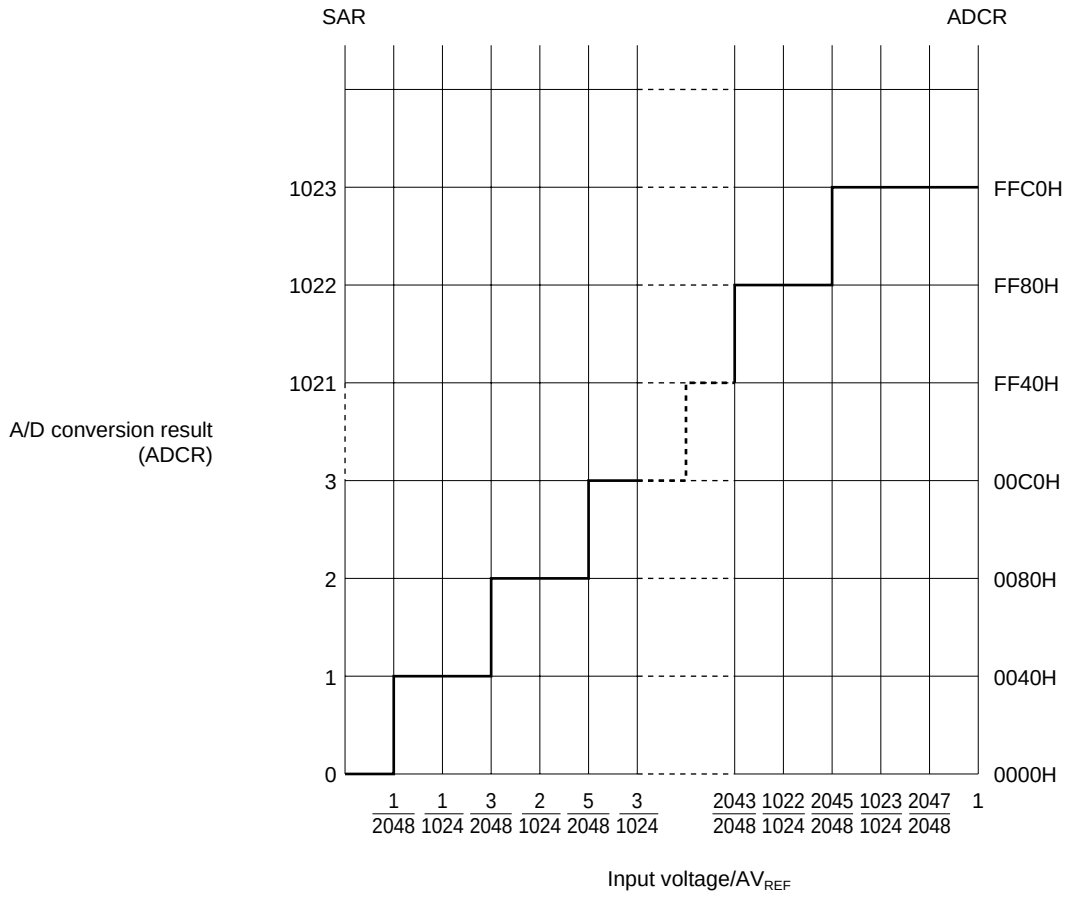
or

$$(ADCR - 0.5) \times \frac{AV_{REF}}{1024} \leq V_{AIN} < (ADCR + 0.5) \times \frac{AV_{REF}}{1024}$$

where, INT( ): Function which returns integer part of value in parentheses
V_{AIN}: Analog input voltage
AV_{REF}: AV_{REF} pin voltage
ADCR: A/D conversion result register (ADCR) value

Figure 13-9 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 12-9: Relationship Between Analog Input Voltage and A/D Conversion Result



12.4.3 A/D converter operation mode

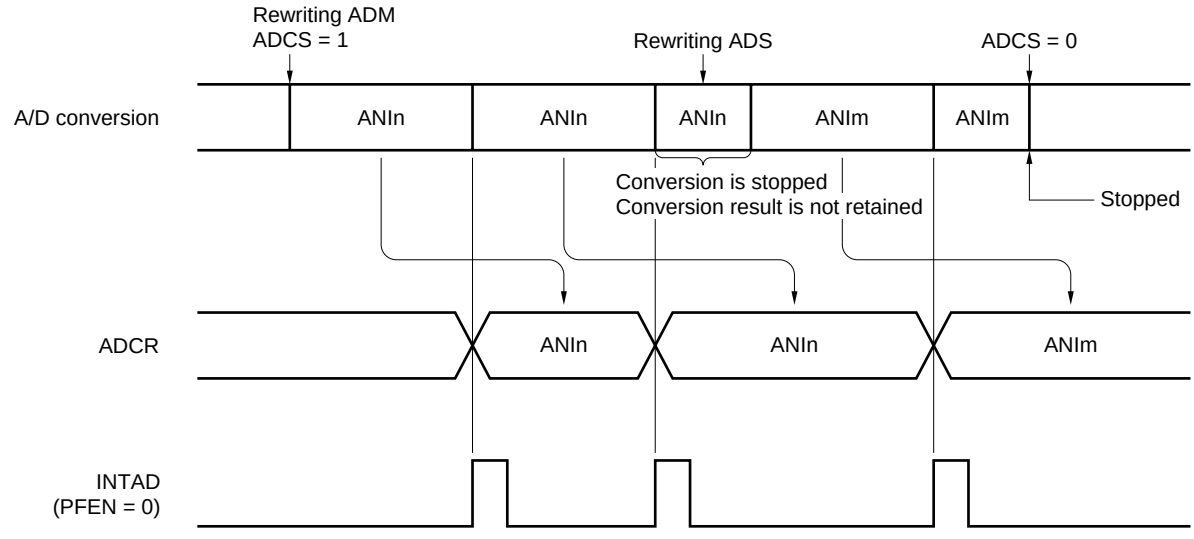
The operation mode of the A/D converter is the select mode. One channel of analog input is selected from ANI0 to ANI7 by the analog input channel specification register (ADS) and A/D conversion is executed.
In addition, the following two functions can be selected by setting of bit 7 (PFEN) of the power-fail comparison mode register (PFM).

- Normal 10-bit A/D converter (PFEN = 0)
- Power-fail detection function (PFEN = 1)

(1) A/D conversion operation (when PFEN = 0)

By setting bit 7 (ADCS) of the A/D converter mode register (ADM) to 1 and bit 7 (PFEN) of the power-fail comparison mode register (PFM) to 0, the A/D conversion operation of the voltage, which is applied to the analog input pin specified by the analog input channel specification register (ADS), is started.
When A/D conversion has been completed, the result of the A/D conversion is stored in the A/D conversion result register (ADCR), and an interrupt request signal (INTAD) is generated. Once the A/D conversion has started and when one A/D conversion has been completed, the next A/D conversion operation is immediately started. The A/D conversion operations are repeated until new data is written to ADS.
If ADM, ADS, the power-fail comparison mode register (PFM), and the power-fail comparison threshold register (PFT) are rewritten during A/D conversion, the A/D conversion operation under execution is stopped and restarted from the beginning.
If 0 is written to ADCS during A/D conversion, A/D conversion is immediately stopped. At this time, the conversion result is undefined.

Figure 12-10: A/D Conversion Operation



Remarks: 1. n = 0 to 7

2. m = 0 to 7

(2) Power-fail detection function (when PFEN = 1)

By setting bit 7 (ADCS) of the A/D converter mode register (ADM) to 1 and bit 7 (PFEN) of the power-fail comparison mode register (PFM) to 1, the A/D conversion operation of the voltage applied to the analog input pin specified by the analog input channel specification register (ADS) is started.

When the A/D conversion has been completed, the result of the A/D conversion is stored in the A/D conversion result register (ADCR), the values are compared with power-fail comparison threshold register (PFT), and an interrupt request signal (INTAD) is generated under the condition specified by bit 6 (PFCM) of PFM.

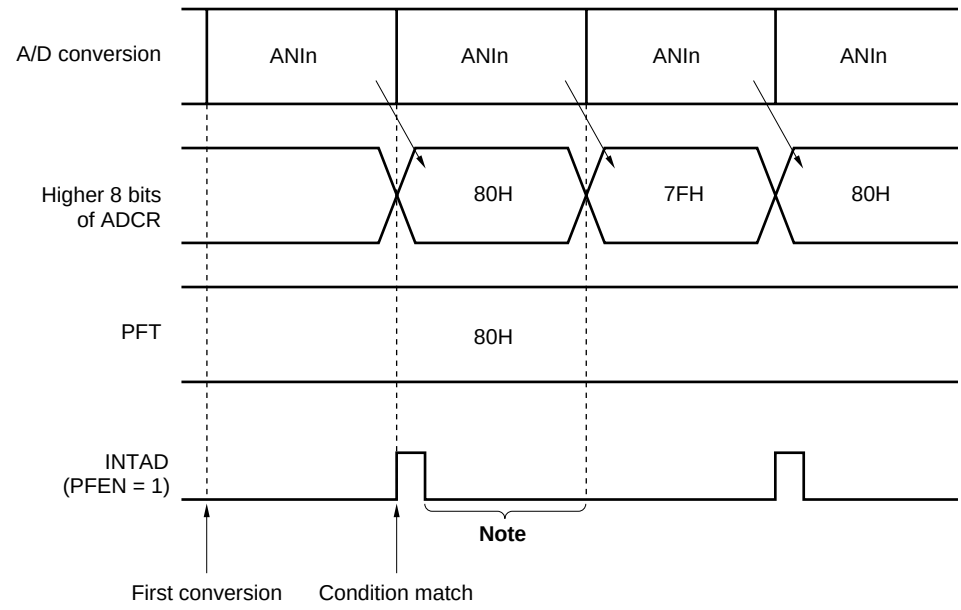
<1> When PFEN = 1 and PFCM = 0

The higher 8 bits of ADCR and PFT values are compared when A/D conversion ends and INTAD is only generated when the higher 8 bits of ADCR ≥ PFT.

<2> When PFEN = 1 and PFCM = 1

The higher 8 bits of ADCR and PFT values are compared when A/D conversion ends and INTAD is only generated when the higher 8 bits of ADCR < PFT.

Figure 12-11: Power-Fail Detection (When PFEN = 1 and PFCM = 0)



Note: If the conversion result is not read before the end of the next conversion after INTAD is output, the result is replaced by the next conversion result.

Remark: n = 0 to 7

The setting methods are described below.

- When used as A/D conversion operation

<1> Set bit 0 (ADCE) of the A/D converter mode register (ADM) to 1.
<2> Select the channel and conversion time using bits 2 to 0 (ADS2 to ADS0) of the analog input channel specification register (ADS) and bits 5 to 3 (FR2 to FR0) of ADM.
<3> Set bit 7 (ADCS) of ADM to 1.
<4> An interrupt request signal (INTAD) is generated.
<5> Transfer the A/D conversion data to the A/D conversion result register (ADCR).
<Change the channel>
<6> Change the channel using bits 2 to 0 (ADS2 to ADS0) of ADS.
<7> An interrupt request signal (INTAD) is generated.
<8> Transfer the A/D conversion data to the A/D conversion result register (ADCR).
<Complete A/D conversion>
<9> Clear ADCS to 0.
<10> Clear ADCE to 0.

Cautions: 1. Make sure the period of <1> to <3> is 14 µs or more.

2. It is no problem if the order of <1> and <2> is reversed.

3. <1> can be omitted. However, do not use the first conversion result after <3> in this case.

4. The period from <4> to <7> differs from the conversion time set using bits 5 to 3 (FR2 to FR0) of ADM. The period from <6> to <7> is the conversion time set using FR2 to FR0.

- When used as power-fail function

<1> Set bit 7 (PFEN) of the power-fail comparison mode register (PFM).
<2> Set power-fail comparison condition using bit 6 (PFCM) of PFM.
<3> Set bit 0 (ADCE) of the A/D converter mode register (ADM) to 1.
<4> Select the channel and conversion time using bits 2 to 0 (ADS2 to ADS0) of the analog input channel specification register (ADS) and bits 5 to 3 (FR2 to FR0) of ADM.
<5> Set a threshold value to the power-fail comparison threshold register (PFT).
<6> Set bit 7 (ADCS) of ADM to 1.
<7> Transfer the A/D conversion data to the A/D conversion result register (ADCR).
<8> The higher 8 bits of ADCR and PFT are compared and an interrupt request signal (INTAD) is generated if the conditions match.
<Change the channel>
<9> Change the channel using bits 2 to 0 (ADS2 to ADS0) of ADS.
<10> Transfer the A/D conversion data to the A/D conversion result register (ADCR).
<11> The higher 8 bits of ADCR and the power-fail comparison threshold register (PFT) are compared and an interrupt request signal (INTAD) is generated if the conditions match.
<Complete A/D conversion>
<12> Clear ADCS to 0.
<13> Clear ADCE to 0.

Cautions: 1. Make sure the period of <3> to <6> is 14 µs or more.

2. It is no problem if the order of <3>, <4>, and <5> is changed.

3. <3> must not be omitted if the power-fail function is used.

4. The period from <7> to <11> differs from the conversion time set using bits 5 to 3 (FR2 to FR0) of ADM. The period from <9> to <11> is the conversion time set using FR2 to FR0.

12.5 How to Read A/D Converter Characteristics Table

Here, special terms unique to the A/D converter are explained.

(1) Resolution

This is the minimum analog input voltage that can be identified. That is, the percentage of the analog input voltage per bit of digital output is called 1LSB (Least Significant Bit). The percentage of 1LSB with respect to the full scale is expressed by %FSR (Full Scale Range).

1LSB is as follows when the resolution is 10 bits.

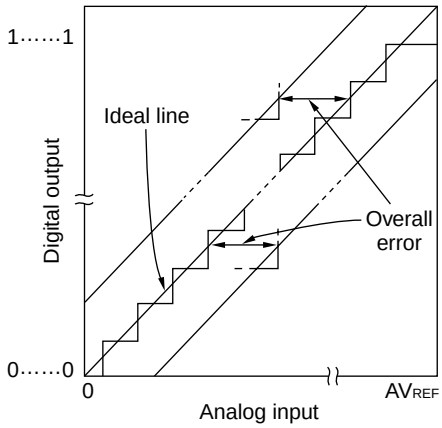
$1LSB = 1/2^{10} = 1/1024$
$= 0.098\%FSR$

Accuracy has no relation to resolution, but is determined by overall error.

(2) Overall error

This shows the maximum error value between the actual measured value and the theoretical value.
Zero-scale error, full-scale error, integral linearity error, and differential linearity errors that are combinations of these express the overall error.
Note that the quantization error is not included in the overall error in the characteristics table.

Figure 12-12: Overall Error

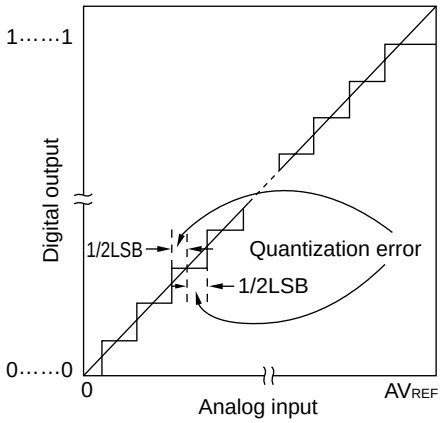


(3) Quantization error

When analog values are converted to digital values, a ±1/2LSB error naturally occurs. In an A/D converter, an analog input voltage in a range of ±1/2LSB is converted to the same digital code, so a quantization error cannot be avoided.

Note that the quantization error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

Figure 12-13: Quantization Error

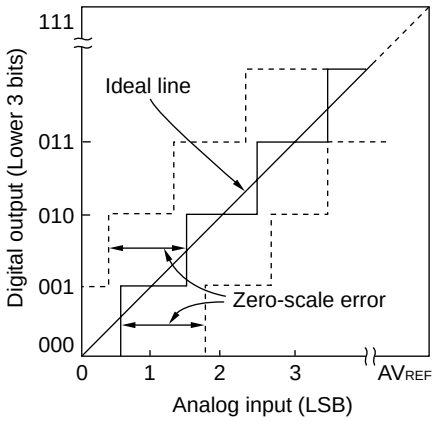


(4) Zero-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (1/2LSB) when the digital output changes from 0.....000 to 0.....001.

If the actual measurement value is greater than the theoretical value, it shows the difference between the actual measurement value of the analog input voltage and the theoretical value (3/2LSB) when the digital output changes from 0……001 to 0……010.

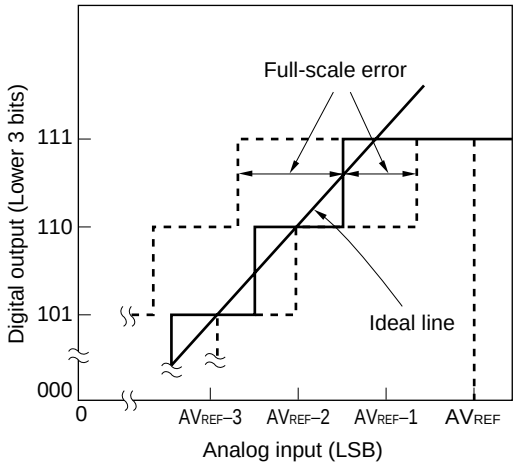
Figure 12-14: Zero-Scale Error



(5) Full-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (Full-scale - 3/2LSB) when the digital output changes from 1.....110 to 1.....111.

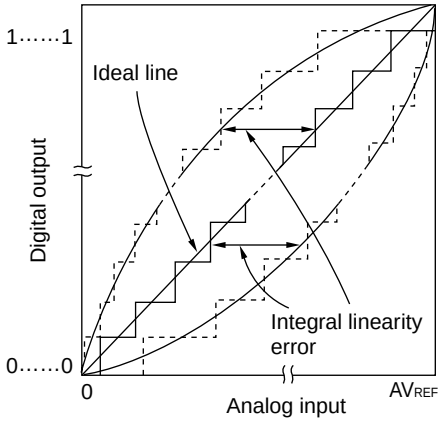
Figure 12-15: Full-Scale Error



(6) Integral linearity error

This shows the degree to which the conversion characteristics deviate from the ideal linear relationship. It expresses the maximum value of the difference between the actual measurement value and the ideal straight line when the zero-scale error and full-scale error are 0.

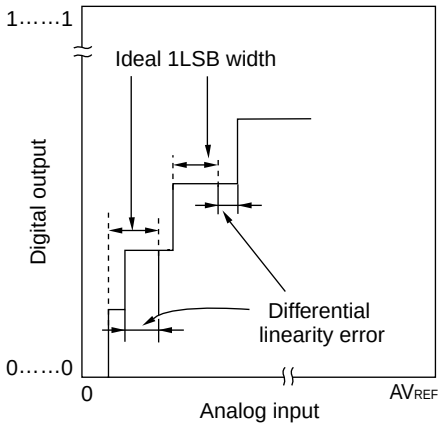
Figure 12-16: Integral Linearity Error



(7) Differential linearity error

While the ideal width of code output is 1LSB, this indicates the difference between the actual measurement value and the ideal value.

Figure 12-17: Differential Linearity Error



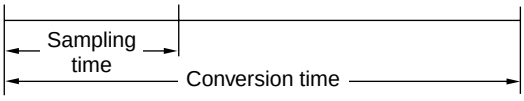
(8) Conversion time

This expresses the time from the start of sampling to when the digital output is obtained.
The sampling time is included in the conversion time in the characteristics table.

(9) Sampling time

This is the time the analog switch is turned on for the analog voltage to be sampled by the sample & hold circuit.

Figure 12-18: Sampling time

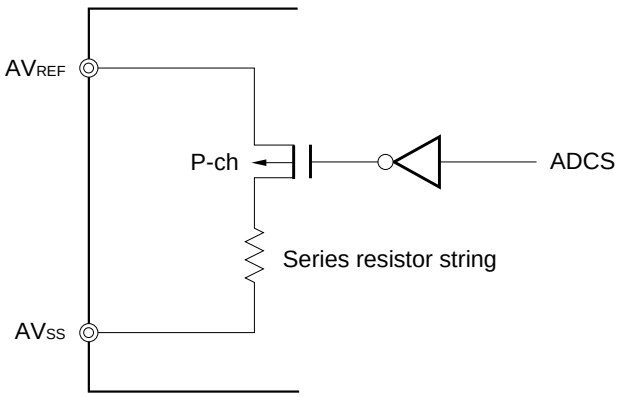


12.6 Cautions for A/D Converter

(1) Operating current in standby mode

The A/D converter stops operating in the standby mode. At this time, the operating current can be reduced by clearing bit 7 (ADCS) of the A/D converter mode register (ADM) to 0.
Figure 12-19 shows the circuit configuration of the series resistor string.

Figure 12-19: Circuit Configuration of Series Resistor String



(2) Input range of ANI0 to ANI7

Observe the rated range of the ANI0 to ANI7 input voltage. If a voltage of AV_{REF} or higher and AV_{SS} or lower (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

(3) Conflicting operations

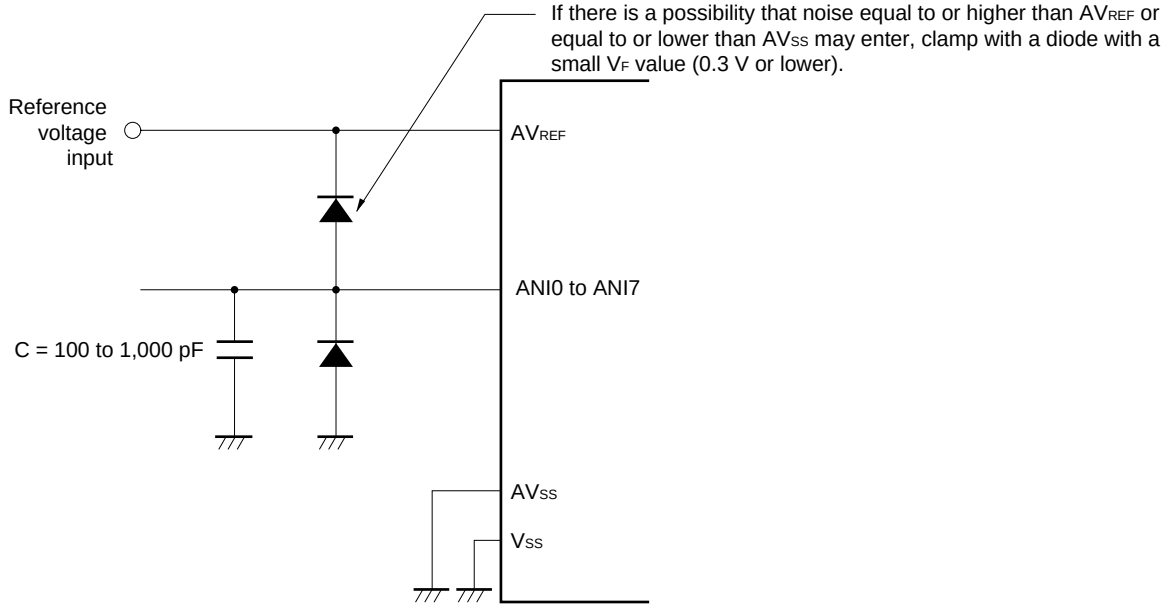
<1> Conflict between A/D conversion result register (ADCR) write and ADCR read by instruction upon the end of conversion.
ADCR read has priority. After the read operation, the new conversion result is written to ADCR.

<2> Conflict between ADCR write and A/D converter mode register (ADM) write or analog input channel specification register (ADS) write upon the end of conversion.
ADM or ADS write has priority. ADCR write is not performed, nor is the conversion end interrupt signal (INTAD) generated.

(4) Noise counter measures

To maintain the 10-bit resolution, attention must be paid to noise input to the AV_{REF} pin and pins ANI0 to ANI7. Because the effect increases in proportion to the output impedance of the analog input source, it is recommended that a capacitor be connected externally, as shown in Figure 12-19, to reduce noise.

Figure 12-20: Analog Input Pin Connection



(5) ANI0/P20 to ANI7/P27

<1> The analog input pins (ANI0 to ANI7) are also used as input port pins (P20 to P27). When A/D conversion is performed with any of ANI0 to ANI7 selected, do not access port 2 while conversion is in progress; otherwise the conversion resolution may be degraded.

<2> If a digital pulse is applied to the pins adjacent to the pins currently used for A/D conversion, the expected value of the A/D conversion may not be obtained due to coupling noise. Therefore, do not apply a pulse to the pins adjacent to the pin undergoing A/D conversion.

(6) Input impedance of ANI0 to ANI7 pins

In this A/D converter, the internal sampling capacitor is charged and sampling is performed for approx. one sixth of the conversion time.
Since only the leakage current flows other than during sampling and the current for charging the capacitor also flows during sampling, the input impedance fluctuates and has no meaning.
To perform sufficient sampling, however, it is recommended to make the output impedance of the analog input source 10 k. or lower, or attach a capacitor of around 100 pF to the ANI0 to ANI7 pins (see Figure 12-20).

(7) AV_{REF} pin input impedance

A series resistor string of several tens of 10 kΩ is connected between the AV_{REF} and AV_{SS} pins.
Therefore, if the output impedance of the reference voltage source is high, this will result in a series connection to the series resistor string between the AV_{REF} and AV_{SS} pins, resulting in a large reference voltage error.

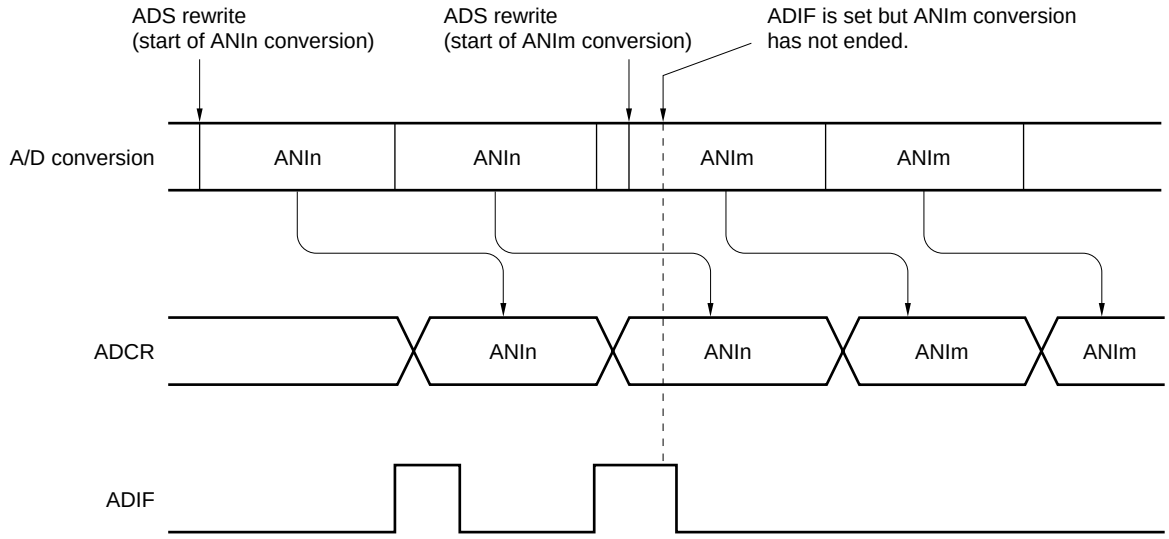
(8) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the analog input channel specification register (ADS) is changed.

Therefore, if an analog input pin is changed during A/D conversion, the A/D conversion result and ADIF for the pre-change analog input may be set just before the ADS rewrite. Caution is therefore required since, at this time, when ADIF is read immediately after the ADS rewrite, ADIF is set despite the fact A/D conversion for the post-change analog input has not ended.

When A/D conversion is stopped and then resumed, clear ADIF before the A/D conversion operation is resumed.

Figure 12-21: Timing of A/D Conversion End Interrupt Request Generation



Remarks: 1. n = 0 to 7

2. m = 0 to 7

(9) Conversion results just after A/D conversion start

The first A/D conversion value immediately after A/D conversion starts may not fall within the rating range if the ADCS bit is set to 1 within 14 µs after the ADCE bit was set to 1, or if the ADCS bit is set to 1 with the ADCE bit = 0. Take measures such as polling the A/D conversion end interrupt request (INTAD) and removing the first conversion result.

(10) A/D conversion result register (ADCR) read operation

When a write operation is performed to the A/D converter mode register (ADM) and analog input channel specification register (ADS), the contents of ADCR may become undefined. Read the conversion result following conversion completion before writing to ADM and ADS. Using a timing other than the above may cause an incorrect conversion result to be read.

(11) A/D converter sampling time and A/D conversion start delay time

The A/D converter sampling time differs depending on the set value of the A/D converter mode register (ADM). The delay time exists until actual sampling is started after A/D converter operation is enabled.

When using a set in which the A/D conversion time must be strictly observed, care is required for the contents shown in Figure 12-22 and Table 12-3.

Figure 12-22: Timing of A/D Converter Sampling and A/D Conversion Start Delay

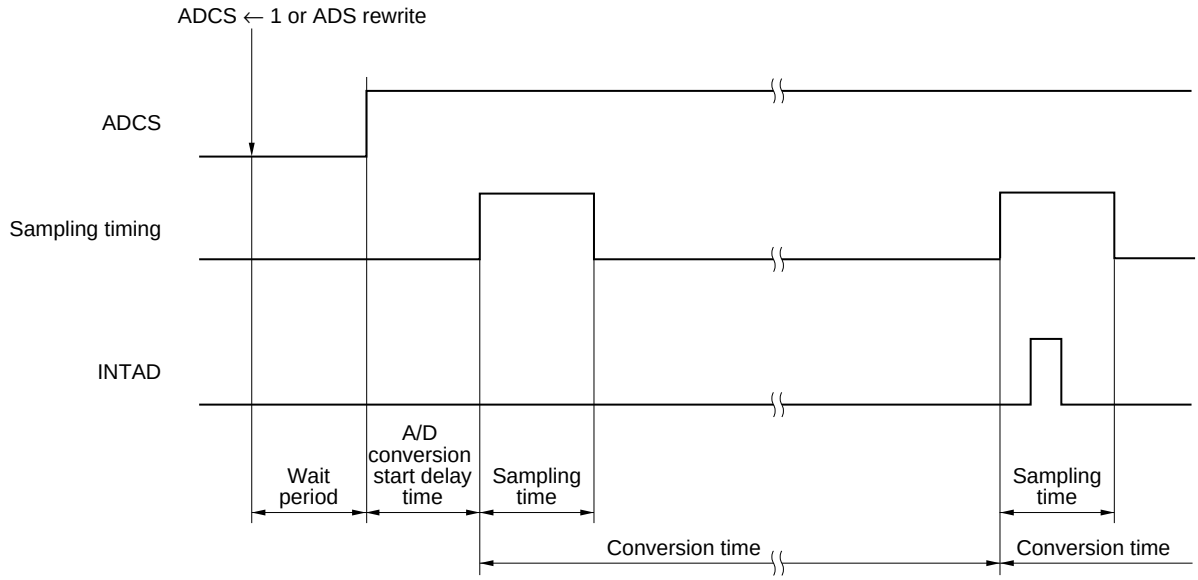


Table 12-3: A/D converter sampling time and A/D conversion start delay time (ADM set value)

FR2	FR1	FR0	Conversion Time	Sampling Time	A/D Conversion Start Delay Time[Note]
0	0	0	$288/f_{PRS}$	$40/f_{PRS}$	$4/f_{PRS}$
0	0	1	$240/f_{PRS}$	$32/f_{PRS}$	$4/f_{PRS}$
0	1	0	$192/f_{PRS}$	$24/f_{PRS}$	$4/f_{PRS}$
1	0	0	$144/f_{PRS}$	$20/f_{PRS}$	$2/f_{PRS}$
1	0	1	$120/f_{PRS}$	$16/f_{PRS}$	$2/f_{PRS}$
1	1	0	$96/f_{PRS}$	$12/f_{PRS}$	$2/f_{PRS}$
Other than above			Setting prohibited	-	-

Note: The A/D conversion start delay time is the time after wait period. For the wait function, see **Chapter 29 ”Cautions For Wait (µPD780822B Subseries)” on page 591**.

Remark: f_{PRS}: Peripheral hardware oscillation frequency

(12) Register generating wait cycle

Do not read data from the ADCR register and do not write data to the ADM, ADS, PFM, and PFT registers while the CPU is operating on the subsystem clock and while oscillation of the clock input to X1 is stopped.

(13) Internal equivalent circuit

The equivalent circuit of the analog input block is shown below.

Figure 12-23: Internal Equivalent Circuit of ANIn Pin

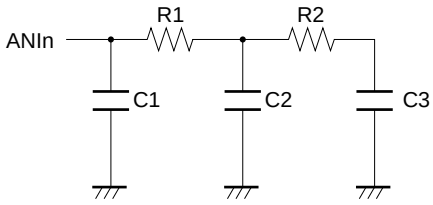


Table 12-4: Resistance and Capacitance Values of Equivalent Circuit (Reference Values)

AV_{REF}	R1	R2	C1	C2	C3
4.5 V	4 kΩ	2.7 kΩ	8 pF	1.4 pF	2 pF

Remarks: 1. The resistance and capacitance values shown in Table 12-4are not guaranteed values.

2. n = 0 to 7

12.7 Cautions on Emulation performed with IE-78K0-NS-A Emulator and IE-780822-NS-EM1 Emulation Board

The address of the special function register ADCR on the In-Circuit Emulator IE-78K0-NS-A differs from the device.

Table 12-5: Different SFR Setting for IE-780822-NS-EM1 Emulation Board

SFR	Real chip address	In-Circuit-Emulator address
ADCR	0xFF8C	0xFF9E
ADCRL	0xFF8C	0xFF9E
ADCRH	0xFF8D	0xFF9F

Note: The address of the special function register ADCR on the In-Circuit Emulator QB-780822 is the same as on the device.

[MEMO]

Chapter 13 Serial Interfaces CSI10 and CSI11 (µPD780822B Subseries)

The µPD780822B Subseries incorporates serial interfaces CSI10 and CSI11.

13.1 Functions of Serial Interfaces CSI10 and CSI11

Serial interfaces CSI10 and CSI11 have the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial communication is not performed and can enable a reduction in the power consumption.
For details, see **13.4.1 ”Operation stop mode” on page 305**.

(2) 3-wire serial I/O mode (MSB/LSB-first selectable)

This mode is used to communicate 8-bit data using three lines: a serial clock line ($\overline{\text{SCK1n}}$) and two serial data lines (SI1n and SO1n).
The processing time of data communication can be shortened in the 3-wire serial I/O mode because transmission and reception can be simultaneously executed.
In addition, whether 8-bit data is communicated with the MSB or LSB first can be specified, so this interface can be connected to any device.
The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface.
For details, see **13.4.2 ”3-wire serial I/O mode” on page 306**.

13.2 Configuration of Serial Interfaces CSI10 and CSI11

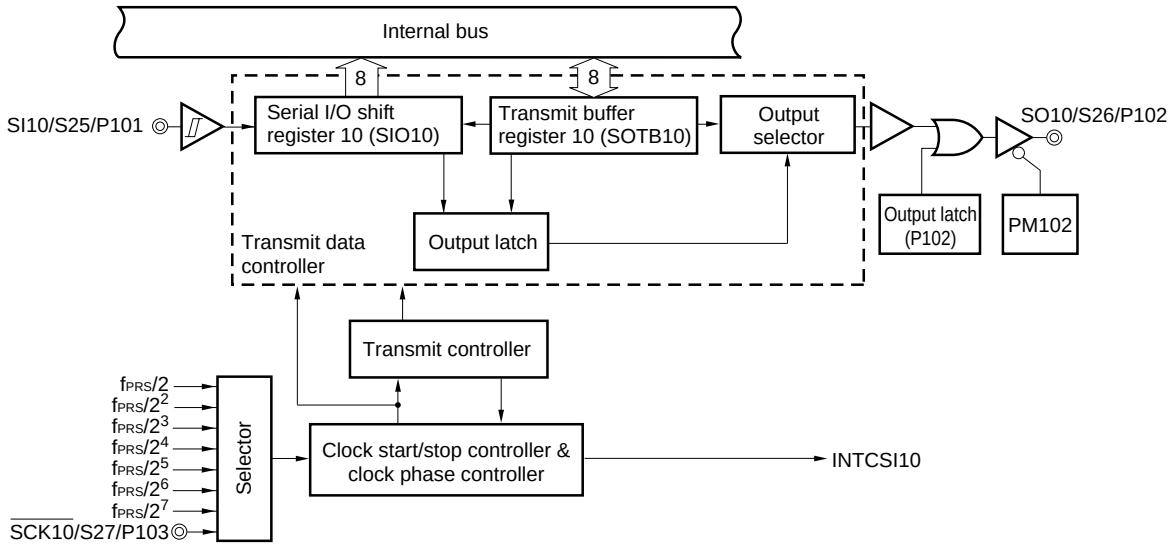
Serial interfaces CSI10 and CSI11 consist of the following hardware.

Table 13-1: Configuration of Serial Interfaces CSI10 and CSI11

Item	Configuration
Registers	Transmit buffer register 1n (SOTB1n) Serial I/O shift register 1n (SIO1n) Transmit controller Clock start/stop controller & clock phase controller
Control registers	Serial operation mode register 1n (CSIM1n) Serial clock selection register 1n (CSIC1n) Port mode register 10 (PM10) or port mode register 11 (PM11) Port register 10 (P10) or port register 11 (P11)

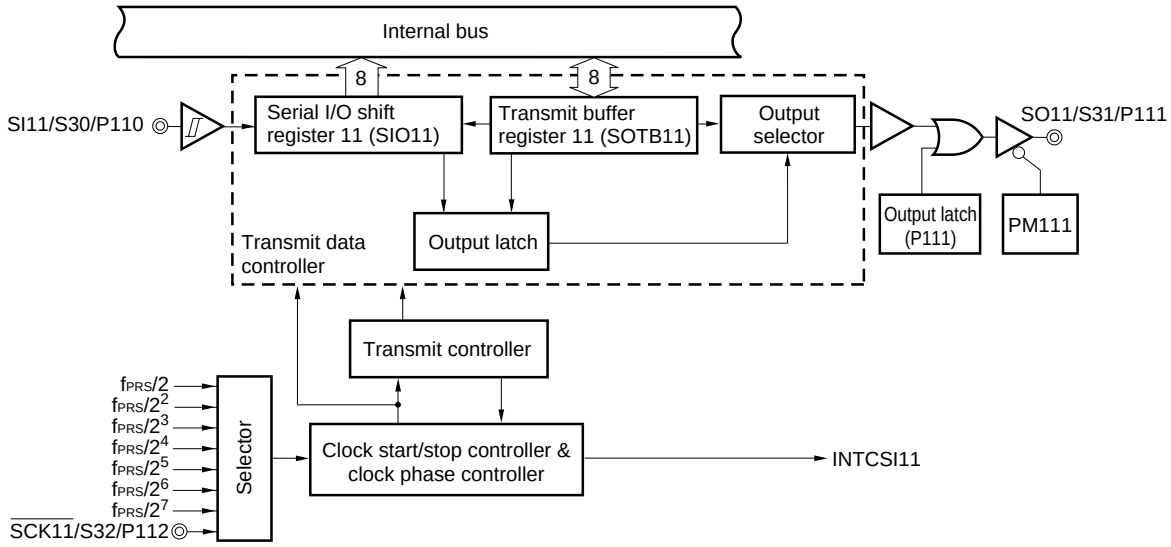
Remark: n = 0, 1

Figure 13-1: Block Diagram of Serial Interface CSI10



Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

Figure 13-2: Block Diagram of Serial Interface CSI11



Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

(1) Transmit buffer register 1n (SOTB1n)

This register sets the transmit data.

Transmission/reception is started by writing data to SOTB1n when bit 7 (CSIE1n) and bit 6 (TRMD1n) of serial operation mode register 1n (CSIM1n) is 1.

The data written to SOTB1n is converted from parallel data into serial data by serial I/O shift register 1n, and output to the serial output pin (SO1n).

SOTB1n can be written or read by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears this register to 00H.

Caution: Do not access SOTB1n when CSOT1n = 1 (during serial communication).

(2) Serial I/O shift register 1n (SIO1n)

This is an 8-bit register that converts data from parallel data into serial data and vice versa.

This register can be read by an 8-bit memory manipulation instruction.

Reception is started by reading data from SIO1n if bit 6 (TRMD1n) of serial operation mode register 1n (CSIM1n) is 0.

During reception, the data is read from the serial input pin (SI1n) to SIO1n.

$\overline{\text{RESET}}$ input clears this register to 00H.

Caution: Do not access SIO1n when CSOT1n = 1 (during serial communication).

Remark: n = 0, 1

13.3 Registers Controlling Serial Interfaces CSI10 and CSI11

Serial interfaces CSI10 and CSI11 are controlled by the following four registers.

- Serial operation mode register 1n (CSIM1n)
- Serial clock selection register 1n (CSIC1n)
- Port mode register 10 (PM10) or port mode register 11 (PM11)
- Port register 10 (P10) or port register 11 (P11)

13.3.1 Serial operation mode register 1n (CSIM1n)

CSIM1n is used to select the operation mode and enable or disable operation.
CSIM1n can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Remark: n = 0, 1

Figure 13-3: Format of Serial Operation Mode Register 10 (CSIM10) (1/2)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM10	CSIE10	TRMD10	0	DIR10	0	0	0	CSOT10	R/W **Note 1**	FF10H	00H

CSIE10	Operation control in 3-wire serial I/O mode
0	Disables operation**Note 2** and asynchronously resets the internal circuit**Note 3**
1	Enables operation

TRMD10**Note 4**	Transmit/receive mode control
0**Note 5**	Receive mode (transmission disabled)
1	Transmit/receive mode

DIR10**Note 6**	First bit specification
0	MSB
1	LSB

Figure 13-3: Format of Serial Operation Mode Register 10 (CSIM10) (2/2)

CSOT10	Communication status flag
0	Communication is stopped
1	Communication is in progress

Notes: 1. Bit 0 is a read-only bit.

2. When using as a general-purpose port, see **Caution 3** of **Figure 13-5, "Format of Serial Clock Selection Register 10 (CSIC10) (1/2)," on page 301** and **Table 13-2, "Relationship Between Register Settings and Pins for Serial interface CSI10," on page 307**.

3. Bit 0 (CSOT10) of CSIM10 and serial I/O shift register 10 (SIO10) are reset.

4. Do not rewrite TRMD10 when CSOT10 = 1 (during serial communication).

5. The SO10 output is fixed to the low level when TRMD10 is 0. Reception is started when data is read from SIO10.

6. Do not rewrite DIR10 when CSOT10 = 1 (during serial communication).

Caution: Be sure to clear bit 5 to 0.

Figure 13-4: Format of Serial Operation Mode Register 11 (CSIM11)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM11	CSIE11	TRMD11	0	DIR11	0	0	0	CSOT11	R/W **Note 1**	FF60H	00H

CSIE11	Operation control in 3-wire serial I/O mode
0	Disables operation**Note 2** and asynchronously resets the internal circuit**Note 3**
1	Enables operation

TRMD11**Note 4**	Transmit/receive mode control
0**Note 5**	Receive mode (transmission disabled)
1	Transmit/receive mode

DIR11**Note 6**	First bit specification
0	MSB
1	LSB

CSOT11	Communication status flag
0	Communication is stopped
1	Communication is in progress

Notes: 1. Bit 0 is a read-only bit.

2. When using as a general-purpose port, see **Caution 3** of **Figure 13-6, "Format of Serial Clock Selection Register 11 (CSIC11) (1/2)," on page 303** and **Table 13-3, "Relationship Between Register Settings and Pins for Serial interface CSI11," on page 308**.

3. Bit 0 (CSOT11) of CSIM11 and serial I/O shift register 11 (SIO11) are reset.

4. Do not rewrite TRMD11 when CSOT11 = 1 (during serial communication).

5. The SO11 output is fixed to the low level when TRMD11 is 0. Reception is started when data is read from SIO11.

6. Do not rewrite DIR11 when CSOT11 = 1 (during serial communication).

13.3.2 Serial clock selection register 1n (CSIC1n)

This register specifies the timing of the data transmission/reception and sets the serial clock.
CSIC1n can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Remark: n = 0, 1

Figure 13-5: Format of Serial Clock Selection Register 10 (CSIC10) (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIC10	0	0	0	CKP10	DAP10	CKS102	CKS101	CKS100	R/W	FF11H	00H

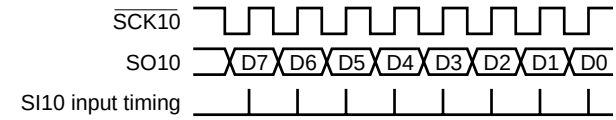
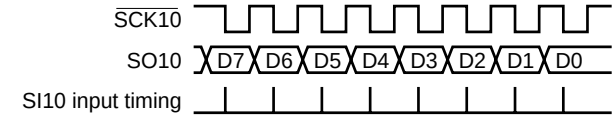
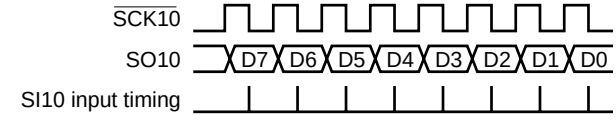
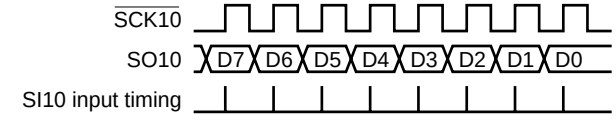
CKP10	DAP10	Specification of data transmission/reception timing	Type
0	0		1
0	1		2
1	0		3
1	1		4

Figure 13-5: Format of Serial Clock Selection Register 10 (CSIC10) (2/2)

CKS102	CKS101	CKS100	CSI10 serial clock selection	Mode
0	0	0	$f_{PRS}/2$ (8 MHz)	Master mode
0	0	1	$f_{PRS}/2^2$ (4 MHz)	Master mode
0	1	0	$f_{PRS}/2^3$ (2 MHz	Master mode
0	1	1	$f_{PRS}/2^4$ (1 MHz)	Master mode
1	0	0	$f_{PRS}/2^5$ (500 kHz)	Master mode
1	0	1	$f_{PRS}/2^6$ (250 kHz)	Master mode
1	1	0	$f_{PRS}/2^7$ (125 kHz)	Master mode
1	1	1	External clock input to $\overline{SCK10}$	Slave mode

Cautions: 1. When the internal low-speed oscillator clock is selected as the clock supplied to the CPU, the clock of the internal low-speed oscillator is divided and supplied as the serial clock. At this time, the operation of serial interface CSI10 is not guaranteed.

2. Do not write to CSIC10 while CSIE10 = 1 (operation enabled).

3. Clear CKP10 to 0 to use $\overline{SCK10}$/S27/P103, SI10/S25/P101, and SO10/S26/P102 as general-purpose port pins.

4. The phase type of the data clock is type 1 after reset.

Remarks: 1. Figures in parentheses are for operation with $f_{PRS} = f_X = 16$ MHz

2. f_{PRS}: Peripheral hardware clock oscillation frequency

Figure 13-6: Format of Serial Clock Selection Register 11 (CSIC11) (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIC11	0	0	0	CKP11	DAP11	CKS112	CKS111	CKS110	R/W	FF61H	00H

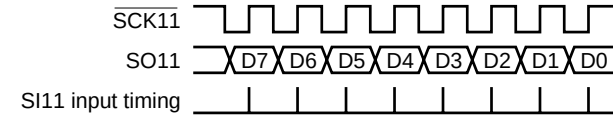
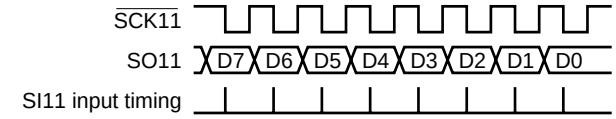
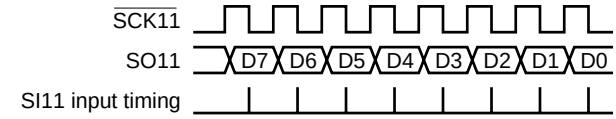
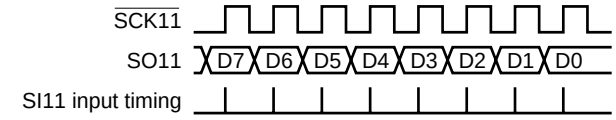
CKP11	DAP11	Specification of data transmission/reception timing	Type
0	0		1
0	1		2
1	0		3
1	1		4

Figure 13-6: Format of Serial Clock Selection Register 11 (CSIC11) (2/2)

CKS112	CKS111	CKS110	CSI11 serial clock selection	Mode
0	0	0	$f_{PRS}/2$ (8 MHz)	Master mode
0	0	1	$f_{PRS}/2^2$ (4 MHz)	Master mode
0	1	0	$f_{PRS}/2^3$ (2 MHz	Master mode
0	1	1	$f_{PRS}/2^4$ (1 MHz)	Master mode
1	0	0	$f_{PRS}/2^5$ (500 kHz)	Master mode
1	0	1	$f_{PRS}/2^6$ (250 kHz)	Master mode
1	1	0	$f_{PRS}/2^7$ (125 kHz)	Master mode
1	1	1	External clock input to $\overline{SCK11}$	Slave mode

Cautions: 1. When the internal low-speed oscillator clock is selected as the clock supplied to the CPU, the clock of the internal low-speed oscillator is divided and supplied as the serial clock. At this time, the operation of serial interface CSI10 is not guaranteed.

2. Do not write to CSIC11 while CSIE11 = 1 (operation enabled).

3. Clear CKP11 to 0 to use $\overline{SCK11}$/S23/P112, SI11/S30/P110, and SO11/S31/P111 as general-purpose port pins.

4. The phase type of the data clock is type 1 after reset.

Remarks: 1. Figures in parentheses are for operation with $f_{PRS} = f_X$ = 16 MHz

2. f_{PRS}: Peripheral hardware clock oscillation frequency

13.3.3 Port mode registers 10 and 11 (PM10, PM11)

These registers set port 10 and 11 input/output in 1-bit units.
When using $\overline{SCK10}$/S27/P103 and $\overline{SCK11}$/S32/P112 as the clock output pins of the serial interface, and SO10/S26/P102 and SO11/S31/P111 as the data output pins, clear PM103, PM112, PM102, PM111, and the output latches of P103, P112, P102, and P111 to 0.
When using $\overline{SCK10}$/S27/P103 and $\overline{SCK11}$/S32/P112 as the clock input pins of the serial interface, SI10//S26/P102 and SI11/S30/P110 as the data input pins, set PM102, PM112, PM103, and PM110 to 1. At this time, the output latches of PM102, PM112, PM103, and PM110 may be 0 or 1.
PM10 and PM11 can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{RESET}$ input sets these registers to FFH.

13.4 Operation of Serial Interfaces CSI10 and CSI11

Serial interfaces CSI10 and CSI11 can be used in the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

13.4.1 Operation stop mode

Serial communication is not executed in this mode. Therefore, the power consumption can be reduced. In addition, the SI10/S25/P101, SO10/S26/P102, $\overline{\text{SCK10}}$/S27/P103, SI11/S30/P110, SO11/S31/P111 and $\overline{\text{SCK11}}$/S32/P112 pins can be used as ordinary I/O port pins in this mode.

(1) Register used

The operation stop mode is set by serial operation mode register 1n (CSIM1n).
To set the operation stop mode, clear bit 7 (CSIE1n) of CSIM1n to 0.

CSIM1n can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears CSIM1n to 00H.

Remark: n = 0, 1

Figure 13-7: Serial operation mode register 1n (CSIM1n) Format

(a) Serial operation mode register 10 (CSIM10)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM10	CSIE10	TRMD10	0	DIR10	0	0	0	CSOT10	R/W	FF10H	00H

CSIE10	Operation control in 3-wire serial I/O mode
0	Disables operation and asynchronously resets the internal circuit**Note**

Note: Bit 0 (CSOT10) of CSIM10 and serial I/O shift register 10 (SIO10) are reset.

(b) Serial operation mode register 11 (CSIM11)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM11	CSIE11	TRMD11	0	DIR11	0	0	0	CSOT11	R/W	FF60H	00H

CSIE11	Operation control in 3-wire serial I/O mode
0	Disables operation and asynchronously resets the internal circuit**Note**

Note: Bit 0 (CSOT11) of CSIM11 and serial I/O shift register 11 (SIO11) are reset.

13.4.2 3-wire serial I/O mode

The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface.
In this mode, communication is executed by using three lines: the serial clock ($\overline{SCK1n}$), serial output (SO1n), and serial input (SI1n) lines.

(1) Registers used:

- Serial operation mode register 1n (CSIM1n)
- Serial clock selection register 1n (CSIC1n)
- Port mode register 10 (PM10) or port mode register 11 (PM11)
- Port register 10 (P10) or port register 11 (P11)

The basic procedure of setting an operation in the 3-wire serial I/O mode is as follows.

<1> Set the CSIC1n register (see **Figures 13-5** and **13-6**).
<2> Set bits 0, 4 and 6 (CSOT1n, DIR1n, and TRMD1n) of the CSIM1n register (see **Figures 13-3** and **13-4**).
<3> Set bit 7 (CSIE1n) of the CSIM1n register to 1. → Transmission/reception is enabled.
<4> Write data to transmit buffer register 1n (SOTB1n). → Data transmission/reception is started.
Read data from serial I/O shift register 1n (SIO1n). → Data reception is started.

Caution: Take relationship with the other party of communication when setting the port mode register and port register.

Remark: n = 0, 1

The relationship between the register settings and pins is shown below.

Table 13-2: Relationship Between Register Settings and Pins for Serial interface CSI10

CSIE10	TRMD10	PM101	P101	PM102	P102	PM103	P103	CSI10 Operation	Pin Function		
									SI10/S25/P101	SO10/S26/P102	$\overline{SCK10}$/S27/P103
0	×	×**Note 1**	×**Note 1**	×**Note 1**	×**Note 1**	×**Note 1**	×**Note 1**	Stop	S25/P101	S26/P102	S27/P103**Note 2**
1	0	1	×	×**Note 1**	×**Note 1**	1	×	Slave reception-**Note 3**	SI10	S26/P102	$\overline{SCK10}$ (input)**Note 3**
1	1	×**Note 1**	×**Note 1**	0	0	1	×	Slave transmission**Note 3**	S25/P101	SO10	$\overline{SCK10}$ (input)**Note 3**
1	1	1	×	0	0	1	×	Slave transmission/reception-**Note 3**	SI10	SO10	$\overline{SCK10}$ (input)**Note 3**
1	0	1	×	×**Note 1**	×**Note 1**	0	1	Master reception	SI10	S26/P102	$\overline{SCK10}$ (output)
1	1	×**Note 1**	×**Note 1**	0	0	0	1	Master transmission	S25/P101	SO10	$\overline{SCK10}$ (output)
1	1	1	×	0	0	0	1	Master transmission/reception	SI10	SO10	$\overline{SCK10}$ (output)

Notes: 1. Can be set as port function.

2. To use $\overline{SCK10}$/S27/P103 as port pins, clear CKP10 to 0.

3. To use the slave mode, set CKS102, CKS101, and CKS100 to 1, 1, 1.

Remark:

×:	don’t care
CSIE10:	Bit 7 of serial operation mode register 10 (CSIM10)
TRMD10:	Bit 6 of CSIM10
CKP10:	Bit 4 of serial clock selection register 10 (CSIC10)
CKS102, CKS101, CKS100:	Bits 2 to 0 of CSIC10
PM10×:	Port mode register
P10×:	Port output latch

Table 13-3: Relationship Between Register Settings and Pins for Serial interface CSI11

CSIE11	TRMD11	PM110	P110	PM111	P111	PM112	P112	CSI10 Operation	Pin Function		
									SI11/S30/P110	SO11/S31/P111	$\overline{\text{SCK11}}$/S32/P112
0	×	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	Stop	S30/P110	S31/P111	S32/P112Note 2
1	0	1	×	×Note 1	×Note 1	1	×	Slave reception-Note 3	SI10	S31/P111	$\overline{\text{SCK10}}$ (input)Note 3
1	1	×Note 1	×Note 1	0	0	1	×	Slave transmis-sionNote 3	S30/P110	SO10	$\overline{\text{SCK10}}$ (input)Note 3
1	1	1	×	0	0	1	×	Slave transmis-sion/reception-Note 3	SI10	SO10	$\overline{\text{SCK10}}$ (input)Note 3
1	0	1	×	×Note 1	×Note 1	0	1	Master reception	SI10	S31/P111	$\overline{\text{SCK10}}$ (output)
1	1	×Note 1	×Note 1	0	0	0	1	Master transmis-sion	S30/P110	SO10	$\overline{\text{SCK10}}$ (output)
1	1	1	×	0	0	0	1	Master transmis-sion/reception	SI10	SO10	$\overline{\text{SCK10}}$ (output)

Notes: 1. Can be set as port function.

2. To use $\overline{\text{SCK11}}$/S32/P112 as port pins, clear CKP11 to 0.

3. To use the slave mode, set CKS112, CKS111, and CKS110 to 1, 1, 1.

Remark: ×: don't care
CSIE11: Bit 7 of serial operation mode register 11 (CSIM11)
TRMD11: Bit 6 of CSIM11
CKP11: Bit 4 of serial clock selection register 11 (CSIC11)
CKS112, CKS111, CKS110: Bits 2 to 0 of CSIC11
PM11×: Port mode register
P11×: Port output latch

(2) Communication operation

In the 3-wire serial I/O mode, data is transmitted or received in 8-bit units. Each bit of the data is transmitted or received in synchronization with the serial clock.

Data can be transmitted or received if bit 6 (TRMD1n) of serial operation mode register 1n (CSIM1n) is 1. Transmission/reception is started when a value is written to transmit buffer register 1n (SOTB1n). In addition, data can be received when bit 6 (TRMD1n) of serial operation mode register 1n (CSIM1n) is 0.

Reception is started when data is read from serial I/O shift register 1n (SIO1n).

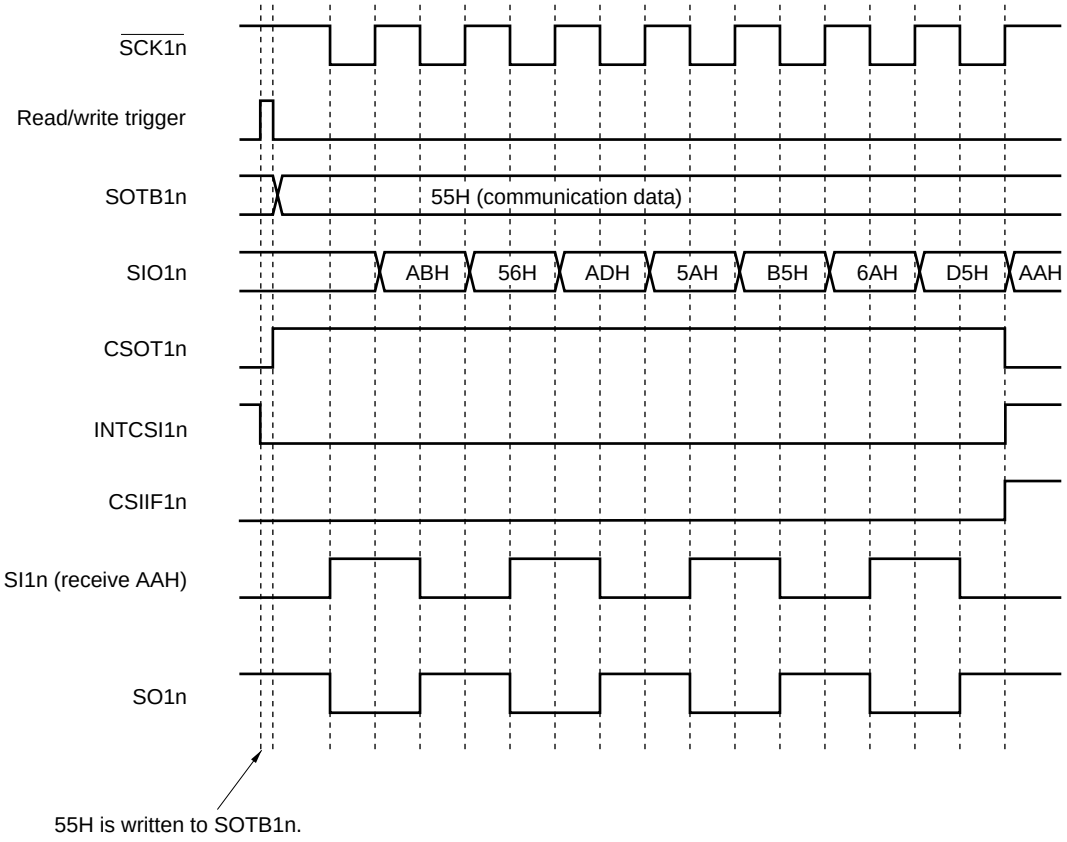
After communication has been started, bit 0 (CSOT1n) of CSIM1n is set to 1. When communication of 8-bit data has been completed, a communication completion interrupt request flag (CSIIF1n) is set, and CSOT1n is cleared to 0. Then the next communication is enabled.

Caution: Do not access the control register and data register when CSOT1n = 1 (during serial communication).

Remark: n = 0, 1

Figure 13-8: Timing in 3-Wire Serial I/O Mode (1/2)

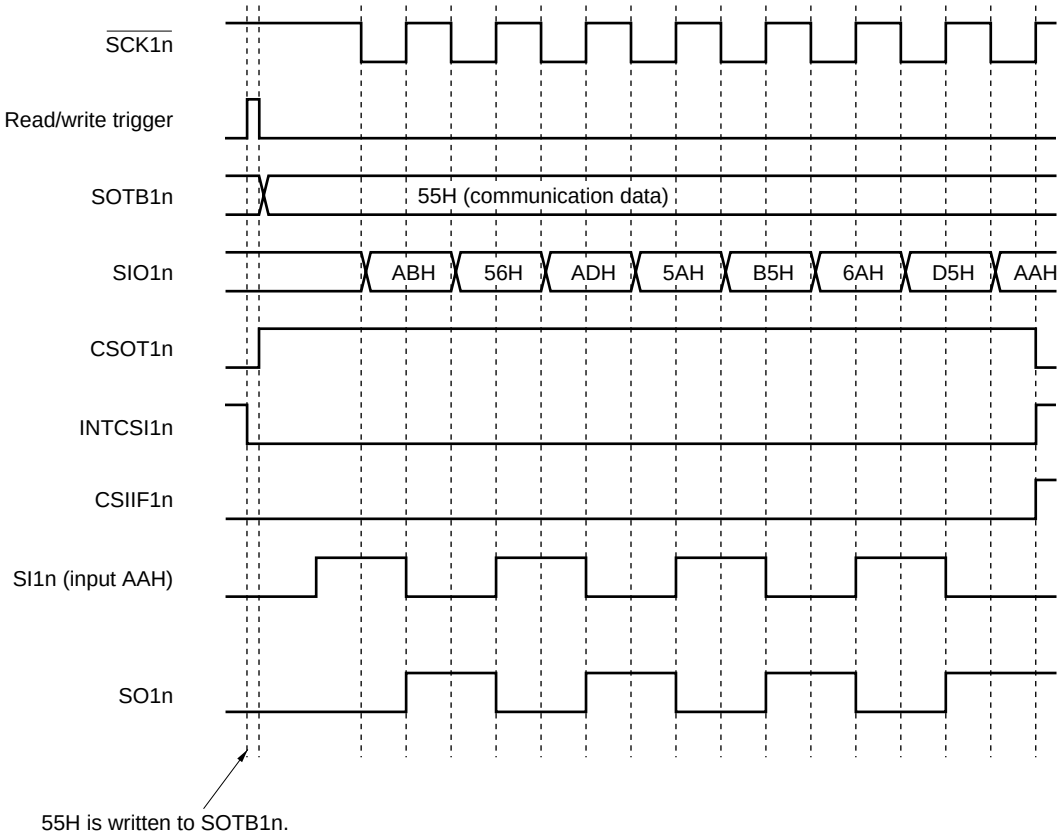
(a) Transmission/reception timing (Type 1; TRMD1n = 1, DIR1n = 0, CKP1n = 0, DAP1n = 0)



Remark: n = 0, 1

Figure 13-8: Timing in 3-Wire Serial I/O Mode (2/2)

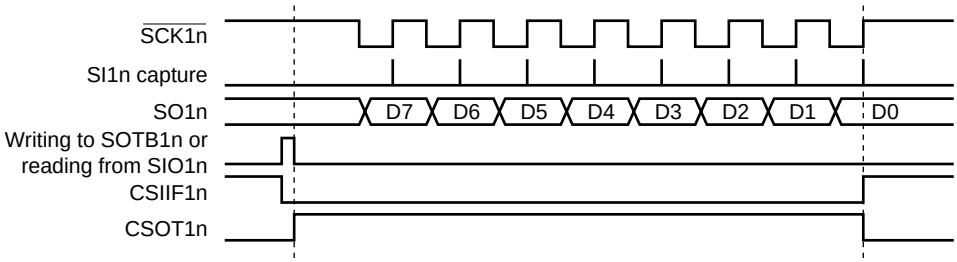
(b) Transmission/reception timing (Type 2; TRMD1n = 1, DIR1n = 0, CKP1n = 0, DAP1n = 1)



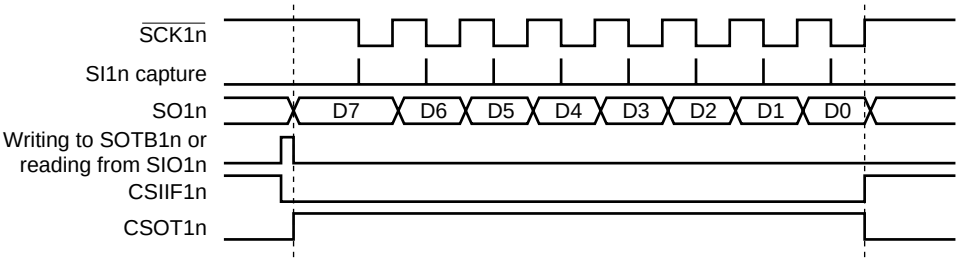
Remark: n = 0, 1

Figure 13-9: Timing of Clock/Data Phase

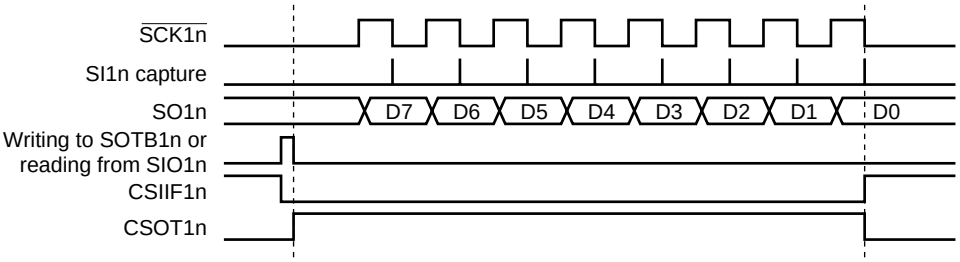
(a) Type 1; CKP1n = 0, DAP1n = 0



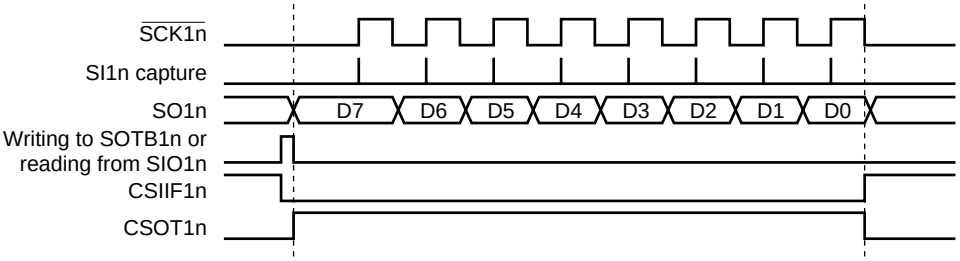
(b) Type 2; CKP1n = 0, DAP1n = 1



(c) Type 3; CKP1n = 1, DAP1n = 0



(d) Type 4; CKP1n = 1, DAP1n = 1



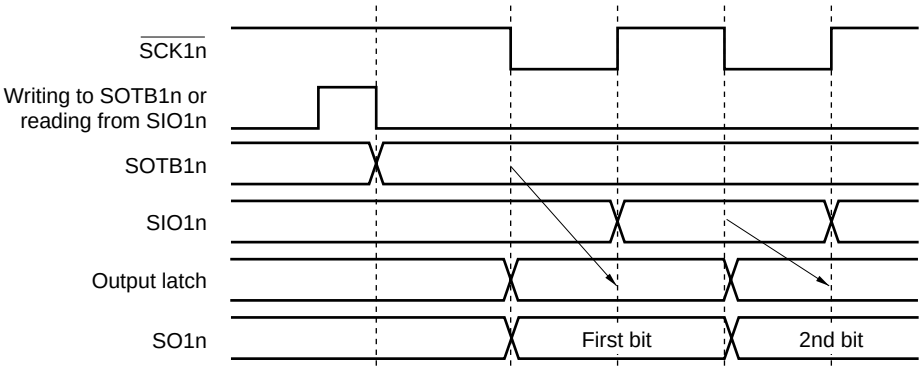
Remark: n = 0, 1

(3) Timing of output to SO1n pin (first bit)

When communication is started, the value of transmit buffer register 1n (SOTB1n) is output from the SO1n pin. The output operation of the first bit at this time is described below.

Figure 13-10: Output Operation of First Bit

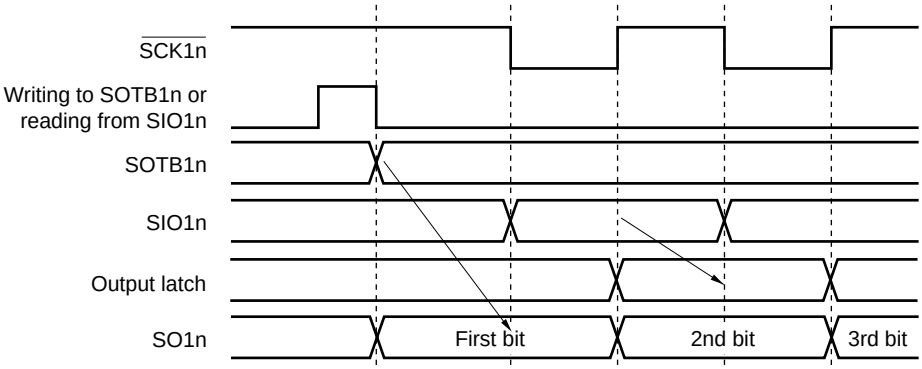
(a) When CKP1n = 0, DAP1n = 0 (or CKP1n = 1, DAP1n = 0)



The first bit is directly latched by the SOTB1n register to the output latch at the falling (or rising) edge of $\overline{SCK1n}$, and output from the SO1n pin via an output selector. Then, the value of the SOTB1n register is transferred to the SIO1n register at the next rising (or falling) edge of $\overline{SCK1n}$, and shifted one bit. At the same time, the first bit of the receive data is stored in the SIO1n register via the SI1n pin.

The second and subsequent bits are latched by the SIO1n register to the output latch at the next falling (or rising) edge of $\overline{SCK1n}$, and the data is output from the SO1n pin.

(b) When CKP1n = 0, DAP1n = 1 (or CKP1n = 1, DAP1n = 1)



The first bit is directly latched by the SOTB1n register at the falling edge of the write signal of the SOTB1n register or the read signal of the SIO1n register, and output from the SO1n pin via an output selector. Then, the value of the SOTB1n register is transferred to the SIO1n register at the next falling (or rising) edge of $\overline{SCK1n}$, and shifted one bit. At the same time, the first bit of the receive data is stored in the SIO1n register via the SI1n pin.

The second and subsequent bits are latched by the SIO1n register to the output latch at the next rising (or falling) edge of $\overline{SCK1n}$, and the data is output from the SO1n pin.

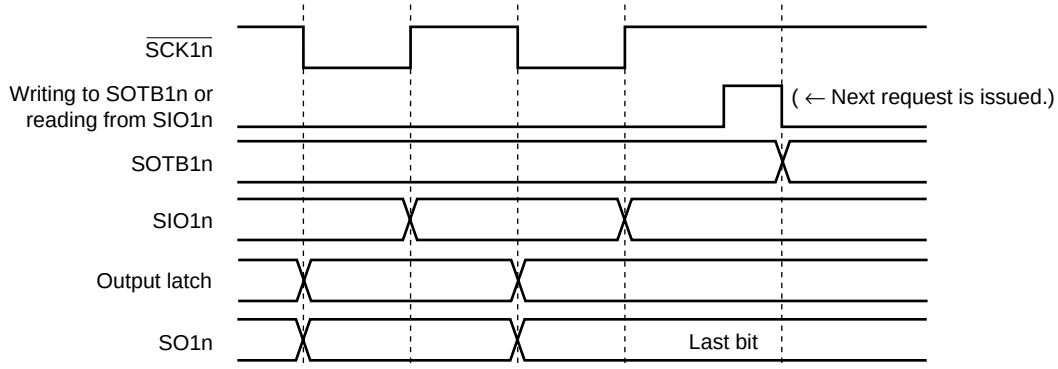
Remark: n = 0, 1

(4) Output value of SO1n pin (last bit)

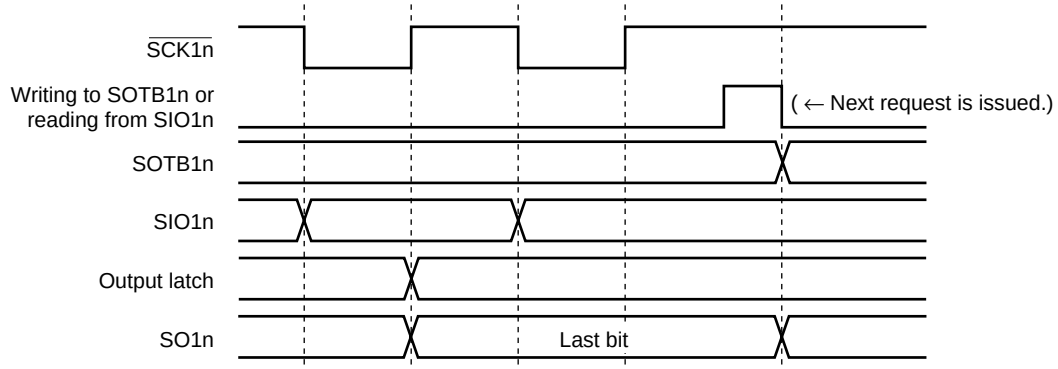
After communication has been completed, the SO1n pin holds the output value of the last bit.

Figure 13-11: Output Value of SO1n Pin (Last Bit)

(a) Type 1; when CKP1n = 0 and DAP1n = 0 (or CKP1n = 1, DAP1n = 0)



(b) Type 2; when CKP1n = 0 and DAP1n = 1 (or CKP1n = 1, DAP1n = 1)



Remark: n = 0, 1

(5) SO1n output

The status of the SO1n output is as follows if bit 7 (CSIE1n) of serial operation mode register 1n (CSIM1n) is cleared to 0.

Table 13-4: SO1n Output Status

TRMD1n	DAP1n	DIR1n	SO1n Output
TRMD1n = 0[Note]	-	-	Outputs low level[Note].
TRMD1n = 1	DAP1n = 0	-	Value of SO1n latch (low-level output)
	DAP1n = 1	DIR1n = 0	Value of bit 7 of SOTB1n
		DIR1n = 1	Value of bit 0 of SOTB1n

Note: Status after reset

Caution: If a value is written to TRMD1n, DAP1n, and DIR1n, the output value of SO1n changes.

Remark: n = 0, 1

Chapter 14 Serial Interface UART6 (µPD780822B Subseries)

14.1 Functions of Serial Interface UART6

Serial interface UART6 has the following two modes.

(1) Operation stop mode

This mode is used when serial communication is not executed and can enable a reduction in the power consumption.
For details, see **14.4.1 "Operation stop mode" on page 331**.

(2) Asynchronous serial interface (UART) mode

This mode supports the LIN (Local Interconnect Network)-bus. The functions of this mode are outlined below.
For details, see **14.4.2 "Asynchronous serial interface (UART) mode" on page 332** and **14.4.3 Dedicated baud rate generator**.

- Two-pin configuration TXD6: Transmit data output pin
  RXB6: Receive data input pin
- Data length of communication data can be selected from 7 or 8 bits.
- Dedicated internal 8-bit baud rate generator allowing any baud rate to be set
- Transmission and reception can be performed independently.
- Twelve operating clock inputs selectable
- MSB- or LSB-first communication selectable
- Inverted transmission operation
- Synchronous break field transmission from 13 to 20 bits
- More than 11 bits can be identified for synchronous break field reception (SBF reception flag provided).

Cautions: 1. TXD6 output inversion function inverts only the transmission side and not the reception side. To use this function, the reception side must be ready for reception of inverted data.

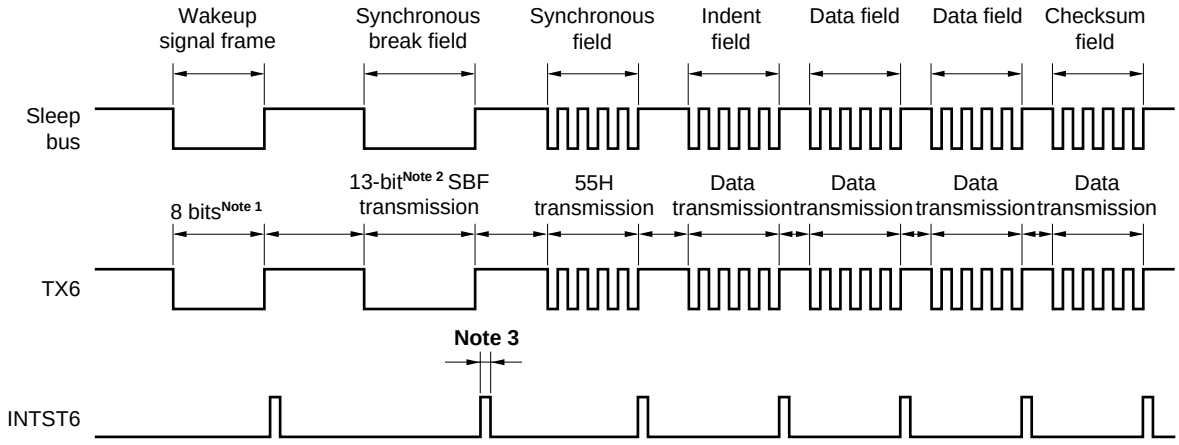
2. If clock supply to serial interface UART6 is not stopped (e.g., in the HALT mode), normal operation continues. If clock supply to serial interface UART6 is stopped (e.g., in the STOP mode), each register stops operating, and holds the value immediately before clock supply was stopped. The TXD6 pin also holds the value immediately before clock supply was stopped and outputs it. However, the operation is not guaranteed after clock supply is resumed. Therefore, reset the circuit so that POWER6 = 0, RXE6 = 0, and TXE6 = 0.

3. If data is continuously transmitted, the communication timing from the stop bit to the next start bit is extended two operating clocks of the macro. However, this does not affect the result of communication because the reception side initializes the timing when it has detected a start bit. Do not use the continuous transmission function if the interface is incorporated in LIN.

Remark: LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol intended to aid the cost reduction of an automotive network.
LIN communication is single-master communication, and up to 15 slaves can be connected to one master.
The LIN slaves are used to control the switches, actuators, and sensors, and these are connected to the LIN master via the LIN network.
Normally, the LIN master is connected to a network such as CAN (Controller Area Network).
In addition, the LIN bus uses a single-wire method and is connected to the nodes via a transceiver that complies with ISO9141.
In the LIN protocol, the master transmits a frame with baud rate information and the slave receives it and corrects the baud rate error. Therefore, communication is possible when the baud rate error in the slave is ±15% or less.

Figures 14-1 and 14-2 outline the transmission and reception operations of LIN.

Figure 14-1: LIN Transmission Operation



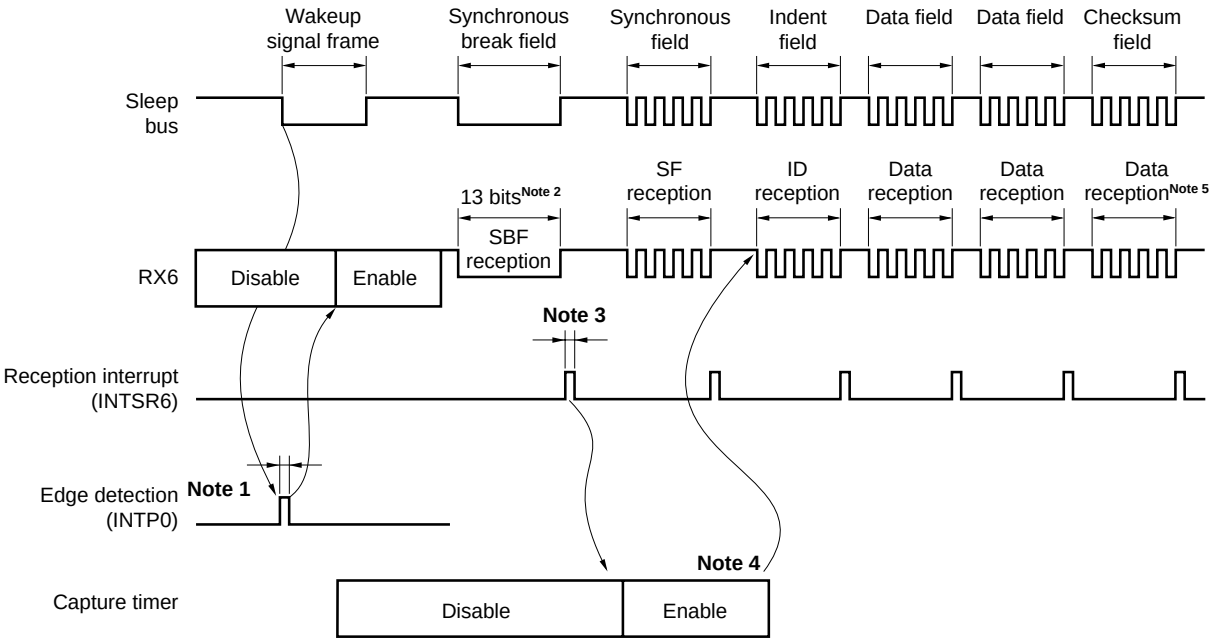
Notes: 1. The wakeup signal frame is substituted by 80H transmission in the 8-bit mode.

2. The synchronous break field is output by hardware. The output width is adjusted by baud rate generator control register 6 (BRGC6) (see **14.4.2 (h) SBF transmission**).

3. INTST6 is output on completion of each transmission. It is also output when SBF is transmitted.

Remark: The interval between each field is controlled by software.

Figure 14-2: LIN Reception Operation



Notes: 1. The wakeup signal is detected at the edge of the pin, and enables UART6 and sets the SBF reception mode.

2. Reception continues until the STOP bit is detected. When an SBF with low-level data of 11 bits or more has been detected, it is assumed that SBF reception has been completed correctly, and an interrupt signal is output. If an SBF with low-level data of less than 11 bits has been detected, it is assumed that an SBF reception error has occurred. The interrupt signal is not output and the SBF reception mode is restored.

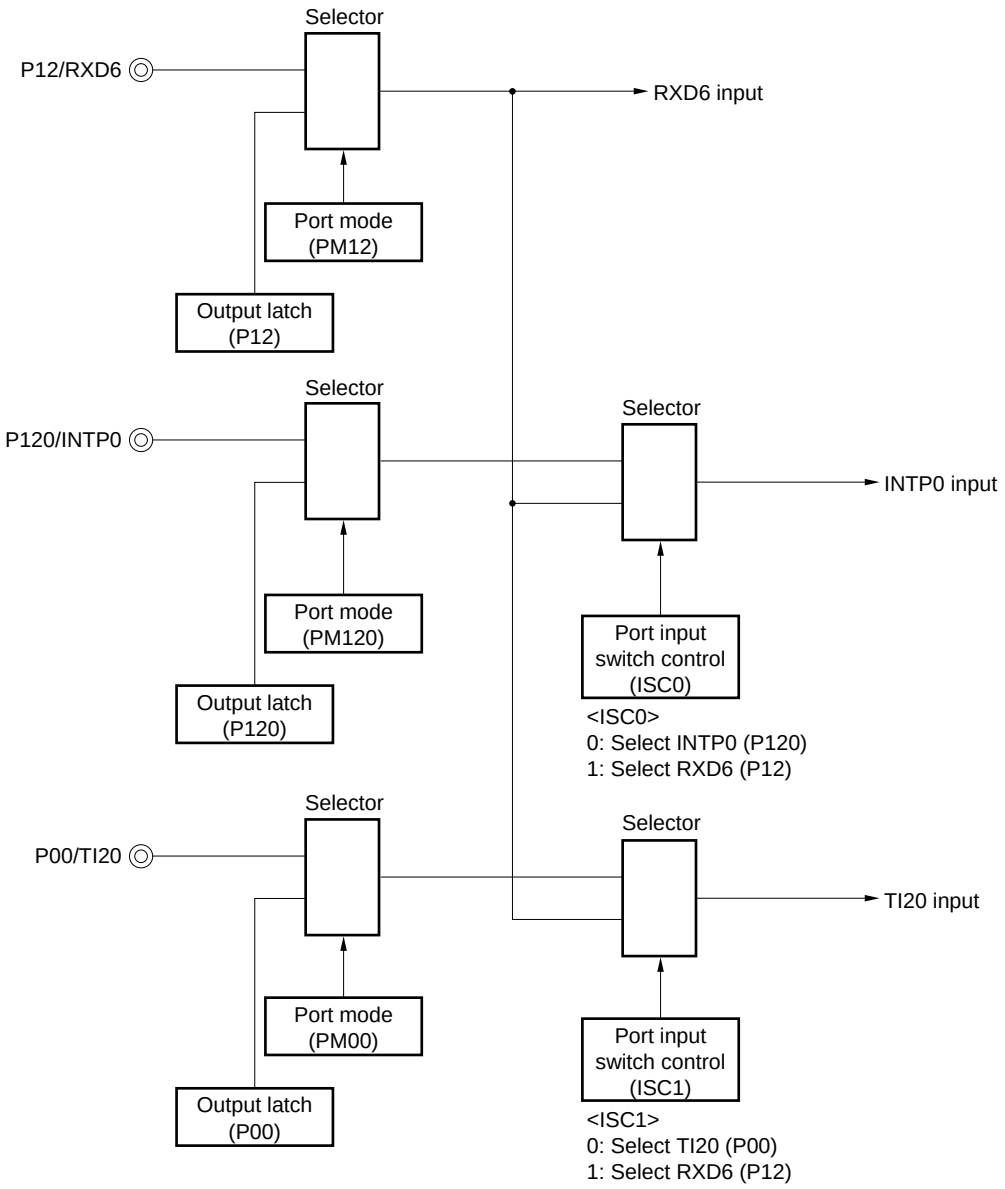
3. If SBF reception has been completed correctly, an interrupt signal is output. This SBF reception completion interrupt enables the capture timer. Detection of errors OVE6, PE6, and FE6 is suppressed, and error detection processing of UART communication and data transfer of the shift register and RXB6 is not performed. The shift register holds the reset value FFH.

4. Calculate the baud rate error from the bit length of the synchronous field, disable UART6 after SF reception, and then re-set baud rate generator control register 6 (BRGC6).

5. Distinguish the checksum field by software. Also perform processing by software to initialize UART6 after reception of the checksum field and to set the SBF reception mode again.

To perform a LIN receive operation, use a configuration like the one shown in Figure 14-3.
The wakeup signal transmitted from the LIN master is received by detecting the edge of the external interrupt (INTP0). The length of the synchronous field transmitted from the LIN master can be measured using the external event capture operation of 16-bit timer/event counter 00, and the baud rate error can be calculated.
The input signal of the reception port input (RXD6) can be input to the external interrupt (INTP0) and 16-bit timer/event counter 20 by port input switch control (ISC0/ISC1), without connecting RXD6 and INTP0/TI000 externally.

Figure 14-3: Port Configuration for LIN Reception Operation



Remark: ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (see **Figure 14-11**)

The peripheral functions used in the LIN communication operation are shown below.
<Peripheral functions used>

- External interrupt (INTP0); wakeup signal detection
  Use: Detects the wakeup signal edges and detects start of communication.
- 16-bit timer/event counter 20 (TI20); baud rate error detection
  Use: Detects the baud rate error (measures the TI20 input edge interval in the capture mode) by detecting the synchronous break field (SBF) length and divides it by the number of bits.
- Serial interface UART6

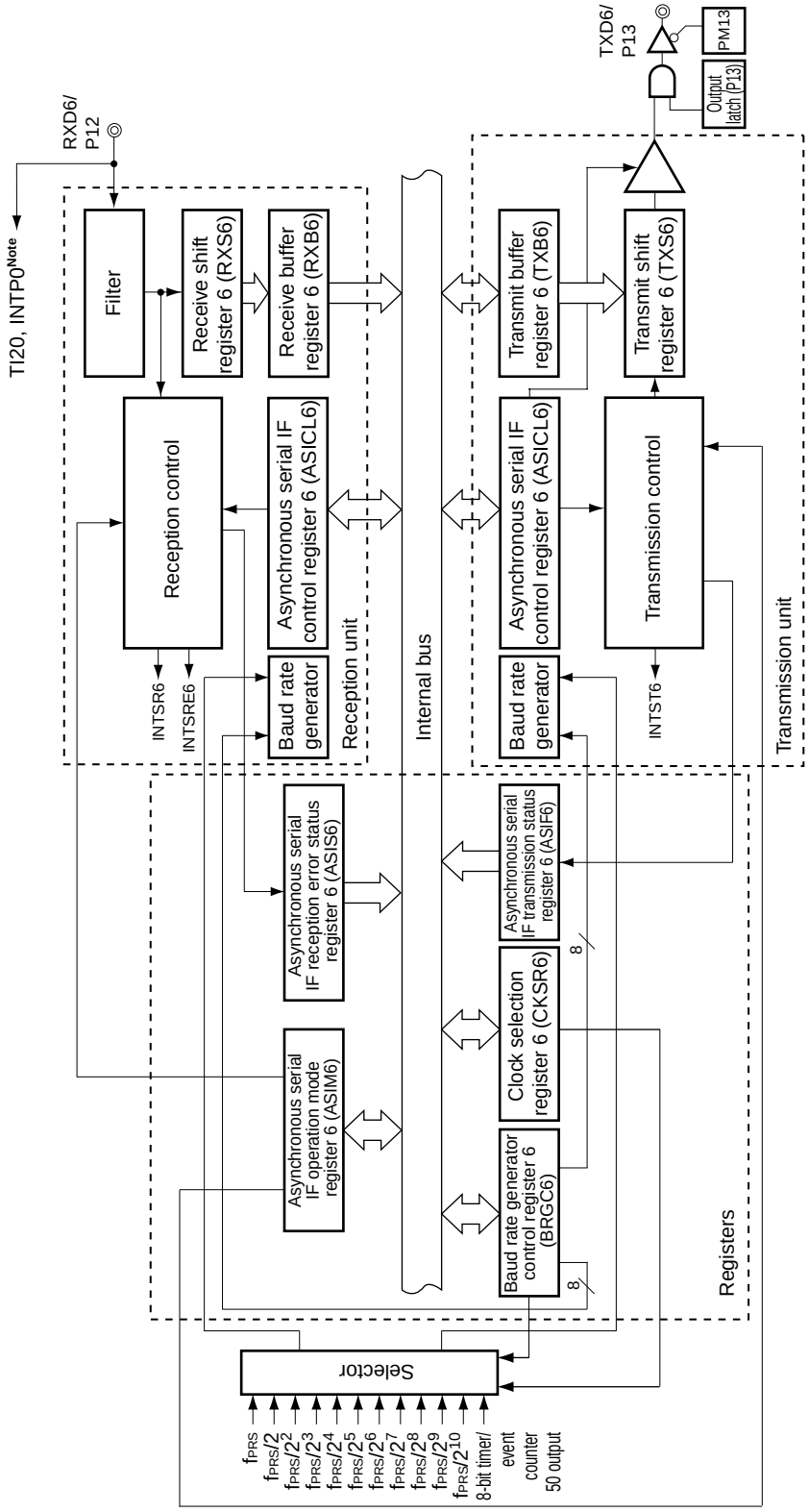
14.2 Configuration of Serial Interface UART6

Serial interface UART6 consists of the following hardware.

Table 14-1: Configuration of Serial Interface UART6

Item	Configuration
Registers	Receive buffer register 6 (RXB6) Receive shift register 6 (RXS6) Transmit buffer register 6 (TXB6) Transmit shift register 6 (TXS6)
Control registers	Asynchronous serial interface operation mode register 6 (ASIM6) Asynchronous serial interface reception error status register 6 (ASIS6) Asynchronous serial interface transmission status register 6 (ASIF6) Clock selection register 6 (CKSR6) Baud rate generator control register 6 (BRGC6) Asynchronous serial interface control register 6 (ASICL6) Input switch control register (ISC) Port mode register 1 (PM1) Port register 1 (P1)

Figure 14-4: Block Diagram of Serial Interface UART6



Note: Selectable with input switch control register (ISC)

Remark: f_{PRS}: Peripheral hardware clock oscillation

(1) Receive buffer register 6 (RXB6)

This 8-bit register stores parallel data converted by receive shift register 6 (RXS6).
Each time 1 byte of data has been received, new receive data is transferred to this register from receive shift register 6 (RXS6). If the data length is set to 7 bits, data is transferred as follows.

- In LSB-first reception, the receive data is transferred to bits 0 to 6 of RXB6 and the MSB of RXB6 is always 0.
- In MSB-first reception, the receive data is transferred to bits 1 to 7 of RXB6 and the LSB of RXB6 is always 0.

If an overrun error (OVE6) occurs, the receive data is not transferred to RXB6.
RXB6 can be read by an 8-bit memory manipulation instruction. No data can be written to this register.
$\overline{\text{RESET}}$ input sets this register to FFH.

(2) Receive shift register 6 (RXS6)

This register converts the serial data input to the RXD6 pin into parallel data.
RXS6 cannot be directly manipulated by a program.

(3) Transmit buffer register 6 (TXB6)

This buffer register is used to set transmit data. Transmission is started when data is written to TXB6. This register can be read or written by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to FFH.

Cautions: 1. Do not write data to TXB6 when bit 1 (TXBF6) of asynchronous serial interface transmission status register 6 (ASIF6) is 1.

2. Do not refresh (write the same value to) TXB6 by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of asynchronous serial interface operation mode register 6 (ASIM6) are 1 or when bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 are 1).

(4) Transmit shift register 6 (TXS6)

This register transmits the data transferred from TXB6 from the TXD6 pin as serial data. Data is transferred from TXB6 immediately after TXB6 is written for the first transmission, or immediately before INTST6 occurs after one frame was transmitted for continuous transmission. Data is transferred from TXB6 and transmitted from the TXD6 pin at the falling edge of the base clock.
TXS6 cannot be directly manipulated by a program.

14.3 Registers Controlling Serial Interface UART6

Serial interface UART6 is controlled by the following nine registers.

- Asynchronous serial interface operation mode register 6 (ASIM6)
- Asynchronous serial interface reception error status register 6 (ASIS6)
- Asynchronous serial interface transmission status register 6 (ASIF6)
- Clock selection register 6 (CKSR6)
- Baud rate generator control register 6 (BRGC6)
- Asynchronous serial interface control register 6 (ASICL6)
- Input switch control register (ISC)
- Port mode register 1 (PM1)
- Port register 1 (P1)

14.3.1 Asynchronous serial interface operation mode register 6 (ASIM6)

This 8-bit register controls the serial communication operations of serial interface UART6.
This register can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to 01H.

Remark: ASIM6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 1).

Figure 14-5: Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) Format(1/2)

	<7>	<6>	<5>	4	3	2	1	0	R/W	Address	After Reset
ASIM6	POWER6	TXE6	RXE6	PS61	PS60	CL6	SL6	ISRM6	R/W	FF50H	01H

POWER6	Enables/disables operation of internal operation clock
0**Note 1**	Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit**Note 2**
1**Note 3**	Enables operation of the internal operation clock

TXE6	Enables/disables transmission
0	Disables transmission (synchronously resets the transmission circuit)
1	Enables transmission

Notes: 1. The output of the TXD6 pin goes high and the input from the RXD6 pin is fixed to the high level when POWER6 = 0.

2. Asynchronous serial interface reception error status register 6 (ASIS6), asynchronous serial interface transmission status register 6 (ASIF6), bit 7 (SBRF6) and bit 6 (SBRT6) of asynchronous serial interface control register 6 (ASICL6), and receive buffer register 6 (RXB6) are reset.

3. Operation of the 8-bit counter output is enabled at the second base clock after 1 is written to the POWER6 bit

Figure 14-5: Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) Format(2/2)

RXE6	Enables/disables reception
0	Disables reception (synchronously resets the reception circuit)
1	Enables reception

PS61	PS60	Transmission operation	Reception operation
0	0	Does not output parity bit	Reception without parity
0	1	Outputs 0 parity	Reception as 0 parity[Note]
1	0	Outputs odd parity	Judges as odd parity
1	1	Outputs even parity	Judges as even parity

CL6	Specifies character length of transmit/receive data
0	Character length of data = 7 bits
1	Character length of data = 8 bits

SL6	Specifies number of stop bits of transmit data
0	Number of stop bits = 1
1	Number of stop bits = 2

ISRM6	Enables/disables occurrence of reception completion interrupt in case of error
0	“INTSRE6” occurs in case of error (at this time, INTSR6 does not occur)
1	“INTSR6” occurs in case of error (at this time, INTSRE6 does not occur)

Note: If “reception as 0 parity” is selected, the parity is not judged. Therefore, bit 2 (PE6) of asynchronous serial interface reception error status register 6 (ASIS6) is not set and the error interrupt does not occur.

Cautions:

1. **At startup, set POWER6 to 1 and then set TXE6 to 1. To stop the operation, clear TXE6 to 0, and then clear POWER6 to 0.**
2. **At startup, set POWER6 to 1 and then set RXE6 to 1. To stop the operation, clear RXE6 to 0, and then clear POWER6 to 0.**
3. **Set POWER6 to 1 and then set RXE6 to 1 while a high level is input to the RXD6 pin. If POWER6 is set to 1 and RXE6 is set to 1 while a low level is input, reception is started.**
4. **Clear the TXE6 and RXE6 bits to 0 before rewriting the PS61, PS60, and CL6 bits.**
5. **Fix the PS61 and PS60 bits to 0 when mounting the device on LIN.**
6. **Make sure that TXE6 = 0 when rewriting the SL6 bit. Reception is always performed with “the number of stop bits = 1”, and therefore, is not affected by the set value of the SL6 bit.**
7. **Make sure that RXE6 = 0 when rewriting the ISRM6 bit.**

14.3.2 Asynchronous serial interface reception error status register 6 (ASIS6)

This register indicates an error status on completion of reception by serial interface UART6. It includes three error flag bits (PE6, FE6, OVE6).
This register is read-only by an 8-bit memory manipulation instruction.
RESET input clears this register to 00H if bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 0. 00H is read when this register is read.

Figure 14-6: Asynchronous Serial Interface Reception Error Status Register 6 (ASIS6) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ASIS6	0	0	0	0	0	PE6	FE6	OVE6	R	FF53H	00H

PE6	Status flag indicating parity error
0	If POWER6 = 0 and RXE6 = 0, or if ASIS6 register is read
1	If the parity of transmit data does not match the parity bit on completion of reception

FE6	Status flag indicating framing error
0	If POWER6 = 0 and RXE6 = 0, or if ASIS6 register is read
1	If the stop bit is not detected on completion of reception

OVE6	Status flag indicating overrun error
0	If POWER6 = 0 and RXE6 = 0, or if ASIS6 register is read
1	If receive data is set to the RXB register and the next reception operation is completed before the data is read

Cautions: 1. The operation of the PE6 bit differs depending on the set values of the PS61 and PS60 bits of asynchronous serial interface operation mode register 6 (ASIM6).

2. The first bit of the receive data is checked as the stop bit, regardless of the number of stop bits.

3. If an overrun error occurs, the next receive data is not written to receive buffer register 6 (RXB6) but discarded.

4. If data is read from ASIS6, a wait cycle is generated. Do not read data from ASIS6 when the CPU is operating on the subsystem clock and the X1 input clock is stopped. For details, see Chapter 29 "Cautions For Wait (µPD780822B Subseries)" on page 591.

14.3.3 Asynchronous serial interface transmission status register 6 (ASIF6)

This register indicates the status of transmission by serial interface UART6. It includes two status flag bits (TXBF6 and TXSF6).
Transmission can be continued without disruption even during an interrupt period, by writing the next data to the TXB6 register after data has been transferred from the TXB6 register to the TXS6 register.
This register is read-only by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H if bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 0. 00H is read when this register is read.

Figure 14-7: Asynchronous Serial Interface Transmission Status Register 6 (ASIF6) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ASIF6	0	0	0	0	0	0	TXBF6	TXSF6	R	FF55H	00H

TXBF6	Transmit buffer data flag
0	If POWER6 = 0 or TXE6 = 0, or if data is transferred to transmit shift register 6 (TXS6)
1	If data is written to transmit buffer register 6 (TXB6) (if data exists in TXB6)

TXSF6	Transmit shift register data flag
0	If POWER6 = 0 or TXE6 = 0, or if the next data is not transferred from transmit buffer register 6 (TXB6) after completion of transfer
1	If data is transferred from transmit buffer register 6 (TXB6) (if data transmission is in progress)

Cautions: 1. To transmit data continuously, write the first transmit data (first byte) to the TXB6 register. Be sure to check that the TXBF6 flag is “0”. If so, write the next transmit data (second byte) to the TXB6 register. If data is written to the TXB6 register while the TXBF6 flag is “1”, the transmit data cannot be guaranteed.

2. To initialize the transmission unit upon completion of continuous transmission, be sure to check that the TXSF6 flag is “0” after generation of the transmission completion interrupt, and then execute initialization. If initialization is executed while the TXSF6 flag is “1”, the transmit data cannot be guaranteed.

14.3.4 Clock selection register 6 (CKSR6)

This register selects the base clock of serial interface UART6.
CKSR6 can be set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Remark: CKSR6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 1).

Figure 14-8: Clock Selection Register 6 (CKSR6) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
CKSR6	0	0	0	0	TPS63	TPS62	TPS61	TPS60	R/W	FF56H	00H

TPS63	TPS62	TPS61	TPS60	Base clock (f_{XCLK6}) selection
0	0	0	0	f_{PRS} (16 MHz)
0	0	0	1	$f_{PRS}/2$ (8 MHz)
0	0	1	0	$f_{PRS}/2^2$ (4 MHz)
0	0	1	1	$f_{PRS}/2^3$ (2 MHz)
0	1	0	0	$f_{PRS}/2^4$ (1 MHz)
0	1	0	1	$f_{PRS}/2^5$ (500 kHz)
0	1	1	0	$f_{PRS}/2^6$ (250 kHz)
0	1	1	1	$f_{PRS}/2^7$ (125 kHz)
1	0	0	0	$f_{PRS}/2^8$ (62.5 kHz)
1	0	0	1	$f_{PRS}/2^9$ (31.25 kHz)
1	0	1	0	$f_{PRS}/2^{10}$ (15.63 kHz)
1	0	1	1	TM50 output**Note**
Other than above				Setting prohibited

Note: To select the output of TM50 as the base clock, start the operation by setting 8-bit timer/event counter 50 so that the duty is 50% of the output in the PWM mode (bit 6 (TMC506) of the TMC50 register = 1), and then set TPS63, TPS62, TPS61, and TPS60 to 1, 0, 1, and 1, respectively. It is not necessary to enable the TO50 pin as a timer output pin (bit 0 (TOE50) of the TMC register may be 0 or 1).

Cautions: 1. When the internal low-speed oscillator clock is selected as the clock to be supplied to the CPU, the clock of the internal low-speed oscillator is divided and supplied as the count clock. If the base clock is the internal low-speed oscillator clock, the operation of serial interface UART6 is not guaranteed.

2. Make sure POWER6 = 0 when rewriting TPS63 to TPS60.

Remarks: 1. Figures in parentheses are for operation with $f_{PRS} = f_X = 16$ MHz

2. f_{PRS}: Peripheral hardware clock oscillation frequency

14.3.5 Baud rate generator control register 6 (BRGC6)

This register sets the division value of the 8-bit counter of serial interface UART6.
BRGC6 can be set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to FFH.

Remark: BRGC6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 1).

Figure 14-9: Baud Rate Generator Control Register 6 (BRGC6) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
BRGC6	MDL67	MDL66	MDL65	MDL64	MDL63	MDL62	MDL61	MDL60	R/W	FF57H	FFH

MDL67	MDL66	MDL65	MDL64	MDL63	MDL62	MDL61	MDL60	k	Output clock selection of 8-bit counter
0	0	0	0	0	×	×	×	×	Setting prohibited
0	0	0	0	1	0	0	0	8	$f_{XCLK6}/8$
0	0	0	0	1	0	0	1	9	$f_{XCLK6}/9$
0	0	0	0	1	0	1	0	10	$f_{XCLK6}/10$
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	0	252	$f_{XCLK6}/252$
1	1	1	1	1	1	0	1	253	$f_{XCLK6}/253$
1	1	1	1	1	1	1	0	254	$f_{XCLK6}/254$
1	1	1	1	1	1	1	1	255	$f_{XCLK6}/255$

Cautions: 1. Make sure that bit 6 (TXE6) and bit 5 (RXE6) of the ASIM6 register = 0 when rewriting the MDL67 to MDL60 bits.

2. The baud rate is the output clock of the 8-bit counter divided by 2.

Remarks: 1. f_{XCLK6}: Frequency of base clock selected by the TPS63 to TPS60 bits of CKSR6 register

2. k: Value set by MDL67 to MDL60 bits (k = 8, 9, 10,..., 255)

3. ×: Don't care

14.3.6 Asynchronous serial interface control register 6 (ASICL6)

This register controls the serial communication operations of serial interface UART6.
ASICL6 can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to 16H.

Caution: ASICL6 can be refreshed (the same value is written) by software during a communication operation (when bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1 or bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 1). Note, however, that communication is started by the refresh operation because bit 6 (SBRT6) of ASICL6 is cleared to 0 when communication is completed (when an interrupt signal is generated).

Figure 14-10: Asynchronous Serial Interface Control Register 6 (ASICL6) Format (1/2)

	<7>	<6>	5	4	3	2	1	0	R/W	Address	After Reset
ASICL6	SBRF6	SBRT6	SBTT6	SBL62	SBL61	SBL60	DIR6	TXDLV6	R/W **Note**	FF58H	16H

SBRF6	SBF reception status flag
0	If POWER6 = 0 and RXE6 = 0 or if SBF reception has been completed correctly
1	SBF reception in progress

SBRT6	SBF reception trigger
0	-
1	SBF reception trigger

SBTT6	SBF transmission trigger
0	-
1	SBF transmission trigger

SBL62	SBL61	SBL60	SBF transmission output width control
1	0	1	SBF is output with 13-bit length.
1	1	0	SBF is output with 14-bit length.
1	1	1	SBF is output with 15-bit length.
0	0	0	SBF is output with 16-bit length.
0	0	1	SBF is output with 17-bit length.
0	1	0	SBF is output with 18-bit length.
0	1	1	SBF is output with 19-bit length.
1	0	0	SBF is output with 20-bit length.

Note: Bits 2 to 5 and 7 are read-only.

Figure 14-10: Asynchronous Serial Interface Control Register 6 (ASICL6) Format (2/2)

DIR6	First bit specification
0	MSB
1	LSB

TXDLV6	Enables/disables inverting TXD6 output
0	Normal output of TXD6
1	Inverted output of TXD6

Cautions:

1. **In the case of an SBF reception error, return the mode to the SBF reception mode and hold the status of the SBRF6 flag.**
2. **Before setting the SBRT6 bit, make sure that bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 1.**
3. **The read value of the SBRT6 bit is always 0. SBRT6 is automatically cleared to 0 after SBF reception has been correctly completed.**
4. **Before setting the SBTT6 bit to 1, make sure that bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1.**
5. **The read value of the SBTT6 bit is always 0. SBTT6 is automatically cleared to 0 at the end of SBF transmission.**
6. **Before rewriting the DIR6 and TXDLV6 bits, clear the TXE6 and RXE6 bits to 0.**

14.3.7 Input switch control register (ISC)

The input switch control register (ISC) is used to receive a status signal transmitted from the master during LIN (Local Interconnect Network) reception. The input signal is switched by setting ISC.
This register can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 14-11: Input Switch Control Register (ISC) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ISC	0	CKSWD	CKSWT1	CKSWT0	CKUCKS	CKCL	ISC1	ISC0	R/W	FF4AH	00H

ISC1	TI20 input source selection
0	TI20 (P00)
1	RXD6 (P12)

ISC0	INTP0 input source selection
0	INTP0 (P120)
1	RXD6 (P12)

Remark: In this part only the ISC1 and ISC0 selection feature is described. The input switch control register has more selection features. Please refer to the **Figure 5-8, "Input Switch Control Register (ISC) Format (1/2)," on page 134**.

14.3.8 Port mode register 1 (PM1)

This register sets port 1 input/output in 1-bit units.
When using the P13/TXD6 pin for serial interface data output, clear PM13 to 0 and set the output latch of P13 to 1.
When using the P12/RXD6 pin for serial interface data input, set PM12 to 1. The output latch of P12 at this time may be 0 or 1.
PM1 can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to FFH.

Figure 14-12: Port Mode Register 1 (PM1) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	R/W	FF21H	FFH

PM1n	P1n pin I/O mode selection (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

14.4 Operation of Serial Interface UART6

Serial interface UART6 has the following two modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode

14.4.1 Operation stop mode

In this mode, serial communication cannot be executed; therefore, the power consumption can be reduced. In addition, the pins can be used as ordinary port pins in this mode. To set the operation stop mode, clear bits 7, 6, and 5 (POWER6, TXE6, and RXE6) of ASIM6 to 0.

(1) Register used

The operation stop mode is set by asynchronous serial interface operation mode register 6 (ASIM6).
ASIM6 can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to 01H.

Figure 14-13: Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) Format

	<7>	<6>	<5>	4	3	2	1	0	R/W	Address	After Reset
ASIM6	POWER6	TXE6	RXE6	PS61	PS60	CL6	SL6	ISRM6	R/W	FF50H	01H

POWER6	Enables/disables operation of internal operation clock
0**Note 1**	Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit**Note 2**

TXE6	Enables/disables transmission
0	Disables transmission (synchronously resets the transmission circuit)

RXE6	Enables/disables reception
0	Disables reception (synchronously resets the reception circuit)

Notes: 1. The output of the TXD6 pin goes high and the input from the RXD6 pin is fixed to the high level when POWER6 = 0.

2. Asynchronous serial interface reception error status register 6 (ASIS6), asynchronous serial interface transmission status register 6 (ASIF6), bit 7 (SBRF6) and bit 6 (SBRT6) of asynchronous serial interface control register 6 (ASICL6), and receive buffer register 6 (RXB6) are reset.

Caution: Clear POWER6 to 0 after clearing TXE6 and RXE6 to 0 to set the operation stop mode. To start the operation, set POWER6 to 1, and then set TXE6 and RXE6 to 1.

Remark: To use the RXD6/P12 and TXD6/P13 pins as general-purpose port pins, see **Chapter 4 "Port Functions (µPD780822B Subseries)" on page 97**.

14.4.2 Asynchronous serial interface (UART) mode

In this mode, data of 1 byte is transmitted/received following a start bit, and a full-duplex operation can be performed.
A dedicated UART baud rate generator is incorporated, so that communication can be executed at a wide range of baud rates.

(1) Registers used:

- Asynchronous serial interface operation mode register 6 (ASIM6)
- Asynchronous serial interface reception error status register 6 (ASIS6)
- Asynchronous serial interface transmission status register 6 (ASIF6)
- Clock selection register 6 (CKSR6)
- Baud rate generator control register 6 (BRGC6)
- Asynchronous serial interface control register 6 (ASICL6)
- Input switch control register (ISC)
- Port mode register 1 (PM1)
- Port register 1 (P1)

The basic procedure of setting an operation in the UART mode is as follows.

<1> Set the CKSR6 register (see **Figure 14-8**).
<2> Set the BRGC6 register (see **Figure 14-9**).
<3> Set bits 0 to 4 (ISRM6, SL6, CL6, PS60, PS61) of the ASIM6 register (see **Figure 14-5**).
<4> Set bits 0 and 1 (TXDLV6, DIR6) of the ASICL6 register (see **Figure 14-10**).
<5> Set bit 7 (POWER6) of the ASIM6 register to 1.
<6> Set bit 6 (TXE6) of the ASIM6 register to 1. → Transmission is enabled.
Set bit 5 (RXE6) of the ASIM6 register to 1. → Reception is enabled.
<7> Write data to transmit buffer register 6 (TXB6). → Data transmission is started.

Caution: Take relationship with the other party of communication when setting the port mode register and port register.

The relationship between the register settings and pins is shown below.

Table 14-2: Relationship Between Register Settings and Pins

POWER6	TXE6	RXE6	PM13	P13	PM12	P12	UART6 Operation	Pin Function	
								TXD6/P13	RXD6/P12
0	0	0	×**Note**	×**Note**	×**Note**	×**Note**	Stop	P13	P12
1	0	1	×**Note**	×**Note**	1	×	Reception	P13	RXD6
	1	0	0	1	×**Note**	×**Note**	Transmission	TXD6	P12
	1	1	0	1	1	×	Transmission/ reception	TXD6	RXD6

Note: Can be set as port function.

Remark: ×: don’t care
POWER6: Bit 7 of asynchronous serial interface operation mode register 6 (ASIM6)
TXE6: Bit 6 of ASIM6
RXE6: Bit 5 of ASIM6
PM1×: Port mode register
P1×: Port output latch

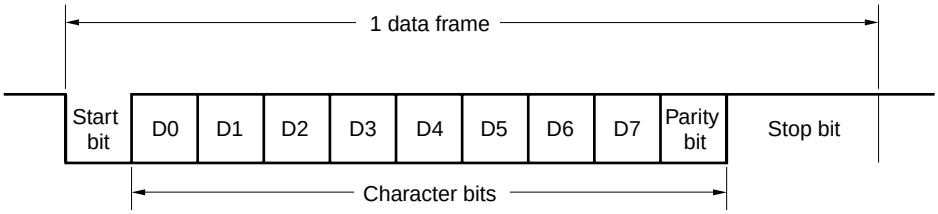
(2) Communication operation

(a) Format and waveform example of normal transmit/receive data

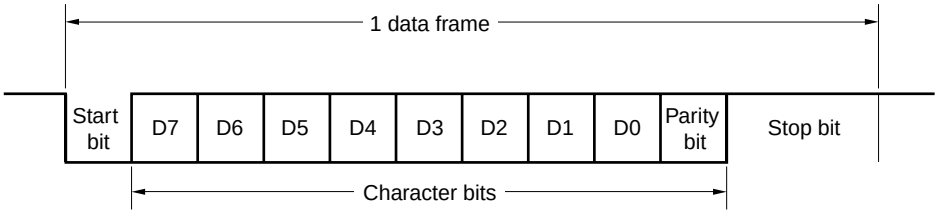
Figures 14-14 and 14-15 show the format and waveform example of the normal transmit/receive data.

Figure 14-14: Format of Normal UART Transmit/Receive Data

(a) LSB-first transmission/reception



(b) MSB-first transmission/reception



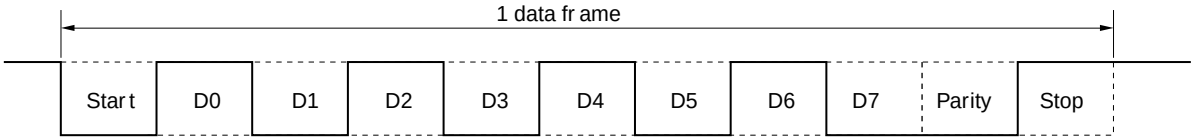
One data frame consists of the following bits.

- Start bit... 1 bit
- Character bits... 7 or 8 bits
- Parity bit... Even parity, odd parity, 0 parity, or no parity
- Stop bit... 1 or 2 bits

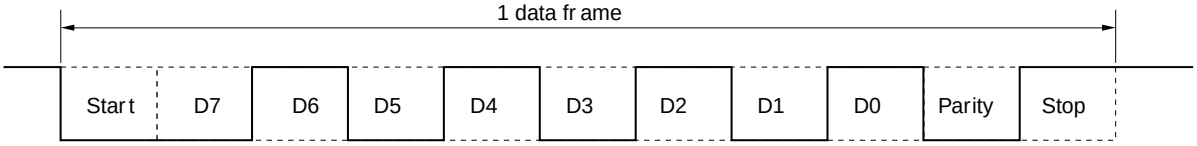
The character bit length, parity, and stop bit length in one data frame are specified by asynchronous serial interface operation mode register 6 (ASIM6).
Whether data is communicated with the LSB or MSB first is specified by bit 1 (DIR6) of asynchronous serial interface control register 6 (ASICL6).
Whether the TXD6 pin outputs normal or inverted data is specified by bit 0 (TXDLV6) of ASICL6.

Figure 14-15: Example of Normal UART Transmit/Receive Data Waveform

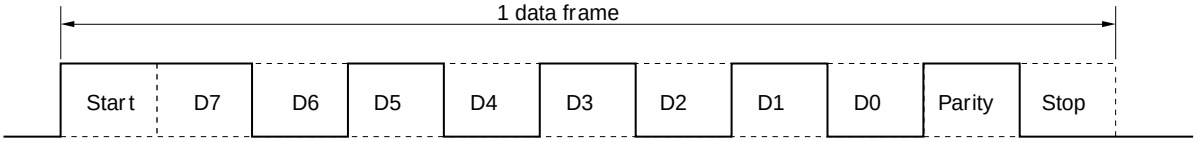
(a) Data length: 8 bits, LSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



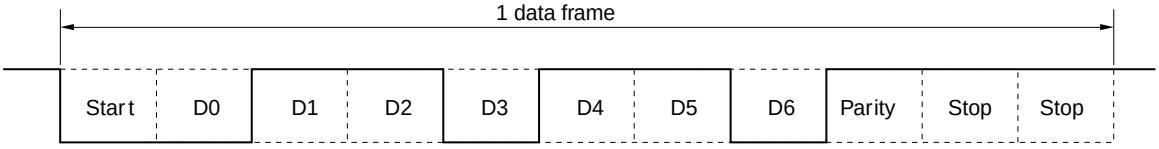
(b) Data length: 8 bits, MSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



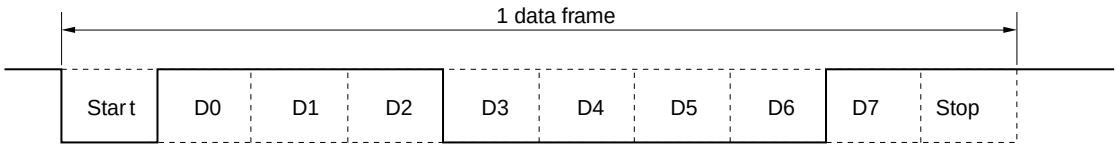
(c) Data length: 8 bits, MSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H, TXD6 pin inverted output



(d) Data length: 7 bits, LSB first, Parity: Odd parity, Stop bit: 2 bits, Communication data: 36H



(e) Data length: 8 bits, LSB first, Parity: None, Stop bit: 1 bit, Communication data: 87H



(b) Parity types and operation

The parity bit is used to detect a bit error in communication data. Usually, the same type of parity bit is used on both the transmission and reception sides. With even parity and odd parity, a 1-bit (odd number) error can be detected. With zero parity and no parity, an error cannot be detected.

Caution: Fix the PS61 and PS60 bits to 0 when the device is incorporated in LIN.

- **Even parity**
  - **Transmission**

    Transmit data, including the parity bit, is controlled so that the number of bits that are “1” is even.
    The value of the parity bit is as follows.

    If transmit data has an odd number of bits that are “1”: 1

    If transmit data has an even number of bits that are “1”: 0

  - **Reception**

    The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is odd, a parity error occurs.

- **Odd parity**
  - **Transmission**

    Unlike even parity, transmit data, including the parity bit, is controlled so that the number of bits that are “1” is odd.

    If transmit data has an odd number of bits that are “1”: 0

    If transmit data has an even number of bits that are “1”: 1

  - **Reception**

    The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is even, a parity error occurs.

- **0 parity**
  The parity bit is cleared to 0 when data is transmitted, regardless of the transmit data.
  The parity bit is not detected when the data is received. Therefore, a parity error does not occur regardless of whether the parity bit is “0” or “1”.

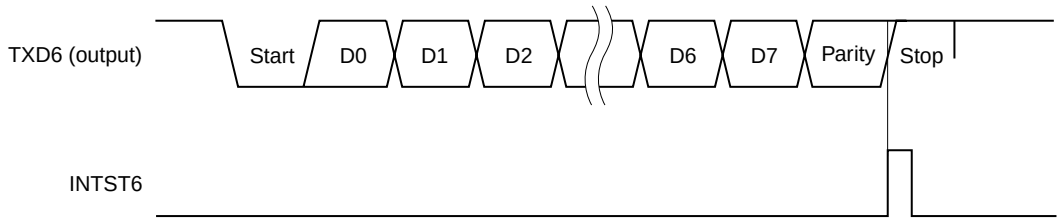
- **No parity**
  No parity bit is appended to the transmit data.
  Reception is performed assuming that there is no parity bit when data is received. Because there is no parity bit, a parity error does not occur.

(c) Normal transmission

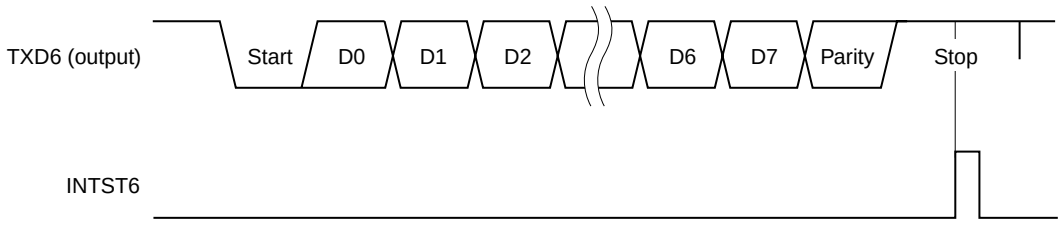
The TXD6 pin outputs a high level when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1. If bit 6 (TXE6) of ASIM6 is then set to 1, transmission is enabled. Transmission can be started by writing transmit data to transmit buffer register 6 (TXB6). The start bit, parity bit, and stop bit are automatically appended to the data.
When transmission is started, the data in TXB6 is transferred to transmit shift register 6 (TXS6). After that, the data is sequentially output from TXS6 to the TXD6 pin. When transmission is completed, the parity and stop bits set by ASIM6 are appended and a transmission completion interrupt request (INTST6) is generated.
Transmission is stopped until the data to be transmitted next is written to TXB6.
Figure 14-16 shows the timing of the transmission completion interrupt request (INTST6). This interrupt occurs as soon as the last stop bit has been output.

Figure 14-16: Normal Transmission Completion Interrupt Request Timing

(a) Stop bit length: 1



(b) Stop bit length: 2



(d) Continuous transmission

The next transmit data can be written to transmit buffer register 6 (TXB6) as soon as transmit shift register 6 (TXS6) has started its shift operation. Consequently, even while the INTST6 interrupt is being serviced after transmission of one data frame, data can be continuously transmitted and an efficient communication rate can be realized. In addition, the TXB6 register can be efficiently written twice (2 bytes) without having to wait for the transmission time of one data frame, by reading bit 0 (TXSF6) of asynchronous serial interface transmission status register 6 (ASIF6) when the transmission completion interrupt has occurred.
To transmit data continuously, be sure to reference the ASIF6 register to check the transmission status and whether the TXB6 register can be written, and then write the data.

Cautions: 1. The TXBF6 and TXSF6 flags of the ASIS register change from "10" to "11", and to "01" during continuous transmission. To check the status, therefore, do not use a combination of the TXBF6 and TXSF6 flags for judgment. Read only the TXBF6 flag when executing continuous transmission.

2. When the device is incorporated in a LIN, the continuous transmission function cannot be used. Make sure that asynchronous serial interface transmission status register 6 (ASIF6) is 00H before writing transmit data to transmit buffer register 6 (TXB6).

TXBF6	Writing to TXB6 Register
0	Writing enabled
1	Writing disabled

Caution: To transmit data continuously, write the first transmit data (first byte) to the TXB6 register. Be sure to check that the TXBF6 flag is "0". If so, write the next transmit data (second byte) to the TXB6 register. If data is written to the TXB6 register while the TXBF6 flag is "1", the transmit data cannot be guaranteed.

The communication status can be checked using the TXSF6 flag.

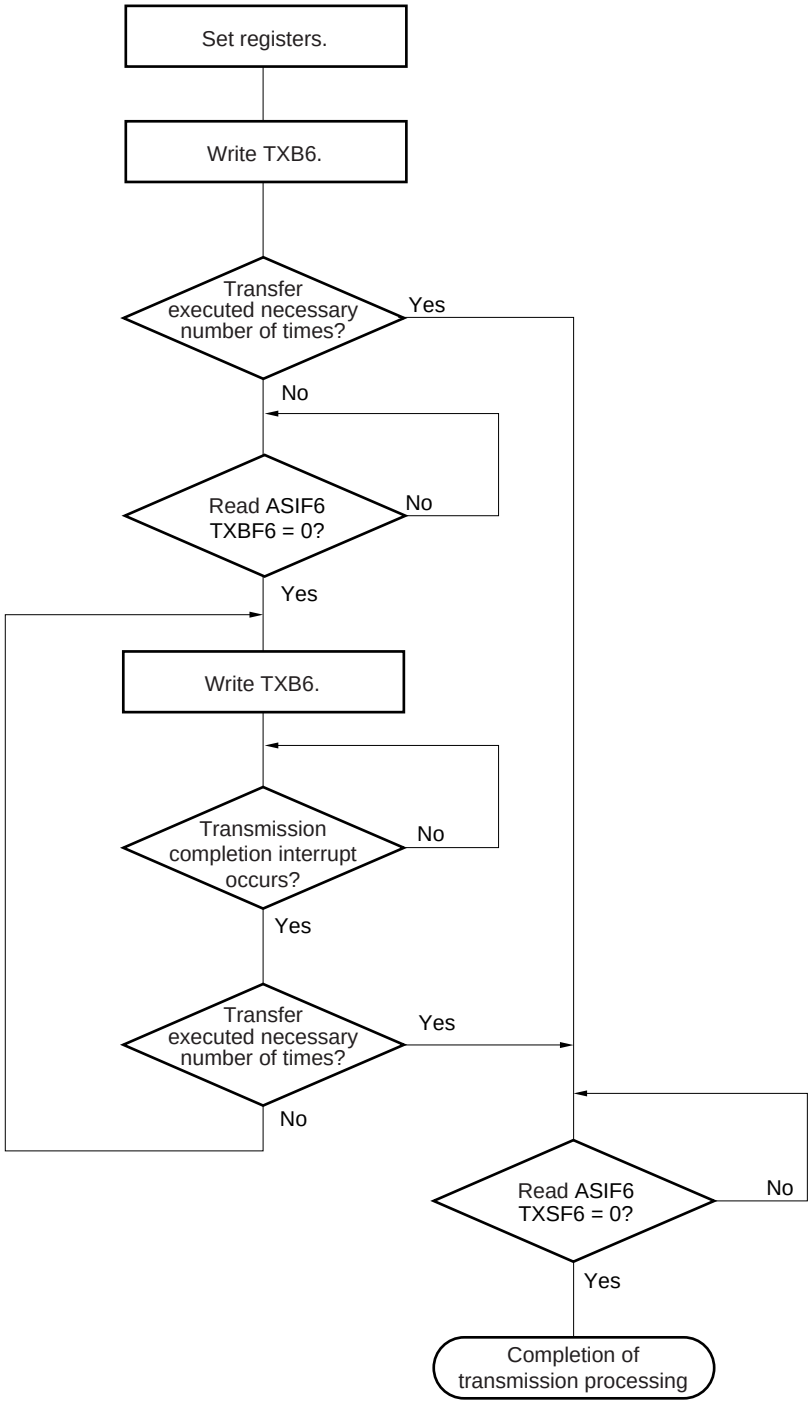
TXSF6	Transmission Status
0	Transmission is completed.
1	Transmission is in progress.

Cautions: 1. To initialize the transmission unit upon completion of continuous transmission, be sure to check that the TXSF6 flag is "0" after generation of the transmission completion interrupt, and then execute initialization. If initialization is executed while the TXSF6 flag is "1", the transmit data cannot be guaranteed.

2. During continuous transmission, an overrun error may occur, which means that the next transmission was completed before execution of INTST6 interrupt servicing after transmission of one data frame. An overrun error can be detected by developing a program that can count the number of transmit data and by referencing the TXSF6 flag.

Figure 14-17 shows an example of the continuous transmission processing flow.

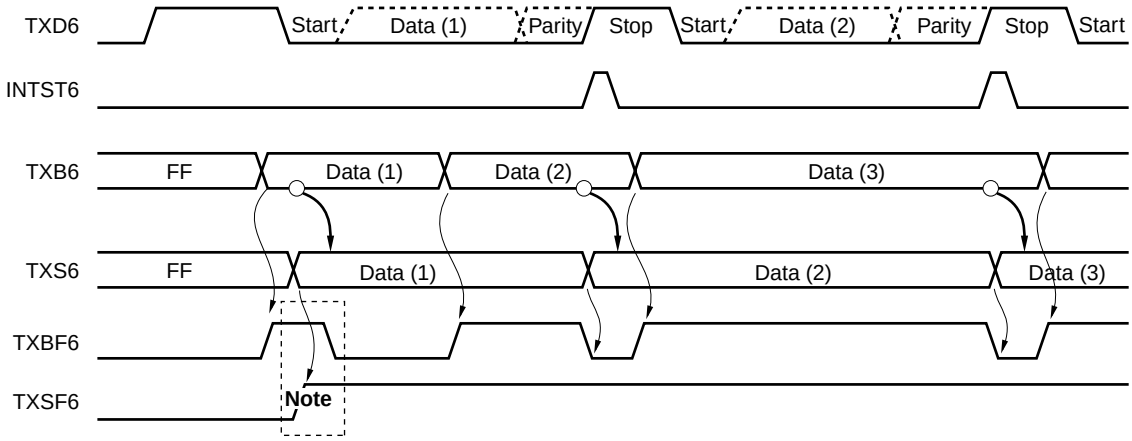
Figure 14-17: Example of Continuous Transmission Processing Flow



Remark: TXB6: Transmit buffer register 6
ASIF6: Asynchronous serial interface transmission status register 6
TXBF6: Bit 1 of ASIF6 (transmit buffer data flag)
TXSF6: Bit 0 of ASIF6 (transmit shift register data flag)

Figure 14-18 shows the timing of starting continuous transmission, and Figure 15-18 shows the timing of ending continuous transmission.

Figure 14-18: Timing of Starting Continuous Transmission

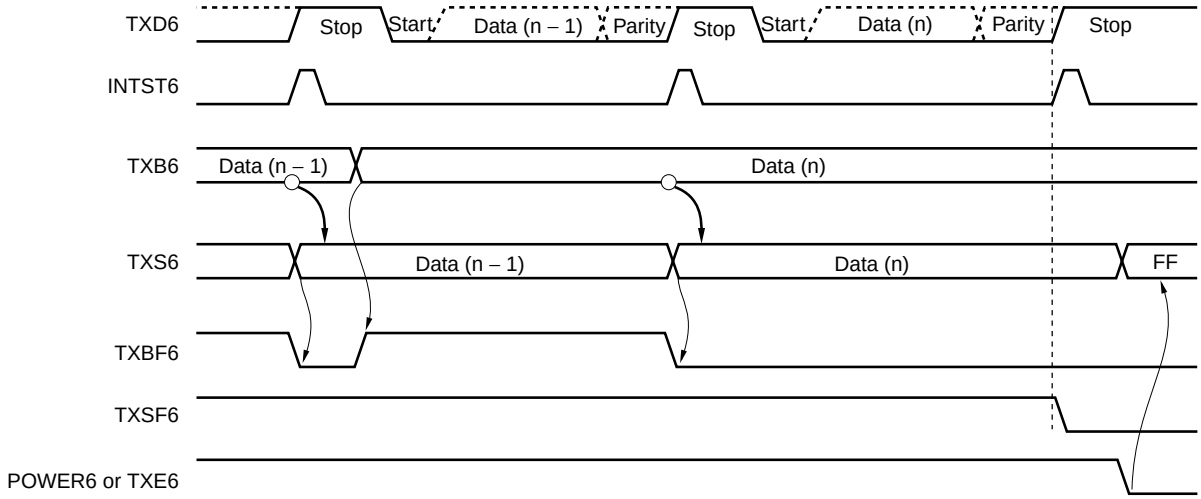


Note: When ASIF6 is read, there is a period in which TXBF6 and TXSF6 = 1, 1. Therefore, judge whether writing is enabled using only the TXBF6 bit.

Remark:

TXD6:	TXD6 pin (output)
INTST6:	Interrupt request signal
TXB6:	Transmit buffer register 6
TXS6:	Transmit shift register 6
ASIF6:	Asynchronous serial interface transmission status register 6
TXBF6:	Bit 1 of ASIF6
TXSF6:	Bit 0 of ASIF6

Figure 14-19: Timing of Ending Continuous Transmission



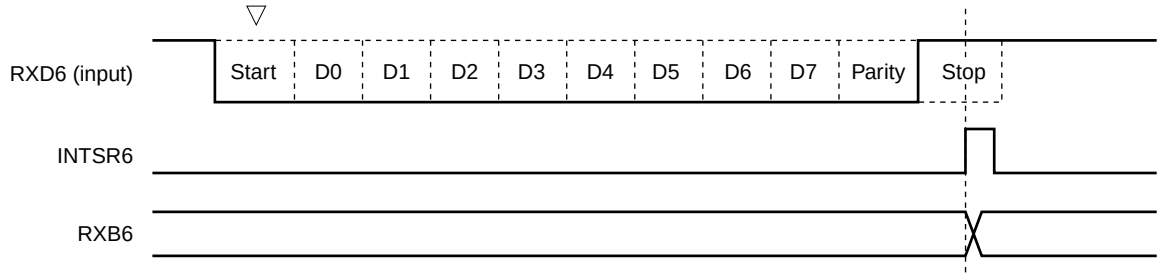
Remark:

TXD6:	TXD6 pin (output)
INTST6:	Interrupt request signal
TXB6:	Transmit buffer register 6
TXS6:	Transmit shift register 6
ASIF6:	Asynchronous serial interface transmission status register 6
TXBF6:	Bit 1 of ASIF6
TXSF6:	Bit 0 of ASIF6
POWER6:	Bit 7 of asynchronous serial interface operation mode register (ASIM6)
TXE6:	Bit 6 of asynchronous serial interface operation mode register (ASIM6)

(e) Normal reception

Reception is enabled and the RXD6 pin input is sampled when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and then bit 5 (RXE6) of ASIM6 is set to 1.
The 8-bit counter of the baud rate generator starts counting when the falling edge of the RXD6 pin input is detected. When the set value of baud rate generator control register 6 (BRGC6) has been counted, the RXD6 pin input is sampled again (in Figure 14-20). If the RXD6 pin is low level at this time, it is recognized as a start bit.
When the start bit is detected, reception is started, and serial data is sequentially stored in the receive shift register (RXS6) at the set baud rate. When the stop bit has been received, the reception completion interrupt (INTSR6) is generated and the data of RXS6 is written to receive buffer register 6 (RXB6). If an overrun error (OVE6) occurs, however, the receive data is not written to RXB6.
Even if a parity error (PE6) occurs while reception is in progress, reception continues to the reception position of the stop bit, and an error interrupt (INTSR6/INTSRE6) is generated on completion of reception.

Figure 14-20: Reception Completion Interrupt Request Timing



Cautions: 1. Be sure to read receive buffer register 6 (RXB6) even if a reception error occurs. Otherwise, an overrun error will occur when the next data is received, and the reception error status will persist.

2. Reception is always performed with the “number of stop bits = 1”. The second stop bit is ignored.

3. Be sure to read asynchronous serial interface reception error status register 6 (ASIS6) before reading RXB6.

(f) Reception error

Three types of errors may occur during reception: a parity error, framing error, or overrun error. If the error flag of asynchronous serial interface reception error status register 6 (ASIS6) is set as a result of data reception, a reception error interrupt request (INTSR6/INTSRE6) is generated.
Which error has occurred during reception can be identified by reading the contents of ASIS6 in the reception error interrupt servicing (INTSR6/INTSRE6) (see **Figure 14-6, "Asynchronous Serial Interface Reception Error Status Register 6 (ASIS6) Format," on page 324**).
The contents of ASIS6 are reset to 0 when ASIS6 is read.

Table 14-3: Cause of Reception Error

Reception Error	Cause
Parity error	The parity specified for transmission does not match the parity of the receive data.
Framing error	Stop bit is not detected.
Overrun error	Reception of the next data is completed before data is read from receive buffer register 6 (RXB6).

The error interrupt can be separated into reception completion interrupt (INTSR6) and error interrupt (INTSRE6) by clearing bit 0 (ISRM6) of asynchronous serial interface operation mode register 6 (ASIM6) to 0.

Figure 14-21: Reception Error Interrupt

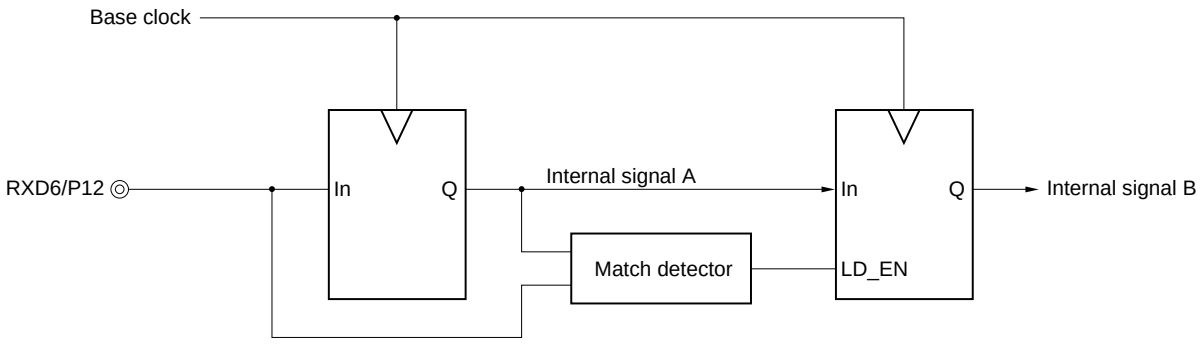
(a) If ISRM6 is cleared to 0 (reception completion interrupt (INTSR6) and error interrupt (INTSRE6) are separated)

(b) If ISRM6 is set to 1 (error interrupt is included in INTSR6)

(g) Noise filter of receive data

The RXD6 signal is sampled with the base clock output by the prescaler block.
If two sampled values are the same, the output of the match detector changes, and the data is sampled as input data.
Because the circuit is configured as shown in Figure 14-22, the internal processing of the reception operation is delayed by two clocks from the external signal status.

Figure 14-22: Noise Filter Circuit



(h) SBF transmission

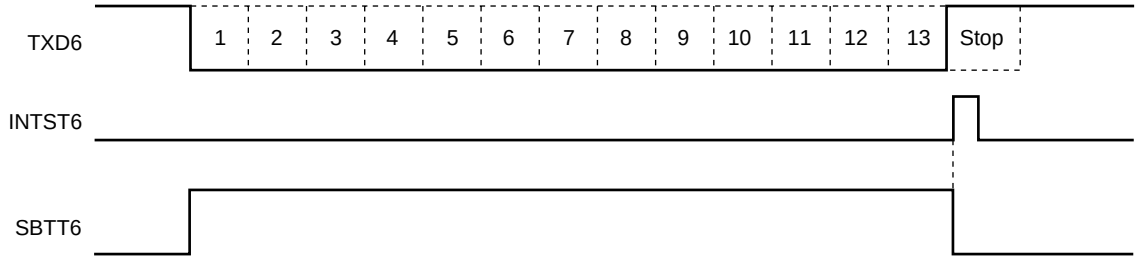
When the device is incorporated in LIN, the SBF (Synchronous Break Field) transmission control function is used for transmission. For the transmission operation of LIN, see **Figure 14-1, "LIN Transmission Operation," on page 316**.

The TXD6 pin outputs a high level when bit 7 (POWER6) of asynchronous serial interface mode register 6 (ASIM6) is set to 1. Transmission is enabled when bit 6 (TXE6) of ASIM6 is set to 1 next time, and SBF transmission operation is started when bit 5 (SBTT6) of asynchronous serial interface control register 6 (ASICL6) is set to 1.

After transmission has been started, the low levels of bits 13 to 20 (set by bits 4 to 2 (SBL62 to SBL60) of ASICL6) are output. When SBF transmission is complete, a transmission completion interrupt request (INTST6) is issued, and SBTT6 is automatically cleared. After SBF transmission is completed, the normal transmission mode is restored.

SBF transmission is stopped until the data to be transmitted next is written to transmit buffer register 6 (TXB6) or SBTT6 is set to 1.

Figure 14-23: SBF Transmission



Remark: TXD6: TXD6 pin (output)
INTST6: Transmission completion interrupt request
SBTT6: Bit 5 of asynchronous serial interface control register 6 (ASICL6)

(i) SBF reception

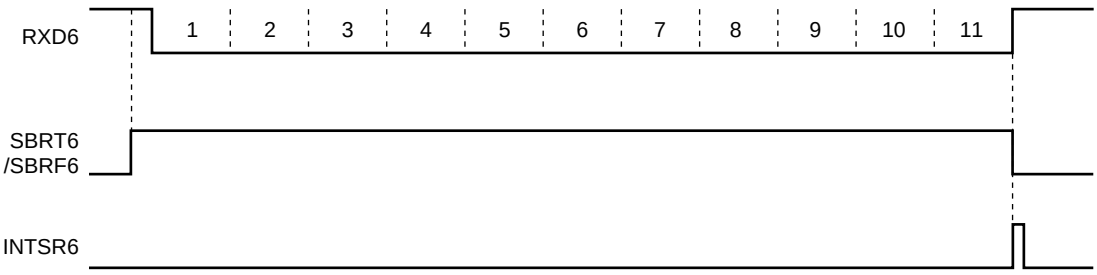
When the device is incorporated in LIN, the SBF (Synchronous Break Field) reception control function is used for reception. For the reception operation of LIN, see **Figure 14-2, "LIN Reception Operation," on page 317**.

Reception is enabled when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and then bit 5 (RXE6) of ASIM6 is set to 1. SBF reception is enabled when bit 6 (SBRT6) of asynchronous serial interface control register 6 (ASICL6) is set to 1. In the SBF reception enabled status, the RxD6 pin is sampled and the start bit is detected in the same manner as the normal reception enable status.

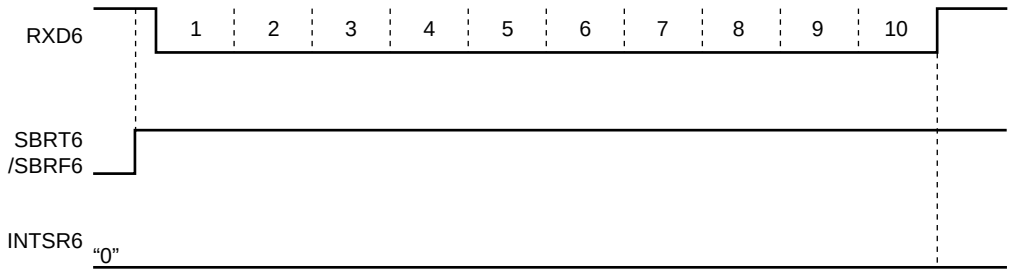
When the start bit has been detected, reception is started, and serial data is sequentially stored in the receive shift register 6 (RXS6) at the set baud rate. When the stop bit is received and if the width of SBF is 11 bits or more, a reception completion interrupt request (INTSR6) is generated as normal processing. At this time, the SBRF6 and SBRT6 bits are automatically cleared, and SBF reception ends. Detection of errors, such as OVE6, PE6, and FE6 (bits 0 to 2 of asynchronous serial interface reception error status register 6 (ASIS6)) is suppressed, and error detection processing of UART communication is not performed. In addition, data transfer between receive shift register 6 (RXS6) and receive buffer register 6 (RXB6) is not performed, and the reset value of FFH is retained. If the width of SBF is 10 bits or less, an interrupt does not occur as error processing after the stop bit has been received, and the SBF reception mode is restored. In this case, the SBRF6 and SBRT6 bits are not cleared.

Figure 14-24: SBF Reception

(a) Normal SBF reception (stop bit is detected with a width of more than 10.5 bits)



(b) SBF reception error (stop bit is detected with a width of 10.5 bits or less)



Remark: RXD6: RXD6 pin (input)
SBRT6: Bit 6 of asynchronous serial interface control register 6 (ASICL6)
SBRF6: Bit 7 of ASICL6
INTSR6: Reception completion interrupt request

14.4.3 Dedicated baud rate generator

The dedicated baud rate generator consists of a source clock selector and an 8-bit programmable counter, and generates a serial clock for transmission/reception of UART6.
Separate 8-bit counters are provided for transmission and reception.

(1) Configuration of baud rate generator

- Base clock

  The clock selected by bits 3 to 0 (TPS63 to TPS60) of clock selection register 6 (CKSR6) is supplied to each module when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is 1. This clock is called the base clock and its frequency is called f_{XCLK6}. The base clock is fixed to low level when POWER6 = 0.

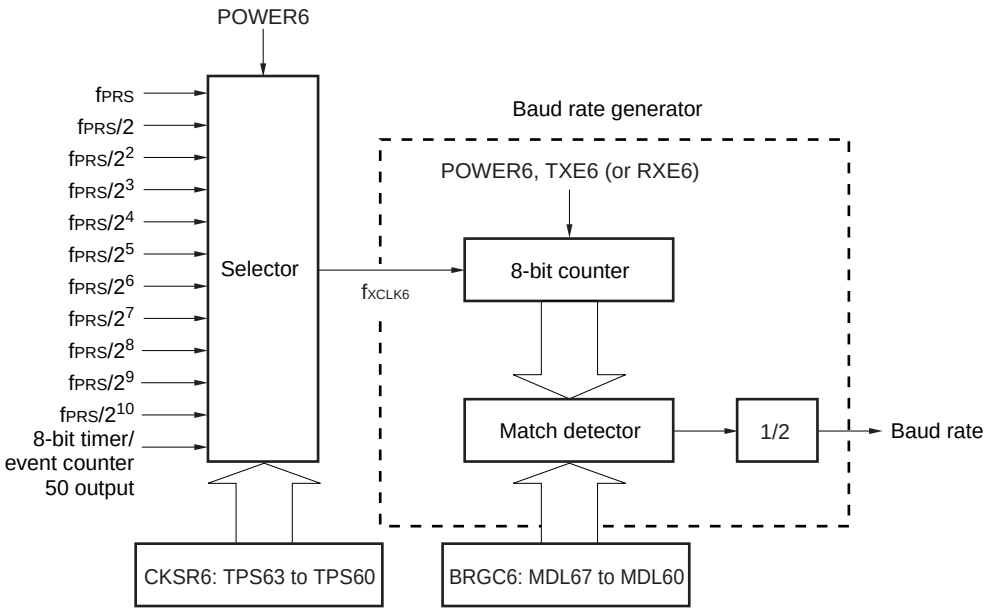
- Transmission counter

  This counter stops operation, cleared to 0, when bit 7 (POWER6) or bit 6 (TXE6) of asynchronous serial interface operation mode register 6 (ASIM6) is 0.
  It starts counting when POWER6 = 1 and TXE6 = 1.
  The counter is cleared to 0 when the first data transmitted is written to transmit buffer register 6 (TXB6).
  If data are continuously transmitted, the counter is cleared to 0 again when one frame of data has been completely transmitted. If there is no data to be transmitted next, the counter is not cleared to 0 and continues counting until POWER6 or TXE6 is cleared to 0.

- Reception counter

  This counter stops operation, cleared to 0, when bit 7 (POWER6) or bit 5 (RXE6) of asynchronous serial interface operation mode register 6 (ASIM6) is 0.
  It starts counting when the start bit has been detected.
  The counter stops operation after one frame has been received, until the next start bit is detected.

Figure 14-25: Configuration of Baud Rate Generator



Remark: POWER6: Bit 7 of asynchronous serial interface operation mode register 6 (ASIM6)
TXE6: Bit 6 of ASIM6
RXE6: Bit 5 of ASIM6
CKSR6: Clock selection register 6
BRGC6: Baud rate generator control register 6
f_{PRS}: Peripheral hardware clock oscillation frequency

(2) Generation of serial clock

A serial clock can be generated by using clock selection register 6 (CKSR6) and baud rate generator control register 6 (BRGC6).
Select the clock to be input to the 8-bit counter by using bits 3 to 0 (TPS63 to TPS60) of CKSR6.
Bits 7 to 0 (MDL67 to MDL60) of BRGC6 can be used to select the division value of the 8-bit counter.

(a) Baud rate

The baud rate can be calculated by the following expression.

$$\text{Baud rate} = \frac{f_{XCLK6}}{2 \times k} \text{ [bps]}$$

f_{XCLK6}: Frequency of base clock selected by TPS63 to TPS60 bits of CKSR6 register
k: Value set by MDL67 to MDL60 bits of BRGC6 register (k = 8, 9, 10,..., 255)

(b) Error of baud rate

The baud rate error can be calculated by the following expression.

$$\text{Error (\%)} = \left[\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (correct baud rate)}} - 1\right] \times 100 \text{ [\%]}$$

Cautions: 1. Keep the baud rate error during transmission to within the permissible error range at the reception destination.

2. Make sure that the baud rate error during reception satisfies the range shown in (4) Permissible baud rate range during reception.

Example:

Frequency of base clock = 16 MHz = 16,000,000 Hz
Set value of MDL67 to MDL60 bits of BRGC6 register = 00110100B (k = 52)
Target baud rate = 153600 bps

Baud rate = 16 M/(2 × 52)
= 16000000/(2 × 52) = 153,846 [bps]

Error = (153846/153600 - 1) × 100
= +0.16 [%]

(3) Example of setting baud rate

Table 14-4: Set Data of Baud Rate Generator

Baud Rate [bps]	f_{PRS} = 16 MHz				f_{PRS} = 10.0 MHz				f_{PRS} = 8.38 MHz			
	TPS63 to TPS60	k	Calculated Value	ERR[%]	TPS63 to TPS60	k	Calculated Value	ERR[%]	TPS63 to TPS60	k	Calculated Value	ERR[%]
600	8H	52	601	0.17	6H	130	601	0.16	6H	109	601	0.11
1200	7H	52	1202	0.17	5H	130	1202	0.16	5H	109	1201	0.11
2400	6H	52	2404	0.17	4H	130	2404	0.16	4H	109	2403	0.11
4800	5H	52	4808	0.17	3H	130	4808	0.16	3H	109	4805	0.11
9600	4H	52	9615	0.16	2H	130	9615	0.16	2H	109	9610	0.11
10400	4H	48	10417	0.16	2H	120	10417	0.16	2H	101	10371	0.28
19200	3H	52	19231	0.16	1H	130	19231	0.16	1H	109	19200	0.11
31250	3H	32	31250	0.00	1H	80	31250	0.00	0H	134	31268	0.06
38400	2H	52	38462	0.16	0H	130	38462	0.16	0H	109	38440	0.11
76800	1H	52	76923	0.16	0H	65	76923	0.16	0H	55	76182	-0.80
115200	0H	69	115942	0.64	0H	43	116279	0.94	0H	36	116388	1.03
153600	0H	52	153846	0.16	0H	33	151515	-1.36	0H	27	155185	1.03

Remark: TPS63 to TPS60: Bits 3 to 0 of clock selection register 6 (CKSR6) (setting of base clock (f_{XCLK6}))

k: Value set by MDL67 to MDL60 bits of baud rate generator control register 6 (BRGC6) (k = 8, 9, 10,..., 255)

f_{PRS}: Peripheral hardware clock oscillation frequency

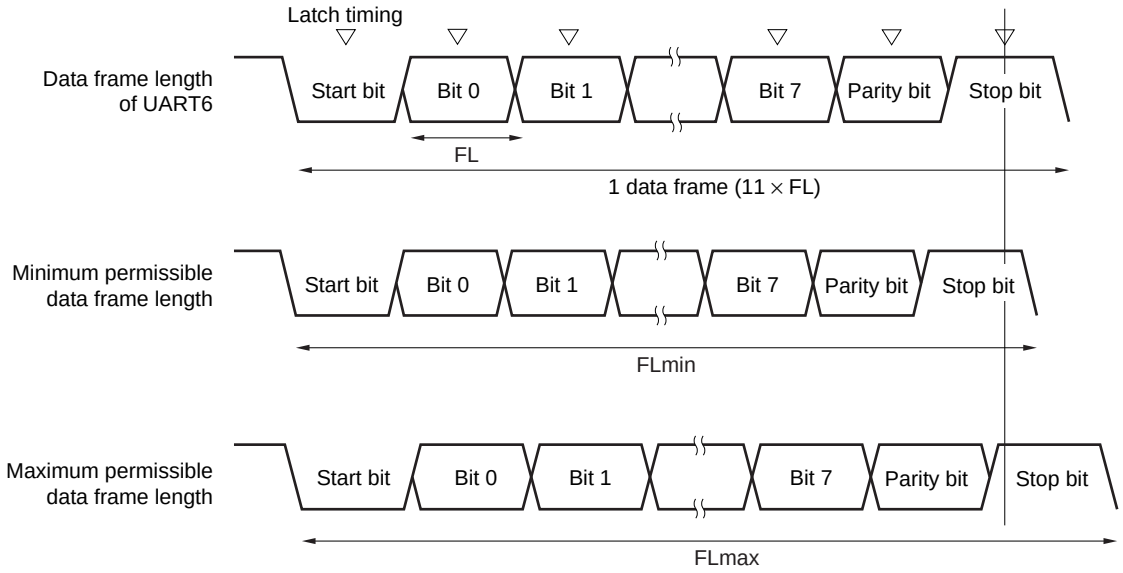
ERR: Baud rate error

(4) Permissible baud rate range during reception

The permissible error from the baud rate at the transmission destination during reception is shown below.

Caution: Make sure that the baud rate error during reception is within the permissible error range, by using the calculation expression shown below.

Figure 14-26: Permissible Baud Rate Range During Reception



As shown in Figure 14-26, the latch timing of the receive data is determined by the counter set by baud rate generator control register 6 (BRGC6) after the start bit has been detected. If the last data (stop bit) meets this latch timing, the data can be correctly received.
Assuming that 11-bit data is received, the theoretical values can be calculated as follows.

$$FL = (Brate)^{-1}$$

Brate: Baud rate of UART6
k: Set value of BRGC6
FL: 1-bit data length
Margin of latch timing: 2 clocks

Minimum permissible data frame length:

$$FLmin = 11 \times FL - \frac{k - 2}{2k} \times FL = \frac{21k + 2}{2k} FL$$

Therefore, the maximum receivable baud rate at the transmission destination is as follows:

$$BRmax = (FLmin/11)^{-1} = \frac{22k}{21k + 2} \ Brate$$

Similarly, the maximum permissible data frame length can be calculated as follows:

$$\frac{10}{11} \times FLmax = 11 \times FL - \frac{k + 2}{2 \times k} \times FL = \frac{21k - 2}{2 \times k} FL$$

$$FLmax = \frac{21k - 2}{20k} FL \times 11$$

Therefore, the minimum receivable baud rate at the transmission destination is as follows:

$$BRmin = (FLmax/11)^{-1} = \frac{20k}{21k - 2} \ Brate$$

The permissible baud rate error between UART6 and the transmission destination can be calculated from the above minimum and maximum baud rate expressions, as follows.

Table 14-5: Maximum/Minimum Permissible Baud Rate Error

Division Ratio (k)	Maximum Permissible Baud Rate Error	Minimum Permissible Baud Rate Error
8	+3.53%	-3.61%
20	+4.26%	-4.31%
50	+4.56%	-4.58%
100	+4.66%	-4.67%
255	+4.72%	-4.73%

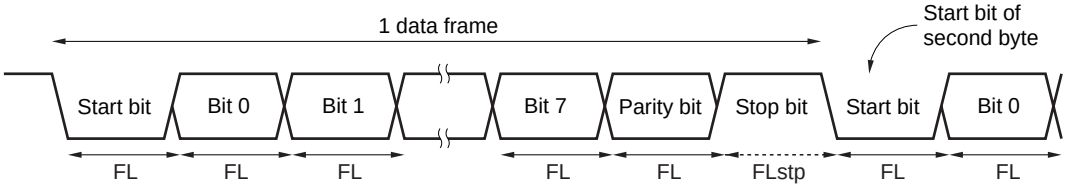
Remarks: 1. The permissible error of reception depends on the number of bits in one frame, input clock frequency, and division ratio (k). The higher the input clock frequency and the higher the division ratio (k), the higher the permissible error.

2. K.: Set value of BRGC6

(5) Data frame length during continuous transmission

When data is continuously transmitted, the data frame length from a stop bit to the next start bit is extended by two clocks of base clock from the normal value. However, the result of communication is not affected because the timing is initialized on the reception side when the start bit is detected.

Figure 14-27: Data Frame Length During Continuous Transmission



Where the 1-bit data length is FL, the stop bit length is FLstp, and base clock frequency is f_{XCLK6}, the following expression is satisfied.

$$FLstp = FL + 2/f_{XCLK6}$$

Therefore, the data frame length during continuous transmission is:

$$\text{Data frame length} = 11 \times FL + 2/f_{XCLK6}$$

[MEMO]

Chapter 15 CAN Controller (µPD780822B Subseries)

Table 15-1: Outline of the Function

Feature	Details
Protocol	CAN2.0 with active extended frame capability (ISO 11898)
Baud rate	Max. 1 MBaud at 16 MHz clock supply
Bus line control	CMOS in / out for external transceiver
Clock	Selected by register
Data storage	CPU RAM area with shared access DCAN uses up to 288 byte of RAM Unused bytes can be used by CPU for other tasks
Message organisation	Received messages will be stored in RAM area depending on message identifier Transmit messages have two dedicated buffers in RAM area
Message number	One input receive shadow buffer (not readable by user) Up to 16 receive message objects including 2 masks Two transmit channels
Message sorting	Unique identifier on all 16 receive message objects Up to 2 message objects with mask Global mask for all messages
DCAN protocol	SFR access for general control
Interrupt	Transmit interrupt for each channel One receive interrupt with enable control for each message One error interrupt
Time functions	Support of time stamp and global time system Programmable single shot mode
Diagnostic	Readable error counters “Valid protocol activity flag” for verification of bus connection “Receive only” mode for automatic baudrate detection
Power down modes	Sleep mode: Wake up from CAN bus Stop mode: No wake-up from CAN bus

15.1 CAN Protocol

CAN is an abbreviation of "Controller Area Network", and is a class C high speed multiplexed communication protocol. CAN is specified by Bosch in the CAN specification 2.0 from September 1991 and is standardized in ISO-11898 (International Organization for Standardization) and SAE (Society of Automotive Engineers).

15.1.1 Protocol Mode Function

(1) Standard format mode

- This mode supports an 11-bit message identifier thus making it possible to differentiate between 2048 types of messages.

(2) Extended format mode

- In the extended format mode, the identifier has 29 bits. It is built by the standard identifier (11 bits) and an extended identifier (18 bits).
- When the IDE bits of the arbitration field is "recessive", the frame is sent in the extended format mode.
- When a message in extended format mode and a remote frame in standard format mode are simultaneously transmitted, the node transmitting the message with the standard mode wins the arbitration.

(3) Bus values

- The bus can have one of two complementary logical values: "dominant" or "recessive". During simultaneous transmission of "dominant" and "recessive" bits, the resulting bus value will be "dominant" (non destructive arbitration).
- For example, in case of a wired-AND implementation of the bus, the "dominant" level would be represented by a logical "0" and the "recessive" level by a logical “1”. This specific representation is used in this manual.
- Physical states (e.g. electrical voltage, light) that represent the logical levels are not given in this document.

15.1.2 Message Format

The CAN protocol message supports different types of frames. The types of frames are listed below:

- Data frame: Carries the data from a transmitter to the receiver.
- Remote frame: Transmission demand frame from the requesting node.
- Error frame: Frame sent on error detection.
- Overload frame: Frame sent when a data or remote frame would be overwritten by the next one before the receiving node could process it. The reception side did not finish its operations on the reception of the previously received frame yet.

15.1.3 Data Frame / Remote Frame

Figure 15-1: Data Frame

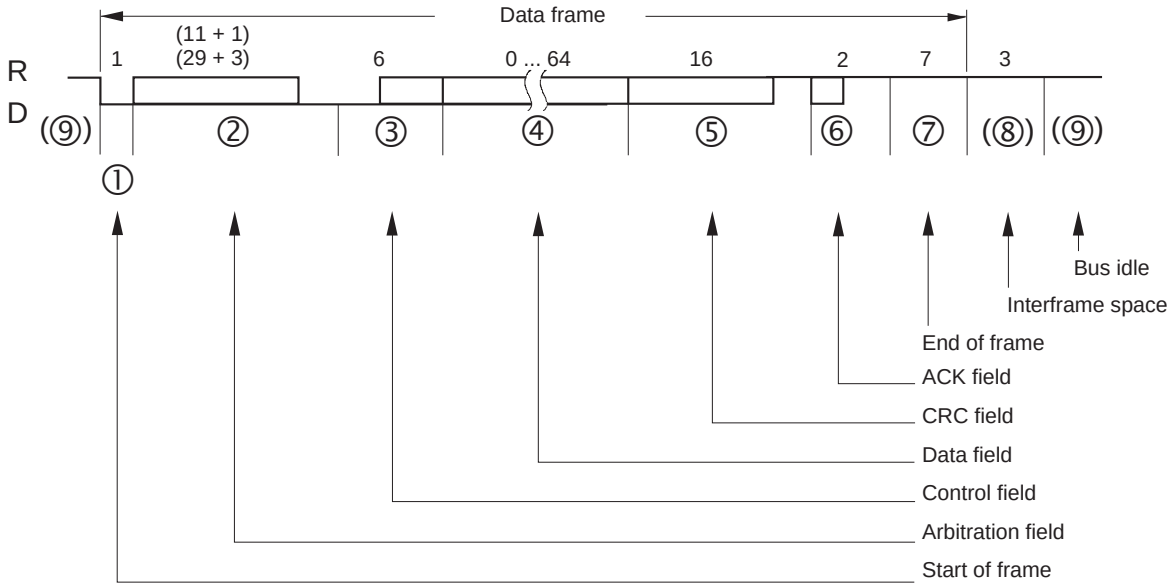
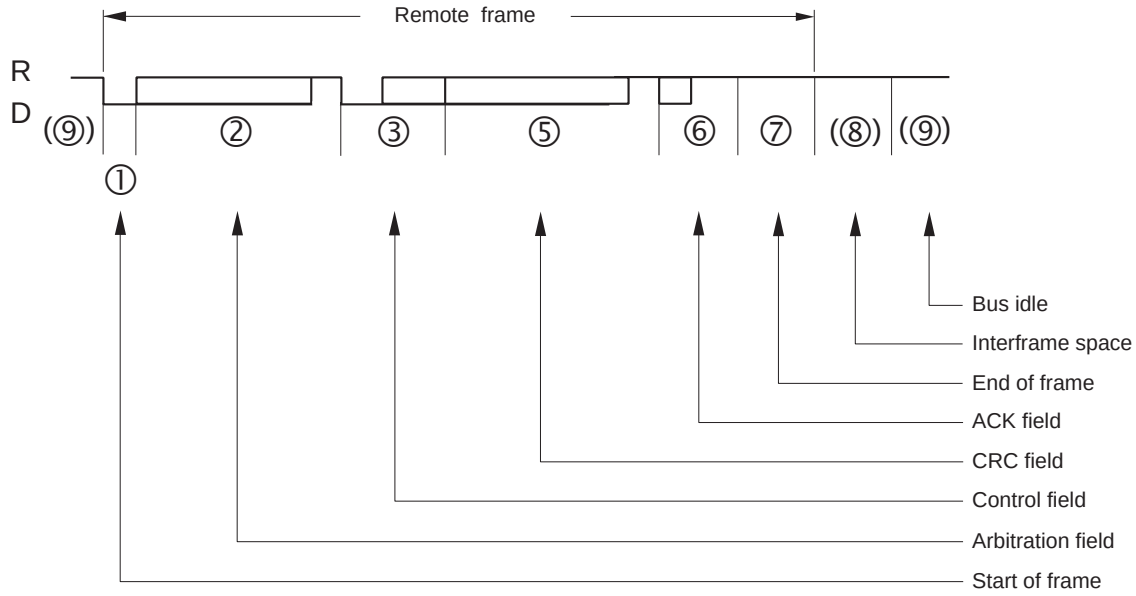


Figure 15-2: Remote Frame

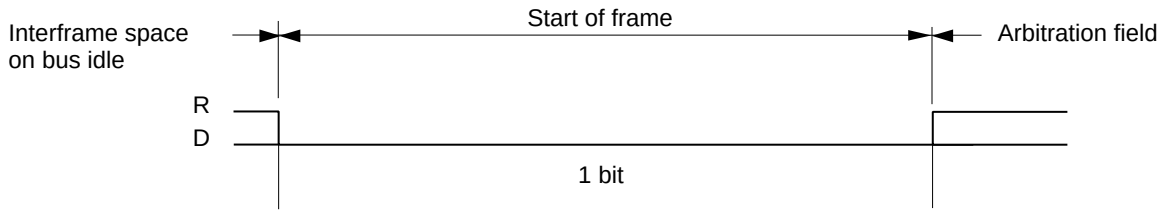


Remark: This frame is transmitted when the reception node requests transmission. Data field is not transmitted even if the data length code ≠ '0' in the control field.

15.1.4 Description of each field

(1) "R" indicates recessive level. "D" indicates dominant level.
Start of frame: The start of data frame and remote frame are indicated.

Figure 15-3: Data Frame



- The start of frame (SOF) is denoted by the falling edge of the bus signal.
- Reception continues when 'Dominant level' is detected at the sample point.
- The bus becomes idle state when 'Recessive level' is detected at a sample point.

(2) Arbitration field: Sets priority, specifies data frame or remote frame, and defines the protocol mode.

Figure 15-4: Arbitration Field/Standard Format Mode

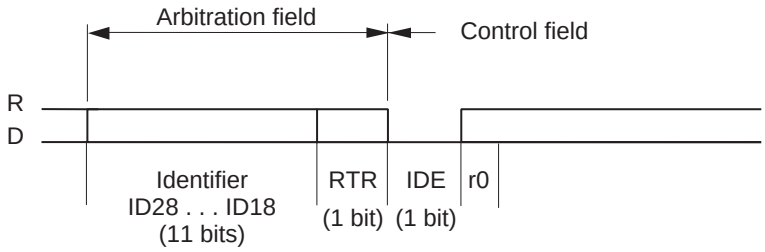
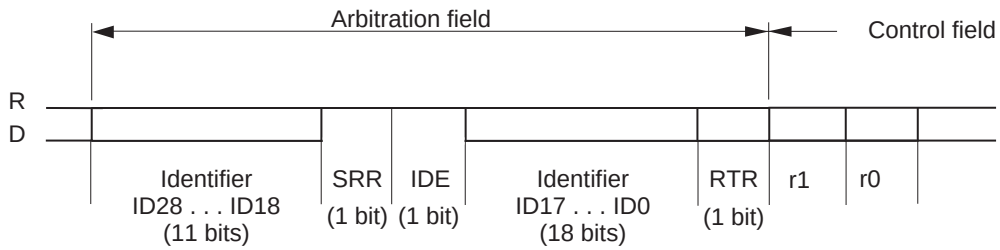


Figure 15-5: Arbitration Field/Extended Format Mode



- ID28 - ID0 is the identifier.
- The identifier is transmitted with MSB at first position.
- Substitute Remote Request (SRR) is only used in extended format mode and is always recessive.

Table 15-2: Bit Number of the Identifier

Protocol Mode Identifier	Number
Standard format mode	11 bits
Extended format mode	29 bits

Table 15-3: RTR Setting

Frame Type	RTR Bit
Data frame	0
Remote frame	1

Table 15-4: Mode Setting

Protocol Mode	IDE Bit
Standard format mode	0
Extended format mode	1

(3) Control field: The data byte number DLC in the data field specifies the number of data bytes in the current frame (DLC=0 to 8).

Figure 15-6: Control Field (Standard Format Mode)

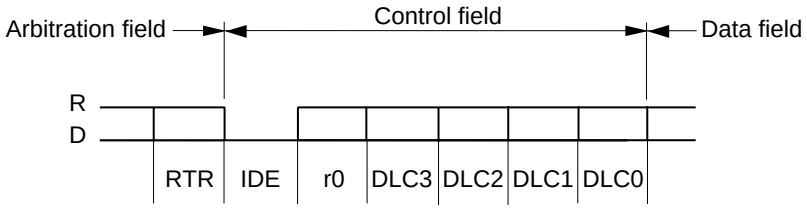
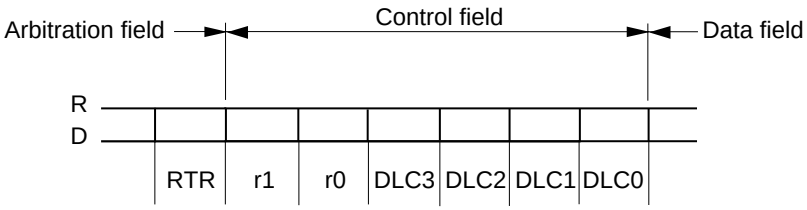


Figure 15-7: Control Field (Extended Format Mode)



- The bits r0 and r1 are reserved bits for future use and are recommended to be recessive.

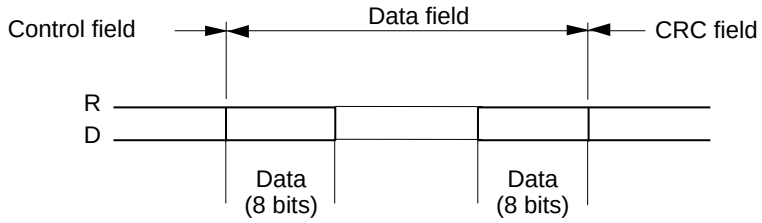
Table 15-5: Data Length Code Setting

Data Length Code				
DLC3	DLC2	DLC1	DLC0	Number of Data Bytes
0	0	0	0	0
0	0	0	1	1
.	.	.	.	
.	.	.	.	
0	1	1	1	7
1	X	X	X	8

Remark: In case of a remote frame, the data field is not generated even if data length code ≠ '0'.

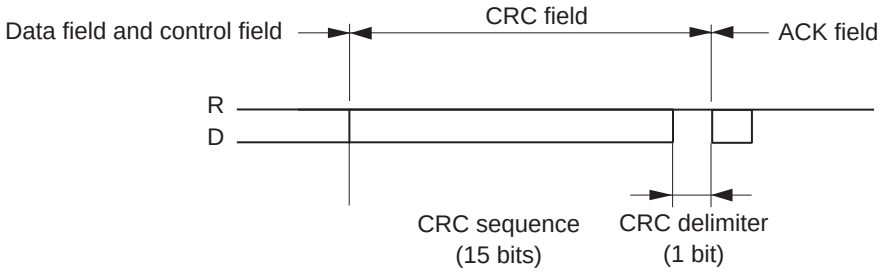
(4) Data field: This field carries the data bytes to be sent. The number of data bytes is defined by the DLC value.

Figure 15-8: Data Field



(5) CRC field: This field consists of a 15-bit CRC sequence to check the transmission error and a CRC delimiter.

Figure 15-9: CRC Field



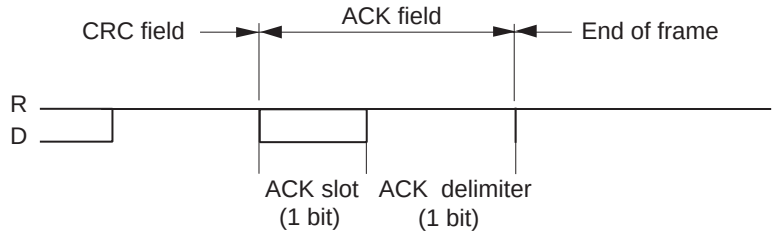
- 15 bits CRC generation polynomial is expressed by

$$P(X) = X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1.$$

- Transmission node: Transmits the CRC sequence calculated from the start of frame, arbitration field, control field and data field eliminating stuff bits.
- Reception node: The CRC received will be compared with the CRC calculated in the receiving node. For this calculation the stuff bits of the received CRC are eliminated. In case these do not match, the node issues an error frame.

(6) ACK field: For check of normal reception.

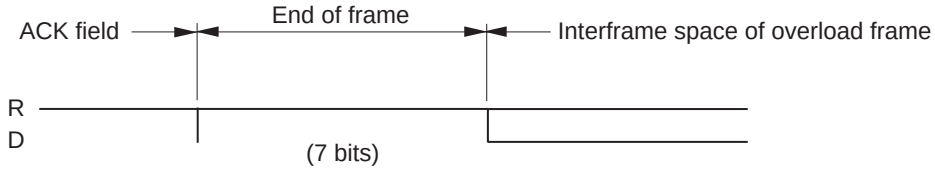
Figure 15-10: ACK Field



- Receive node sets the ACK slot to dominant level if no error was detected.

(7) End of frame: Indicates the end of the transmission/reception.

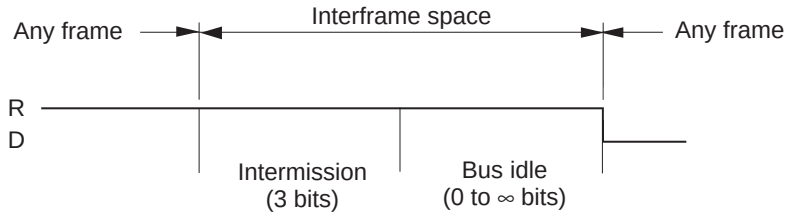
Figure 15-11: End of Frame



(8) Interframe space: This sequence is inserted after data frames, remote frames, error frames, and overload frames in the serial bitstream on the bus to indicate start or end of a frame. The length of the interframe space depends on the error state (active or passive) of the node.

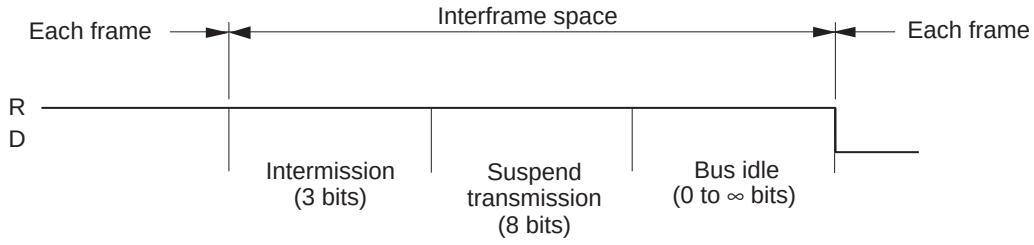
(a) Error active: Consists of 3 bits intermission and bus idle.

Figure 15-12: Interframe Space/Error Active



(b) Error passive: Consists of 3 bits intermission, suspend transmission and bus idle.

Figure 15-13: Interframe Space/Error Passive



Remark: The nominal value of the intermission field is 3 bits. However, transmission nodes may start immediately a transmission already in the 3rd bit of this field when a dominant level is detected.

Table 15-6: Operation in the Error State

Error State	Operation
Error active	Any node in this state is able to start a transmission whenever the bus is idle.
Error passive	Any node in this state has to wait for 11 consecutive recessive bits before initiating a transmission.

15.1.5 Error Frame

- This frame is sent from a node if an error is detected.
- The type of an Error Frame is defined by its error flag: ACTIVE ERROR FLAG or PASSIVE ERROR FLAG. Which kind of flag a node transmits after detecting an error condition depends on the internal count of the error counters of each node.

Figure 15-14: Error Frame

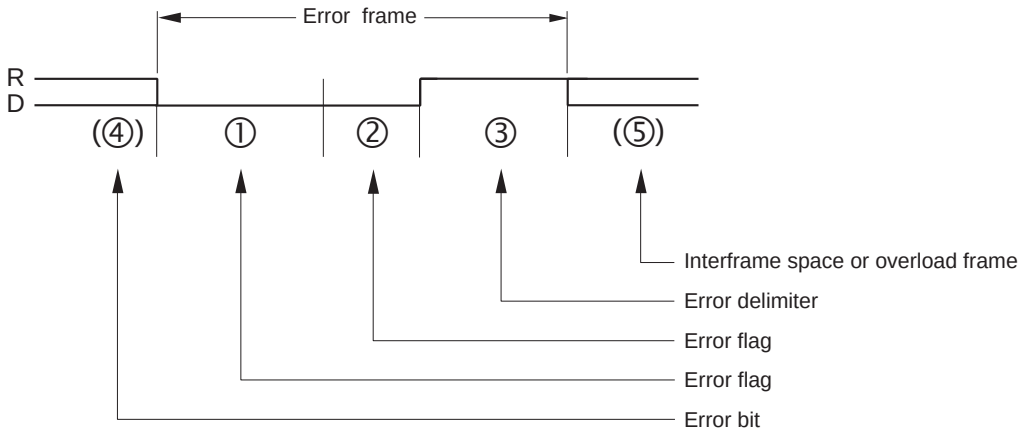


Table 15-7: Definition of each Field

No.	Name	Bit Number	Definition
1	Error flag	6	Error active node: sends 6 bits dominant level continuously. Error passive node: sends 6 bits recessive level continuously.
2	Error flag superpositioning	0 to 6	Nodes receiving an “error flag” detect bit stuff errors and issue error flags’ themselves.
3	Error delimiter	8	Sends 8 bits recessive level continuously. In case of monitoring dominant level at 8th bit, an overload frame is transmitted after the next bit.
4	Erroneous bit	-	An error frame is transmitted continuously after the bit where the error has occurred (in case of a CRC error, transmission continues after the ACK delimiter).
5	Interframe space/ overload frame	3/14 20 MAX	Interframe space or overload frame continues.

15.1.6 Overload Frame

- This frame is started at the first bit of the intermission when the reception node is busy with exploiting the receive operation and is not ready for further reception.
- When a bit error is detected in the intermission, also an overload frame is sent following the next bit after the bit error detection.
- Detecting a dominant bit during the 3rd bit of intermission will be interpreted as START OF FRAME.
- At most two OVERLOAD FRAMEs may be generated to delay the next DATA FRAME or REMOTE FRAME.

Figure 15-15: Overload Frame

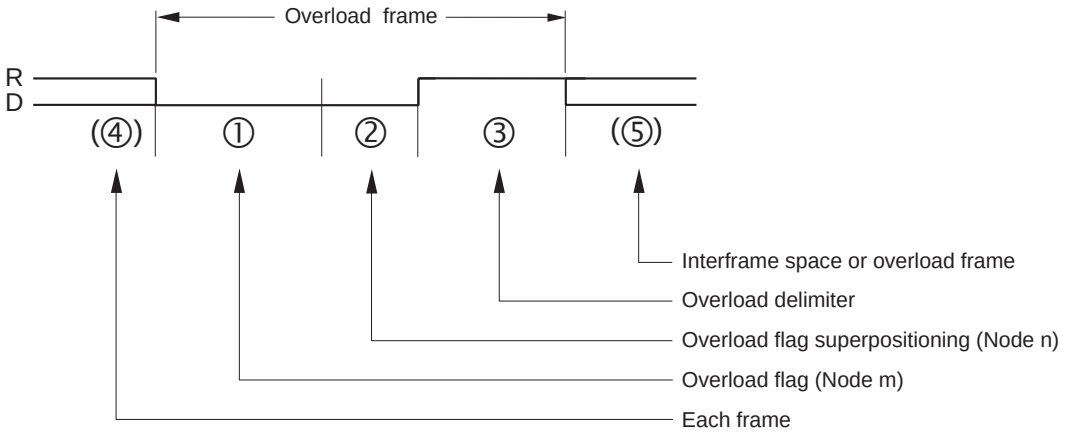


Table 15-8: Definition of each Frame

No.	Name	Bit Number	Definition
1	Overload flag	6	Sent 6 bits dominant level continuously.
2	Overload flag from any node	0 to 6	A node that receives an overload flag in the interframe space. Issues an overload flag.
3	Overload delimiter	8	Sends 8 bits recessive level continuously. In case of monitoring dominant level at 8th bit, an overload frame is transmitted after the next bit.
4	Any frame	-	Output following the end of frame, error delimiter and overload delimiter.
5	Interframe space/ overload frame	3/14 20 MAX	Interframe space or overload frame continues.

15.2 Function

15.2.1 Arbitration

If two or more nodes happen to start transmission in coincidence, the access conflict is solved by a bit-wise arbitration mechanism during transmission of the ARBITRATION FIELD.

(1) When a node starts transmission:

- During bus idle, the node having the output data can transmit.

(2) When more than one node starts transmission:

- The node with the lower identifier wins the arbitration.
- Any transmitting node compares its output arbitration field and the data level on the bus.
- It looses arbitration, when it sends recessive level and reads dominant from bus.

Table 15-9: Arbitration

Level Detection	Status of Arbitrating Node
Conformity of Level	Continuous Transmission
Non-conformity of Level	The data output is stopped from the next bit and reception operation starts.

(3) Priority of data frame and remote frame:

- When a data frame and remote frame with the same message identifier are on the bus, the data frame has priority because its RTR bit carries 'Dominant level'. The data frame wins the arbitration.

15.2.2 Bit Stuffing

When the same level continues for more than 5 bits, bit stuffing (insert 1 bit with inverse level) takes place.

- Due to this a resynchronization of the bit timing can be done at least every 10 bits.
- Nodes detecting an error condition send an error frame, violating the bit stuff rule and indicating this message to be erroneous for all nodes.

Table 15-10: Bit Stuffing

Transmission	During the transmission of a data frame and a remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, 1 bit level with reverse level of data is inserted before the following bit.
Reception	During the reception of a data frame and a remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, the reception is continued by deleting the next bit.

15.2.3 Multi Master

As the bus priority is determined by the identifier, any node can be the bus master.

15.2.4 Multi Cast

Any message can be received by any node (broadcast).

15.2.5 Sleep Mode/Stop Function

This is a function to put the CAN controller in waiting mode to achieve low power consumption. The SLEEP mode of the DCAN complies to the method described in ISO 11898.
Additional to this SLEEP mode, which can be woken up by bus activities, the STOP mode is fully controlled by the CPU device.

15.2.6 Error Control Function

(1) Error types

Table 15-11: Error Types

Type	Description of Error		Detection State	
	Detection Method	Detection Condition	Transmission/ Reception	Field/Frame
Bit error	Comparison of output level and level on the bus (except stuff bit)	Disagreement of both levels	Transmission/ reception node	Bit that output data on the bus at the start of frame to the end of frame, error frame and overload frame.
Stuff error	Check of the reception data at the stuff bit	6 consecutive bits of the same output level	Transmission/ reception node	Start of frame to CRC sequence
CRC error	Comparison of the CRC generated from the reception data and the received CRC sequence	Disagreement of CRC	Reception node	Start of frame to data field
Form error	Field/frame check of the fixed format	Detection of the fixed format error	Reception node	CRC delimiter ACK field End of frame Error frame Overload frame
ACK error	Check of the ACK slot by the transmission node	Detection of recessive level in ACK slot	Transmission node	ACK slot

(2) Output timing of the error frame

Table 15-12: Output Timing of the Error Frame

Type	Output timing
Bit error, stuff error, form error, ACK error	Error frame is started at the next bit timing following the detected error
Error passive	CRC error Error frame is started at the next bit timing following the ACK delimiter

(3) Measures when error occurs

- Transmission node re-transmits the data frame or the remote frame after the error frame.
- The CAN standard (ISO-11898) allows a programmable suppression of this re-transmission. It is called single shot mode.

(4) Error state

(a) Types of error state

- Three types of error state: These are error active, error passive and bus off.
- The transmission error counter (TEC) and the reception error counter (REC) control the error state.
- The error counters are incremented on each error occurrence (refer to Table Table 15-14, “Error Counter,” on page 370).
- If the value of error counter exceeds 96, warning level for error passive state is reached.
- When only one node is active at start-up, it may not receive an acknowledgment on a transmitted message. This will increment TEC until error passive state is reached. The bus off state will not be reached because for this specific condition TEC will not increment any more if values greater than 127 are reached.
- A node in bus off state will not issue any dominant level on the CAN transmit pin. The reception of messages is not affected by the bus off state.

Table 15-13: Types of Error

Type	Operation	Value of Error Counter	Output Error Flag Type
Error active	Transmission/ reception	0 to 127	Active error flag (6 bits of dominant level continue)
Error passive	Transmission	128 to 255	Passive error flag (6 bits of recessive level continue)
	Reception	128 or more	
Bus off	Transmission	more than 255	Communication cannot be made
	Reception	-	Does not exist

(b) Error counter

- Error counter counts up when an error has occurred, and counts down upon successful transmission and reception. The error counters are updated during the first bit of an error flag.

Table 15-14: Error Counter

State	Transmission Error Counter (TEC)	Reception Error Counter (REC)
Reception node detects an error (except bit error in the active error flag or overload flag).	No change	+1
Reception node detects dominant level following the error flag of the own error frame.	No change	+8
Transmission node transmits an error flag. Exception: 1. ACK error is detected in the error passive state and dominant level is not detected in the passive error flag sent. 2. Stuff error generation in arbitration field.	+8	No change
Bit error detection during active error flag and overload flag when transmitting node is in error active state.	+8	No change
Bit error detection during active error flag and overload flag when receiving node is in error active state.	No change	+8
When the node detects fourteen continuous dominant bits counted from the beginning of the active error flag or the overload flag, and every time, eight subsequent dominant bits after that are detected. Every time when the node detects eight continuous dominant bits after the passive error flag.	+8	+8
When the transmitting node has completed to sent without error.	-1 (-0 when error counter = 0)	No change
When the reception node has completed to receive without error.	No change	-1 (1 ≤REC ≤127) –0 (REC = 0) 119-127 (REC > 127)

(c) Overload frame

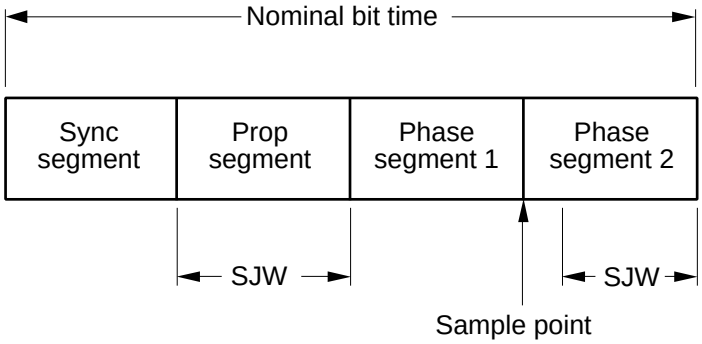
- In case the recessive level of first intermission bit is driven to dominant level, an overload frame occurs on the bus. Upon detection of an overload frame any transmit request will be postponed until the bus becomes idle.

15.2.7 Baud Rate Control Function

(1) Nominal bit time (8 to 25 time quanta)

- Definition of 1 data bit time is as follows.

Figure 15-16: Nominal Bit Time (8 to 25 Time Quanta)



[1 Minimum time for one time/quantum (TQ) = 1/fx]

- Sync segment: In this segment the bit synchronization is performed.
- Prop segment: This segment absorbs delays of the output buffer, the CAN bus and the input buffer. Prop segment time =(output buffer delay) + (CAN bus delay) + (input buffer delay).
- Phase segment 1/2: These segments compensate the data bit time error. The larger the size measured in TQ is, the larger is the tolerable error.
- The synchronization jump width (SJW) specifies the synchronization range. The SJW is programmable. SJW can have less or equal number of TQ as phase segment 2.

Table 15-15: Segment Name and Segment Length

Segment Name	Segment Length (allowed Number of TQs)
Sync segment (Synchronization segment)	1
Prop segment (Propagation segment)	Programmable 1 to 8
Phase segment 1 (Phase buffer segment 1)	Programmable 1 to 8
Phase segment 2 (Phase buffer segment 2)	Maximum of phase segment 1 and the IPT **Note**
SJW	Programmable 1 to 4

Note: IPT = Information Processing Time. It needs to be less than or equal to 2 TQ.

(2) Adjusting synchronization of the data bit

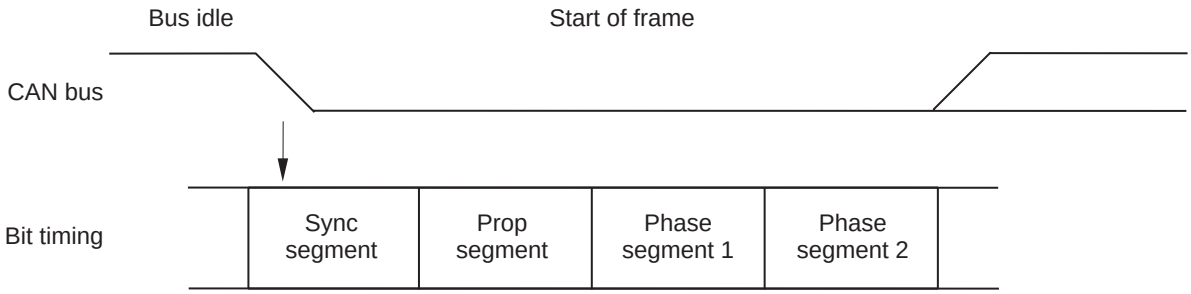
- The transmission node transmits data synchronized to the transmission node bit timing.
- The reception node adjusts synchronization at recessive to dominant edges on the bus. Depending on the protocol this synchronization can be a hard or soft synchronization.

(a) Hard synchronization

This type of synchronization is performed when the reception node detects a start of frame in the bus idle state.

- When the node detects a falling edge of a SOF, the current time quanta becomes the synchronization segment. The length of the following segments are defined by the values programmed into the SYNC0 and SYNC1 registers.

Figure 15-17: Adjusting Synchronization of the Data Bit

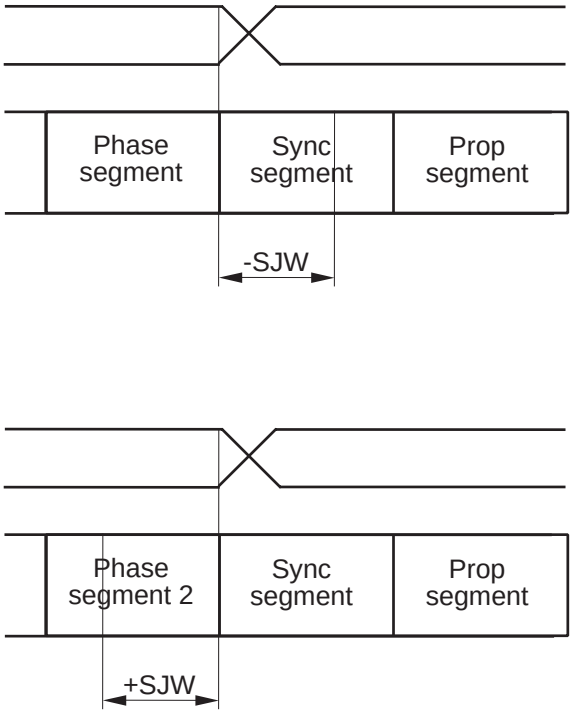


(b) Soft synchronization

When a recessive to dominant level change on the bus is detected, a soft synchronization is performed.

- If the phase error is larger than the programmed SJW value, the node will adjust the timing by applying this SJW-value. Full synchronization is achieved by subsequent adjustments on the next recessive to dominant edge(s).
- These errors that are equal or less of the programmed SJW are corrected instantly and full synchronization is achieved already for the next bit.
- The TQ at which the edge occurs becomes sync segment forcibly, if the phase error is less than or equal to SJW.

Figure 15-18: Bit Synchronization



15.2.8 State Shift Chart

Figure 15-19: Transmission State Shift Chart

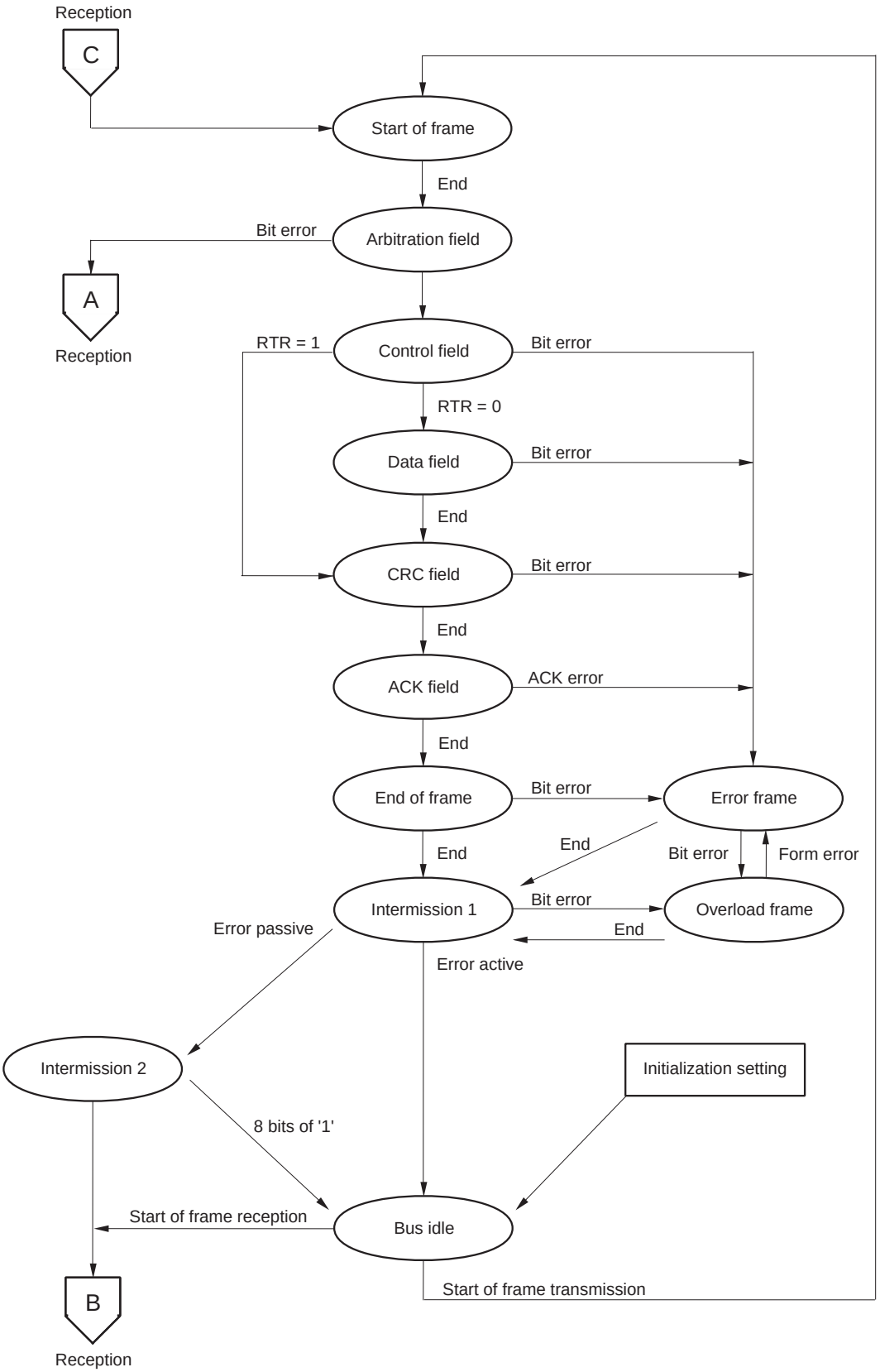


Figure 15-20: Reception State Shift Chart

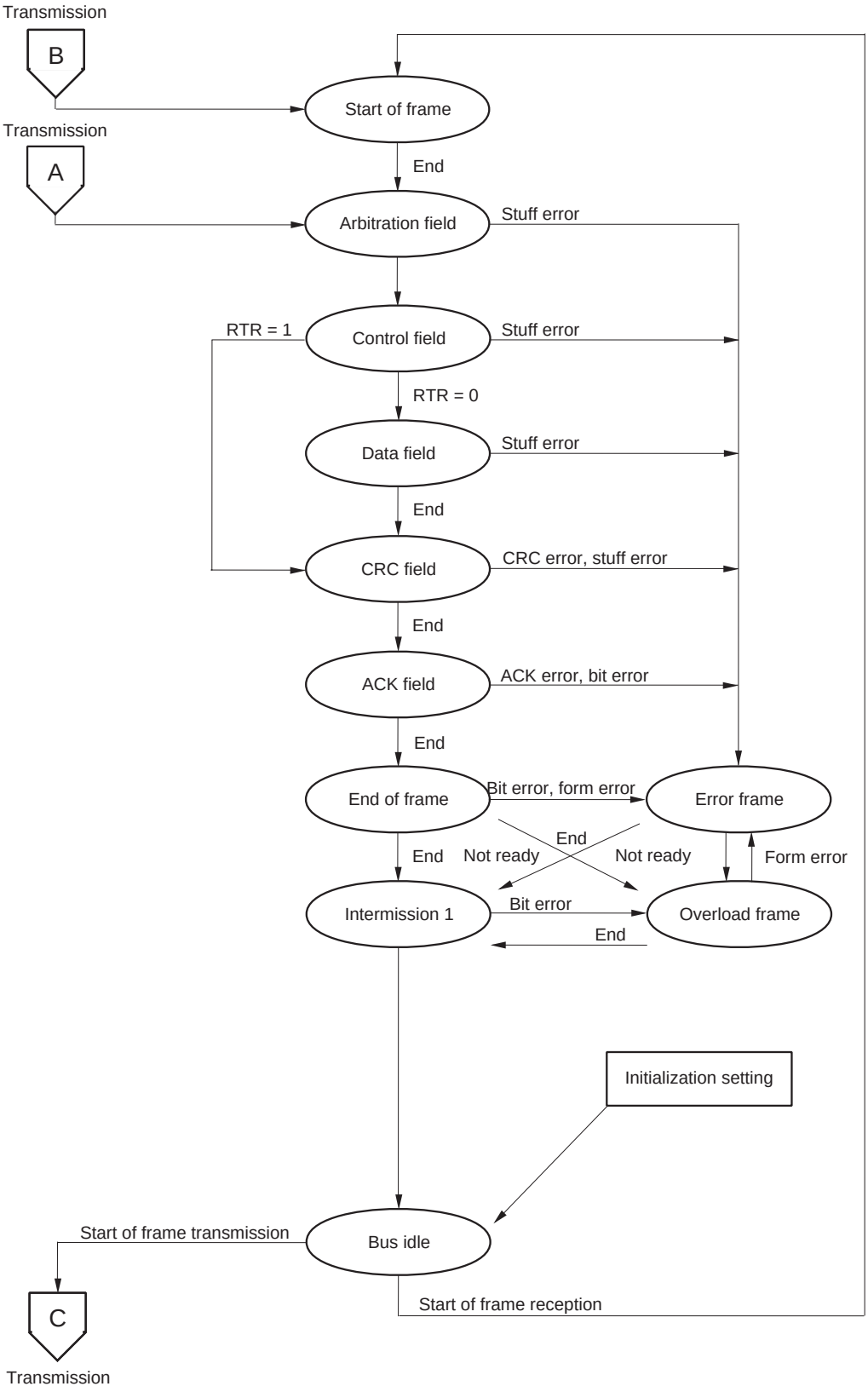
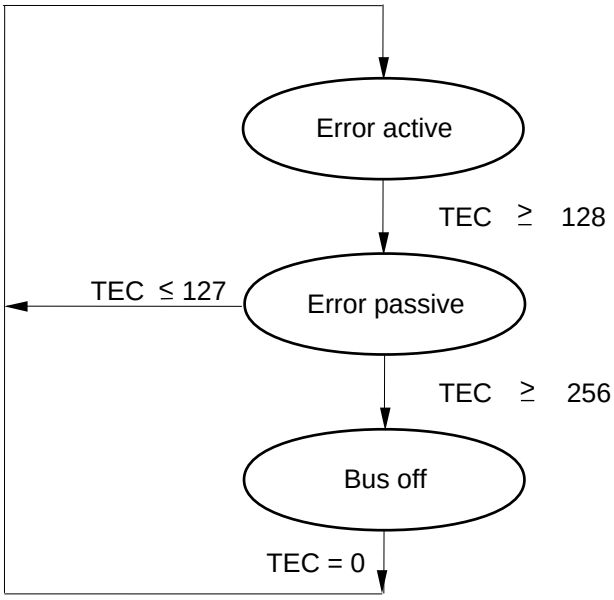


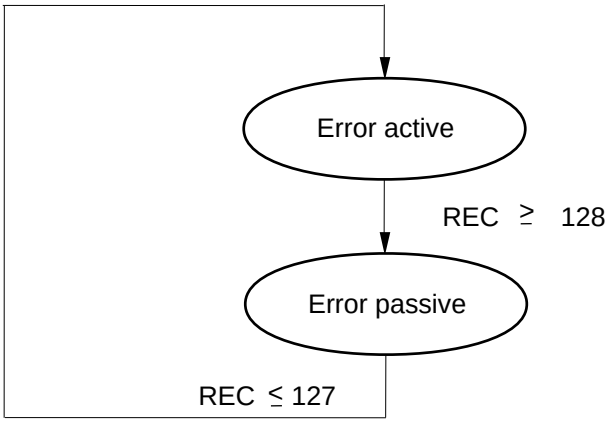
Figure 15-21: Error State Shift Chart

(a) Transmission



TEC = Transmission error counter

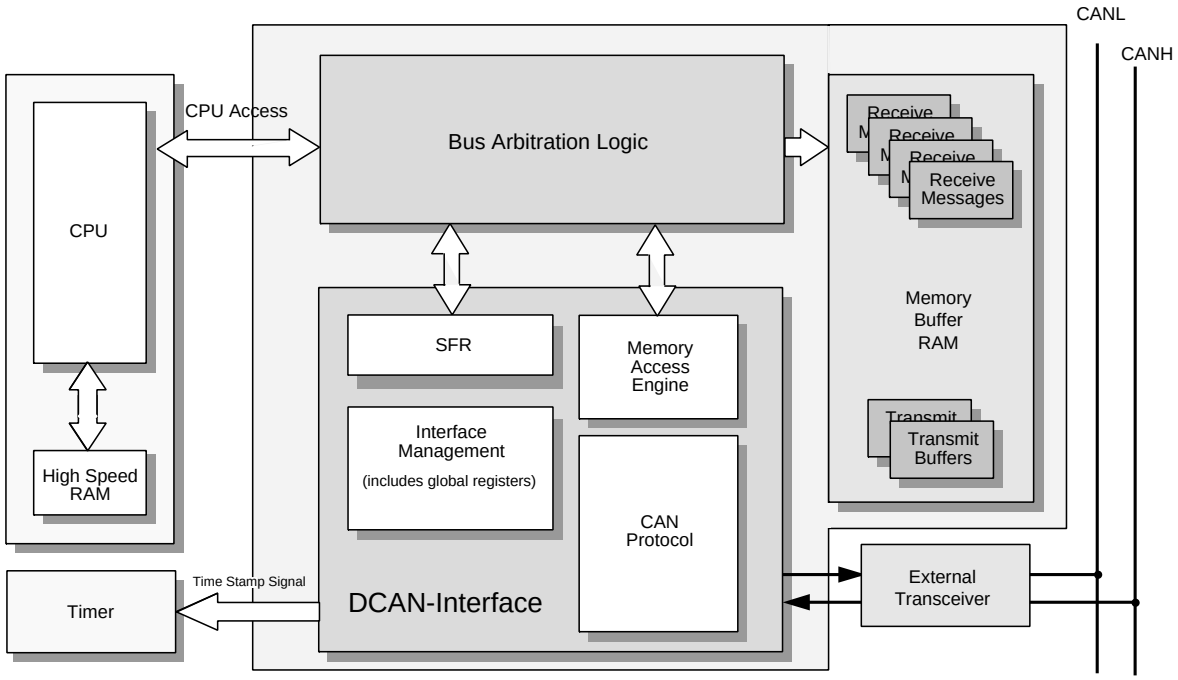
(b) Reception



REC = Reception error counter

15.3 Outline Description

Figure 15-22: Structural Block Diagram

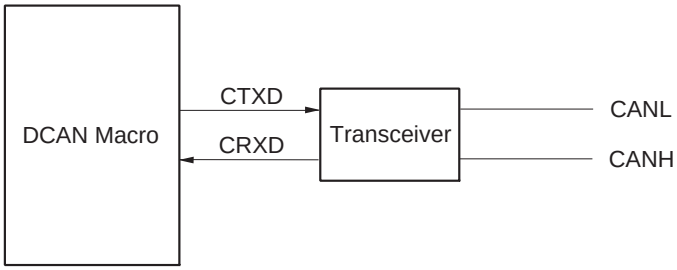


This interface part handles all protocol activities by hardware in the CAN protocol part. The memory access engine fetches information for the CAN protocol transmission from the dedicated RAM area to the CAN protocol part or compares and sorts incoming information and stores it into predefined RAM areas.
The DCAN interfaces directly to the RAM area that is accessible by the DCAN and by the CPU.
The DCAN part works with an external bus transceiver which converts the transmit data and receive data lines to the electrical characteristics of the CAN bus itself.

15.4 Connection with Target System

The DCAN Macro has to be connected to the CAN bus with an external transceiver.

Figure 15-23: Connection to the CAN Bus



15.5 CAN Controller Configuration

The CAN-module consists of the following hardware
.

Table 15-16: CAN Configuration

Item	Configuration
Message definition	In RAM area
CAN input/output	1 (CTXD) 1 (CRXD)
Control registers	DCAN control register (DCANC) CAN control register (CANC) Transmit control register (TCR) Receive message register (RMES) Redefinition control register (REDEF) CAN error status register (CANES) Transmit error counter (TEC) Receive error counter (REC) Message count register (MCNT) Bit rate prescaler (BRPRS) Synchronous control register 0 (SNYC0) Synchronous control register 1 (SYNC1) Mask control register (MASKC)

15.6 Special Function Register for CAN-module

Table 15-17: SFR Definitions

Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
			1-bit	8-bit	16-bit	
DCAN control register	DCANC	R/W	×	×	-	00H
CAN control register	CANC	R/W	×	×	-	01H
Transmit control register	TCR	R/W	-	×	-	00H
Receive message register	RMES	R	-	×	-	00H
Redefinition control register	REDEF	R/W	×	×	-	00H
CAN error status register	CANES	R/W	-	×	-	00H
Transmit error counter	TEC	R	-	×	-	00H
Receive error counter	REC	R	-	×	-	00H
Message count register	MCNT	R	-	×	-	C0H
Bit rate prescaler	BRPRS	R/W	-	×	-	00H
Synchronous control register 0	SYNC0	R/W	-	×	-	18H
Synchronous control register 1	SYNC1	R/W	-	×	-	0EH
Mask control register	MASKC	R/W	-	×	-	00H

The following SFR bits can be accessed with 1-bit instructions. The other SFR registers have to be accessed with 8-bit instructions.

Table 15-18: SFR Bit Definitions

Name	Description	Bit
SOFE	Start of frame enable	CANC.4
SLEEP	Sleep mode	CANC.2
INIT	Initialize	CANC.0
DEF	Redefinition enable	REDEF.7

15.7 Message and Buffer Configuration

Table 15-19: Message and Buffer Configuration

Address Note 2	Register Name	R/W	After Reset
00xH	Transmit buffer 0	R/W	Note 1
01xH	Transmit buffer 1		
02xH	Receive message 0 / Mask 0		
03xH	Receive message 1		
04xH	Receive message 2 / Mask 1		
05xH	Receive message 3		
06xH	Receive message 4		
07xH	Receive message 5		
08xH	Receive message 6		
09xH	Receive message 7		
0AxH	Receive message 8		
0BxH	Receive message 9		
0CxH	Receive message 10		
0DxH	Receive message 11		
0ExH	Receive message 12		
0FxH	Receive message 13		
10xH	Receive message 14		
11xH	Receive message 15		

Notes: 1. Contents is undefined, because data resides in normal RAM area.

2. This address is an offset to the RAM area starting address defined with CADD0/1 in the message count register (MCNT).

15.8 Transmit Buffer Structure

The DCAN has two independent transmit buffers. The two buffers have a 16 byte data structure for standard and extended frames with the ability to send up to 8 data bytes per message. The structure of the transmit buffer is similar to the structure of the receive buffers. The CPU can use addresses that are specified as “unused” in the transmit buffer layout. As well the CPU may use unused ID addresses, unused data addresses[Note], and an unused transmit buffer of the DCAN for its own purposes. The control bits, the identification and the message data have to be stored in the message RAM area.

The transmission control is done by the TCR register. A transmission priority selection allows the customer to realize an application specific priority selection. After the priority selection the transmission can be started by setting the TXRQn bit (n = 0, 1).

In the case that both transmit buffers are used, the transmit priorities can be set. For this purpose the DCAN has the TXP bit in the TCR register. The application software has to set this priority before the transmission is started.

The two transmit buffers supply two independent interrupt lines for an interrupt controller.

Note: Message objects that need less than 8 data byte (DLC < 8) may use the remaining bytes (8 - DLC) for application purposes.

15.9 Transmit Message Format

Table 15-20: Transmit Message Format

Name	Address[Note]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TCON	n0H	IDE	RTR	0	0	DLC3	DLC2	DLC1	DLC0
	n1H	Unused							
IDTX0	n2H	ID standard part							
IDTX1	n3H	ID standard part			0	0	0	0	0
IDTX2	n4H	ID extended part							
IDTX3	n5H	ID extended part							
IDTX4	n6H	ID extended part		0	0	0	0	0	0
	n7H	Unused							
DATA0	n8H	Message data byte 0							
DATA1	n9H	Message data byte 1							
DATA2	nAH	Message data byte 2							
DATA3	nBH	Message data byte 3							
DATA4	nCH	Message data byte 4							
DATA5	nDH	Message data byte 5							
DATA6	nEH	Message data byte 6							
DATA7	nFH	Message data byte 7							

Note: This address is a relative offset to the starting address of the transmit buffer.

(1) Transmit Message Definition

The memory location labelled TCON includes the information of the RTR bit and the bits of the control field of a data or remote frame.

TCON is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets TCON to an undefined value.

Figure 15-24: Transmit Message Definition Bits

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCON	IDE	RTR	0	0	DLC3	DLC2	DLC1	DLC0	xxx0H	undefined	R/W

IDE	Identifier Extension Select
0	Transmit standard frame message; 11 bit identifier
1	Transmit extended frame message; 29 bit identifier

RTR	Remote Transmission Select
0	Transmit data frames
1	Transmit remote frames

DLC3	DLC2	DLC1	DLC0	Data Length Code Selection of Transmit Message
0	0	0	0	0 data bytes
0	0	0	1	1 data bytes
0	0	1	0	2 data bytes
0	0	1	1	3 data bytes
0	1	0	0	4 data bytes
0	1	0	1	5 data bytes
0	1	1	0	6 data bytes
0	1	1	1	7 data bytes
1	0	0	0	8 data bytes
Others than above				**Note**

Remark: The control field describes the format of frame that is generated and its length. The reserved bits of the CAN protocol are always sent in dominant state (0).

Note: The data length code selects the number of bytes which have to be transmitted. Valid entries for the data length code (DLC) are 0 to 8. If a value greater than 8 is selected, 8 bytes are transmitted in the data frame. The Data Length Code is specified in DLC3 through DLC0.

(2) Transmit Identifier Definition

These memory locations set the message identifier in the arbitration field of the CAN protocol.

IDTX0 to IDTX4 register can be set with an 1-bit or an 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets IDTX0 to IDTX4 to an undefined value.

Figure 15-25: Transmit Identifier (IDTX0 to IDTX4) Registers

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDTX0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxx2H	undefined	R/W
IDTX1	ID20	ID19	ID18	0	0	0	0	0	xxx3H	undefined	R/W
IDTX2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	undefined	R/W
IDTX3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	undefined	R/W
IDTX4	ID1	ID0	0	0	0	0	0	0	xxx6H	undefined	R/W

Remark: If a standard frame is defined by the IDE bit in the TCON byte then IDTX0 and IDTX1 are used only. IDTX2 to IDTX4 are free for use by the CPU for application needs.

(3) Transmit Data Definition

These memory locations set the transmit message data of the data field in the CAN frame.

DATA0 to DATA7 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to an undefined value.

Figure 15-26: Transmit Data

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DATA0									xxx8H	undefined	R/W
DATA1									xxx9H	undefined	R/W
DATA2									xxxAH	undefined	R/W
DATA3									xxxBH	undefined	R/W
DATA4									xxxCH	undefined	R/W
DATA5									xxxDH	undefined	R/W
DATA6									xxxEH	undefined	R/W
DATA7									xxxFH	undefined	R/W

Remark: Unused data bytes that are not used by the definition in the DLC bits in the TCON byte are free for use by the CPU for application needs.

15.10 Receive Buffer Structure

The DCAN has up to 16 receive buffers. The number of used buffers is defined by the MCNT register. Unused receive buffers can be used as application RAM for the CPU. The received data is stored directly in this RAM area.

The 16 buffers have a 16 byte data structure for standard and extended frames with a capacity of up to 8 data bytes per message. The structure of the receive buffer is similar to the structure of the transmit buffers. The semaphore bits DN and MUC enable a secure reception detection and data handling. For the first 8 receive message buffers the successful reception is mirrored by the DN-flags in the RMES register.

The receive interrupt request can be enabled or disabled for each used buffer separately.

15.11 Receive Message Format

Table 15-21: Receive Message Format

Name	Address[Note 1]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IDCON	n0H	0	0	0	0	0	ENI	RTR	IDE
DSTAT	n1H	DN	MUC	R1	R0	DLC			
IDREC0	n2H	ID standard part							
IDREC1	n3H	ID standard part			0	0	0	0	RTR_{REC}[Note 2]
IDREC2	n4H	ID extended part							
IDREC3	n5H	ID extended part							
IDREC4	n6H	ID extended part		0	0	0	0	0	0
	n7H	unused							
DATA0	n8H	Message data byte 0							
DATA1	n9H	Message data byte 1							
DATA2	nAH	Message data byte 2							
DATA3	nBH	Message data byte 3							
DATA4	nCH	Message data byte 4							
DATA5	nDH	Message data byte 5							
DATA6	nEH	Message data byte 6							
DATA7	nFH	Message data byte 7							

Notes: 1. This address is a relative offset to the start address of the receive buffer.

2. RTR_{REC} is the received value of the RTR message bit when this buffer is used together with a mask function.
By using the mask function a successfully received identifier overwrites the bytes IDREC0 and IDREC1 for standard frame format and IDREC0 to IDREC4 for extended frame format.

For the RTR_{REC} bit exist two modes:

- RTR bit in the MCON byte of the dedicated mask is set to 0. In this case RTR_{REC} will always be written to 0 together with the update of the IDn bits in IDREC1. The received frame type (data or remote) is defined by the RTR bit in IDCON of the buffer.
- RTR bit in the MCON byte of the dedicated mask is set to 1 (data and remote frames are accepted). In this case the RTR bit in IDCON has no meaning. The received message type passed the mask is shown in RTR_{REC}.

If a buffer is not assigned to a mask function (mask 1, mask 2 or global mask) the bytes IDREC0 to IDREC4 are only read for comparing. During initialization the RTR_{REC} should be defined to 0.

(1) Receive control bits definition

The memory location labelled IDCON defines the kind of frame (data or remote frame with standard or extended format) that is monitored for the associated buffer. Notification by the receive interrupt upon successful reception can be selected for each receive buffer separately.

IDCON can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IDCON to an undefined value.

Figure 15-27: Control bits for Receive Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDCON	0	0	0	0	0	ENI	RTR	IDE	xxx0H	undefined	R/W

IDE	Identifier Extension Select
0	Receive standard frame message; 11-bit identifier
1	Receive extended frame message; 29-bit identifier

RTR	Remote Transmission Select
0	Receive data frames
1	Receive remote frames

ENI	Enable Interrupt on Receive**Note**
0	No interrupt generated
1	Generate receive interrupt after reception of valid message

The control bits define the type of message that is transferred in the associated buffer if this type of message appears on the bus.
This byte will never be written by the DCAN. Only the host CPU can change this byte.

Note: The user has to define with the ENI bit if he wants to set a receive interrupt request when new data is received in this buffer.

(2) Receive status bits definition

The memory location labelled DSTAT sets the receive status bits of the arbitration field of the CAN protocol.

DSTAT can be set with an 1-bit or an 8-bit memory manipulation instruction.
RESET input sets DSTAT to an undefined value.

Figure 15-28: Receive Status Bits (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DSTAT	DN	MUC	R1	R0	DLC3	DLC2	DLC1	DLC0	xxx1H	undefined	R/W

The receive status reflects the current status of a message. It signals whether new data is stored or if the DCAN currently transfers data into this buffer.
In addition the data length of the last transferred data and the reserved bits of the protocol are shown.

DN	Data New
0	No change in data
1	Data changed

The DCAN-module sets DN twice. At first when it starts storing a message from the shadow buffer into the receive buffer and secondly when it finished the operation.

The CPU needs to clear this bit, to signal by itself that it has read the data. During initialization of the receive buffers the DN-bit should also be cleared. Otherwise the CPU gets no information on an update of the buffer after a successful reception.

MUC	Memory Update
0	CAN does not access data part
1	CAN is transferring new data to message buffer

The DCAN-module sets MUC when it starts transferring a message into the buffer and clears the MUC bit when the transfer is finished.

R1	Reserved Bit 1
0	Reserved bit 1 of received message was “0”
1	Reserved bit 1 of received message was “1”

R0	Reserved Bit 0
0	Reserved bit 0 of received message was “0”
1	Reserved bit 0 of received message was “1”

Figure 15-28: Receive Status Bits (2/2)

DLC3	DLC2	DLC1	DLC0	Data Length Code Selection of Receive Message
0	0	0	0	0 data bytes
0	0	0	1	1 data bytes
0	0	1	0	2 data bytes
0	0	1	1	3 data bytes
0	1	0	0	4 data bytes
0	1	0	1	5 data bytes
0	1	1	0	6 data bytes
0	1	1	1	7 data bytes
1	0	0	0	8 data bytes
Others than above				**Note**

DSTAT is written by the DCAN two times during message storage:
At the first access to this buffer DN = 1, MUC = 1, reserved bits and DLC are written.
At the last access to this buffer DN = 1, MUC = 0, reserved bits and DLC are written.

Note: Valid entries for the data length code are 0 to 8. If a value higher than 8 is received, 8 bytes are stored in the message buffer frame together with the data length code received in the DLC of the message.

(3) Receive Identifier Definition

These memory locations define the receive identifier of the arbitration field of the CAN protocol.

IDREC0 to IDREC4 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IDREC0 to IDREC4 to an undefined value.

Figure 15-29: Receive Identifier (IDREC0 to IDREC4) Registers

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDREC0	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	xxx2H	undefined	R/W
IDREC1	ID20	ID19	ID18	0	0	0	0	RTR_{REC}	xxx3H	undefined	R/W
IDREC2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	undefined	R/W
IDREC3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	undefined	R/W
IDREC4	ID1	ID0	0	0	0	0	0	0	xxx6H	undefined	R/W

The identifier of the receive message has to be defined during the initialization of the DCAN.
The DCAN uses this data for the comparison with the identifiers received on the CAN bus. For normal message buffers without mask function this data is only read by the DCAN for comparison. In combination with a mask function this data is overwritten by the received ID that has passed the mask.

The identifier of the receive messages should not be changed without being in the initialization phase or setting the receive buffer to redefinition in the RDEF register, because the change of the contents can happen at the same time when the DCAN uses the data for comparison. This can cause inconsistent data stored in this buffer and also the ID-part can be falsified in case of using mask function.

Remarks: 1. The unused parts of the identifier (IDREC1 bit 4 - 0 always and IDREC4 bit 5 - 0 in case of extended frame reception) may be written by the DCAN to “0”. They are not released for other use by the CPU.

2. RTR_{REC} is the received value of the RTR message bit when this buffer is used together with a mask function.
By using the mask function a successfully received identifier overwrites the IDREC0 and IDREC1 registers for standard frame format and the IDREC0 to IDREC4 registers for extended frame format.
For the RTR_{REC} bit exists two modes:

- RTR bit in the MCON register of the dedicated mask is set to “0”. In this case RTR_{REC} bit will always be written to “0” together with the update of the IDn bits (n = 18 to 20) in IDREC1. The received frame type (data or remote) is defined by the RTR bit in IDCON of the buffer.
- RTR bit in the MCON register of the dedicated mask is set to “1” (data and remote frames are accepted). In this case the RTR bit in IDCON register has no meaning. The received message type passed the mask is shown in RTR_{REC} bit.

If a buffer is not dedicated to a mask function (mask 1, mask 2 or global mask) the IDREC0 to IDREC4 registers are only read for comparing. All receive identifiers should be defined to “0” before the application sets up its specific values.

(4) Receive Message Data Part

These memory locations set the receive message data part of the CAN protocol.

DATA0 to DATA7 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to an undefined value.

Figure 15-30: Receive Data

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DATA0									xxx8H	undefined	R/W
DATA1									xxx9H	undefined	R/W
DATA2									xxxAH	undefined	R/W
DATA3									xxxBH	undefined	R/W
DATA4									xxxCH	undefined	R/W
DATA5									xxxDH	undefined	R/W
DATA6									xxxEH	undefined	R/W
DATA7									xxxFH	undefined	R/W

The DCAN stores received data bytes in this memory area. Only those data bytes which are actually received and match with the identifier are stored in the receive buffer memory area.

If the DLC is less than eight, the DCAN will not write additional bytes exceeding the DLC value up to eight. The DCAN stores a maximum of 8 bytes (according to the CAN protocol rules) even when the received DLC is greater than eight.

15.12 Mask Function

Table 15-22: Mask Function

Name	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MCON	n0H							RTR	
	n1H	Unused							
MREC0	n2H	ID standard part							
MREC1	n3H	ID standard part			0	0	0	0	0
MREC2	n4H	ID extended part							
MREC3	n5H	ID extended part							
MREC4	n6H	ID extended part		0	0	0	0	0	0
	n7H	Unused							
	n8H	Unused							
	n9H	Unused							
	nAH	Unused							
	nBH	Unused							
	nCH	Unused							
	nDH	Unused							
	nEH	Unused							
	nFH	Unused							

Receive message buffer 0 and buffer 2 can be switched for masked operation with the mask control register (MASKC). In this case the message does not hold message identifier and data of the frame. Instead, it holds identifier and RTR mask information for masked compare operations for the next higher message buffer number. In case the global mask is selected, it keeps mask information for all higher message buffer numbers.

A mask does not store any information about identifier length. Therefore the same mask can be used for both types of frames (standard and extended) during global mask operation.

All unused bytes can be used by the CPU for application needs.

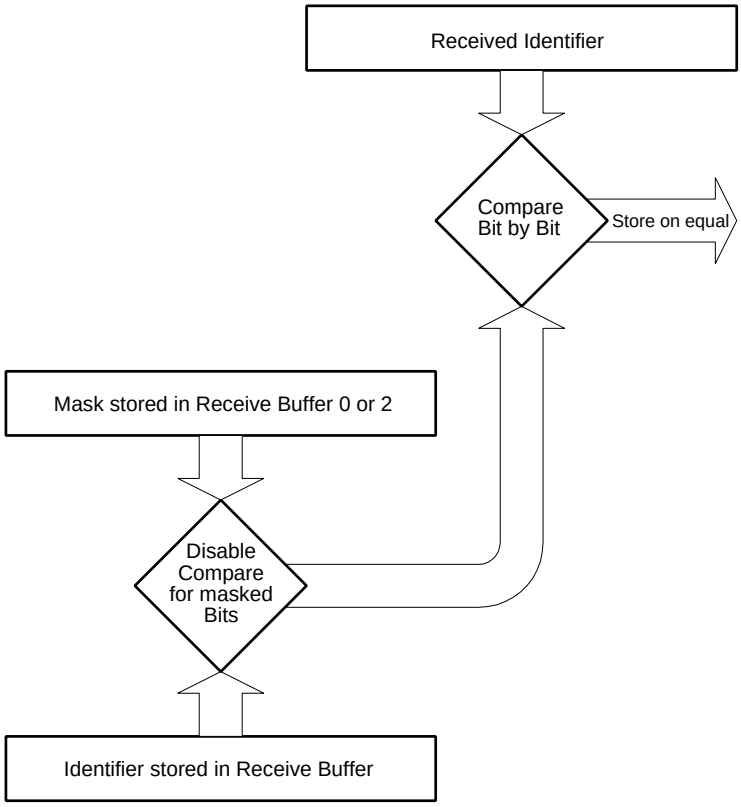
(1) Identifier Compare with Mask

The identifier compare with mask provides the possibility to exclude some bits from the comparison process. That means each bit is ignored when the corresponding bit in the mask definition is set to one.

The setup of the mask control register (MASKC) defines which receive buffer is used as a mask and which receive buffer uses which mask for comparison.

The mask does not include any information about the identifier type to be masked. This has to be defined within the dedicated receive buffer. Therefore a global mask can serve for standard receive buffers at the same time as for extended receive buffer.

Figure 15-31: Identifier Compare with Mask



This function implements the so called basic-CAN behaviour.

In this case the type of identifier is fixed to standard or extended by the setup of the IDE bit in the receive buffer. The comparison of the RTR bit can also be masked. It is possible to receive data and remote frames on the same masked receive buffer.

The following information is stored in the receive buffer:

- Identifier (11 or 29 bit as defined by IDE bit)
- Remote bit (RTR_{REC}) if both frames types (data or remote) can be received by this buffer
- Reserved bits
- Data length code (DLC)
- Data bytes as defined by DLC

Caution: All writes into the DCAN memory are byte accesses. Unused bits in the same byte will be written zero. Unused bytes will not be written and are free for application use by the CPU.

(2) Mask Identifier Control Register (MCON)

The memory location labelled MCON sets the mask identifier control bit of the CAN protocol.

MCON can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MCON to an undefined value.

Figure 15-32: Control Bits for Mask Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MCON	0	0	0	0	0	0	RTR	0	xxx0H	undefined	R/W

RTR	Remote Transmission Select
0	Check RTR bit of received message **Note 1**
1	Receive message independent from RTR bit **Note 2**

Notes: 1. For RTR = 0 the received frame type (data or remote) is defined by the RTR bit in IDCON of the dedicated buffer. In this case RTR_{REC} will always be written to “0” together with the update of the IDn bits (n = 18 to 20) in IDREC1.

2. In case RTR in MCON is set to “1”, RTR bit in IDCON of the dedicated receive buffer has no meaning. The received message type passed the mask is shown in the RTR_{REC} bit.

(3) Mask Identifier Definition

These memory locations set the mask identifier definition of the DCAN.

MREC0 to MREC4 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MREC0 to MREC4 to an undefined value.

Figure 15-33: Mask Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MREC0	MID28	MID27	MID26	MID25	MID24	MID23	MID22	MID21	xxx2H	undefined	R/W
MREC1	MID20	MID19	MID18	0	0	0	0	0	xxx3H	undefined	R/W
MREC2	MID17	MID16	MID15	MID14	MID13	MID12	MID11	MID10	xxx4H	undefined	R/W
MREC3	MID9	MID8	MID7	MID6	MID5	MID4	MID3	MID2	xxx5H	undefined	R/W
MREC4	MID1	MID0	0	0	0	0	0	0	xxx6H	undefined	R/W

MIDn	Mask Identifier Bit (n = 0...28)
0	Check IDn bit in IDREC0 through IDREC4 of received message
1	Receive message independent from IDn bit

15.13 Operation of the CAN Controller

15.13.1 DCAN control register (DCANC)

The DCAN control register (DCANC) is an 8-bit register that enables or disables the operation of the CAN controller.
This register is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DCANC to 00H.

Figure 15-34: DCAN Control Register (DCANC) Format

Symbol	7	6	5	4	3	2	1	<0>	Address	After Reset
DCANC	0	0	0	0	0	0	0	DCANEN	FFB0H	00H
	R	R	R	R	R	R	R	R/W		

DCANEN	DCAN Controller Function
0	Operation disabled
1	Operation enabled

Caution: To operate the CAN controller, set DCANEN to 1 (operation enabled) first (when initializing the µPD780822B Subseries).

15.13.2 CAN control register (CANC)

The operational modes are controlled via the CAN control register CANC.
CANC can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CANC to 01H.

Figure 15-35: CAN Control Register (1/2)

Symbol	7	6	5	<4>	3	<2>	1	<0>	Address	After Reset
CANC	RXF	TXF	0	SOFE	SOFSEL	SLEEP	STOP	INIT	FFB1H	01H
	R	R	R	R/W	R/W	R/W	R/W	R/W		

CANC.5 has always to be written as 0.

INIT	Request status for operational modes
0	Normal operation
1	Initialization mode

The INIT is the request bit to control the DCAN. INIT starts and stops the CAN protocol activities. Due to bus activities disabling the DCAN is not allowed any time. Therefore changing the INIT bit must not have an immediate effect to the CAN protocol activities. Setting the INIT bit is a request only.
The INITSTAT bit in the CANES register reflects if the request has been granted. The registers MCNT, SYNC0, SYNC1, and MASKC are write protected while INIT is cleared independently of INITSTAT. Any write to these registers when INIT is set and the initialisation mode is not confirmed by the INITSTAT bit can have unexpected behaviour to the CAN bus.

STOP	Stop Mode Selection
0	Normal sleep operation / Sleep mode is released when a transition on the CAN bus is detected
1	Stop operation / Sleep mode is cancelled only by CPU access. No wake up from CAN bus

SLEEP	Sleep/Stop Request for CAN protocol
0	Normal operation
1	CAN protocol goes to sleep or stop mode depending on STOP bit

Figure 15-35: CAN Control Register (2/2)

The clock supply to the DCAN is switched off during initialization, DCAN Sleep, and DCAN Stop mode. All modes are only accepted while CAN protocol is in idle state, whereby the CRXD pin must be recessive (= high level). A sleep or stop request out of idle state is rejected and the WAKE bit in CANES is set. DCAN Sleep and DCAN Stop mode can be requested in the same manner. The only difference is that the DCAN Stop mode prevents the wake up by CAN bus activity.

Caution: The DCAN Sleep or DCAN Stop mode can not be requested as long as the WAKE bit in CANES is set.

The DCAN Sleep mode is cancelled under following conditions:

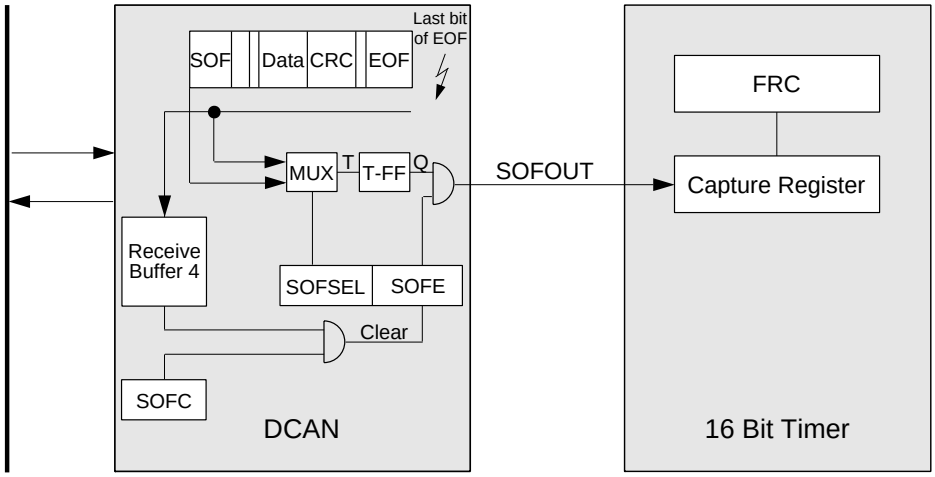
a) CPU clears the SLEEP bit.
b) Any transition while idle state on CAN bus (STOP = 0).
c) CPU sets SLEEP, but CAN protocol is active due to bus activity.

The WAKE bit in CANES is set under condition b) and c).

SOFSEL	Start of Frame Output Function Select
0	Last bit of EOF is used to generate the time stamp
1	SOF is used to generate the time stamp

SOFE	Start of Frame Enable
0	SOFOUT does not change
1	SOFOUT toggles depending on the selected mode

Figure 15-36: DCAN Support



The generation of an SOFOUT signal can be used for time measurements and for global time base synchronization of different CAN nodes as a prerequisite for time triggered communication.

Table 15-23: Possible Setup of the SOFOUT Function

SOFSEL	SOFC	SOFE	SOFOUT Function
x	x	0	Time stamp function disabled
0	x	1	Toggles with each EOF
1	0	1	Toggles with each start of frame on the CAN Bus
1	1	1	Toggles with each start of frame on the CAN bus. Clears SOFE bit when DCAN starts to store a message in receive buffer 4

SOFC is located in the synchronization register SYNC1.

$\overline{\text{RESET}}$ and setting of the INIT bit of CANC register clears the SOFOUT to 0.

Table 15-24: Transmission / Reception Flag

TXF	Transmission Flag
0	No transmission
1	Transmission active on CAN bus

RXF	Reception Flag
0	No data on the CAN bus
1	Reception active on the CAN bus

The TXF and RXF bits of CANC register show the present status of the DCAN macro. This is not the status of the CAN bus. If both bits are cleared, the DCAN macro is in idle state.
RXF and TXF bits are read-only bits. During initialization mode both bits are undefined.

Figure 15-37: Time Stamp Function

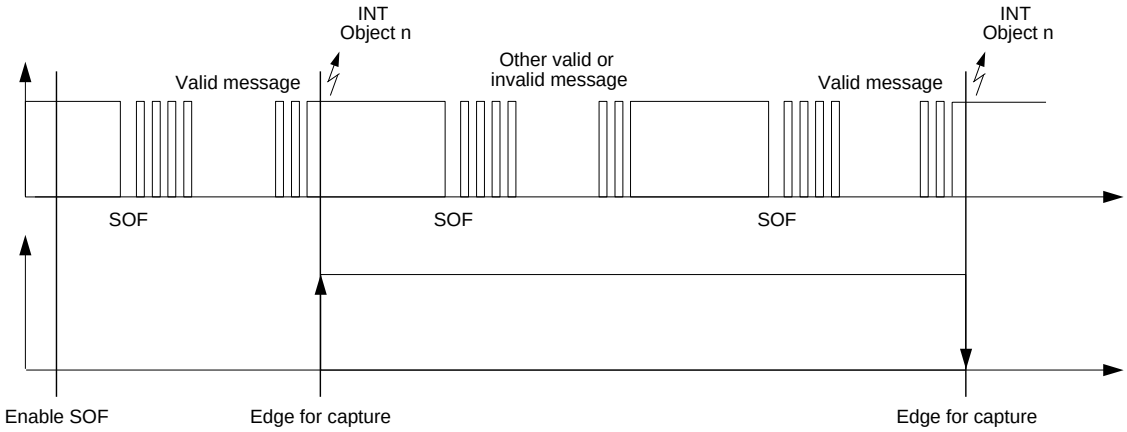


Figure 15-38: SOFOUT Toggle Function

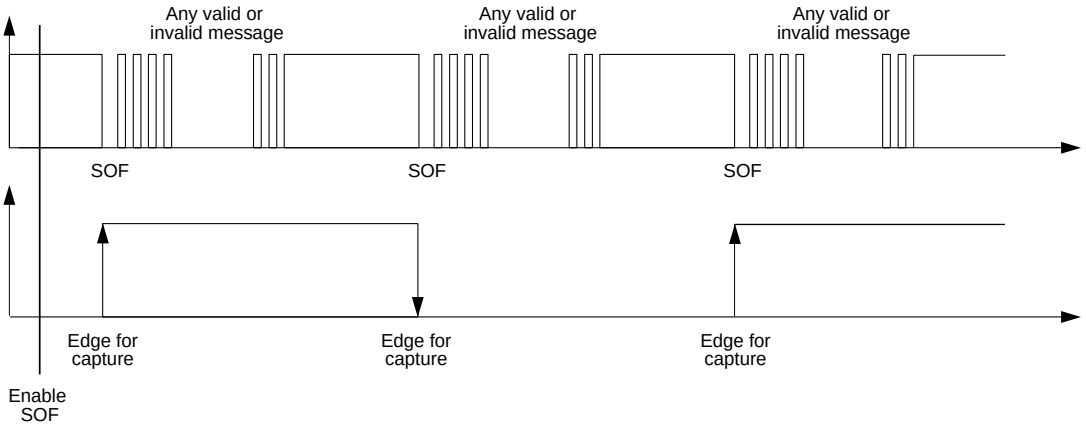
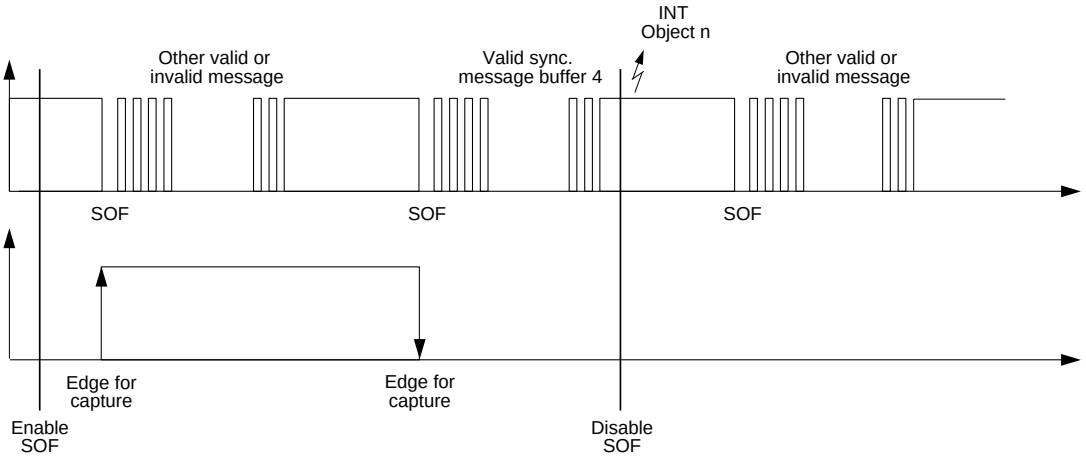


Figure 15-39: Global Time System Function



15.13.3 DCAN Error Status Register (CANES)

This register shows the status of the DCAN.

CANES has to be set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CANES to 00H.
The RESET sets the INIT-bit in CANC register, therefore CANES will be read as 08H after RESET release.

Figure 15-40: CAN Error Status Register (1/3)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
CANES	BOFF	RECS	TECS	0	INITSTATE	VALID	WAKE	OVER	FFB5H	00H
	R	R	R	R	R	R/W	R/W	R/W		

Remark: BOFF, RECS, TECS and INITSTATE are read only bits.

Caution: Don't use bit operations on this SFR.
The VALID, WAKE and OVER bits have a special behavior during CPU write operations:

- **Writing a "0" to them do not change them.**
- **Writing an "1" clears the associated bit.**

This avoids any timing conflicts between CPU access and internal activities. An internal set condition of a bit overrides a CPU clear request at the same time.

BOFF	Bus Off Flag
0	Transmission error counter ≤ 255
1	Transmission error counter > 255

BOFF is cleared after receiving 128 x 11 bits recessive state (Bus idle) or by issuing a hard DCAN reset with the TLRES bit in the MCNTn register **Note**.
An interrupt is generated when the BOFF bit changes its value.

RECS	Reception error counter status
0	Reception error counter < 96
1	Reception error counter ≥ 96 / Warning level for error passive reached

RECS is updated after each reception.
An interrupt is generated when RECS changes its value.

Note: Issuing TLRES bit may violate the minimum recovery time as defined in ISO-11898.

Figure 15-40: CAN Error Status Register (2/3)

TECS	Transmission error counter status
0	Transmission error counter < 96
1	Transmission error counter ≥ 96 / Warning level for error passive reached

TECS is updated after each reception.
An interrupt is generated when TECS changes its value.

INITSTATE	Operational status of the DCAN
0	CAN is in normal operation
1	CAN is stopped and ready to accept new configuration data

INITSTATE changes with a delay to the INIT bit in CANC register. The delay depends on the current bus activity and the time to set all internal activities to inactive state. This time can be several bit times long. While BOFF bit is set, a request to go into the initialization mode by setting the INIT bit is ignored. In this case the INITSTATE bit will not be set until the Bus-off state is left.

VALID	Valid protocol activity detected
0	No valid message detected by the CAN protocol
1	Error free message reception from CAN bus

This bit shows valid protocol activities independent from the message definitions and the RXONLY bit setting in SYNC1n register. VALID is updated after each reception. The VALID bit will be set at the end of the frame when a complete protocol without errors has been detected.

Cautions: 1. The VALID bit is cleared if CPU writes an “1” to it, or when the INIT bit in CANC register is set.

2. Writing a “0” to the valid bit has no influence.

Figure 15-40: CAN Error Status Register (3/3)

WAKE	Wake up Condition
0	Normal operation
1	Sleep mode has been cancelled or sleep/stop mode request was not granted

This bit is set and an error interrupt is generated under the following circumstances:

a) A CAN bus activity occurs during DCAN Sleep mode.

b) Any attempt to set the SLEEP bit in the CAN control register during receive or transmit operation will immediately set the WAKE bit.

The CPU must clear this bit after recognition in order to receive further error interrupts, because the error interrupt line is kept active as long as this bit is set.

Cautions: 1. The WAKE bit is cleared to “0” if CPU writes an “1” to it, or when the INIT bit in CANC register is set.

2. Writing a “0” to the WAKE bit has no influence.

OVER	Overrun Condition
0	Normal operation
1	Overrun occurred during access to RAM

The overrun condition is set whenever the CAN can not perform all RAM accesses that are necessary for comparing and storing received data or fetching transmitted data. Typically, the overrun condition is encountered when the frequency for the macro is too low compared to the programmed baud rate. An error interrupt is generated at the same time.
The DCAN interface will work properly (i. e. no overrun condition will occur) with the following settings: The DCAN clock as defined with the PRM bits in the BRPRS register is set to a minimum of 16 times of the CAN baudrate **and** the selected CPU clock (defined in the PCC register) is set to a minimum of 16 times of the baudrate.

Possible reasons for an overrun condition are:

- Too many messages are defined.
- DMA access to RAM area is too slow compared to the CAN Baudrate.

The possible reactions of the DCAN differ depending on the situation, when the overrun occurs.

Table 15-25: Possible Reactions of the DCAN

Overrun Situation	When detected	DCAN Behavior
Cannot get transmit data.	Next data byte request from protocol. Immediate during the frame.	The frame itself conforms to the CAN specification, but its content is faulty. Corrupted data or ID in the frame. TXRQx bit (x = 0, 1) is not cleared. DCAN will retransmit the correct frame after synchronization to the bus.
Cannot store receive data.	Data storage is ongoing during the six bit of the next frame.	Data in RAM is inconsistent. No receive flags. DN and MUC bit may be set in message.
Cannot get data for ID comparison	ID compare is ongoing during six bits of next frame.	Message is not received and its data is lost.

15.13.4 CAN Transmit Error Counter (TEC)

This register shows the transmit error counter.
TEC register can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets TEC to 00H.

Figure 15-41: Transmit Error Counter

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
TEC	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	FFB6H	00H
	R	R	R	R	R	R	R	R		

The transmit error counter reflects the status of the error counter for transmission errors as it is defined in the CAN protocol according ISO 11898.

15.13.5 CAN Receive Error Counter (REC)

This register shows the receive error counter.
REC can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets REC to 00H.

Figure 15-42: Receive Error Counter

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
REC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	FFB7H	00H
	R	R	R	R	R	R	R	R		

The receive error counter reflects the status of the error counter for reception errors as it is defined in the CAN protocol according ISO 11898.

15.13.6 Message Count Register

This register sets the number of receive message buffers and allocates the RAM area of the receive message buffers, which are handled by the DCAN-module.
MCNT can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MCNT to C0H.

Figure 15-43: Message Count Register (MCNT) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
MCNT	CADD1	CADD0	TLRES	MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	FFB8H	C0H
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		

This register is readable at any time.
Write is only permitted when the CAN is in initialization mode.

MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	Receive Message Count
0	0	0	0	0	Setting prohibited
0	0	0	0	1	1 receive buffer
0	0	0	1	0	2 receive buffer
0	0	0	1	1	3 receive buffer
0	0	1	0	0	4 receive buffer
0	0	1	0	1	5 receive buffer
0	0	1	1	0	6 receive buffer
0	0	1	1	1	7 receive buffer
0	1	0	0	0	8 receive buffer
0	1	0	0	1	9 receive buffer
0	1	0	1	0	10 receive buffer
0	1	0	1	1	11 receive buffer
0	1	1	0	0	12 receive buffer
0	1	1	0	1	13 receive buffer
0	1	1	1	0	14 receive buffer
0	1	1	1	1	15 receive buffer
1	0	0	0	0	16 receive buffer
1	x	x	x	x	Setting prohibited, will be automatically changed to 16

Figure 15-43: Message Count Register (MCNT) (2/2)

TLRES	Reset function for CAN Protocol Machine
0	No Reset is issued
1	Reset of CAN protocol machine is issued if DCAN is in bus off state, DCAN will enter INIT state (CANC.0 = 1 && CANES.3 = 1)

Cautions: 1. Issuing TLRES bit may violate the minimum recovery time as defined in ISO-11898.

2. If no receive buffer is desired, define one receive buffer and disable this buffer with the REDEF function.

CADD1	CADD0	DCAN Address definition
0	0	Setting prohibited
0	1	
1	0	
1	1	F900H to FA1FH (reset value)

15.14 Baudrate Generation

(1) Bit Rate Prescaler Register (BRPRS)

This register sets the clock for the DCAN (internal DCAN clock) and the number of clocks per time quantum (TQ).

BRPRS can be set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets BRPRS to 3FH.

Figure 15-44: Bit Rate Prescaler (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
BRPRS	PRM1	PRM0	BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	FFB9H	00H
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		

The PRMn (n = 0, 1) bits define the clock source for the DCAN operation.
The PRM selector defines the input clock to the DCAN Macro and influences therefore all DCAN activities.

Writing to the BRPRS register is only allowed during initialization mode. Any write to this register when INIT bit is set in CANC register and the initialization mode is not confirmed by the INITSTATE bit of CANES register can cause unexpected behaviour to the CAN bus.

PRM1	PRM0	Input Clock Selector for DCAN Clock
0	0	f_{PRS} is input for DCAN
0	1	$f_{PRS}/2$ is input for DCAN
1	0	$f_{PRS}/4$ is input for DCAN
1	1	CCLK is input for DCAN

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

The BRPRSn bits (n = 0 to 5) define the number of DCAN clocks applied for one TQ.
For BRPRSn (n = 0 to 5) two modes are available depending on the TLMODE bit in the SYNC1 register.

Figure 15-44: Bit Rate Prescaler (2/2)

Setting of BRPRSn (n = 5 to 0) for TLMODE = 0:

BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	Bit Rate Prescaler[Note]
0	0	0	0	0	0	2
0	0	0	0	0	1	4
0	0	0	0	1	0	6
0	0	0	0	1	1	8
.	.	.	.	.	.	.
.	.	.	.	.	.	2 x BRPRSn[5-0] + 2
.	.	.	.	.	.	.
1	1	1	0	1	0	118
1	1	1	0	1	1	120
1	1	1	1	0	0	122
1	1	1	1	0	1	124
1	1	1	1	1	0	126
1	1	1	1	1	1	128

Note: The bit rate prescaler value represents the DCAN clocks per TQ.

Setting of BRPRSn (n = 7 to 0) for TLMODE = 1:

BRPRS7	BRPRS6	BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	Bit Rate Prescaler
0	0	0	0	0	0	0	0	1[Note]
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	3
0	0	0	0	0	0	1	1	4
		.	.	.	.	.	.	.
		.	.	.	.	.	.	BRPRSn[7-0] +1
		.	.	.	.	.	.	.
1	1	1	1	1	0	1	0	123
1	1	1	1	1	0	1	1	124
1	1	1	1	1	1	0	0	125
1	1	1	1	1	1	0	1	126
1	1	1	1	1	1	1	0	127
1	1	1	1	1	1	1	1	128

Note: When using this setting the user needs to assure that phase segment 2 consists of at least 3 TQ. Phase segment 2 is given by the difference of DBT - SPT each measured in units of TQ.

BRPRS7 and BRPRS6 are located in the MASKC register.

(2) Synchronization Control Registers (SYNC0 and SYNC1)

These registers define the CAN bit timing. They define the length of one data bit on the CAN bus, the position of the sample point during the bit timing, and the synchronization jump width. The range of resynchronization can be adapted to different CAN bus speeds or network characteristics. Additionally, some modes related to the baud rate can be selected in SYNC1 register.

SYNC0 and SYNC1 can be read or written with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets SYNC0 to 18H.
$\overline{\text{RESET}}$ input sets SYNC1 to 0EH.

Figure 15-45: Synchronization Control Registers 0 and 1 (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SYNC0	SPT2	SPT1	SPT0	DBT4	DBT3	DBT2	DBT1	DBT0	FFBAH	18H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SYNC1	TLMODE	SOFC	SAMP	0	SJW1	SJW0	SPT4	SPT3	FFBBH	0EH	R/W

The length of a data bit time is programmable via DBT[4-0].

DBT4	DBT3	DBT2	DBT1	DBT0	Data Bit Time
Other than under					Setting prohibited
0	0	1	1	1	8 x TQ
0	1	0	0	0	9 x TQ
0	1	0	0	1	10 x TQ
0	1	0	1	0	11 x TQ
0	1	0	1	1	12 x TQ
0	1	1	0	0	13 x TQ
0	1	1	0	1	14 x TQ
0	1	1	1	0	15 x TQ
0	1	1	1	1	16 x TQ
1	0	0	0	0	17 x TQ
1	0	0	0	1	18 x TQ
1	0	0	1	0	19 x TQ
1	0	0	1	1	20 x TQ
1	0	1	0	0	21 x TQ
1	0	1	0	1	22 x TQ
1	0	1	1	0	23 x TQ
1	0	1	1	1	24 x TQ
1	1	0	0	0	25 x TQ
Other than above					Setting prohibited

Figure 15-45: Synchronization Control Registers 0 and 1 (2/2)

The position of the sample point within the bit timing is defined by SPT0n through SPT4n.

SPT4	SPT3	SPT2	SPT1	SPT0	Sample Point Position
Other than under					Setting prohibited
0	0	0	0	1	2 x TQ
0	0	0	1	0	3 x TQ
0	0	0	1	1	4 x TQ
0	0	1	0	0	5 x TQ
0	0	1	0	1	6 x TQ
0	0	1	1	0	7 x TQ
0	0	1	1	1	8 x TQ
0	1	0	0	0	9 x TQ
0	1	0	0	1	10 x TQ
0	1	0	1	0	11 x TQ
0	1	0	1	1	12 x TQ
0	1	1	0	0	13 x TQ
0	1	1	0	1	14 x TQ
0	1	1	1	0	15 x TQ
0	1	1	1	1	16 x TQ
1	0	0	0	0	17 x TQ
Other than above					Setting prohibited

TLMODE	Resolution of Bit Rate Prescaler
0	1 unit of BRPRS[5-0] in BRPRS register equals 2 DCAN clocks, BRPRS[7-6] in MASKC register are disabled (compatible to older macro versions)
1	1 unit of BRPRS[7-0] in BRPRS and MASKC register equals 2 DCAN clocks, BRPRS[7-6] in MASKC register are enabled**Note**

Note: The user needs to assure that phase segment 2 (TSEG2) consists of at least 3 TQ when using this setting. Phase segment 2 is given by the difference of DBT - SPT each measured in units of TQ.

SJW0 and SJW1 define the synchronization jump width as specified in ISO 11898.

SJW1	SJW0	Synchronisation Jump Width
0	0	1 x TQ
0	1	2 x TQ
1	0	3 x TQ
1	1	4 x TQ

Limits on defining the bit timing

The sample point position needs to be programmed between 3TQ[Note] and 17TQ, which equals a register value of **2** ≤ **SPTxn** ≤ **16** (n = 0, 1; x = 4 to 0).

The number of TQ per bit is restricted to the range from 8TQ to 25TQ, which equals a register value of **7** ≤ **DBTxn** ≤ **24** (n = 0, 1; x = 4 to 0).

The length of phase segment 2 (TSEG2) in TQ is given by the difference of TQ per bit (DBTxn) and the sample point position (SPTxn). Converted to register values the following condition applies:
2 ≤ **DBTxn - SPTxn** ≤ **8** (n = 0, 1; x = 4 to 0).

The number of TQ allocated for soft synchronization must not exceed the number of TQ for phase segment 2, but SJWyn may have as many TQ as phase segment 2:
SJWyn ≤ **DBTxn - SPTxn - 1** (n = 0, 1; x = 4 to 0; y = 0, 1).

Note: Sample point positions of 3 TQ or 4 TQ are for test purposes only. For the minimum number of TQ per bit time, 8TQ, the minimum sample point position is 5 TQ.

Example:

System clock:	$f_X = f_{PRS}$	16 MHz
CAN parameter:	Baud rate	1 MBaud
	Sample Point	75%
	SJW	25%

At first, calculate the overall prescaler value:

$$\frac{f_{PRS}}{\text{Baudrate}} = \frac{f_X}{\text{Baudrate}} = \frac{16\ \text{MHz}}{1\ \text{MBaud}} = 16$$

16 can be split as 1 x 16 or 2 x 8. Other factors can not be mapped to the registers. Only 8 and 16 are valid values for TQ per bit. Therefore the overall prescaler value realized by BRPRSn is 2 or 1 respectively.

With TLMODE = 0 the following register settings apply:

Register value	Description	Bit fields
BRPRSn = 00h	Clock selector = $f_X = f_{PRS}$	PRMn = 00b
		BRPRSx = 000000b
SYNC0n = A7h	CAN Bit in TQ = 8	DBTx = 00111b
	7 < (fx/Baudrate/bit rate prescaler) < 25]	
SYNC1n = 0zzz0100b	sample point 75% = 6 TQ	SPTx = 00101b
	SJW 25% = 2 TQ	SJWy = 01b
	1 TQ equals 2 clocks & BRPRS6, 7 are disabled	TLMODE = 0
	z depends on the setting of: - Number of sampling points - Receive only function - Use of time stamp or global time system	

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency
f_X: X1 input clock oscillation frequency

With TLMODE = 1 the following register settings apply:

Register values	Description	Bit fields
BRPRSn = 00h	Clock selector = $f_X = f_{PRS}$	PRMn = 00b
MASKCn = 00xx xxxxb		BRPRSn = 0000 0000b
SYNC0n = 6Fh	CAN Bit in TQ = 16	DBTn = 01111b
	7 < (fx/Baudrate/bit rate prescaler) < 25]	
SYNC1n = 1zzz 1101b	sample point 75% = 12 TQ:	SPTn = 01011b
	SJW 25% = 4 TQ	SJWn = 11b
	1 TQ equals 1 clock, BRPRS 6, 7 are enabled	TLMODE = 1
	z depends on the setting of: - Number of sampling points - Receive only function - Use of time stamp or global time system	

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency
f_X: X1 input clock oscillation frequency

The VALID bit in CANES reports if the DCAN interface receives any valid message.

SAMP defines the number of sample points per bit as specified in the ISO-11898.

SAMP	Bit Sampling
0	Sample receive data one time at receive point
1	Sample receive data three times and take majority decision at sample point

SOFC works in conjunction with the SOFE and SOFSEL bits in the CAN Control Register CANC. For detailed information please refer to the bit description of that SFR register and the time function mode.

SOFC	Start of Frame Control
0	SOFE bit is independent from CAN bus activities
1	SOFE bit will be cleared when a message for receive message 4 is received and SOF mode is selected

Caution: CPU can read SYNC0/SYNC1 register at any time. Writing to the SYNC0/SYNC1 registers is only allowed during initialization mode. Any write to this register when INIT is set and the initialization mode is not confirmed by the INITSTATE bit can have unexpected behavior to the CAN bus.

15.15 Function Control

15.15.1 Transmit Control

(1) Transmit control register (TCR)

This register controls the transmission of the DCAN-module. The transmit control register (TCR) provides complete control over the two transmit buffers and their status. It is possible to request and abort transmission of both buffers independently.

TCR can be set with a an 8-bit memory manipulation instruction.
RESET input sets TCR to 00H.

Figure 15-46: Transmit Control Register (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
TCR	TXP	0	TXC1	TXC0	TXA1	TXA0	TXRQ1	TXRQ0	FFB2H	00H
	R/W	R	R	R	R/W	R/W	R/W	R/W		

Caution: Don't use bit operations on this register. Also logical operations (read-modify-write) via software may lead to unexpected transmissions. Initiating a transmit request for buffer 1 while TXRQ0 is already set, is simply achieved by writing 02H or 82H. The status of the bits for buffer 0 is not affected by this write operation.

TXP	Transmission Priority
0	Buffer 0 has priority over buffer 1
1	Buffer 1 has priority over buffer 0

The user defines which buffer has to be send first in the case of both request bits are set. If only one buffer is requested by the TXRQn bits (n = 0, 1) bits, TXP bit has no influence.
TXCn (n = 0, 1) shows the status of the first transmission. It is updated when TXRQn (n = 0, 1) is cleared.

TXAn	Transmission Abort Flag
0	Write: normal operation
	Read: no abort pending
1	Write: aborts current transmission request for this buffer n
	Read: abort is pending

TXCn	Transmission Complete Flag
0	Transmit was aborted / no data sent
1	Transmit was complete / abort had no effect

The TXAn bits (n = 0, 1) allow to free a transmit buffer with a pending transmit request. Setting the TXAn bit (n = 0, 1) by the CPU requests the DCAN to empty its buffer by clearing the respective TXRQn bit (n = 0, 1).

Figure 15-46: Transmit Control Register (2/2)

The TXAn bits (n = 0, 1) have a dual function:

1. The CPU can request an abort by writing a “1” into the bit.

2. The DCAN signals whether such an request is still pending. The bit is cleared at the same time when the TXRQn bit (n = 0, 1) is cleared.

The abort process does not affect any rules of the CAN protocol. A frame already started will continue to its end.

An abort operation can cause different results dependent on the time it is issued.

d) When an abort request is recognized by the DCAN before the start of the arbitration for transmit, the TXCn bit (n = 0, 1) is reset showing that the buffer was not send to other nodes.
e) When the abort request is recognized during the arbitration and the arbitration is lost afterwards, the TXCn bit (n = 0, 1) is reset showing that the buffer was not send to other nodes.
f) When the abort request is recognized during frame transmission and the transmission ends with an error afterwards, the TXCn bit (n = 0, 1) is reset showing that the buffer was not send to other nodes.
g) When the abort request is recognized during the frame transmission and transmission ends without error. The TXCn bit (n = 0, 1) is set showing a successful transfer of the data. I.e the abort request was not issued.

In all cases the TXRQn bit and the TXAn bit (n = 0, 1) bit will be cleared at the end of the abort operation, when the transmit buffer is available again.

Cautions: 1. The bits are cleared when the INIT bit in CANC register is set.

2. Writing a 0 to TXAn (n = 0, 1) bit has no influence

3. Do not perform read-modify-write operations on TCR.

The TXCn bit (n = 0, 1) are updated at the end of every frame transmission or abort.

TXRQn	Transmission Request Flag
0	Write: no influence
	Read: transmit buffer is free
1	Write: request transmission for buffer n
	Read: transmit buffer is occupied by former transmit request

The transmit request bits are checked by the DCAN immediately before the frame is started. The order in which the TXRQn bit (n = 0, 1) will be set does not matter as long as the first requested frame is not started on the bus.
The TXRQn bit (n = 0, 1) have dual function:

- 1. Request the transmission of a transmit buffer.
- 2. Inform the CPU whether a buffer is available or if it is still occupied by a former transmit request.

Setting the transmission request bit requests the DCAN to sent the buffer contents onto the bus. The DCAN clears the bit after completion of the transmission. Completion is either a normal transfer without error or an abort request.

An error during the transmission does not influence the transmit request status. The DCAN will automatically retry the transfer.

Cautions: 1. The bits are cleared when the INIT bit in CANC is set. A transmission already started will be finished but not retransmitted in case of an error.

2. Writing a 0 to TXRQ0 bit has no influence.

3. Do not use bit operations on this register.

4. Do not change data in transmit buffer when the corresponding TXRQ bit is set.

15.15.2 Receive Control

The receive message register mirrors the current status of the first 8 receive buffers. Each buffer has one status bit in this register. This bit is always set when a new message is completely stored out of the shadow buffer into the associated buffer. The CPU can easily find the last received message during receive interrupt handling. The bits in this register always correspond to the DN bit in the data buffers. They are cleared when the CPU clears the DN bit in the data buffer. The register itself is read only.

(1) Receive message register (RMES)

This register shows receptions of messages of the DCAN-module. More than one bit set is possible.

RMES can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets RMES to 00H.

Figure 15-47: Receive Message Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
RMES	DN7	DN6	DN5	DN4	DN3	DN2	DN1	DN0	FFB3H	00H
	R	R	R	R	R	R	R	R		

This register is read only and it is cleared when the INIT bit in CANC register is set.

DN	Data New Bit for Message n (n = 0...7)
0	No message received on message n or CPU has cleared DN bit in message n
1	Data received in message n that was not acknowledged by the CPU

DN0 bit has no meaning when receive buffer 0 is configured for mask operation in the mask control register.
DN2 bit has no meaning when receive buffer 2 is configured for mask operation in the mask control register.

15.15.3 Mask Control

The mask control register defines whether the DCAN compares all identifier bits or if some bits are not used for comparison. This functionality is provided by the use of the mask information. The mask information defines for each bit of the identifier whether it is used for comparison or not. The DCAN uses a receive buffer for this information, when it is enabled by the mask control register. In this case this buffer is not used for normal message storage. Unused bytes can be used for application needs.

(1) Mask control register (MASKC)

This register controls the mask function applied to any received message.

MASKC can be written with an 8-bit memory manipulation instruction.
RESET input sets MASKC to 00H.

Figure 15-48: Mask Control Register (1/2)

Symbol	7 Note	6 Note	5	4	3	2	1	0	Address	After Reset
MASKC	BRPRS7	BRPRS6	SSHT	AL	0	GLOBAL	MSK1	MSK0	FFBCH	00H
	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W		

Note: BRPRS[7 - 6] are only enable if TLMODE is set to 1.

Caution: This register is readable at any time. Writing to the MASKC register is only allowed during initialization mode. Any write to this register when INIT bit is set and the initialization mode is not confirmed by the INITSTATE bit can have unexpected behavior to the CAN bus.

MSK0	Mask 0 Enable
0	Receive buffer 0 and 1 in normal operation
1	Receive buffer 0 is mask for buffer 1

MSK1	Mask 1 Enable
0	Receive buffer 2 and 3 in normal operation
1	Receive buffer 2 is mask for buffer 3

GLOBAL	Enable Global Mask
0	Normal operation
1	Highest defined mask is active for all following buffers

Figure 15-48: Mask Control Register (2/2)

SSHT	AL	Function
0	x	Single shot mode disabled
1	0	Single shot mode enabled; no re-transmission when an error occurs. Transmit message will not be queued for a second transmit request when the arbitration was lost
1	1	Single shot mode enabled; no re-transmission when an error occurs. Transmit message will be queued for a second transmit request when the arbitration was lost.

BRPRS7	BRPRS6	Prescaler values
0	0	Selects 0 - 64 DCAN clocks per time quanta
0	1	Selects 65 - 128 DCAN clocks per time quanta
1	0	Selects 129 - 192 DCAN clocks per time quanta
1	1	Selects 193 - 256 DCAN clocks per time quanta

The following table shows which compare takes place for the different receive buffers. The ID in this table always represents the ID stored in the mentioned receive buffer. The table also shows which buffers are used to provide the mask information and therefore do not receive messages. A global mask can be used for standard and extended frames at the same time. The frame type is only controlled by the IDE bit of the receiving buffer.

Table 15-26: Mask Operation Buffers

GLOBAL	MSK1	MSK0	Receive Buffer					Operation
			0	1	2	3	4-15	
X	0	0	Compare ID	Compare ID	Compare ID	Compare ID	Compare ID	Normal
0	0	1	Mask0	Compare ID & mask0	Compare ID	Compare ID	Compare ID	One mask
0	1	0	Compare ID	Compare ID	Mask1	Compare ID & mask1	Compare ID	One mask
0	1	1	Mask0	Compare ID & mask0	Mask1	Compare ID & mask1	Compare ID	Two masks
1	0	1	Mask0	Compare ID & mask0	Compare ID & mask0	Compare ID	& mask0	Global mask
1	1	0	Compare ID	Compare ID	Mask1	Compare ID	& mask1	Two normal, rest global mask
1	1	1	Mask0	Compare ID & mask0	Mask1	Compare ID	& mask1	One mask, rest global mask

Priority of receive buffers during compare

It is possible that more than one receive buffer is configured to receive a particular message. For this case an arbitrary rule for the storage of the message into one of several matching receive buffers becomes effective. The priority of a receive buffers depends on its type defined by the setup of the mask register in first place and its number in second place.

The rules for priority are:

- All non-masked receive buffers have a higher priority than the masked receive buffer.
- Lower numbered receive buffers have higher priority.

Examples:

1. All RX buffers are enabled to receive the same standard identifier 0x7FFH. Result: the message with identifier 0x7FFH is stored in RX0.
2. In difference to the previous set up, the mask option is set for RX2. Again the message 0x7FFH is stored in buffer in RX0.
3. If additionally RX0 is configured as a mask, the message will be stored in RX4.

15.15.4 Special Functions

(1) Redefinition control register (REDEF)

This register controls the redefinition of an identifier of a received buffer.

REDEF can be written with an 1-bit or an 8-bit memory manipulation instruction.
RESET input sets REDEF to 00H.

Figure 15-49: Redefinition Control Register (1/2)

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset
REDEF	DEF	0	0	0	SEL3	SEL2	SEL1	SEL0	FFB4H	00H
	R/W	R	R	R	R/W	R/W	R/W	R/W		

The redefinition register provides a way to change identifiers and other control information for one receive buffer, without disturbing the operation of the other buffers.

DEF	Redefine Permission Bit
0	Normal operation
1	Receive operation for selected message is disabled. CPU can change definition data for this message.

This bit is cleared when INIT bit in CANC is set.

Figure 15-49: Redefinition Control Register (2/2)

SEL3	SEL2	SEL1	SEL0	Buffer selection (n =0...15)
0	0	0	0	Buffer 0 is selected for redefinition
0	0	0	1	Buffer 1 is selected for redefinition
0	0	1	0	Buffer 2 is selected for redefinition
0	0	1	1	Buffer 3 is selected for redefinition
0	1	0	0	Buffer 4 is selected for redefinition
0	1	0	1	Buffer 5 is selected for redefinition
0	1	1	0	Buffer 6 is selected for redefinition
0	1	1	1	Buffer 7 is selected for redefinition
1	0	0	0	Buffer 8 is selected for redefinition
1	0	0	1	Buffer 9 is selected for redefinition
1	0	1	0	Buffer 10 is selected for redefinition
1	0	1	1	Buffer 11 is selected for redefinition
1	1	0	0	Buffer 12 is selected for redefinition
1	1	0	1	Buffer 13 is selected for redefinition
1	1	1	0	Buffer 14 is selected for redefinition
1	1	1	1	Buffer 15 is selected for redefinition
Other than above				Setting prohibited

Cautions: 1. Keep special programming sequence. Failing to do so can cause inconsistent data or loss of receive data.

2. Do not change DEF bit and SEL bit at the same time. Change SEL bit only when DEF bit is cleared.

3. Write first SEL with DEF cleared. Write than SEL with DEF, or use bit manipulation instruction. Only clear DEF bit by keeping SEL or use bit manipulation instruction.

Setting the redefinition bit removes the selected receive buffer from the list of possible ID hits during identifier comparisons.

Setting the DEF bit will not have immediate effect, if DCAN is preparing to store or is already in progress of storing a received message into the particular buffer. In this case the redefinition request is ignored for the currently processed message.

The application should monitor the DN flag before requesting the redefinition state for a particular buffer. A DN flag set indicates a new message that arrived or a new message that is in progress of being stored to that buffer. The application should be prepared to receive a message immediately after redefinition state was set. The user can identify this situation because the data new bit (DN) in the receive buffer will be set. This is of special importance if it is used together with a mask function because in this case the DCAN also writes the identifier part of the message to the receive buffer. Then the application needs to re-write the configuration of the message buffer.

15.16 Interrupt Information

15.16.1 Interrupt Vectors

The DCAN peripheral supports four interrupt sources as shown in the following table.

Table 15-27: Interrupt Sources

Function	Source	Interrupt Flag
Error	Error counter Overrun error Wake up	CEIF
Receive	Received frame is valid	CRIF
Transmit buffer 0	TXRQ0 is cleared	CTIF0
Transmit buffer 1	TXRQ1 is cleared	CTIF1

15.16.2 Transmit Interrupt

The transmit interrupt is generated when all following conditions are fulfilled:

- The transmit interrupt 0 is generated when TXRQ0 bit is cleared.
- The transmit interrupt 1 is generated when TXRQ1 bit is cleared.

Clearing of these bits releases the buffer for writing a new message into it. This event can occur due to a successful transmission or due to an abort of a transmission. Only the DCAN can clear this bit. The CPU can only request to clear the TXRQn bit by setting the ABORTn bit (n = 0, 1).

15.16.3 Receive Interrupt

The receive interrupt is generated when all of the following conditions are fulfilled:

- CAN protocol part marks received frame valid.
- The received frame passes the acceptance filter. In other words, a message buffer with an identifier/mask combination fits to the received frame.
- The memory access engine successfully stored data in the message buffer.
- The message buffer is marked for interrupt generation with ENI bit set.

The memory access engine can delay the interrupt up to the 7th bit of the next frame because of its compare and store operations.

15.16.4 Error Interrupt

The error interrupt is generated when any of the following conditions are fulfilled:

- Transmission error counter (BOFF) changes its state.
- Transmission error counter status (TECS) changes its state.
- Reception error counter status (RECS) changes its state.
- Overrun during RAM access (OVER) becomes active.
- The wake-up condition (WAKE) becomes active.

The wake-up condition activates an internal signal to the interrupt controller. In order to receive further error interrupts generated by other conditions, the CPU needs to clear the WAKE bit in CANES register every time a wake-up condition was recognized.
No further interrupt can be detected by the CPU as long as the WAKE bit is set.

15.17 Influence of the standby Function of the CAN Controller

15.17.1 CPU Halt Mode

The CPU halt mode is possible in conjunction with DCAN Sleep mode.

15.17.2 CPU Stop Mode

The DCAN stops any activity when its clock supply stops due to a CPU Stop mode issued. This may cause an erroneous behaviour on the CAN bus. Entering the CPU Stop Mode is not allowed when the DCAN is in normal mode, i.e. online to the CAN bus.

The DCAN will reach an overrun condition, when it receives clock supply again.

CPU Stop mode is possible when the DCAN was set to initialization state, sleep mode or stop mode beforehand. Note that the CPU will not be started again if the DCAN Stop mode was entered previously.

15.17.3 DCAN Sleep Mode

The DCAN Sleep mode is intended to lower the power consumption during phases where no communication is required.
The CPU requests the DCAN Sleep mode. The DCAN will signal with the WAKE bit, if the request was granted or if it is not possible to enter the sleep mode due to ongoing bus activities.
After a successful switch to the DCAN Sleep mode, the CPU can safely go into halt, watch or stop mode. However, the application needs to be prepared that the DCAN cancels the sleep mode any time due to bus activities. If the wake-up interrupt is serviced, the CPU Stop mode has not to be issued. Otherwise the CPU will not be released from CPU Stop mode even when there is ongoing bus activity.
The wake-up is independent from the clock. The release time for the CPU Stop mode of the device is of no concern because the DCAN synchronizes again to the CAN bus after clock supply has started.

The following example sketches the general approach on how to enter the DCAN Sleep mode. Note that the function may not return for infinite time when the CAN bus is busy. The user may apply time out controls to avoid excessive run-times.

Code example:

```
DCAN_Sleep_Mode(void){
CANES = 0x02;                   // clear Wake bit
CANC = 0x04                     // request DCAN Sleep mode
while (CANES & 0x02)            // check if DCAN Sleep mode was accepted
        {
        CANES = 0x02;           // try again to get DCAN asleep
        CANC = 0x04;
        }
}
```

The following code example assures a safe transition into CPU Stop mode for all timing scenarios of a suddenly occurring bus activity. The code prevents that the CPU gets stuck with its oscillator stopped despite CAN bus activity.

Code example:

```
........                        //any application code

DCAN_Sleep_Mode;                //request and enter DCAN sleep mode

........                        //any application code

DI();                           //disable interrupts
NOP;Note
NOP;
if (wakeup_interrupt_occurred == FALSE)
                                // the variable wakeup_interrupt occurred
                                // needs to be initialized at system reset
                                // and it needs to be set TRUE when servicing
                                // the wake-up interrupt.
        {
        CPU_STOP;               //enter CPU Stop mode
        }
NOP:Note
NOP:
NOP;
EI();                           // enable interrupts
.........                       // resume with application code
```

Note: The interrupt acknowledge needs some clock cycles (depends on host core). In order to prevent that the variable wakeup_interrupt_occurred is already read before DI(); becomes effective some NOP-instruction have to be inserted. As well the number of NOP-instructions after the CPU Stop instruction is dependent on the host core. The given example is tailored for 78K0.

15.17.4 DCAN Stop Mode

The CPU requests this mode from DCAN. The procedure equals the request for DCAN Sleep mode. The DCAN will signal with the WAKE bit, if the request was granted or if it is not possible to enter the DCAN Stop mode due to ongoing bus activities.

After a successful switch to the DCAN Stop mode, the CPU can safely go into halt, watch or stop mode without any precautions. The DCAN can only be woken up by the CPU. Therefore the CPU needs to clear the SLEEP bit in the CANC register.

This mode reduces the power consumption of the DCAN to a minimum.

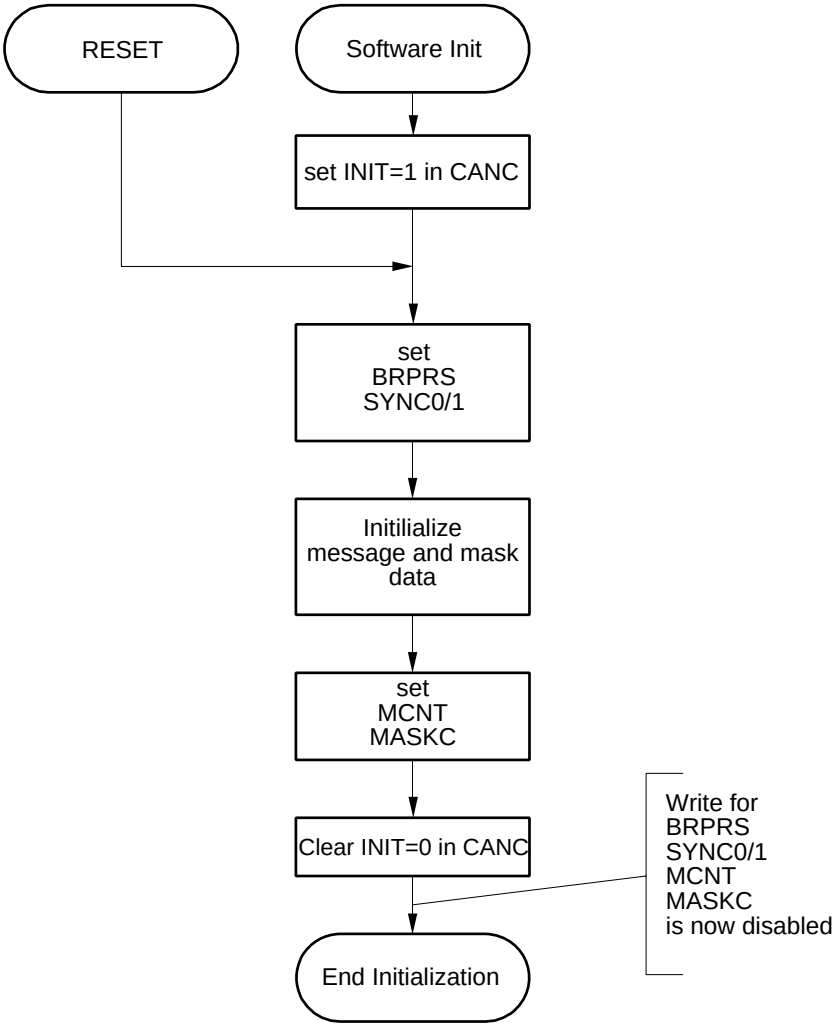
Code example:

```
DCAN_Stop_Mode(void){
CANES = 0x02;                       // clear Wake bit
CANC = 0x06                         // request DCAN Stop mode
while (CANES & 0x02)                // check if DCAN Stop mode was accepted
        {
        CANES = 0x02;               // try again to get DCAN into stop mode
        CANC = 0x06;
        }
}
```

15.18 Functional Description by Flowcharts

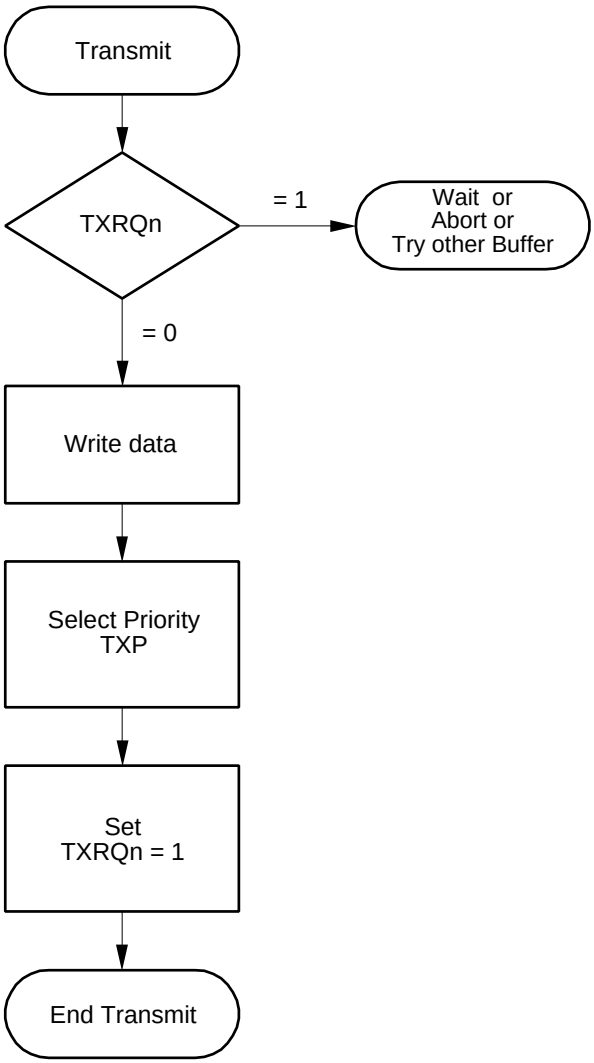
15.18.1 Initialization

Figure 15-50: Initialization Flow Chart



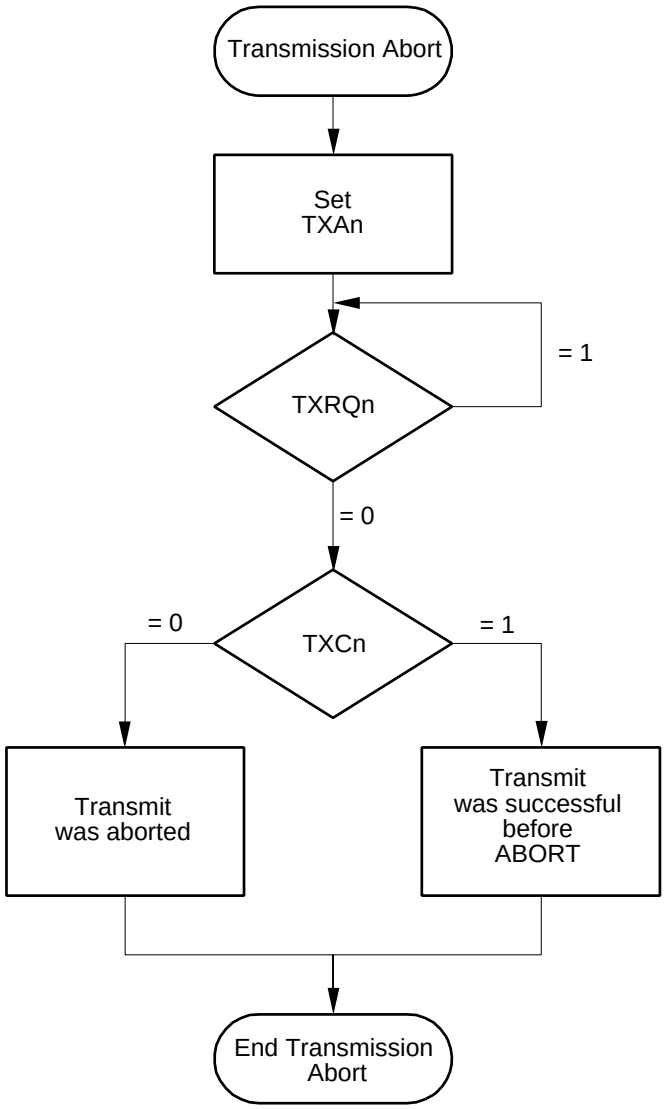
15.18.2 Transmit Preparation

Figure 15-51: Transmit Preparation



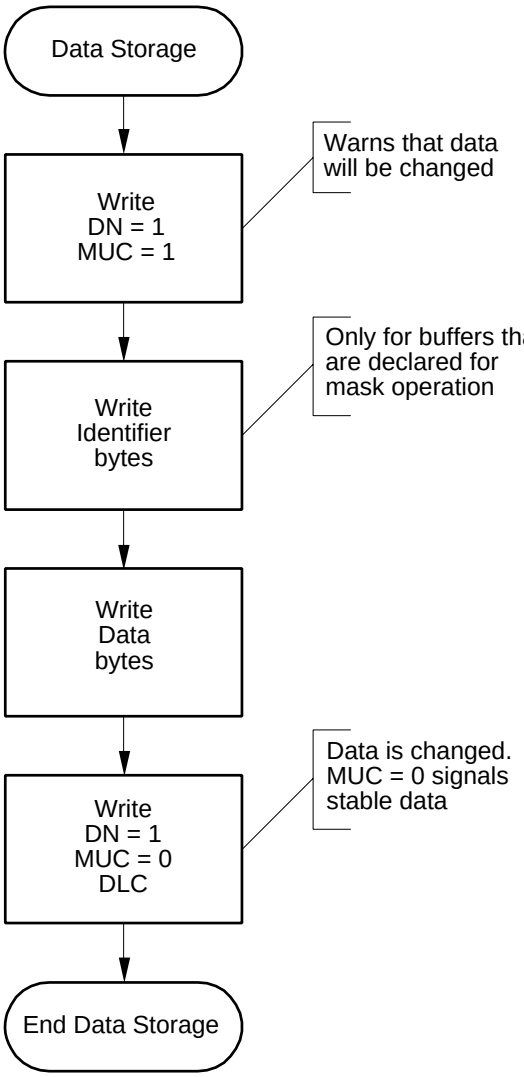
15.18.3 Abort Transmit

Figure 15-52: Transmit Abort



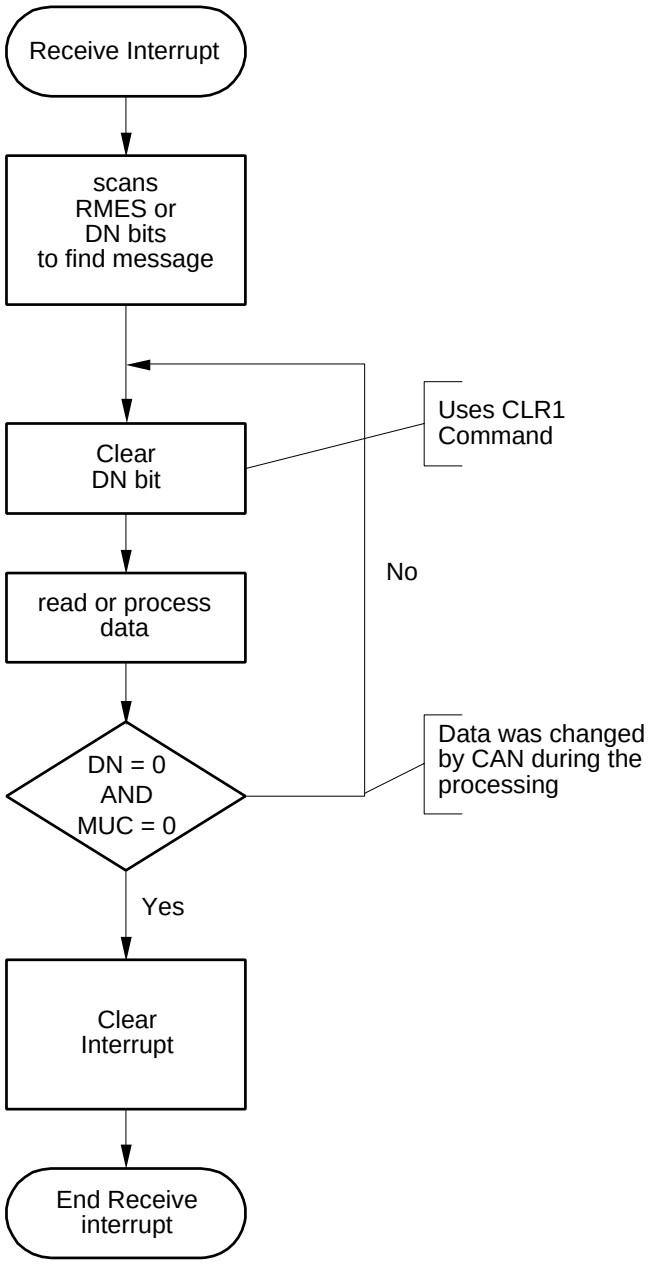
15.18.4 Handling by the DCAN

Figure 15-53: Handling of Semaphore Bits by DCAN-Module



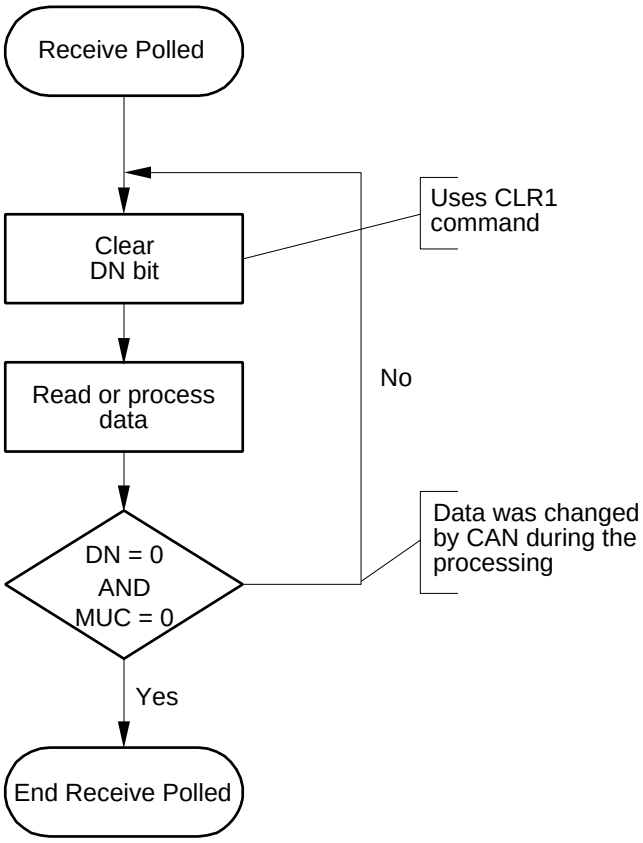
15.18.5 Receive Event Oriented

Figure 15-54: Receive with Interrupt, Software Flow



15.18.6 Receive Task Oriented

Figure 15-55: Receive, Software Polling



15.19 CAN Controller Precautions

(1) Resynchronization

According to the CAN protocol specification (BOSCH CAN specification, version 2.0, Sept. 1991, part A, chapter 8) a CAN node has to perform a soft-synchronization, when acting as a transmitter sending a dominant bit, if a recessive to dominant edge occurs after the sample point within phase segment 2. This scenario is only encountered in case of a disturbance. For this case the soft-synchronization is not performed by the implementations listed below. Due to this, the nominal length of an error frame, that follows this disturbance, can be extended by the amount of time quanta, allocated for the synchronization jump width.

Chapter 16 LCD Controller/Driver (µPD780822B Subseries)

16.1 LCD Controller/Driver Functions

The functions of the LCD controller/driver incorporated in the µPD780822B Subseries are listed below.

(1) Automatic output of segment signals and common signals is possible by automatic reading of the display data memory.
(2) Display mode
    - 1/4 duty (1/3 bias)
(3) Any of four frame frequencies can be selected in each display mode.
(4) Maximum of 34 segment signal outputs (S0 to S33); 4 common signal outputs (COM0 to COM3). All segment outputs can be switched to input/output ports.
S0/P77 - S7/P70, S8/P87 - S15/P80, S16/P97 - S23/P90, S24/P100 - S29/P105, and S30/P110 - S33/P113 are bitwise switchable.

The maximum number of displayable pixels is shown in Table 16-1.

Table 16-1: Maximum Number of Display Pixels

Bias Method	Time Division	Common Signals Used	Maximum Number of Display Pixels
1/3	4	COM0 to COM3	136 (34 segments x 4 commons)

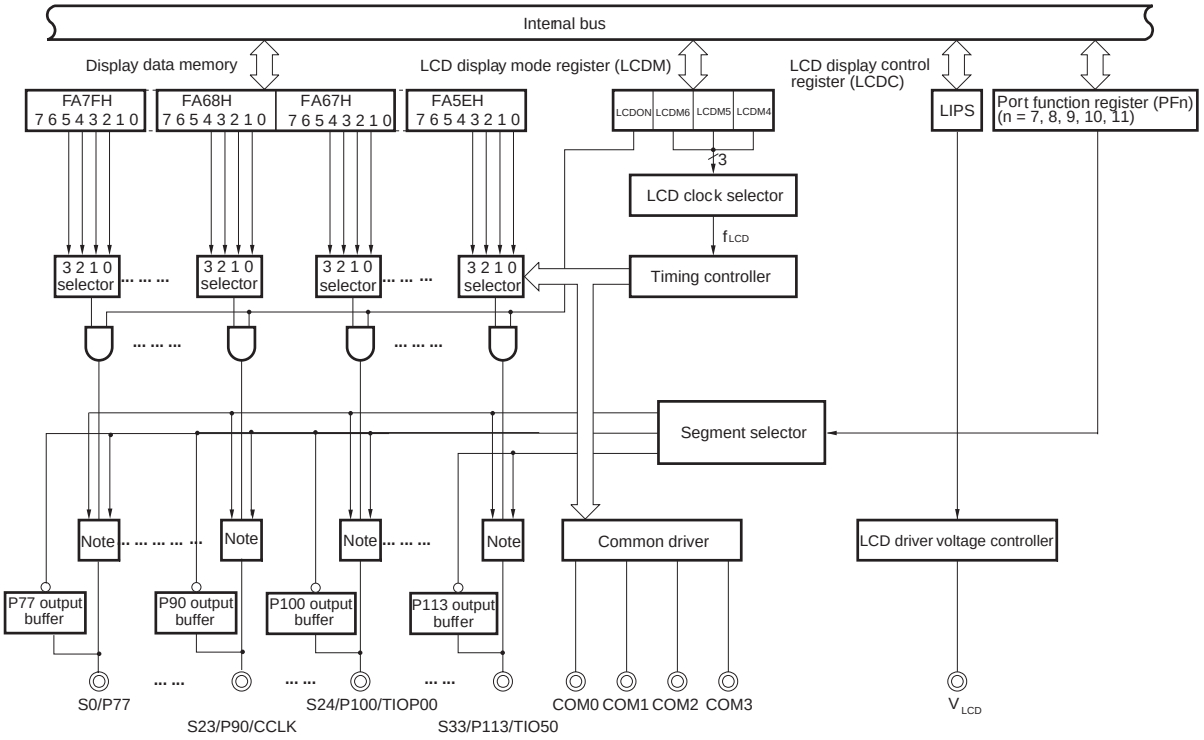
16.2 LCD Controller/Driver Configuration

The LCD controller/driver consists of the following hardware.

Table 16-2: LCD Controller/Driver Configuration

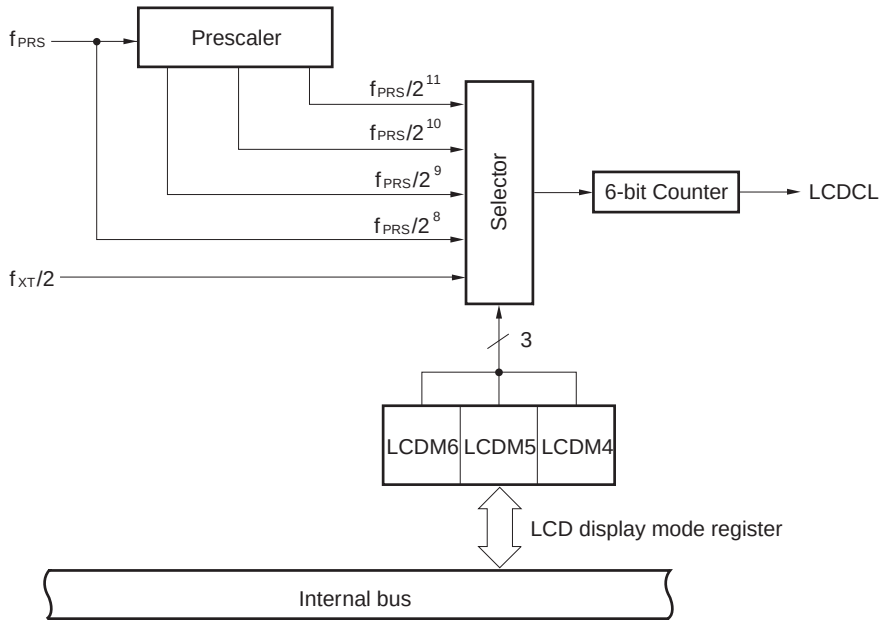
Item	Configuration
Display outputs	Segment signals: 34
	Segment signal with alternate function: 34
	Common signals: 4 (COM0 to COM3)
Control registers	LCD display mode register (LCDM)
	LCD display control register (LCDC)

Figure 16-1: LCD Controller/Driver Block Diagram



Remark: Segment driver

Figure 16-2: LCD Clock Select Circuit Block Diagram



Remarks: **1.** LCDCL: LCD clock

2. f_{PRS}: Peripheral hardware clock oscillation frequency

3. f_{XT}: Subsystem clock operation frequency

16.3 LCD Controller/Driver Control Registers

The LCD controller/driver is controlled by the following two registers.

- LCD display mode register (LCDM)
- LCD display control register (LCDC)

(1) LCD display mode register (LCDM)

This register enables/disables the LCD and selects the LCD clock.
LCDM is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears LCDM to 00H.

Figure 16-3: LCD Display Mode Register (LCDM) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
LCDM	LCDON	LCDM6	LCDM5	LCDM4	0	0	0	0	FF90H	00H	R/W

LCDON	LCD Enable/Disable
0	Display off (all segment outputs are non-select signal outputs)
1	Display on

LCDM6	LCDM5	LCDM4	LCD Clock Selection		
				f_{PRS} = 10 MHz	f_{PRS} = 16 MHz
0	0	0	$f_{PRS}/2^{17}$	76 Hz	122 Hz
0	0	1	$f_{PRS}/2^{16}$	153 Hz	244 Hz
0	1	0	$f_{PRS}/2^{15}$	305 Hz	488 Hz
0	1	1	$f_{PRS}/2^{14}$	610 Hz	977 Hz
1	0	0	$f_{XT}/2^{7}$	256 Hz	
Other than above			Setting prohibited		

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

(2) LCD display control register (LCDC)

This register sets cutoff of the current flowing to split resistors for LCD drive voltage generation and switchover between segment output and input/output port functions.
LCDC is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears LCDC to 00H.

Figure 16-4: LCD Display Control Register (LCDC) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
LCDC	1	0	0	0	0	0	0	LIPS	FF92H	80H	R/W

LIPS	LCD Driving Power Supply Selection
0	LCD drive power not supplied
1	LCD drive power supplied from V_{DD}

Cautions: 1. Always write “1” to bit 7.
Always write”0” to bit 1 to 6.

2. If the LCD drive voltage is supplied from an external device, make sure, that the LIPS-bit of the LCDC register is set to 0.
Otherwise the device may be damaged.

16.4 LCD Controller/Driver Settings

LCD controller/driver settings should be performed as shown below.

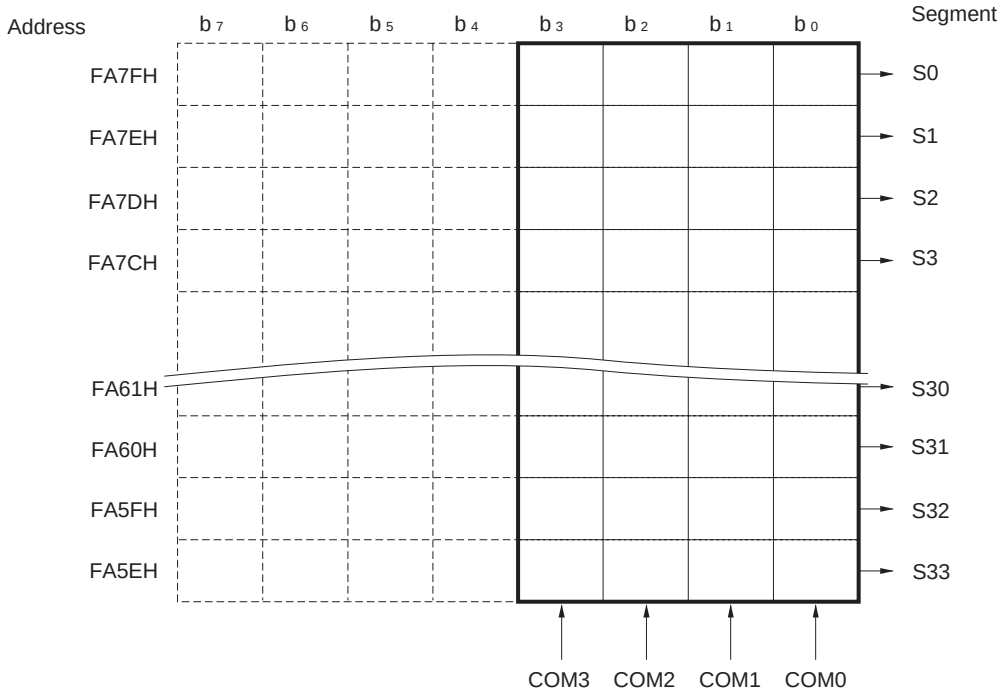
<1> Set the initial value in the display data memory (FA5EH to FA7FH).
<2> Set the pins to be used as segment outputs in port function registers (PF7, PF8, PF9, PF10, PF11).
<3> Set the LCD power supply in the LCD display control register (LCDC).
<4> Set the LCD clock in the LCD display mode register (LCDM).

Next, set data in the display data memory according to the display contents.

16.5 LCD Display Data Memory

The LCD display data memory is mapped onto addresses FA5EH to FA7FH. The data stored in the LCD display data memory can be displayed on an LCD panel by the LCD controller/driver.
Figure 16-5 shows the relationship between the LCD display data memory contents and the segment outputs/common outputs.
Any area not used for display can be used as normal RAM[Note].

Figure 16-5: Relationship between LCD Display Data Memory Contents and Segment/Common Outputs



Caution: The upper 4 bits of the LCD display data memory do not incorporate memory. Be sure to set them to 0.

Remark: The data of S0 is stored at the highest address in the LCD display data memory.

Note: $\overline{\text{RESET}}$ clears the LCD Display Data Memory to 00H.

16.6 Common Signals and Segment Signals

An individual pixel on an LCD panel lights when the potential difference of the corresponding common signal and segment signal reaches or exceeds a given voltage (the LCD drive voltage V_{LCD}). The light goes off when the potential difference becomes V_{LCD} or lower.
As an LCD panel deteriorates if a DC voltage is applied in the common signals and segment signals, it is driven by AC voltage.

(1) Common signals

For common signals, the selection timing order is as shown in Table 16-3, and operations are repeated with these as the cycle.

Table 16-3: COM Signals

COM Signal / Time Division	COM0	COM1	COM2	COM3
4-Time Division				

(2) Segment signals

Segment signals correspond to a 34-byte LCD display data memory (FA5EH to FA7FH). Each display data memory bit 0, bit 1, bit 2, and bit 3 is read in synchronization with the COM0, COM1, COM2 and COM3 timings respectively, and if the value of the bit is 1, it is converted to the selection voltage. If the value of the bit is 0, it is converted to the non-selection voltage and send to a segment pin (S0 to S33) (S33 to S0 have an alternate function as input/output port pins).
Consequently, it is necessary to check what combination of front surface electrodes (corresponding to the segment signals) and rear surface electrodes (corresponding to the common signals) of the LCD panel to be used to form the display pattern. Then write a bit data corresponding on a one-to-one basis with the pattern to be displayed.
Bits 4 to 7 are fixed at 0.

(3) Common signal and segment signal output waveforms

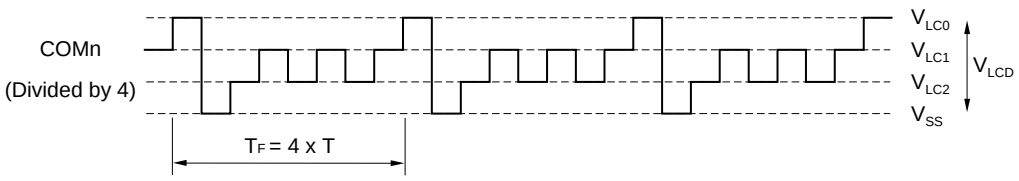
The voltages shown in Table 16-4 are output in the common signals and segment signals.
The $\pm V_{LCD}$ ON voltage is only produced when the common signal and segment signal are both at the selection voltage; other combinations produce the OFF voltage.

Table 16-4: LCD Drive Voltage

Common	Segment	Select Level	Non-Select Level
		V_{SS1}, V_{LC0}	V_{LC1}, V_{LC2}
Select Level	V_{LC0}, V_{SS1}	$-V_{LCD}$, $+V_{LCD}$	-1/3 V_{LCD}, +1/3 V_{LCD}
Non-Select Level	V_{LC2}, V_{LC1}	-1/3 V_{LCD}, +1/3 V_{LCD}	-1/3 V_{LCD}, +1/3 V_{LCD}

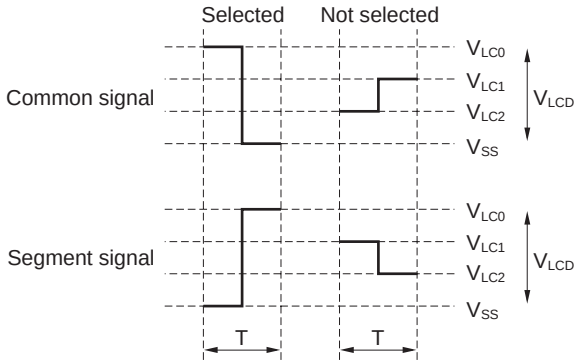
Figure 16-6 shows the common signal waveform, and Figure 16-7 shows the common signal and segment signal voltages and phases.

Figure 16-6: Common Signal Waveform



Remark: T : One LCDCL cycle
TF : Frame frequency

Figure 16-7: Common Signal and Segment Signal Voltages and Phases



Remark: T : One LCDCL cycle

16.7 Supplying LCD Drive Voltage V_{LC0}, V_{LC1}, and V_{LC2}

The µPD780822B Subseries have a split resistor to create an LCD drive voltage, and the drive voltage is fixed to 1/3 bias.
To supply various LCD drive voltages, internal V_{DD} or external V_{LCD} supply voltage can be selected.

Table 16-5: LCD Drive Voltage Supply

LCD Drive Voltage \ Bias Method	1/3 Bias Method
V_{LC0}	V_{LC0}
V_{LC1}	2/3 V_{LC0}
V_{LC2}	1/3 V_{LC0}

Figure 16-8 shows an example of supplying an LCD drive voltage from an internal source according to Table 16-5.

Figure 16-8: Example of Connection of LCD Drive Power Supply (1/2)

(a) To supply LCD drive voltage from V_{DD}

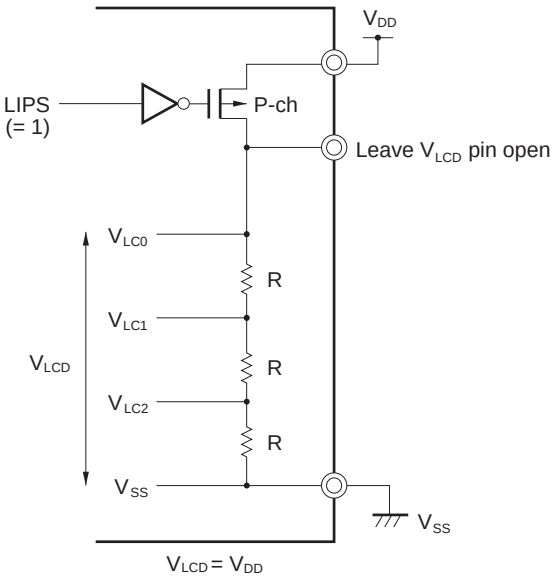
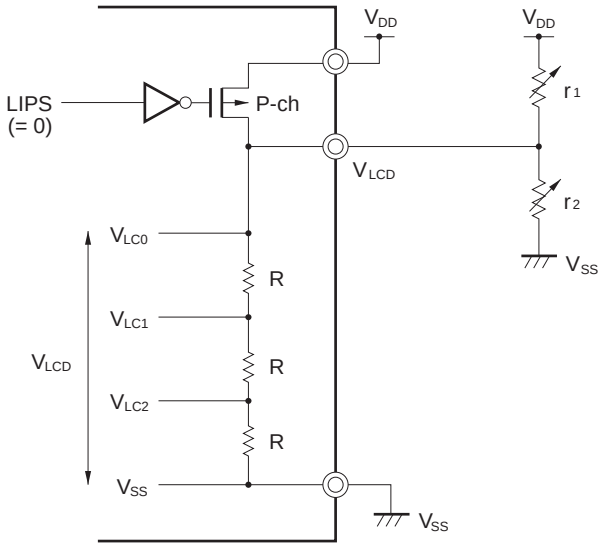


Figure 16-8: Example of Connection of LCD Drive Power Supply (2/2)

(b) To supply LCD drive voltage from external source



$$V_{LCD} = \frac{3R \ r_2}{3R \ r_2 + 3R \ r_1 + r_1 \ r_2} \times V_{DD}$$

Caution: If the LCD drive voltage is supplied from an external source, make sure, that the LIPS-bit of the LCDC register is set to 0.
Otherwise the device may be damaged.

16.8 Display Mode

16.8.1 4-time-division display example

Figure 16-10, “4-Time-Division LCD Panel Connection Example,” on page 443 shows the relationship between a 4-time-division type 10-digit LCD panel and the display pattern shown in Figure 16-9, “4-Time-Division LCD Display Pattern and Electrode Connections,” on page 442 and the µPD780822B Subseries segment signals (S0 to S19) in conjunction with common signals (COM0 to COM3). The display example is “1234567890”. The display data memory contents (addresses FA7FH to FA6CH) correspond to this.
An explanation is given here taking the example of the 5th digit “6” (❻). In accordance with the display pattern in Figure 16-9, “4-Time-Division LCD Display Pattern and Electrode Connections,” on page 442, selection and non-selection voltages must be send to pins S8 and S9 as shown in Table 16-6 at the COM0 to COM3 common signal timings.

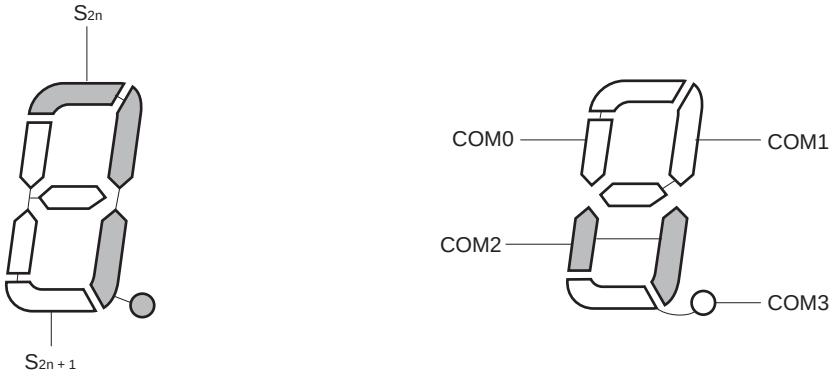
Table 16-6: Selection and Non-Selection Voltages (COM0 to COM3)

Segment / Common	S8	S9
COM0	S	S
COM1	NS	S
COM2	S	S
COM3	NS	S

Remark: S: Selection, NS: Non-selection

From this, it can be seen that 0101 (COM0 is LSB) must be prepared in the display data memory (address FA77H) corresponding to S8.
Examples of the LCD drive waveforms between S8 and the COM0 and COM1 signals are shown in Figure 16-11, “4-Time-Division LCD Drive Waveform Examples (1/3 Bias Method),” on page 444 (for the sake of simplicity, waveforms for COM2 and COM3 have been omitted). When S8 carries the selection voltage at the COM0 selection timing, it can be seen that the $+V_{LCD}/-V_{LCD}$ AC square wave, which is the LCD illumination (ON) level, is generated.

Figure 16-9: 4-Time-Division LCD Display Pattern and Electrode Connections



Remark: n = 0 to 9

Figure 16-10: 4-Time-Division LCD Panel Connection Example

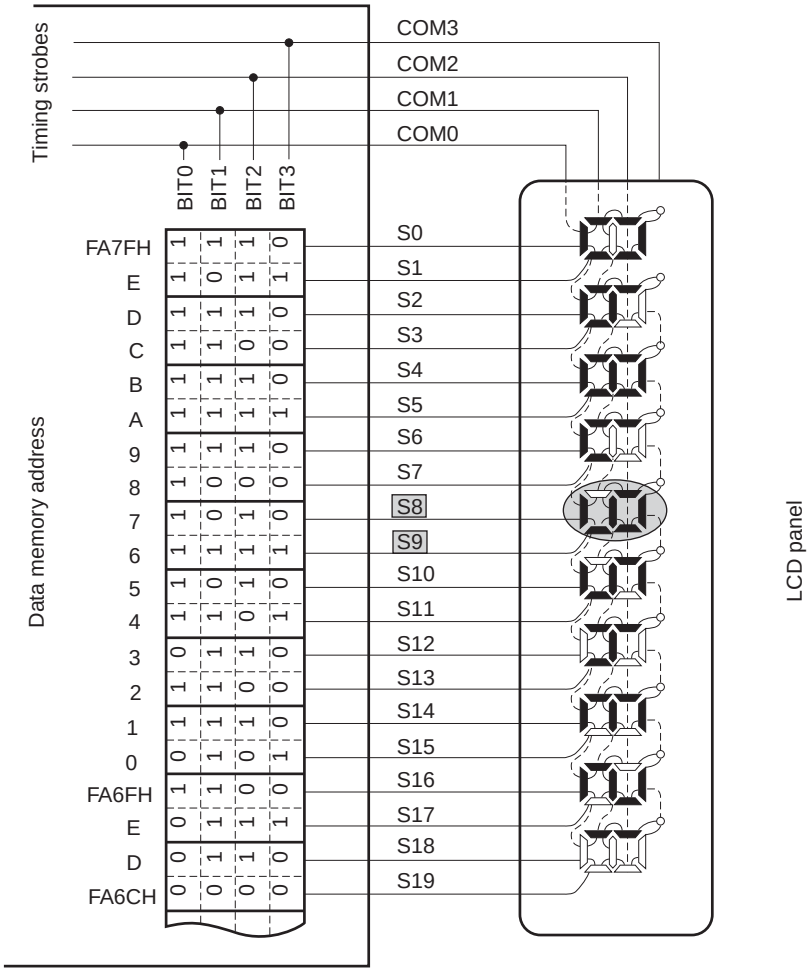
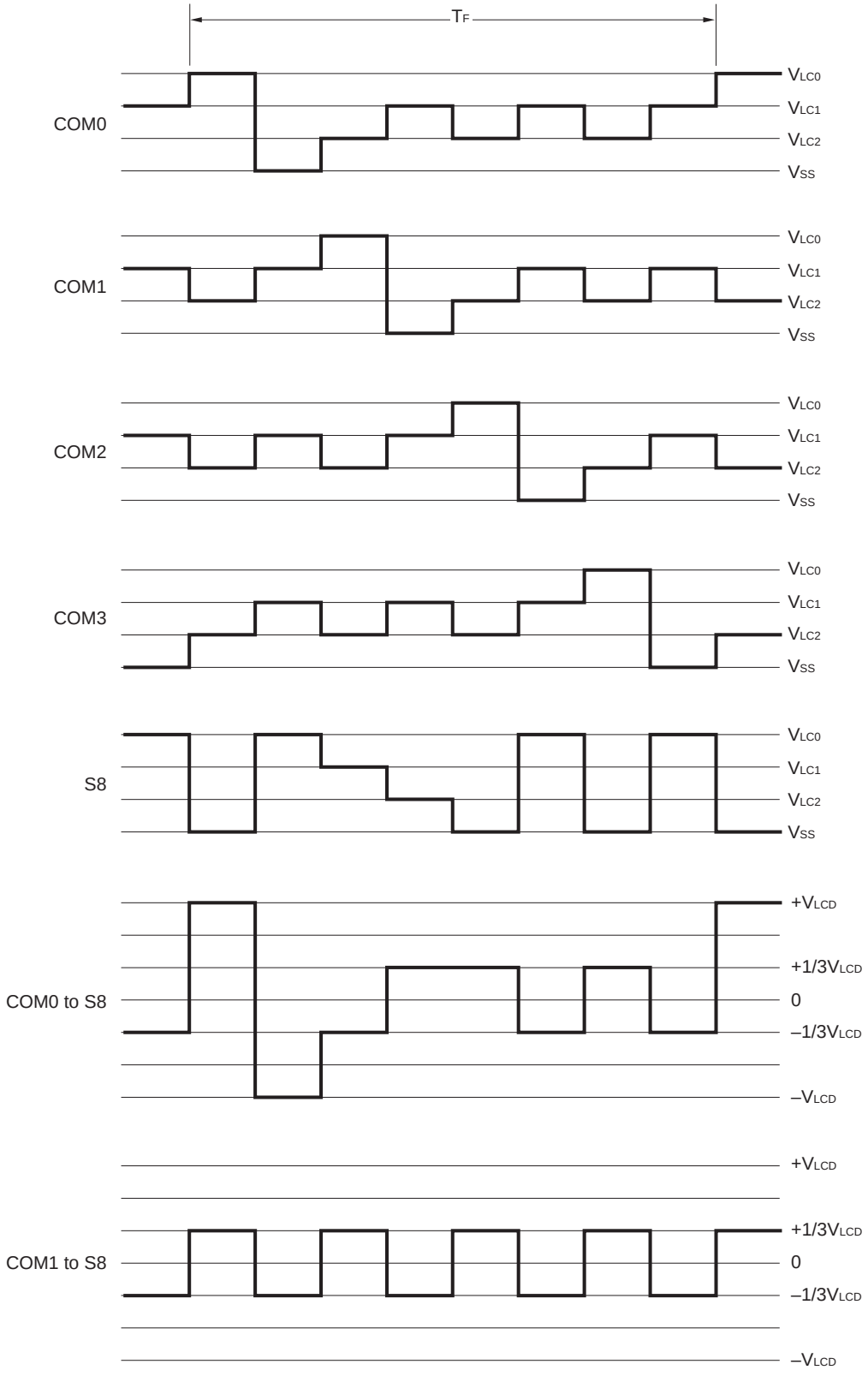


Figure 16-11: 4-Time-Division LCD Drive Waveform Examples (1/3 Bias Method)



16.9 Cautions on Emulation performed with IE-78K0-NS-A Emulator and IE-780822-NS-EM1 Emulation Board

LCD emulation on the Emulator IE-78K0-NS-A is done by µPD780338 device.
For LCD RAM, there are no differences between real-chip and emulator from address and data setting point of view.
For the LCD registers LCDM and LCDC, addresses are re-mapped, but data settings are not the same. Please refer to following tables:

SFR	Real-chip address	Emulator address
LCDM	0xFF90	0xFF90
LCDC	0xFF92	0xFF92
LCD RAM	0xFF7F to 0xFF5E	0xFF7F to 0xFF5E

SFR	Real-chip data setting	Emulator data setting (from µPD780338)
LCDM	Real-chip definition	LCDM3
LCDC	Real-chip definition	LCDC3

To perform debugging with the emulator, the used SFRs are LCDM3 and LCDC3.
The bit settings on these registers are different than the bit settings on LCDM and LCDC (µPD780822B Subseries real-chip).

SFR settings required (LCDM, LCDC) for emulation:

(1) LCD display mode register 3 (LCDM3)

This register enables or disables display, controls the booster circuit.
This register can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the value of this register to 00H.

Figure 16-12: LCD Display Mode Register 3(LCDM3) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
LCDM3	LCDON	SCOC	VLCON	0	0	0	0	0	FF90H	00H	R/W

LCDON	Display Control (enables output of display data)
0	Display OFF (all segment outputs are non-reset signals outputs)
1	Display ON

SCOC	Output Control of segment/common pins
0	Outputs GND level to segment/common pins
1	Outputs select signal to segment/common pins

VLCON	Booster Circuit Control
0	Stops booster circuit
1	Operates booster circuit

Cautions: 1. Always set bits 4, 3, 2, 1, 0 to 0.

2. Set the LCDON, SCOC and VLCON bits in the following sequence:

- **To display LCD while LCD booster circuit stops:**
  - **(1) Set VLCON to 1. All the segment and common pins are in the GND output mode (SCOC = 0).**
  - **(2) Set VLCON to 1 and wait 500 ms or longer with software.**
  - **(3) Set SCOC to 1. All the segment and common pins output an unselect waveform and are in unselect display mode.**
  - **(4) Set LCDON to 1. The value of the display RAM is reflected on the segment output waveform, and all the segment and common pins are in select display mode.**
- **To stop LCD booster circuit while LCD displays:**
  - **(1) Clear VLCON to 0. All the segment and common pins are in unselect display mode.**
  - **(2) Clear SCOC to 0. All the segment and common pins are in the GND output mode**
  - **(3) Clear VLCON to 1. The LCD booster circuit stops.**

(2) LCD clock control register 3 (LCDC3)

This register is used to select a LCD frame frequency.
It is set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the value of this register to 00H.

Figure 16-13: LCD Clock Control Register 3 (LCDC3) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
LCDC3	0	0	0	0	1	1	LCDC31	LCDC30	FF92H	00H	R/W

LCDC31	LCDC30	Selection of reference clock generator frame frequency
0	0	$f_{PRS}/2^{14}$ (610 Hz)
0	1	$f_{PRS}/2^{15}$ (305 Hz)
1	0	$f_{PRS}/2^{16}$ (153 Hz)
1	1	$f_{PRS}/2^{17}$ (76 Hz)

Cautions: 1. Always set bits 7, 6, 5, 4 to 0 and bits 3 and 2 to 1.

2. Do not re-write LCDC3 while LCD is operating.
Be sure to set this register while LCDON = 0, SCOC = 0 and VLCON = 0.

Remark: Figures in parentheses are for operation with f_{PRS} = 10 MHz.

[MEMO]

Chapter 17 Sound Generator (µPD780822B Subseries)

17.1 Sound Generator Function

The sound generator has the function to operate an external speaker. The following two signals are supplied by the sound generator.

(1) Basic cycle output signal (with/without amplitude)

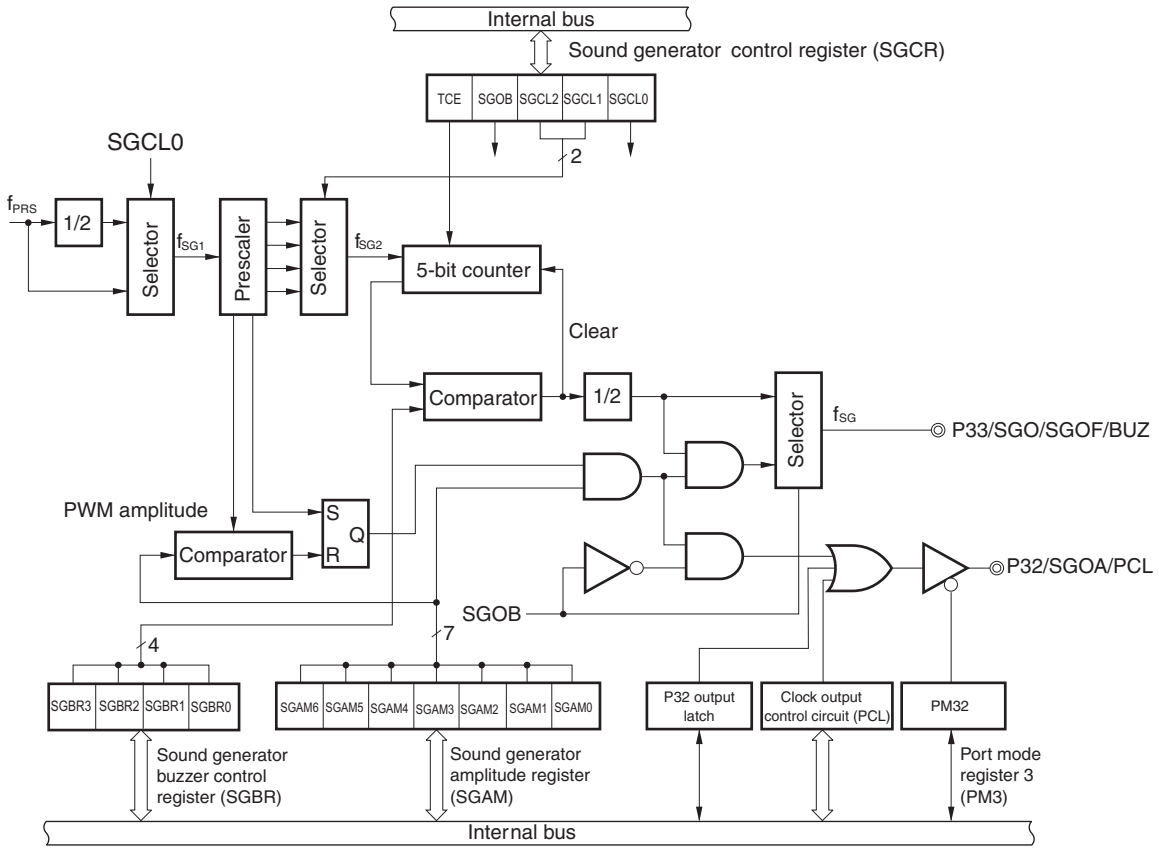
A buzzer signal with a variable frequency in a range of 0.5 to 3.8 kHz (at $f_X = f_{PRS} = 8.38$ MHz) can be created. The amplitude of the basic cycle output signal can be varied by ANDing the basic cycle output signal with the 7-bit-resolution PWM signal, to achieve control of the volume.

(2) Amplitude output signal

A PWM signal with a 7-bit resolution for variable amplitude can be generated independently.

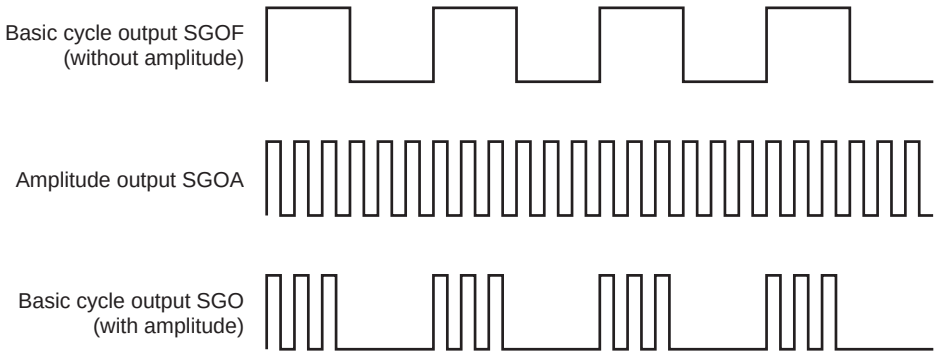
Figure 17-1 shows the sound generator block diagram and Figure 17-2 shows the concept of each signal.

Figure 17-1: Sound Generator Block Diagram



Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

Figure 17-2: Concept of Each Signal



17.2 Sound Generator Configuration

The sound generator consists of the following hardware.

Table 17-1: Sound Generator Configuration

Item	Configuration
Counter	8 bits × 1, 5 bits × 1
SG output	SGO/SGOF (with/without append bit of basic cycle output) SGOA (amplitude output)
Control register	Sound generator control register (SGCR) Sound generator buzzer control register (SGBR) Sound generator amplitude register (SGAM)

17.3 Sound Generator Control Registers

The following three types of registers are used to control the sound generator.

- Sound generator control register (SGCR)
- Sound generator buzzer control register (SGBR)
- Sound generator amplitude control register (SGAM)

(1) Sound generator control register (SGCR)

SGCR is a register which sets up the following four types.

- Controls sound generator output
- Selects output of sound generator
- Selects sound generator input frequency f_{SG1}
- Selects 5-bit counter input frequency f_{SG2}

SGCR is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears SGCR to 00H.
Figure 17-3 shows the SGCR format.

Figure 17-3: Sound Generator Control Register (SGCR) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SGCR	TCE	0	0	0	SGOB	SGCL2	SGCL1	SGCL0	FFC0H	00H	R/W

TCE	Sound Generator Operation Control
0	Timer operation stopped, SGOF/SGO and SGOA for low-level output Operation stopped (sound generator output signal disable, SGO = 0)
1	Operation enable

Caution: Before setting the TCE bit, set all the other bits.

Remark: SGOF : Basic cycle signal (without amplitude)
SGO : Basic cycle signal (with amplitude)
SGOA : Amplitude signal

SGOB	Sound Generator Output Selection
0	Sound generator operation with separated frequency / amplitude output (SGOF and SGOA outputs selected)
1	Sound generator operation with composed frequency / amplitude output (SGO and PCL outputs selected)

SGCL2	SGCL1	5-Bit Counter Input Frequency f_{SG2} Selection
0	0	$f_{SG2} = f_{SG1}/2^5$
0	1	$f_{SG2} = f_{SG1}/2^6$
1	0	$f_{SG2} = f_{SG1}/2^7$
1	1	$f_{SG2} = f_{SG1}/2^8$

Figure 17-3: Sound Generator Control Register (SGCR) Format (2/2)

SGCL0	Sound Generator Input Frequency Selection
0	$f_{SG1} = f_{PRS}/2$
1	$f_{SG1} = f_{PRS}$

Cautions: 1. When rewriting SGCR to other data, stop the timer operation (TCE = 0) beforehand.

2. Bits 4 to 6 must be set to 0.

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency.

Maximum and minimum values of the buzzer output frequency are shown in the Table.

Table 17-2: Maximum and Minimum Values of the Buzzer Output Frequency

SGCL2	SGCL1	SGCL0	f_{SG2}	Maximum and Minimum Values of Buzzer Output			
				f_{PRS} = 8 MHz		f_{PRS} = 16 MHz	
				Max. (kHz)	Min. (kHz)	Max. (kHz)	Min. (kHz)
0	0	0	$f_{PRS}/2^6$	3.676	1.953	7.353	3.906
0	0	1	$f_{PRS}/2^5$	7.353	3.906	14.706	7.813
0	1	0	$f_{PRS}/2^7$	1.838	0.977	3.676	1.953
0	1	1	$f_{PRS}/2^6$	3.676	1.953	7.353	3.906
1	0	0	$f_{PRS}/2^8$	0.919	0.488	1.838	0.977
1	0	1	$f_{PRS}/2^7$	1.838	0.977	3.676	1.953
1	1	0	$f_{PRS}/2^9$	0.460	0.244	0.919	0.488
1	1	1	$f_{PRS}/2^8$	0.919	0.488	1.838	0.977

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency.

(2) Sound generator buzzer control register (SGBR)

SGBR is a register that sets the basic frequency of the sound generator output signal.
SGBR is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears SGBR to 00H.
Figure 17-4 shows the SGBR format.

Figure 17-4: Sound Generator Buzzer Control Register (SGBR) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SGBR	0	0	0	0	SGBR3	SGBR2	SGBR1	SGBR0	FFC2H	00H	R/W

SGBR3	SGBR2	SGBR1	SGBR0	Sound generator operation control[Note]	
				f_{PRS} = 8 MHz	f_{PRS} = 16 MHz
0	0	0	0	3.676 kHz	7.353 kHz
0	0	0	1	3.472 kHz	6.944 kHz
0	0	1	0	3.289 kHz	6.579 kHz
0	0	1	1	3.125 kHz	6.250 kHz
0	1	0	0	2.976 kHz	5.952 kHz
0	1	0	1	2.841 kHz	5.682 kHz
0	1	1	0	2.717 kHz	5.435 kHz
0	1	1	1	2.604 kHz	5.208 kHz
1	0	0	0	2.500 kHz	5.000 kHz
1	0	0	1	2.404 kHz	4.808 kHz
1	0	1	0	2.315 kHz	4.630 kHz
1	0	1	1	2.232 kHz	4.464 kHz
1	1	0	0	2.155 kHz	4.310 kHz
1	1	0	1	2.083 kHz	4.167 kHz
1	1	1	0	2.016 kHz	4.032 kHz
1	1	1	1	1.953 kHz	3.906 kHz

Note: Output frequency when SGCL2 - SGCL0 = (0, 0, 0)

Cautions: 1. When rewriting SGBR to other data, stop the timer operation (TCE = 0) beforehand.

2. Bits 4 to 7 must be set to 0.

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency.

(3) Sound generator amplitude register (SGAM)

SGAM is a register that sets the amplitude of the sound generator output signal.
SGAM is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears SGAM to 00H.
Figure 17-5 shows the SGAM format.

Figure 17-5: Sound Generator Amplitude Register (SGAM) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SGAM	0	SGAM6	SGAM5	SGAM4	SGAM3	SGAM2	SGAM1	SGAM0	FFC1H	00H	R/W

SGAM6	SGAM5	SGAM4	SGAM3	SGAM2	SGAM1	SGAM0	Amplitude
0	0	0	0	0	0	0	0/128
0	0	0	0	0	0	1	2/128
0	0	0	0	0	1	0	3/128
0	0	0	0	0	1	1	4/128
0	0	0	0	1	0	0	5/128
0	0	0	0	1	0	1	6/128
0	0	0	0	1	1	0	7/128
0	0	0	0	1	1	1	8/128
0	0	0	1	0	0	0	9/128
0	0	0	1	0	0	1	10/128
0	0	0	1	0	1	0	11/128
0	0	0	1	0	1	1	12/128
0	0	0	1	1	0	0	13/128
0	0	0	1	1	0	1	14/128
0	0	0	1	1	1	0	15/128
0	0	0	1	1	1	1	16/128
0	0	1	0	0	0	0	17/128
0	0	1	0	0	0	1	18/128
0	0	1	0	0	1	0	19/128
0	0	1	0	0	1	1	20/128
0	0	1	0	1	0	0	21/128
0	0	1	0	1	0	1	22/128
0	0	1	0	1	1	0	23/128
0	0	1	0	1	1	1	24/128
0	0	1	1	0	0	0	25/128
0	0	1	1	0	0	1	26/128
0	0	1	1	0	1	0	27/128
0	0	1	1	0	1	1	28/128
0	0	1	1	1	0	0	29/128
0	0	1	1	1	0	1	30/128
0	0	1	1	1	1	0	31/128
			¦				¦
1	1	1	1	1	1	1	128/128

Cautions: 1. When rewriting the contents of SGAM, the timer operation does not need to be stopped. However, note that a high level may be output for one period due to rewrite timing.

2. Bit 7 must be set to 0.

17.4 Sound Generator Operations

17.4.1 Sound generator settings

<A>. Choose the sound generator output by SGOB bit.
<B>. Set port 3 in output mode by PM3 register.
<C>. Set the maximum and the minimum values of the buzzer output frequency through SGCL2, SGCL1 and SGCL0.
<D>. Choose the output frequency by SGBR register.
The sound generator output frequency f_{SG} can be calculated by the following expression:

$$f_{SG} = 2^{(SGCL0 - SGCL1 - 2 \times SGCL2 - 7)} \times [f_X / (SGBR + 17)]$$

Remarks: **1.** f_X is the component frequency

2. SGCL2, SGCL1 and SGCL0 values must be replaced by decimals values in the previous expression.

Example:

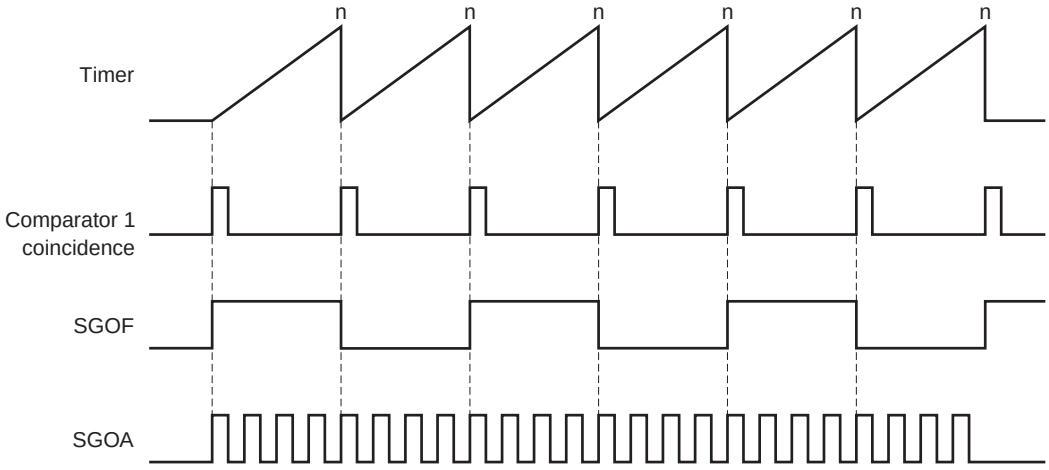
When $f_X = f_{PRS}$ = 8 MHz, SGCL0 to SGCL2 is (1, 0, 0), SGBR0 to SGBR3 is (1, 1, 1, 1) and SGBR = 15. Therefore, the sound generator output frequency is f_{SG} = 3.906 kHz.

<E>. Select the amplitude of the output signal by SGAM register.
<F>. Set TCE to 1 to start the sound generator operation.

17.4.2 To output basic cycle signal SGOF (without amplitude)

Select SGOF output by setting bit 3 (SGOB) of the sound generator control register (SGCR) to “0”.
The basic cycle signal with a frequency specified by the SGCL0 to SGCL2 and SGBR0 to SGBR3 is output.
At the same time, the amplitude signal with an amplitude specified by the SGAM0 to SGAM6 is output from the SGOA pin.

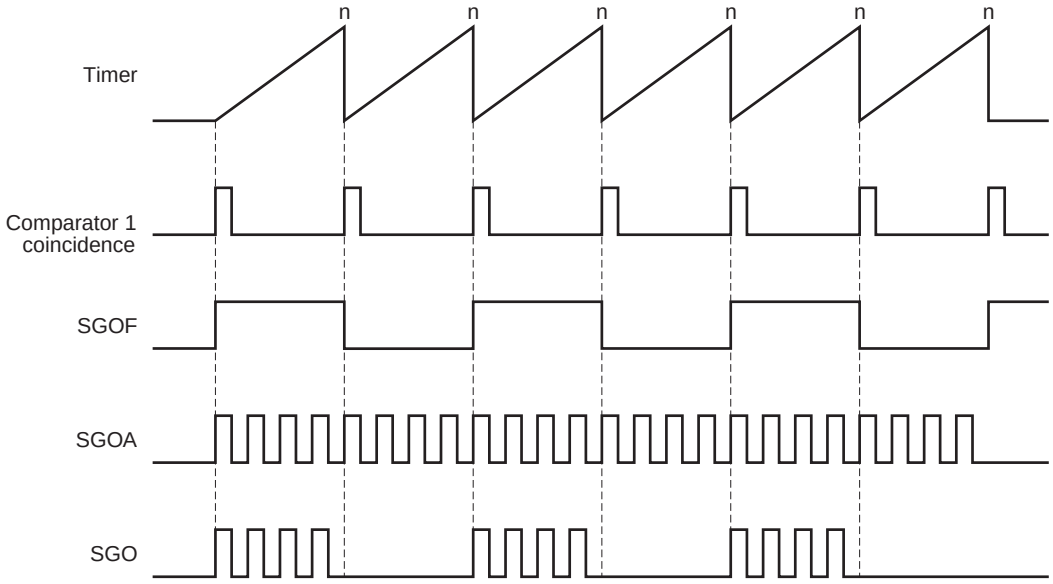
Figure 17-6: Sound Generator Output Operation Timing



17.4.3 To output basic cycle signal SGO (with amplitude)

Select SGO output by setting bit 3 (SGOB) of the sound generator control register (SGCR) to “1”.
The basic cycle signal with a frequency specified by the SGCL0 to SGCL2 and SGBR0 to SGBR3 is output.
When SGO output is selected, the SGOA pin can be used as a PCL output (clock output) or I/O port pin.

Figure 17-7: Sound Generator Output Operation Timing



[MEMO]

Chapter 18 Meter Controller / Driver (µPD780822B Subseries)

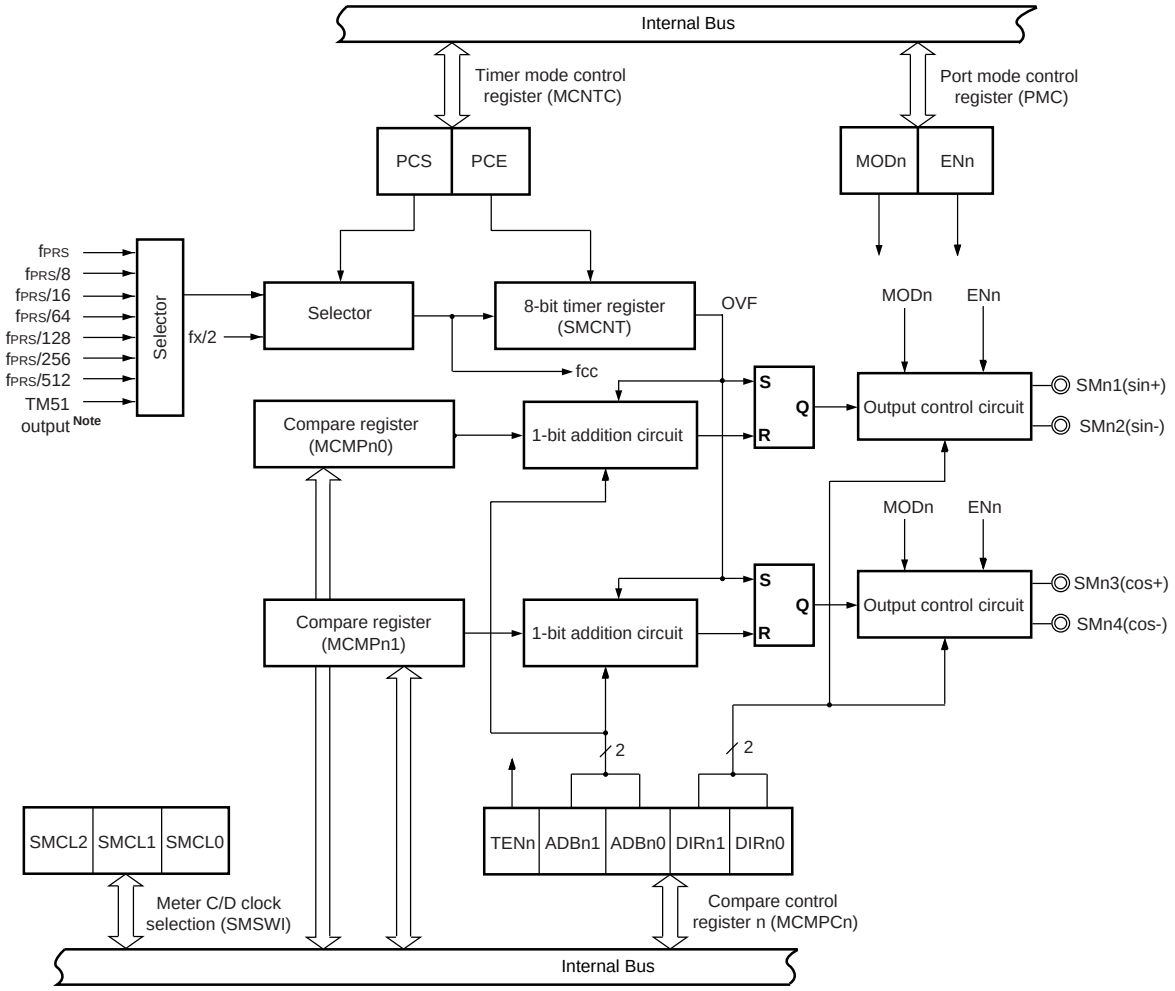
18.1 Meter Controller/Driver Functions

The meter controller/driver is a function to drive a stepping motor for external meter control or cross coil.

- Can set pulse width with a precision of 8 bits
- Can set pulse width with a precision of 8 + 1 bits with 1-bit addition function
- Can drive up to four 360° type meters

Figure 18-1 shows the block diagram of the meter controller/driver.
Figure 18-2 shows 1-bit addition circuit block diagram.

Figure 18-1: Meter Controller/Driver Block Diagram



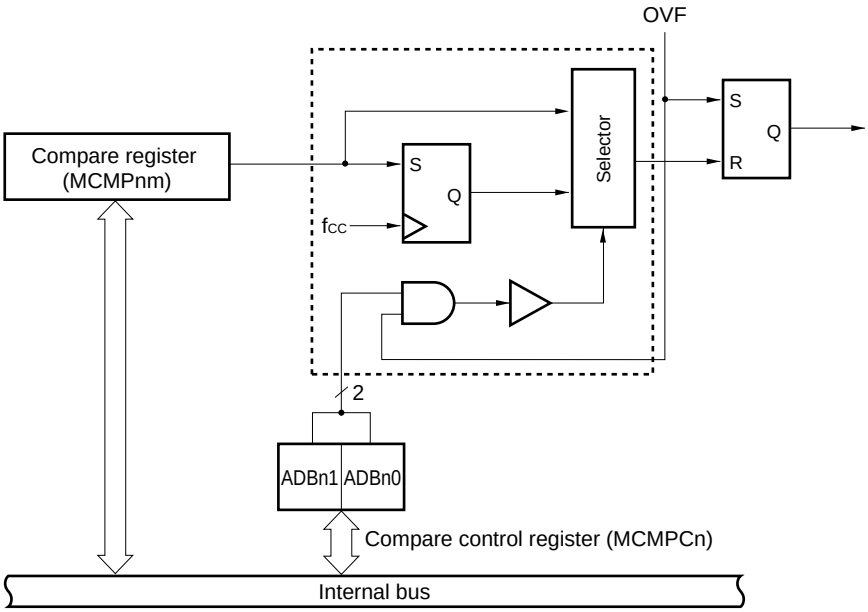
Note: The output signal of the timer TM51 can be used as clock input of the meter controller/driver. for details, please refer to Figure 8-2, “8-Bit Timer/Event Counter 51 Block Diagram,” on page 216.

Remarks: 1. n = 1 to 4

2. f_{PRS}: Peripheral hardware clock oscillation frequency.

The meter controller/driver is a peripheral to control/drive up to four external meters (stepper motor/ cross coil motors).

Figure 18-2: 1-bit Addition Circuit Block Diagram



Remark: n = 1 to 4,
m = 0, 1

18.2 Meter Controller/Driver Configuration

The meter controller/driver consists of the following hardware.

Table 18-1: Meter Controller/Driver Configuration

Item	Configuration
Timer	Free-running up counter (SMCNT): 1 channel
Register	Compare register (MCMPn0, MCMPn1): 8 channels
Control registers	Timer mode control register (MCNTC) Compare control register n (MCMPCn) Port mode control register (PMC) Meter controller/driver clock switch register (SMSWI)
Pulse control circuit	1-bit addition circuit/output control circuit

Remark: n = 1 to 4

(1) Free running up counter (SMCNT)

SMCNT is an 8-bit free running counter. It is also a register that executes an increment at the rising edge of the input clock.
A PWM pulse with a resolution of 8 bits can be created. The duty factor can be set in a range of 0 to 100%.
The count value is cleared in the following cases.

- When $\overline{\text{RESET}}$ signal input
- When counter stops (PCE = 0)

(2) Compare registers nm (MCMPnm)

MCMPnm are 8-bit registers that can rewrite compare values according to the specification by bit 4 (TENn) of the compare control register n (MCMPCn).
$\overline{\text{RESET}}$ input sets these registers to 00H and clears hardware to 0.
MCMPnm are registers that support read/write only for 8-bit access instructions.
MCMPnm continuously compare their value with the SMCNT value. When two values match, a signal on the sin side (m = 0) or on the cos side (m = 1) of the meter n is generated.

Remark: n = 1 to 4
m = 1, 2

(3) Timer mode control register (MCNTC)

MCNTC is an 8-bit register that controls the operation of the free running counter SMCNT (clock selection, counter start/stop).

(4) Compare control register n (MCMPCn)

MCMPCn is an 8-bit register which enables/disables the data transfer from master to slave, controls the output detection of the PWM pins, and the 1-bit addition circuit.
The 1-bit addition circuit repeats 1-bit addition/non-addition to the PWM output alternately upon MCNT overflow, and enables the state of the PWM output between the current compare value and the next compare value.
This circuit is controlled by bits 2 and 3 (ADBn0, ADBn1) of the MCMPCn register.

(5) Port mode control register (PMC)

PMC is an 8-bit register that specifies PWM/PORT output and selects Full bridge or Half bridge mode.

(6) Meter controller/driver clock switch (SMSWI)

The input clock of the meter controller/driver can be selected with the meter controller/driver clock switch. By default the register is set to 00H after $\overline{\text{RESET}}$.
SMSWI is a register that supports read/write only as 8-bit instruction.

(7) Output control circuit

This circuit consists of P-ch and N-ch drivers and can drive a meter in H bridge configuration by connecting a coil. When a meter is driven in half bridge configuration, the unused pins can be used as normal output port pins.

The relation of the duty factor of the PWM signal output from the SMnm pin is indicated by the following expression (n = 1 to 4, m = 0, 1).

$$\text{PWM (duty)} = \frac{\text{Set value of MCMPnm} \times \text{cycle of MCNT count clock}}{255 \times \text{cycle of MCNT count clock}} \times 100\% = \frac{\text{Set value of MCMPnm}}{255} \times 100\%$$

Cautions: 1. MCMPn0 and MCMPn1 cannot be read or written by a 16-bit access instruction.

2. MCMPn0 and MCMPn1 are in master-slave configuration, and SMCNT is compared with a slave register. The PWM pulse is not generated until the first overflow occurs after the counting operation has been started because the compare data is not transferred to the slave.

18.3 Meter Controller/Driver Control Registers

The meter controller/driver is controlled by the following four registers.

- Timer mode control register (MCNTC)
- Compare control register n (MCMPCn)
- Port mode control register (PMC)
- Meter controller/driver clock switch register (SMSWI)

Remark: n = 1 to 4

(1) Timer mode control register (MCNTC)

MCNTC is an 8-bit register that controls the operation of the free-running up counter (SMCNT).
MCNTC is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears MCNTC to 00H.
Figure 18-3 shows the MCNTC format.

Figure 18-3: Timer Mode Control Register (MCNTC) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MCNTC	0	0	PCS	PCE	0	0	0	0	FFBFH	00H	R/W

PCS	Timer Counter Clock Selection
0	Selection via SMSWI register (prescaler switch)
1	$f_{PRS}/2$

PCE	Timer Operation Control
0	Operation stopped (timer value is cleared)
1	Operation enabled

Cautions: 1. When rewriting MCNTC to other data, stop the timer operation (PCE=0) beforehand.

2. Bits 0 to 3, 6, and 7 must be set to 0.

Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

(2) Compare control register (MCMPCn)

MCMPCn is an 8-bit register that controls the operation of the compare register and output direction of the PWM pin.
MCMPCn is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears MCMPCn to 00H.
Figure 18-4 shows the MCMPCn format.

Figure 18-4: Compare Control Register n (MCMPCn) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MCMPCn	0	0	0	TENn	ADBn1	ADBn0	DIRn1	DIRn0	FFCCH to FFCFH	00H	R/W

TENn **Note**	Enables Transfer by Register from Master to Slave
0	Disables data transfer from master to slave. New data can be written.
1	Transfer data from master to slave when SMCNT overflows. New data cannot be written.

ADBn1	Control of 1-bit Addition circuit (cos side of meter n)
0	No 1-bit addition to PWM output
1	1-bit addition to PWM output

ADBn0	Control of 1-bit Addition circuit (sin side of meter n)
0	No 1-bit addition to PWM output
1	1-bit addition to PWM output

Remark: n = 1 to 4

Note: TENn functions as a control bit and status flag.
As soon as the timer overflows and PWM data is output, TENn is cleared to “0” by hardware.

The relation among the DIRn1 and DIRn0 bits of the MCMPCn register and output pin is shown below.

DIRn1	DIRn0	Direction Control Bit			
		SMn1	SMn2	SMn3	SMn4
0	0	PWM	0	PWM	0
0	1	PWM	0	0	PWM
1	0	0	PWM	0	PWM
1	1	0	PWM	PWM	0

Caution: Bits 5 to 7 must be set to 0.

(3) Port mode control register (PMC)

PMC is an 8-bit register that specifies PWM/PORT output.
PMC is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears PMC to 00H.
Figure 18-5 shows the PMC format.

Figure 18-5: Port Mode Control Register (PMC) Format (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PMC	MOD4	MOD3	MOD2	MOD1	EN4	EN3	EN2	EN1	FFCBH	00H	R/W

MOD4	Meter 4 Full/Half Bridge Selection
0	Meter 4 output is full bridge
1	Meter 4 output is half bridge

MOD3	Meter 3 Full/Half Bridge Selection
0	Meter 3 output is full bridge
1	Meter 3 output is half bridge

MOD2	Meter 2 Full/Half Bridge Selection
0	Meter 2 output is full bridge
1	Meter 2 output is half bridge

MOD1	Meter 1 Full/Half Bridge Selection
0	Meter 1 output is full bridge
1	Meter 1 output is half bridge

EN4	Meter 4 Port/PWM Mode Selection
0	Meter 4 output is in port mode
1	Meter 4 output is in PWM mode

EN3	Meter 3 Port/PWM Mode Selection
0	Meter 3 output is in port mode
1	Meter 3 output is in PWM mode

Figure 18-5: Port Mode Control Register (PMC) Format (2/2)

EN2	Meter 2 Port/PWM Mode Selection
0	Meter 2 output is in port mode
1	Meter 2 output is in PWM mode

EN1	Meter 1 Port/PWM Mode Selection
0	Meter 1 output is in port mode
1	Meter 1 output is in PWM mode

Caution: After setting the meter output to PWM mode the according port latch registers (P14 and P15) and port mode registers (PM14 and PM15) must be set to 0.

The relation among the ENn and MODn bits of the PMC register, DIRn1 and DIRn0 bits of the MCMPCn register, and output pins is shown below.

ENn	MODn	DIRn1	DIRn0	SMn1 (sin +)	SMn2 (sin -)	SMn3 (cos +)	SMn4 (cos -)	Mode
0	X	X	X	PORT	PORT	PORT	PORT	Port mode
1	0	0	0	PWM	GND	PWM	GND	PWM mode full bridge
1	0	0	1	PWM	GND	GND	PWM	
1	0	1	0	GND	PWM	GND	PWM	
1	0	1	1	GND	PWM	PWM	GND	
1	1	0	0	PWM	PORT	PWM	PORT	PWM mode half bridge
1	1	0	1	PWM	PORT	PORT	PWM	
1	1	1	0	PORT	PWM	PORT	PWM	
1	1	1	1	PORT	PWM	PWM	PORT	

DIRn1 and DIRn0 address the quadrant of sin and cos. DIRn1 and DIRn0 = 00 through 11 correspond to the 1st to 4th quadrant. The PWM signal is routed to the specific pin with respect to the polarity of the sin/cos of each quadrant.

When ENn = 0, all the output pins are used as port pins regardless of MODn, DIRn1 and DIRn0.
When ENn = 1 and MODn = 0, the full bridge mode is activated, and the pins, which do not output a PWM signal, will output “0”.
When ENn = 1 and MODn = 1, the half bridge mode is activated, and the pins, which do not output a PWM signal, are used as “PORT”.

Caution: DIRn0 and DIRn1 consist of the master and the slave and the output polarity of the PWM output changes at the overflow of SMCNT.

(4) Meter controller/driver clock register (SMSWI)

SMSWI is an 8-bit register that specifies the input clock of the meter controller/driver.
SMSWI is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets SMSWI to 00H.
Figure 18-6 shows the SMSWI format.

Figure 18-6: Meter Controller/Driver Clock Register (SMSWI) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SMSWI	0	0	0	0	0	SMCL2	SMCL1	SMCL0	FFBDH	00H	R/W

SMCL2	SMCL1	SMCL0	Meter Controller/Driver Clock Switch
0	0	0	f_{PRS}
0	0	1	$f_{PRS}/2^3$
0	1	0	$f_{PRS}/2^4$
0	1	1	$f_{PRS}/2^6$
1	0	0	$f_{PRS}/2^7$
1	0	1	$f_{PRS}/2^8$
1	1	0	$f_{PRS}/2^9$
1	1	1	TM51 Output

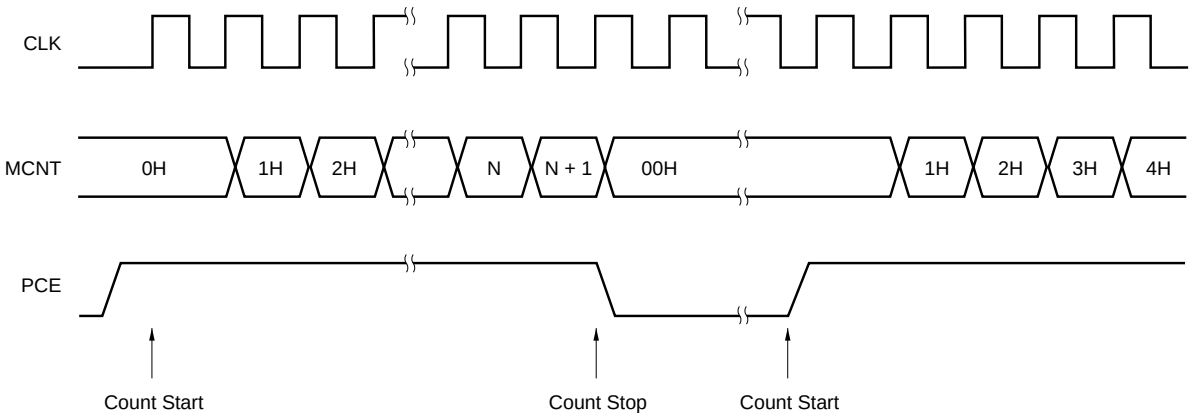
Remark: f_{PRS}: Peripheral hardware clock oscillation frequency

18.4 Meter Controller/Driver Operations

18.4.1 Basic operation of free-running up counter (SMCNT)

The free-running up counter is clocked by the count clock selected by the PCS bit of the time mode control register.
The value of SMCNT is cleared by $\overline{\text{RESET}}$ input.
The counting operation is enabled or disabled by the PCE bit of the timer mode control register (MCNTC).
Figure 18-7 shows the timing from count start to restart.

Figure 18-7: Restart Timing after Count Stop (Count Start → Count Stop → Count Start)



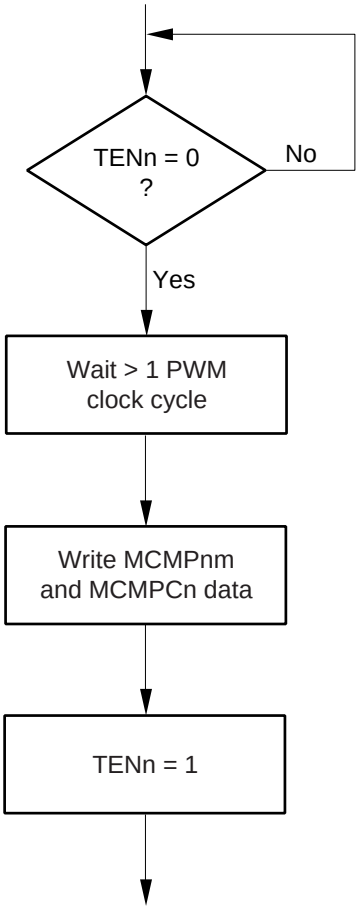
Remark: N = 00H to FFH

18.4.2 Update of PWM data

Confirm that bit 4 (TENn) of MCMPCn is 0, wait for more than one PWM clock cycle (as selected in SMSWI register), and then write to MCMPnm (8-bit PWM data) and to MCMPCn.
At the same time, set TENn to 1.
The data will be automatically transferred to the slave latch when the timer overflows, and the PWM data becomes valid. At the same time, TENn is automatically cleared to 0.

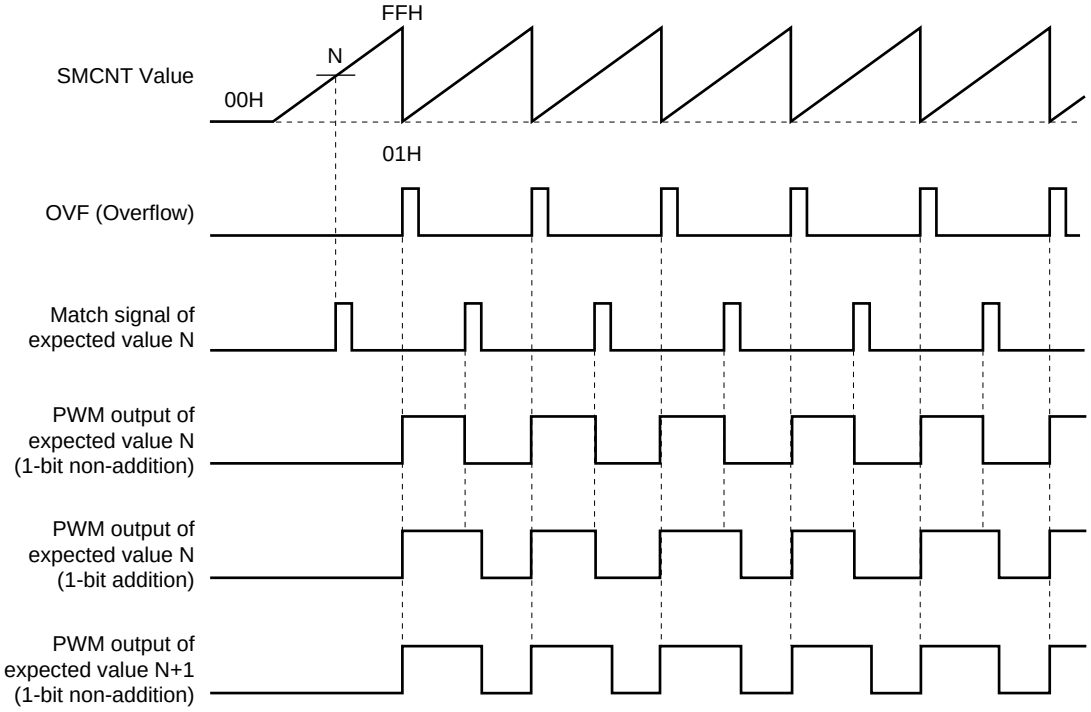
Remark: n = 1 to 4
m = 0, 1

Figure 18-8: Update of PWM data



18.4.3 Operation of 1-bit addition circuit

Figure 18-9: Timing in 1-bit Addition Circuit Operation

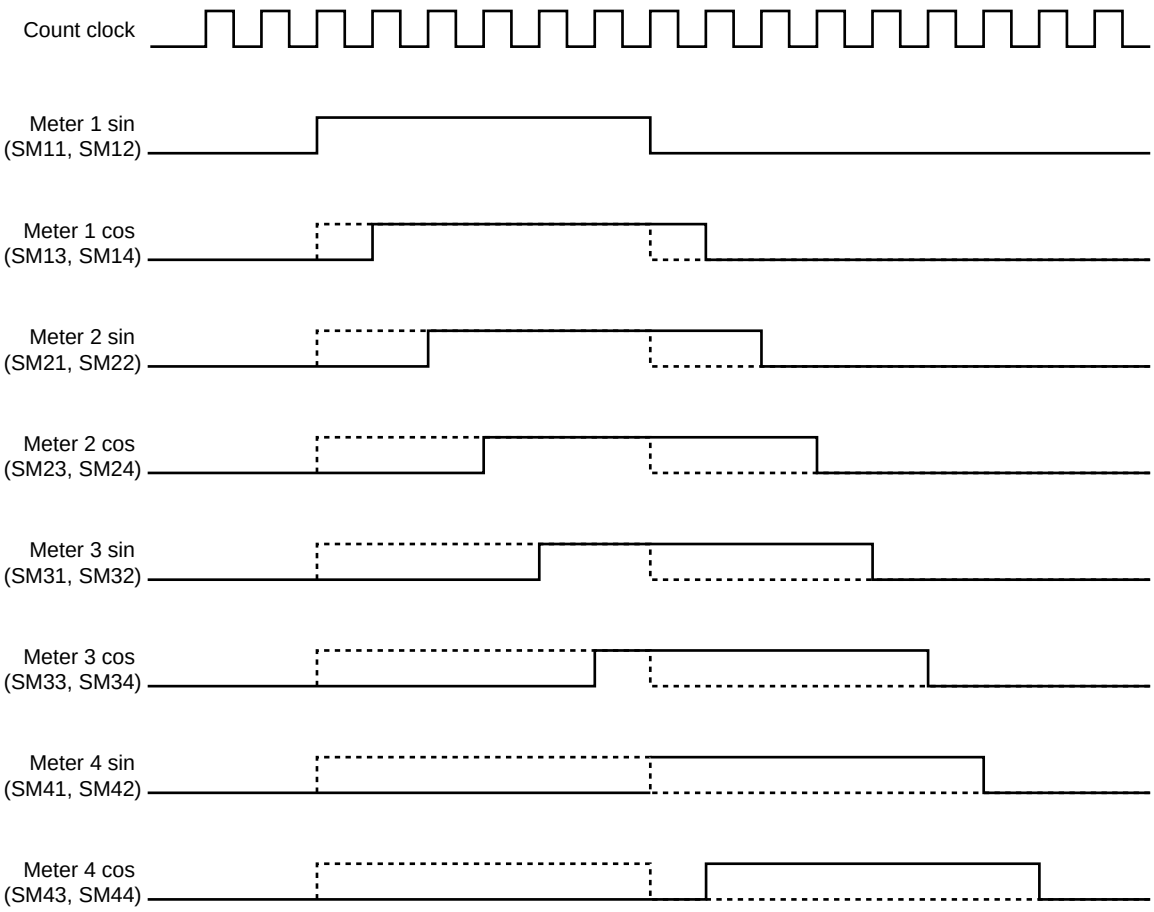


The 1-bit addition mode repeats 1-bit addition/non-addition to the PWM output every second SMCNT overflow. Therefore, the falling edge of the PWM output signal will occur at compare value N and compare value N+1 alternately. A 1-bit addition to the PWM output is applied by setting ADBn of the MCMPCn register to 1. In 1-bit non-addition mode the falling edge of the PWM output signal will always occur at compare value N+1 of SMCNT. A 1-bit non-addition (normal output) is applied by setting ADBn to 0.

Remark: n = 1 to 4

18.4.4 PWM output operation (output with 1 clock shifted)

Figure 18-10: Timing of Output with 1 Clock Shifted



If the wave of sin and cos of meters 1 to 4 rises and falls internally as indicated by the broken line, the SM11 to SM44 pins always shift the count clock by 1 clock. The output signals are generated in order to prevent V_{DD}/GND from fluctuating.

[MEMO]

Chapter 19 Interrupt Functions (µPD780822B Subseries)

19.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into a high interrupt priority group and a low interrupt priority group by setting the priority specify flag register (PR0L, PR0H, PR1L and PR1H).
Multiple high priority interrupts can be applied to low priority interrupts. If two or more interrupts with the same priority are simultaneously generated, each interrupts has a predetermined priority (see Table 19-1, “Interrupt Source List,” on page 474).
A standby release signal is generated.
The maskable interrupt has three sources of external interrupt requests and 25 sources of internal interrupt requests.

(2) Software interrupt

This is a vectored interrupt to be generated by executing the BRK instruction. It is acknowledged even in a disabled state. The software interrupt does not undergo interrupt priority control.

19.2 Interrupt Sources and Configuration

There are a total of 29 interrupt sources: maskable, and software interrupts.

Table 19-1: Interrupt Source List

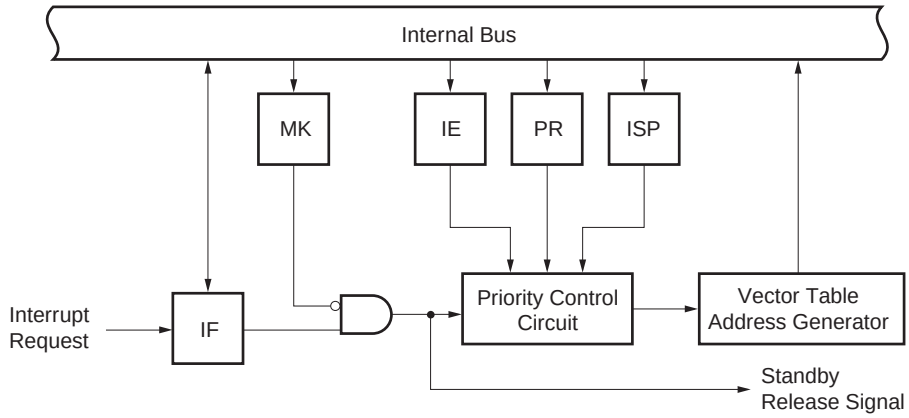
Type	Interrupt Priority Note 1	Interrupt Source		Internal/ External	Vector Address	Basic Structure Type Note 2
		Name	Trigger			
Reset	-		RESET input	External	0000H	-
			Clock Monitor main clock failure detection	Internal		
			Watchdog overflow (NMI mode)			
Maskable	0	INTWDT	Overflow of watchdog timer (interval timer mode)	Internal	0004H	(A)
	1	INTP0	Pin input edge detection	External	0006H	(B)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTOVF	Overflow of 16-bit timer 2	Internal	000EH	(A)
	5	INTTM20	16-bit timer capture register (CR20) match signal		0010H	
	6	INTTM21	16-bit timer capture register (CR21) match signal		0012H	
	7	INTTM22	Generation of 16-bit timer capture register (CR22) match signal		0014H	
	8	INTAD	End of A/D converter conversion		0016H	
	9	INTSRE6	UART6 reception error generation		0018H	
	10	INTSR6	UART6 end of reception		001AH	
	11	INTST6	UART6 end of transfer		001CH	
	12	INTCSI10	CSI1 end of serial transfer (channel 0)		001EH	
	13	INTCSI11	CSI1 end of serial transfer (channel 1)		0020H	
	14	INTTM50	8-bit timer/event counter 50 match signal		0022H	
	15	INTTM51	8-bit timer/event counter 51 match signal		0024H	
	16	INTWTI	Watch timer interval timer overflow		0026H	
	17	INTWT	Watch timer overflow		0028H	
	18	INTCE	CAN Error		002AH	
	19	INTCR	CAN Receive		002CH	
	20	INTCT0	CAN Transmit buffer 0		002EH	
	21	INTCT1	CAN Transmit buffer 1		0030H	
	22	INTTP0CC0	Timer TMP0 capture 0 trigger or compare 0 match		0032H	
	23	INTTP0CC1	Timer TMP0 capture 1 trigger or compare 1 match		0034H	
	24	INTTP1CC0	Timer TMP1 capture 0 trigger or compare 0 match		0036H	
	25	INTTP1CC1	Timer TMP1 capture 1 trigger or compare 1 match		0038H	
	26	INTTP2CC0	Timer TMP2 capture 0 trigger or compare 0 match		003AH	
	27	INTTP2CC1	Timer TMP2 capture 1 trigger or compare 1 match		003CH	
Software	-	BRK	BRK instruction execution	Internal	003EH	(C)

Notes: 1. Default priorities are intended for two or more simultaneously generated maskable interrupt requests. 0 is the highest priority and 28 is the lowest priority.

2. Basic configuration types (A) to (C) correspond to (A) to (C) of Figure 19-1 on page 475.

Figure 19-1: Basic Configuration of Interrupt Function (1/2)

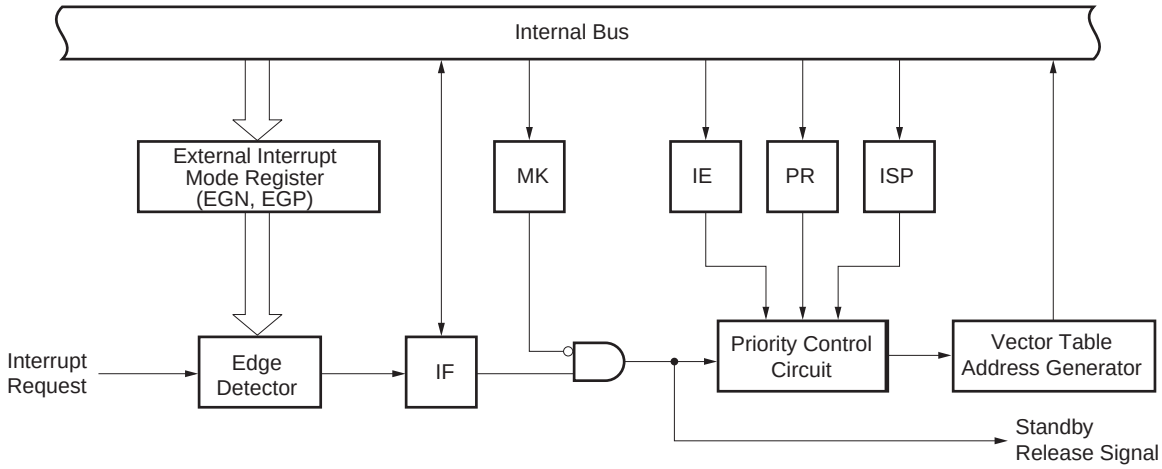
(A) Internal maskable interrupt



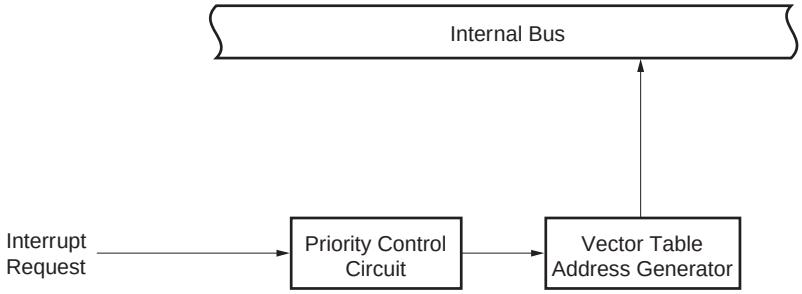
Remark: IF : Interrupt request flag
IE : Interrupt enable flag
ISP : In-service priority flag
MK : Interrupt mask flag
PR : Priority specify flag

Figure 19-1: Basic Configuration of Interrupt Function (2/2)

(B) External maskable interrupt



(C) Software interrupt



Remark: IF : Interrupt request flag
IE : Interrupt enable flag
ISP : In-service priority flag
MK : Interrupt mask flag
PR : Priority specify flag

19.3 Interrupt Function Control Registers

The following five types of registers are used to control the interrupt functions.

- Interrupt request flag register (IF0L, IF0H, IF1L, IF1H)
- Interrupt mask flag register (MK0L, MK0H, MK1L, MK1H)
- Priority specify flag register (PR0L, PR0H, PR1L, PR1H)
- External interrupt mode register (EGP, EGN)
- Program status word (PSW)

Table 19-2 gives a listing of interrupt request flags, interrupt mask flags, and priority specify flags corresponding to interrupt request sources.

Table 19-2: Various Flags Corresponding to Interrupt Request Sources

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag	Priority Specify Flag
INTWDT	WDTIF	WDTMK	WDTPR
INTP0	PIF0	PMK0	PPR0
INTP1	PIF1	PMK1	PPR1
INTP2	PIF2	PMK2	PPR2
INTOVF	OVFIF	OVFMK	OVFPR
INTTM20	TMIF20	TMMK20	TMPR20
INTTM21	TMIF21	TMMK21	TMPR21
INTTM22	TMIF22	TMMK22	TMPR22
INTAD	ADIF	ADMK	ADPR
INTSRE6	SREIF6	SREMK6	SREPR6
INTSR6	SRIF6	SRMK6	SRPR6
INTST6	STIF6	STMK6	STPR6
INTCSI10	CSIIF10	CSIMK10	CSIPR10
INTCSI11	CSIIF11	CSIMK11	CSIPR11
INTM50	TMIF50	TMMK50	TMPR50
INTM51	TMIF51	TMMK51	TMPR51
INTWTI	WTIIF	WTIMK	WTIPR
INTWT	WTIF	WTMK	WTPR
INTCE	CEIF	CEMK	CEPR
INTCR	RRF	CRMK	CRPR
INTCT0	CTIF0	CTMK0	CTPR0
INTCT1	CTIF1	CTMK1	CTPR1
INTTP0CC0	CC0IFP0	CC0MKP0	CC0PRP0
INTTP0CC1	CC1IFP0	CC1MKP0	CC1PRP0
INTTP1CC0	CC0IFP1	CC0MKP1	CC0PRP1
INTTP1CC1	CC1IFP1	CC1MKP1	CC1PRP1
INTTP2CC0	CC0IFP2	CC0MKP2	CC0PRP2
INTTP2CC1	CC1IFP2	CC1MKP2	CC1PRP2

(1) Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H)

The interrupt request flag is set to 1 when the corresponding interrupt request is generated. It is cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon application of $\overline{\text{RESET}}$ input.

IF0L, IF0H, IF1L and IF1H are set with an 1-bit or an 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register IF0, or if IF1L and IF1H are used as a 16-bit register IF1, use a 16-bit memory manipulation instruction for the setting.

$\overline{\text{RESET}}$ input sets these registers to 00H.

Figure 19-2: Interrupt Request Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
IF0L	TMIF21	TMIF20	OVFIF	0	PIF2	PIF1	PIF0	WDTIF	FFE0H	00H	R/W
IF0H	TMIF50	CSIIF11	CSIIF10	STIF6	SRIF6	SREIF6	ADIF	TMIF22	FFE1H	00H	R/W
IF1L	CC0IFP0	CTIF1	CTIF0	CRIF	CEIF	WTIF	WTIIF	TMIF51	FFE2H	00H	R/W
IF1H	0	0	0	CC1IFP2	CC0IFP2	CC1IFP1	CC0IFP1	CC1IFP0	FFE3H	00H	R/W

xxIFx	Interrupt request flag
0	No interrupt request signal
1	Interrupt request signal is generated; interrupt request state

Cautions: 1. Set always 0 in IF0L bit 4.

2. Set always 0 in IF1H bit 7, bit 6 and bit 5.

(2) Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H)

The interrupt mask flag is used to enable/disable the corresponding maskable interrupt service. MK0L, MK0H, MK1L and MK1H are set with an 1-bit or an 8-bit memory manipulation instruction. If MK0L and MK0H are used as a 16-bit register MK0, or if MK1L and MK1H are used as a 16-bit register MK1, use a 16-bit memory manipulation instruction for the setting.
$\overline{\text{RESET}}$ input sets these registers to FFH, except MK1H = DFH.

Figure 19-3: Interrupt Mask Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
MK0L	TMMK21	TMMK20	OVFMK	1	PMK2	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W
MK0H	TMMK50	CSIMK11	CSIMK10	STMK6	SRMK6	SREMK6	ADMK	TMMK22	FFE5H	FFH	R/W
MK1L	CC0MKP0	CTMK1	CTMK0	CRMK	CEMK	WTMK	WTIMK	TMMK51	FFE6H	FFH	R/W
MK1H	1	1	0	CC1MKP2	CC0MKP2	CC1MKP1	CC0MKP1	CC1MKP0	FFE7H	DFH	R/W

xxMKx	Interrupt Servicing Control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions: 1. Set always 1 in MK1L bit 4.

2. Set always 1 in MK1H bit 7 and bit 6.

3. Set always 0 in MK1H bit 5.

(3) Priority specify flag registers (PR0L, PR0H, PR1L, PR1H)

The priority specify flag is used to set the corresponding maskable interrupt priority orders.
The higher priority level is assigned when the corresponding bit in the PR register is reset.
PR0L, PR0H, PR1L and PR1H are set with an 1-bit or an 8-bit memory manipulation instruction. If PR0L and PR0H are used as a 16-bit register PR0, or if PR1L and PR1H are used as a 16-bit register PR1, use a 16-bit memory manipulation instruction for the setting.
$\overline{\text{RESET}}$ input sets these registers to FFH.

Figure 19-4: Priority Specify Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
PR0L	MPR21	MPR20	OVFPR	1	PPR2	PPR1	PPR0	WDTPR	FFE8H	FFH	R/W
PR0H	TMPR50	CSIPR11	CSIPR10	STPR6	SRPR6	SREPR6	ADPR	MPR22	FFE9H	FFH	R/W
PR1L	CC0PRP0	CTPR1	CTPR0	CRPR	CEPR	WTPR	WTIPR	TMPR51	FFEAH	FFH	R/W
PR1H	1	1	1	CC1PRP2	CC0PRP2	CC1PRP1	CC0PRP1	CC1PRP0	FFEBH	FFH	R/W

xxPRx	Priority Level Selection
0	High priority level
1	Low priority level

Cautions: 1. Set always 1 in PR0L bit 4.

2. Set always 1 in PR1H bit 7, bit 6 and bit 5.

(4) External interrupt rising edge enable register (EGP), External interrupt falling edge enable register (EGN)

EGP and EGN specify the valid edge to be detected on pins P00 to P02.
EGP and EGN can be read or written to with an 1-bit or an 8-bit memory manipulation instruction.
These registers are set to 00H when the $\overline{\text{RESET}}$ signal is output.

Figure 19-5: Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
EGP	0	0	0	0	0	EGP2	EGP1	EGP0	FF48H	00H	R/W
EGN	0	0	0	0	0	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	Valid edge of INTPn pin (n = 0 - 2)
0	0	Interrupt disable
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

(5) Program status word (PSW)

The program status word is a register to hold the instruction execution result and the current status for interrupt request. The IE flag to set maskable interrupts (enable/disable) and the ISP flag to control multiple interrupt servicing are mapped.

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, and when the BRK instruction is executed, the contents of PSW automatically is saved onto the stack and the IE flag is reset to 0. If a maskable interrupt request is acknowledged contents of the priority specify flag of the acknowledged interrupt are transferred to the ISP flag. The acknowledged contents of PSW is also saved onto the stack with the PUSH PSW instruction. It is retrieved from the stack with the RETI, RETB, and POP PSW instructions.

$\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 19-6: Program Status Word Format

Symbol	7	6	5	4	3	2	1	0	After Reset	R/W
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY	02H	R/W

ISP	Priority of Interrupt Currently Being Received
0	High-priority interrupt servicing (low-priority interrupt disable)
1	Interrupt request not acknowledged or low-priority interrupt servicing (all-maskable interrupts enable)

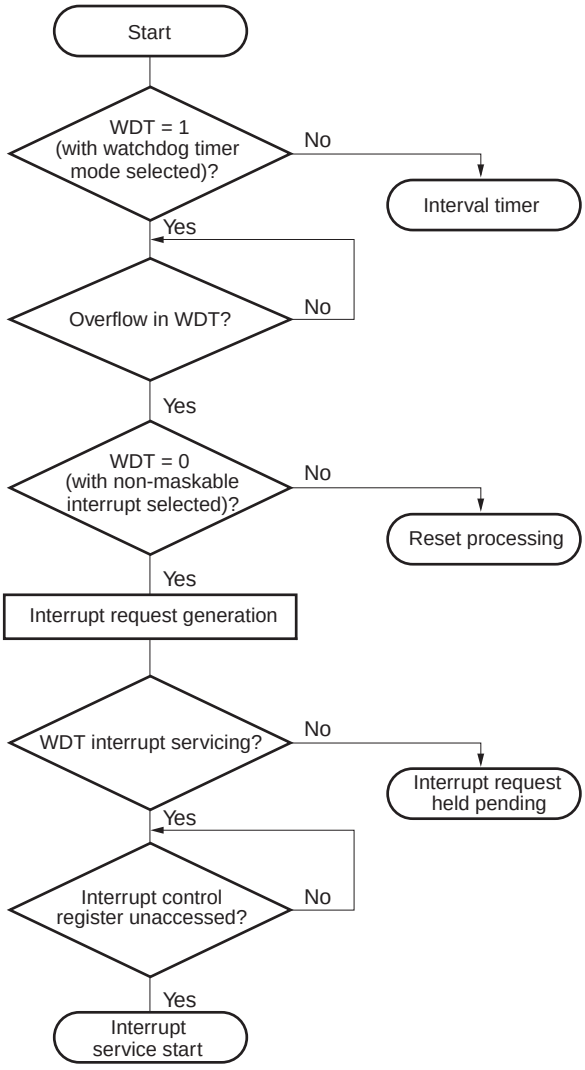
IE	Interrupt Request Acknowledge Enable/Disable
0	Disable
1	Enable

19.4 Interrupt Servicing Operations

19.4.1 Non-maskable interrupt request acknowledge operation

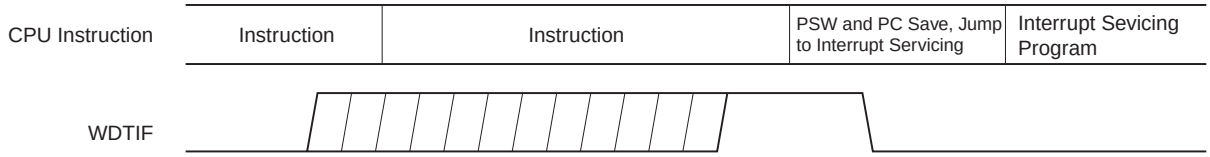
A non-maskable interrupt request is unconditionally acknowledged even if in an interrupt request acknowledge disable state. It does not undergo interrupt priority control and has highest priority over all other interrupts.
If a non-maskable interrupt request is acknowledged, PSW and PC are pushed on the stack. The IE and ISP flags are reset to 0, and the vector table contents are loaded into PC.
A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after the current execution of the non-maskable interrupt servicing program is terminated (following RETI instruction execution) and one main routine instruction is executed.
If a new non-maskable interrupt request is generated twice or more during a non-maskable interrupt service program execution, only one non-maskable interrupt request is acknowledged after termination of the non-maskable interrupt service program execution.

Figure 19-7: Flowchart from Non-Maskable Interrupt Generation to Acknowledge



Remark: WDTM : Watchdog timer mode register
WDT : Watchdog timer

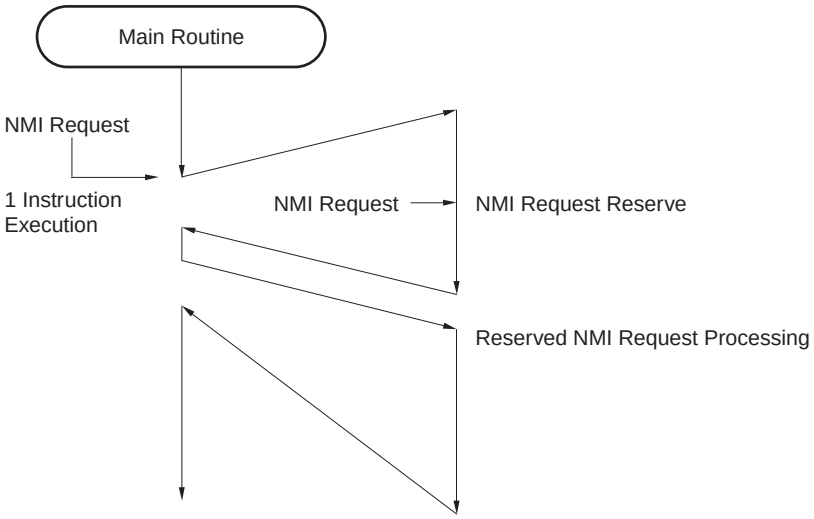
Figure 19-8: Non-Maskable Interrupt Request Acknowledge Timing



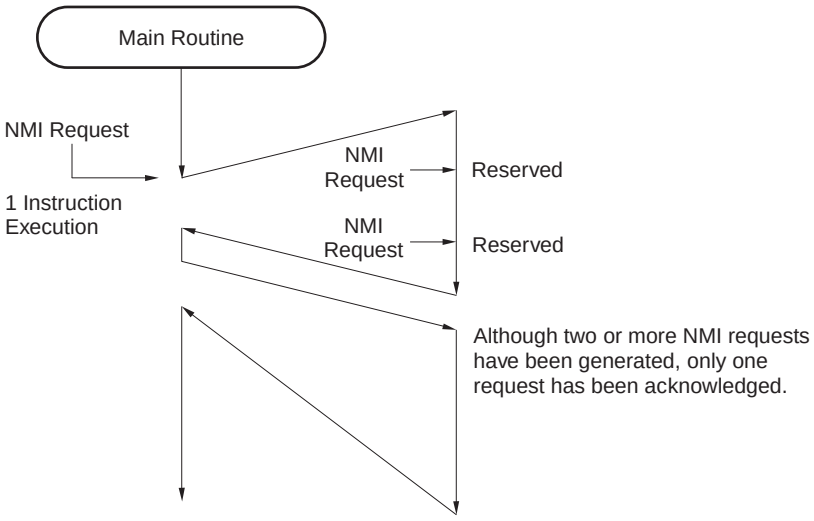
Remark: WDTIF : Watchdog timer interrupt request flag

Figure 19-9: Non-Maskable Interrupt Request Acknowledge Operation

(a) If a new non-maskable interrupt request is generated during non-maskable interrupt servicing program execution



(b) If two non-maskable interrupt requests are generated during non-maskable interrupt servicing program execution



19.4.2 Maskable interrupt request acknowledge operation

A maskable interrupt request becomes acknowledgeable when an interrupt request flag is set to 1 and the interrupt mask (MK) flag is cleared to 0. A vectored interrupt request is acknowledged in an interrupt enable state (with IE flag set to 1). However, a low-priority interrupt request is not acknowledged during high-priority interrupt service (with ISP flag reset to 0).
Wait times from maskable interrupt request generation to interrupt servicing are as follows.

Table 19-3: Times from Maskable Interrupt Request Generation to Interrupt Service

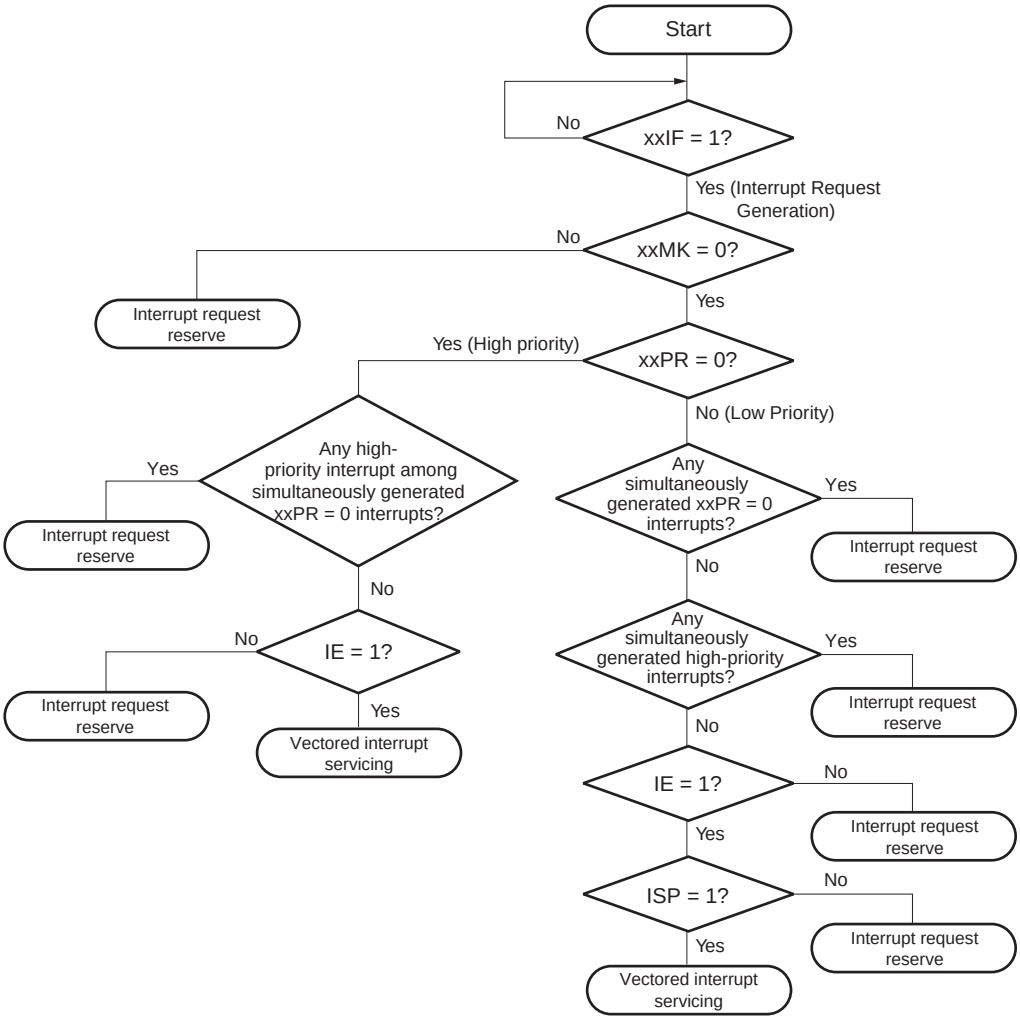
	Minimum Time	Maximum Time[Note]
When xxPRx = 0	7 clocks	32 clocks
When xxPRx = 1	8 clocks	33 clocks

Note: If an interrupt request is generated just before a divide instruction, the wait time is maximized.

Remark: 1 clock: $1/f_{CPU}$
(f_{CPU}: CPU clock)

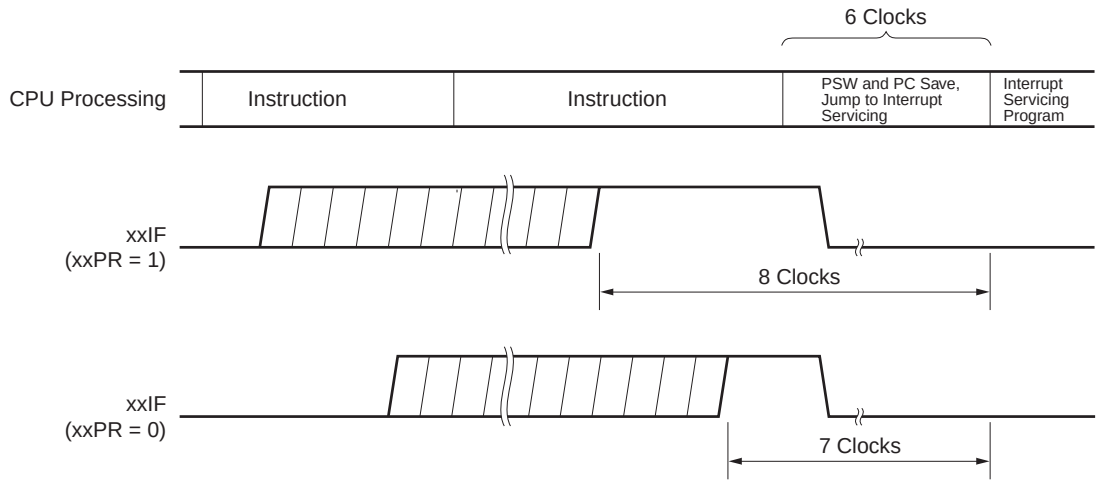
If two or more maskable interrupt requests are generated simultaneously, the request specified for higher priority with the priority specify flag is acknowledged first. If two or more requests are specified for the same priority with the priority specify flag, the interrupt request with the higher default priority is acknowledged first.
Any reserved interrupt requests are acknowledged when they become acknowledgeable.
Figure 19-10, “Interrupt Request Acknowledge Processing Algorithm,” on page 486 shows interrupt request acknowledge algorithms.
When a maskable interrupt request is acknowledged, the contents of program status word (PSW) and program counter (PC) are saved in this order onto the stack. Then, the IE flag is reset (to 0), and the value of the acknowledged interrupt priority specify flag is transferred to the ISP flag. Further, the vector table data determined for each interrupt request is loaded into PC and the program will branch accordingly.
Return from the interrupt is possible with the RETI instruction.

Figure 19-10: Interrupt Request Acknowledge Processing Algorithm



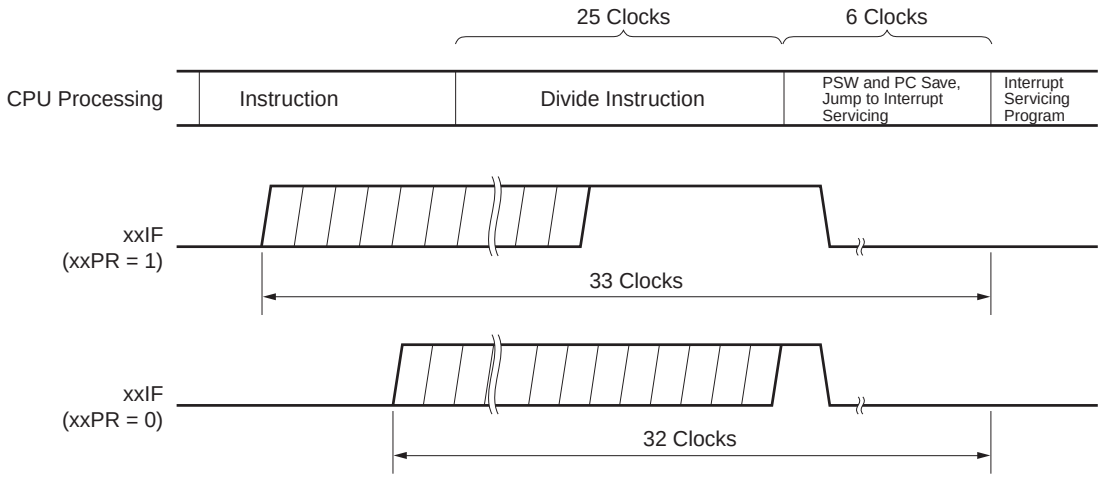
Remark: xxIF : Interrupt request flag
xxMK : Interrupt mask flag
xxPR : Priority specify flag
IE : Flag to control maskable interrupt request acknowledge
ISP : Flag to indicate the priority of interrupt being serviced
(0 = an interrupt with higher priority is being serviced,
1 = interrupt request is not acknowledged or an interrupt with lower priority is being serviced)

Figure 19-11: Interrupt Request Acknowledge Timing (Minimum Time)



Remark: 1 clock: 1/ f_{CPU}
(f_{CPU}: CPU clock)

Figure 19-12: Interrupt Request Acknowledge Timing (Maximum Time)



Remark: 1 clock: 1/ f_{CPU}
(f_{CPU}: CPU clock)

19.4.3 Software interrupt request acknowledge operation

A software interrupt request is acknowledged by BRK instruction execution. Software interrupt cannot be disabled.
If a software interrupt is acknowledged, the contents of program status word (PSW) and program counter (PC) are saved to stacks, in this order. Then the IE flag is reset (to 0), and the contents of the vector tables (003EH and 003FH) are loaded into PC and the program branches accordingly.
Return from the software interrupt is possible with the RETB instruction.

Caution: Do not use the RETI instruction for returning from the software interrupt.

19.4.4 Multiple interrupt servicing

A multiple interrupt service consists in acknowledging another interrupt during the execution of another interrupt routine.
A multiple interrupt service is generated only in the interrupt request acknowledge enable state (IE = 1) (except non-maskable interrupt). As soon as an interrupt request is acknowledged, it enters the acknowledge disable state (IE = 0). Therefore, in order to enable multiple interrupts, it is necessary to set the interrupt enable state by setting the IE flag (1) with the EI instruction during interrupt servicing.
Even in an interrupt enabled state, a multiple interrupt may not be enabled. However, it is controlled according to the interrupt priority. There are two priorities, the default priority and the programmable priority. The multiple interrupt is controlled by the programmable priority control.
If an interrupt request with the same or higher priority than that of the interrupt being serviced is generated, it is acknowledged as a multiple interrupt. In the case of an interrupt with a priority lower than that of the interrupt being processed, it is not acknowledged as a multiple interrupt.
An interrupt request not acknowledged as a multiple interrupt due to interrupt disable or a low priority is reserved and acknowledged following one instruction execution of the main processing after the completion of the interrupt being serviced.
During non-maskable interrupt servicing, multiple interrupts are not enabled.
Table 19-4, “Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing,” on page 489 shows an interrupt request enabled for multiple interrupt during interrupt servicing, and Figure 19-13, “Multiple Interrupt Example (1/2),” on page 490 shows multiple interrupt examples.

Table 19-4: Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing

Maskable Interrupt Request / Interrupt being serviced		Non-maskable Interrupt Request	Maskable Interrupt Request			
			xxPR = 0		xxPR = 1	
			IE = 1	IE = 0	IE = 1	IE = 0
Non-maskable interrupt		D	D	D	D	D
Maskable Interrupt	ISP = 0	E	E	D	D	D
	ISP = 1	E	E	D	E	D
Software interrupt		E	E	D	E	D

Remarks: 1. E : Multiple interrupt enabled

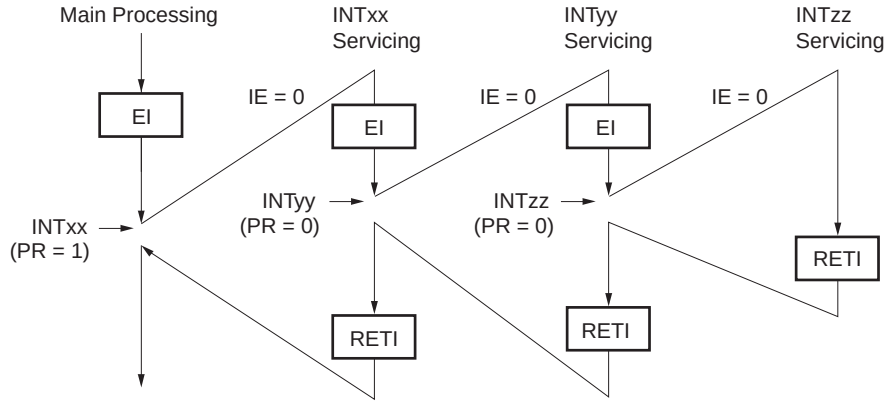
2. D : Multiple interrupt disabled

3. ISP and IE are the flags contained in PSW
ISP = 0 : An interrupt with higher priority is being serviced
ISP = 1 : An interrupt request is not accepted or an interrupt with lower priority is being serviced
IE = 0 : Interrupt request acknowledge is disabled
IE = 1 : Interrupt request acknowledge is enabled

4. xxPR is a flag contained in PR0L, PR0H, and PRIL
xxPR = 0 : Higher priority level
xxPR = 1 : Lower priority level

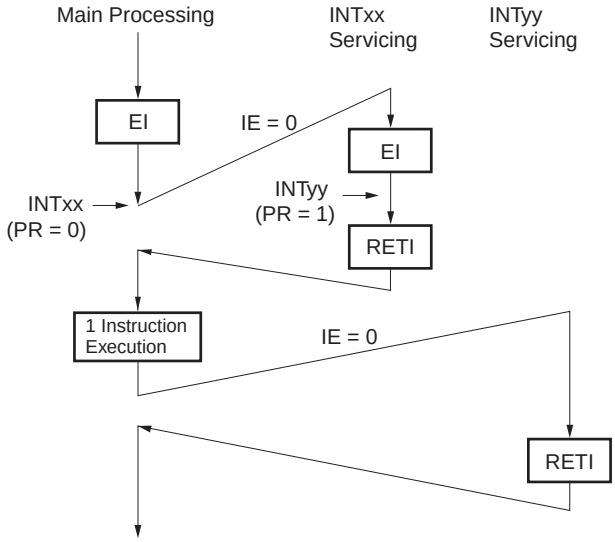
Figure 19-13: Multiple Interrupt Example (1/2)

(a) Example 1. Two multiple interrupts generated



During interrupt INTxx servicing, two interrupt requests, INTyy and INTzz are acknowledged, and a multiple interrupt is generated. An EI instruction is issued before each interrupt request acknowledge, and the interrupt request acknowledge enable state is set.

(b) Example 2. Multiple interrupt is not generated by priority control

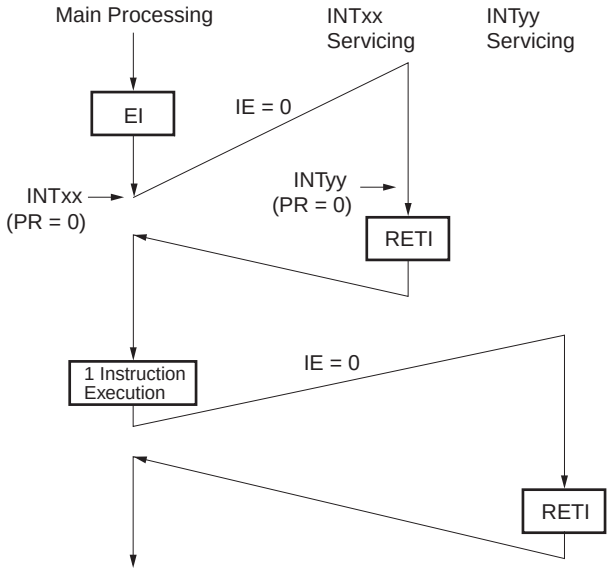


The interrupt request INTyy generated during interrupt INTxx servicing is not acknowledged because the interrupt priority is lower than that of INTxx, and a multiple interrupt is not generated. INTyy request is retained and acknowledged after execution of 1 instruction execution of the main processing.

Remark: PR = 0 : Higher priority level
PR = 1 : Lower priority level
IE = 0 : Interrupt request acknowledge disable

Figure 19-13: Multiple Interrupt Example (2/2)

(c) Example 3. A multiple interrupt is not generated because interrupts are not enabled



Because interrupts are not enabled in interrupt INTxx servicing (an EI instruction is not issued), interrupt request INTyy is not acknowledged, and a multiple interrupt is not generated. The INTyy request is reserved and acknowledged after 1 instruction execution of the main processing.

Remark: PR = 0 : Higher priority level
IE = 0 : Interrupt request acknowledge disable

19.4.5 Interrupt request reserve

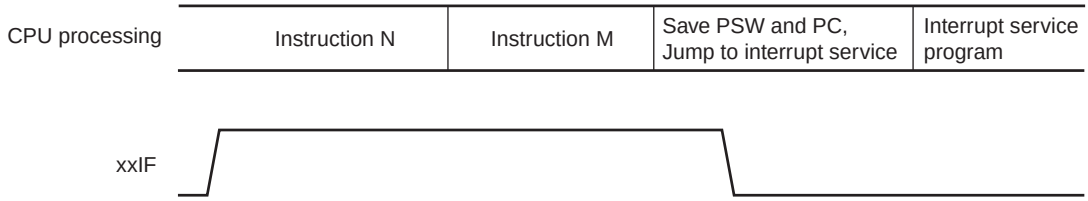
Some instructions may reserve the acknowledge of an instruction request until the completion of the execution of the next instruction even if the interrupt request is generated during the execution.
The following list shows such instructions (interrupt request reserve instruction).

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW.bit, CY
- MOV1 CY, PSW.bit
- AND1 CY, PSW.bit
- OR1 CY, PSW.bit
- XOR CY, PSW.bit
- SET1/CLR1 PSW.bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW.bit, $addr16
- BF PSW.bit, $addr16
- BTCLR PSW.bit, $addr16
- EI
- DI
- Manipulate instructions:
  for IF0L, IF0H, IF1L, MK0L, MK0H, MK1L, PR0L, PR0H, PR1L, EGP, EGN

Caution: BRK instruction is not an interrupt request reserve instruction described above. However, in a software interrupt started by the execution of BRK instruction, the IE flag is cleared to 0. Therefore, interrupt requests are not acknowledged even when a maskable interrupt request is issued during the execution of the BRK instruction. However, non-maskable interrupt requests are acknowledged.

Figure 19-14 shows the interrupt request hold timing.

Figure 19-14: Interrupt Request Hold



Remarks: **1.** Instruction N: Instruction that holds interrupts requests

2. Instruction M: Instructions other than interrupt request pending instruction

3. The xxPR (priority level) values do not affect the operation of xxIF (interrupt request).

[MEMO]

Chapter 20 Standby Function (μPD780822B Subseries)

20.1 Standby Function and Configuration

20.1.1 Standby function

Table 20-1: Relationship Between Operation Clocks in Each Operation Status

Status / Operation Mode	X1 Oscillator		Internal Low-speed Oscillator			Subsystem Clock Oscillator	CPU Clock after Release	Prescaler Clock Supplied to Peripherals	
	MSTOP = 0 MCC = 0	MSTOP = 1 MCC = 1	Note 1	Note 2				MCM0 = 0	MCM0 = 1
				RSTOP = 0	RSTOP = 1				
Reset	Stopped		Stopped			Oscillating	Internal low-speed oscillator	Stopped	
STOP			Oscillating	Oscillating	Stopped		Note 3	Stopped	
HALT	Oscillating	Stopped					Note 4	Internal low-speed oscillator	X1

Notes: 1. When “Cannot be stopped” is selected for internal low-speed oscillator by an option byte / a mask option.

2. When “Can be stopped by software” is selected for internal low-speed oscillator by an option byte / a mask option.

3. Operates using the CPU clock at STOP instruction execution.

4. Operates using the CPU clock at HALT instruction execution.

Caution: The RSTOP setting is valid only when “Can be stopped by software” is set for internal low-speed oscillator by an option byte.

Remark: MSTOP: Bit 7 of the main OSC control register (MOC)
MCC: Bit 7 of the processor clock control register (PCC)
RSTOP: Bit 0 of the internal low-speed oscillator mode register (RCM)
MCM0: Bit 0 of the main clock mode register (MCM)

The standby function is designed to reduce the operating current of the system. The following two modes are available.

(1) HALT mode

HALT instruction execution sets the HALT mode. In the HALT mode, the CPU operation clock is stopped. If the X1 oscillator, internal low-speed oscillator, or subsystem clock oscillator is operating before the HALT mode is set, oscillation of each clock continues. In this mode, the operating current is not decreased as much as in the STOP mode, but the HALT mode is effective for restarting operation immediately upon interrupt request generation and carrying out intermittent operations.

(2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the X1 oscillator stops, stopping the whole system, thereby considerably reducing the CPU operating current.
Because this mode can be cleared by an interrupt request, it enables intermittent operations to be carried out. However, because a wait time is required to secure the oscillation stabilization time after the STOP mode is released, select the HALT mode if it is necessary to start processing immediately upon interrupt request generation.

In either of these two modes, all the contents of registers, flags and data memory just before the standby mode is set are held. The I/O port output latches and output buffer statuses are also held.

Cautions: 1. STOP mode can be used only when CPU is operating on the X1 input clock or internal low-speed oscillator clock. HALT mode can be used when CPU is operating on the X1 input clock, internal low-speed oscillator clock, or subsystem clock. However, when the STOP instruction is executed during internal low-speed oscillator clock operation, the X1 oscillator stops, but internal low-speed oscillator does not stop.

2. When shifting to the STOP mode, be sure to stop the peripheral hardware operation before executing STOP instruction.

3. The following sequence is recommended for operating current reduction of the A/D converter when the standby function is used: First clear bit 7 (ADCS) of the A/D converter mode register (ADM) to 0 to stop the A/D conversion operation, and then execute the HALT or STOP instruction.

4. If the internal low-speed oscillator is operating before the STOP mode is set, oscillation of the internal low-speed oscillator clock cannot be stopped in the STOP mode. However, when the internal low-speed oscillator clock is used as the CPU clock, the CPU operation is stopped for $17/f_R$ (s) after STOP mode is released.

20.1.2 Registers controlling standby function

The standby function is controlled by the following two registers.

- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

Remark: For the registers that start, stop, or select the clock, see **Chapter 5 "Clock Generator (µPD780822B Subseries)" on page 125**.

(1) Oscillation stabilization time counter status register (OSTC)

This is the status register of the X1 input clock oscillation stabilization time counter. If the internal low-speed oscillator clock is used as the CPU clock, the X1 input clock oscillation stabilization time can be checked.

OSTC can be read by an 1-bit or an 8-bit memory manipulation instruction.

Reset release (reset by $\overline{\text{RESET}}$ input, clock monitor, and WDT), the STOP instruction, MSTOP (bit 7 of MOC register) = 1, and MCC (bit 7 of PCC register) = 1 clear OSTC to 00H.

Figure 20-1: Format of Oscillation Stabilization Time Counter Status Register (OSTC)

	7	6	5	4	3	2	<1>	<0>	R/W	Address	After Reset
OSTC	0	0	0	MOST11	MOST13	MOST14	MOST15	MOST16	R	FFA3H	00H

MOST11	MOST13	MOST14	MOST15	MOST16	Oscillation stabilization time status		
						f_X = 10 MHz	f_X = 16 MHz
1	0	0	0	0	$2^{11}/f_{XP}$ min.	204.8 μs min.	128 μs min.
1	1	0	0	0	$2^{13}/f_{XP}$ min.	819.2 μs min.	512 μs min.
1	1	1	0	0	$2^{14}/f_{XP}$ min.	1.64 ms min.	1.02 ms min.
1	1	1	1	0	$2^{15}/f_{XP}$ min.	3.27 ms min.	2.04 ms min.
1	1	1	1	1	$2^{16}/f_{XP}$ min.	6.55 ms min.	4.09 ms min.

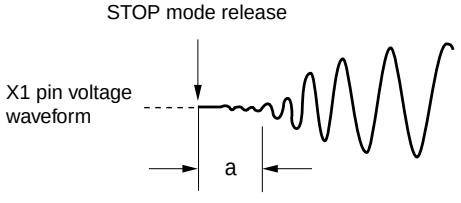
Cautions: 1. After the above time has elapsed, the bits are set to 1 in order from MOST11 and remain 1.

2. If the STOP mode is entered and then released while the internal low-speed oscillator clock is being used as the CPU clock, set the oscillation stabilization time as follows.

• Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS

The X1 oscillation stabilization time counter counts only during the oscillation stabilization time set by OSTS. Therefore, note that only the statuses during the oscillation stabilization time set by OSTS are set to OSTC after STOP mode has been released.

3. The wait time when STOP mode is released does not include the time after STOP mode release until clock oscillation starts ("a" below) regardless of whether STOP mode is released by $\overline{\text{RESET}}$ input or interrupt generation.



Remark: f_X: X1 input clock oscillation frequency

(2) Oscillation stabilization time select register (OSTS)

This register is used to select the X1 oscillation stabilization wait time when STOP mode is released. The wait time set by OSTS is valid only after STOP mode is released when the X1 input clock is selected as the CPU clock. After STOP mode is released when the internal low-speed oscillator clock is selected, check the oscillation stabilization time using OSTC.

OSTS can be set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets OSTS to 05H.

Figure 20-2: Format of Oscillation Stabilization Time Select Register (OSTS)

	7	6	5	4	3	2	<1>	<0>	R/W	Address	After Reset
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	R/W	FFA4H	05H

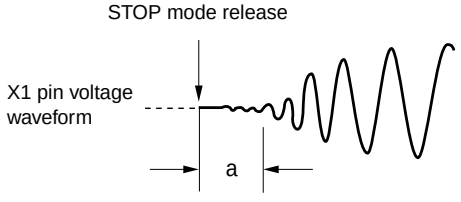
OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection		
				f_X = 10 MHz	f_X = 16 MHz
0	0	1	$2^{11}/f_{XP}$	204.8 μs	128 μs
0	1	0	$2^{13}/f_{XP}$	819.2 μs	512 μs
0	1	1	$2^{14}/f_{XP}$	1.64 ms	1.02 ms
1	0	0	$2^{15}/f_{XP}$	3.27 ms	2.04 ms
1	0	1	$2^{16}/f_{XP}$	6.55 ms	4.09 ms
Other than above			Setting prohibited		

Cautions: 1. If the STOP mode is entered and then released while the internal low-speed oscillator clock is being used as the CPU clock, set the oscillation stabilization time as follows.

• Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS

The X1 oscillation stabilization time counter counts only during the oscillation stabilization time set by OSTS. Therefore, note that only the statuses during the oscillation stabilization time set by OSTS are set to OSTC after STOP mode has been released.

2. The wait time when STOP mode is released does not include the time after STOP mode release until clock oscillation starts (“a” below) regardless of whether STOP mode is released by $\overline{\text{RESET}}$ input or interrupt generation.



Remark: f_{XP}: X1 input clock oscillation frequency

20.2 Standby Function Operation

20.2.1 HALT mode

(1) HALT mode

The HALT mode is set by executing the HALT instruction. HALT mode can be set regardless of whether the CPU clock before the setting was the X1 input clock, internal low-speed oscillator clock, or subsystem clock. The operating states in the HALT mode are shown below.

Table 20-2: Operating States in HALT Mode (1/3)

(a) When HALT Instruction is executed while CPU is Operating on X1 Input Clock

HALT Mode Setting / Item		when Internal Low-speed Oscillator Oscillation continues		when Internal Low-speed Oscillator Oscillation stopped[Note 1]	
		when Subsystem Clock used	when Subsystem Clock not used	when Subsystem Clock used	when Subsystem Clock not used
System clock		Clock supply to the CPU is stopped.			
CPU		Operation stopped			
Port (latch)		Status before HALT mode was set is retained			
16-bit timer TM20		Operable			
16-bit timer TMP0		Operable			
16-bit timer TMP1		Operable		Operable[Note 2]	
16-bit timer TMP2		Operable			
8-bit timer/event counter 50		Operable			
8-bit timer/event counter 51		Operable			
Watch timer		Operable	Operable[Note 2]	Operable	Operable[Note 2]
Watchdog timer	Internal low-speed oscillator cannot be stopped[Note 3]	Operable		-	
	Internal low-speed oscillator can be stopped[Note 3]	Operation stopped			
A/D converter		Operable			
Serial interface	UART6	Operable			
	CSI10	Operable			
	CSI11	Operable			
Clock output/buzzer output controller		Operable	Operable[Note 2]	Operable	Operable[Note 2]
CAN		DCAN has to be in SLEEP mode			
Sound Generator		Operable			
LCD C/D		Operable			
Stepper Motor C/D		Operable			
Clock Monitor		Operable		Operation stopped	
External interrupt		Operable			

Notes: 1. When “Stopped by software” is selected for internal low-speed oscillator by an option byte / a mask option and internal low-speed oscillator is stopped by software (for option bytes, see **Chapter 24 ”Mask Options / Option Bytes (µPD780822B Subseries)” on page 549**).

2. Operable when the X1 input clock is selected.

3. “Internal low-speed oscillator cannot be stopped” or “Internal low-speed oscillator can be stopped by software” can be selected by an option byte.

Table 20-2: Operating States in HALT Mode (2/3)

(b) Operating States in HALT Mode while CPU is Operating on Internal Low-speed Oscillator Clock

HALT Mode Setting / Item		when X1 Input Clock Oscillation continues		when X1 Input Clock Oscillation stopped	
		when Subsystem Clock used	when Subsystem Clock not used	when Subsystem Clock used	when Subsystem Clock not used
System clock		Clock supply to the CPU is stopped.			
CPU		Operation stopped			
Port (latch)		Status before HALT mode was set is retained			
16-bit timer TM20		Operable		Operation stopped	
16-bit timer TMP0		Operable		Operation stopped	
16-bit timer TMP1		Operable		Operation stopped	
16-bit timer TMP2		Operable		Operation stopped	
8-bit timer/event counter 50		Operation not guaranteed when count clock other than TI50 is selected			
8-bit timer/event counter 51		Operation not guaranteed when count clock other than TI51 is selected			
Watch timer		Operable[Note 1]	Operation not guaranteed	Operable[Note 1]	Operation not guaranteed
Watchdog timer	Internal low-speed oscillator cannot be stopped[Note 2]	Operable			
	Internal low-speed oscillator can be stopped[Note 2]	Operation stopped			
A/D converter		Operation not guaranteed			
Serial interface	UART6	Operation not guaranteed when serial clock other than TM50 output is selected during 8-bit timer/event counter 50 operation			
	CSI10	Operation not guaranteed when serial clock other than external $\overline{\text{SCK10}}$ is selected			
	CSI11	Operation not guaranteed when serial clock other than external $\overline{\text{SCK11}}$ is selected			
Clock output/buzzer output controller		Operable[Note 1]	Operation not guaranteed	Operable[Note 1]	Operation not guaranteed
CAN		DCAN has to be in SLEEP mode		Operation stopped	
Sound Generator		Operable		Operation stopped	
LCD C/D		Operable		Operation stopped	
Stepper Motor C/D		Operable		Operation not guaranteed	
Clock Monitor		Operable		Operation stopped	
External interrupt		Operable			

Notes: 1. Operation not guaranteed when other than subsystem clock is selected.

2. “Internal low-speed oscillator cannot be stopped” or “Internal low-speed oscillator can be stopped by software” can be selected by an option byte.

Table 20-2: Operating States in HALT Mode (3/3)

(c) Operating States in HALT Mode while CPU is Operating on Subsystem Clock

HALT Mode Setting / Item		when X1 Input Clock Oscillation continues		when X1 Input Clock Oscillation stopped	
		when internal low-speed oscillator oscillation continues	when internal low-speed oscillator oscillation stopped **Note 1**	when internal low-speed oscillator oscillation continues	when internal low-speed oscillator oscillation stopped **Note 1**
System clock		Clock supply to the CPU is stopped.			
CPU		Operation stopped			
Port (latch)		Status before HALT mode was set is retained			
16-bit timer TM20		Operable		Operation stopped	
16-bit timer TMP0		Operable		Operation stopped	
16-bit timer TMP1		Operable	Operable **Note 2**	Operation stopped	
16-bit timer TMP2		Operable		Operation stopped	
8-bit timer/event counter 50		Operable		Operable only when TI50 is selected as the count clock	
8-bit timer/event counter 51		Operable		Operable only when TI51 is selected as the count clock	
Watch timer		Operable		Operable only when subsystem clock is selected	
Watchdog timer	Internal low-speed oscillator cannot be stopped **Note 2**	Operable	-	Operable	-
	Internal low-speed oscillator can be stopped **Note 2**	Operation stopped			
A/D converter		Operable		Not operable	
Serial interface	UART6	Operable		Operable only when TM50 output is selected as the serial clock during 8-bit timer/event counter 50 operation	
	CSI10	Operable		Operable only when external clock is selected as the serial clock	
	CSI11	Operable		Operable only when external clock is selected as the serial clock	
Clock output/buzzer output controller		Operable		Operable only when subsystem clock is selected	
CAN		DCAN has to be in SLEEP mode		Operation stopped	
Sound Generator		Operable		Operation stopped	
LCD C/D		Operable		Operation stopped	
Stepper Motor C/D		Operable		Operation not guaranteed	Operation stopped
Clock Monitor		Operable		Operation stopped	
External interrupt		Operable			

Notes: 1. When “Stopped by software” is selected for internal low-speed oscillator by an option byte / a mask option and internal low-speed oscillator is stopped by software (for option bytes, see **Chapter 24 ”Mask Options / Option Bytes (µPD780822B Subseries)” on page 549**).

2. “Internal low-speed oscillator cannot be stopped” or “Internal low-speed oscillator can be stopped by software” can be selected by an option byte.

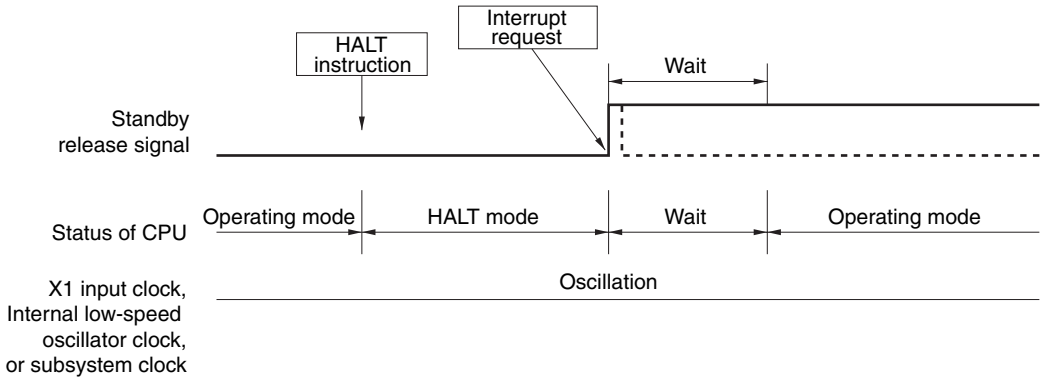
(2) HALT mode release

The HALT mode can be released by the following two sources.

(a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledgement is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgement is disabled, the next address instruction is executed.

Figure 20-3: HALT Mode Release by Interrupt Request Generation



Remarks: 1. The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

2. The wait time is as follows:

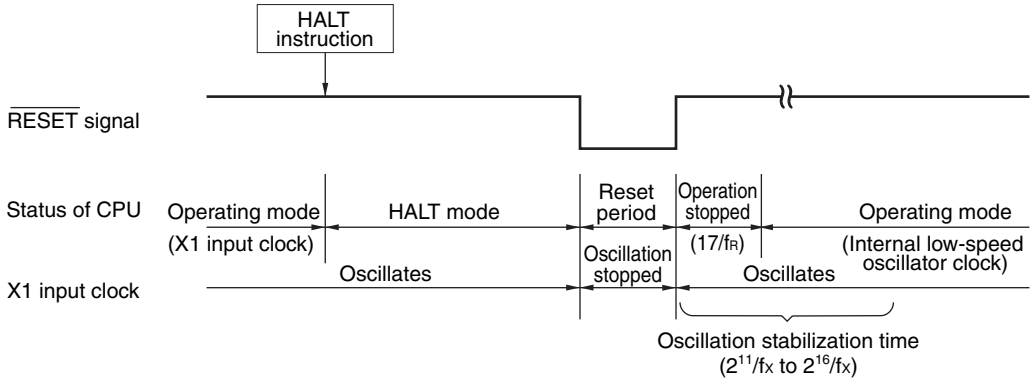
- When vectored interrupt servicing is carried out: 8 or 9 clocks
- When vectored interrupt servicing is not carried out: 2 or 3 clocks

(3) Release by $\overline{\text{RESET}}$ input

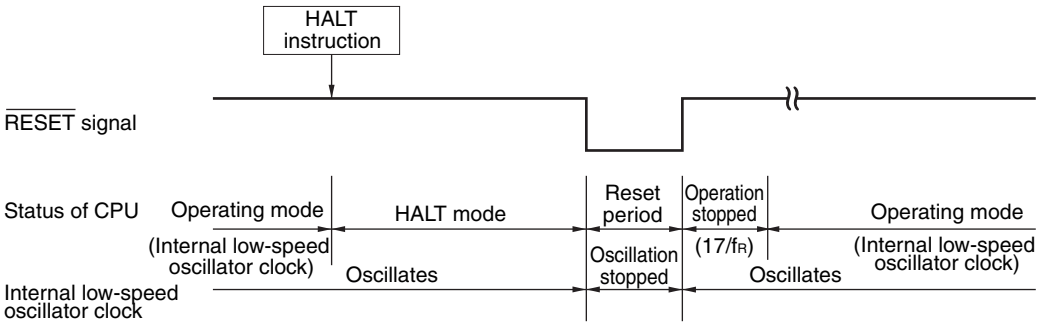
When the $\overline{\text{RESET}}$ signal is input, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

Figure 20-4: HALT Mode Release by $\overline{\text{RESET}}$ Input (1/2)

(a) When X1 input clock is used as CPU clock



(b) When internal low-speed oscillator clock is used as CPU clock

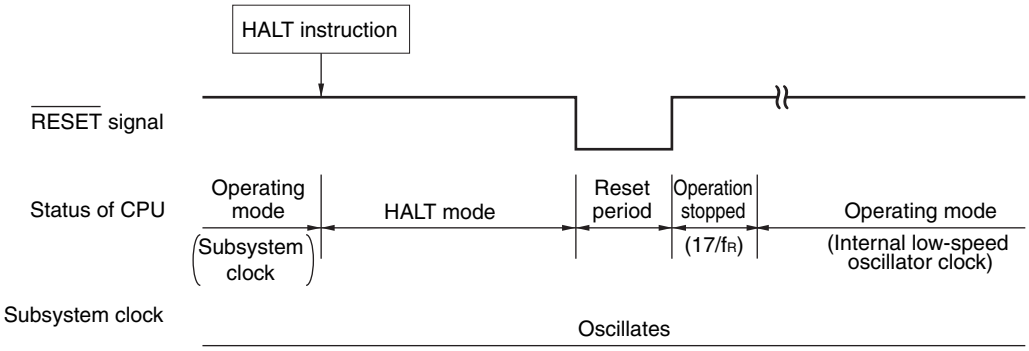


Remarks: 1. f_X: X1 input clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

Figure 20-4: HALT Mode Release by RESET Input (2/2)

(c) When subsystem clock is used as CPU clock



Remark: f_R: Internal low-speed oscillator clock oscillation frequency

Table 20-3: Operation in Response to Interrupt Request in HALT Mode

Release Source	MK××	PR××	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt servicing execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt servicing execution
	1	×	×	×	HALT mode held
$\overline{\text{RESET}}$ input	-	-	×	×	Reset processing

Remark: ×: don't care

20.2.2 STOP mode

(1) STOP mode setting and operating states

The STOP mode is set by executing the STOP instruction, and it can be set when the CPU clock before the setting was the X1 input clock or internal low-speed oscillator clock.

Caution: Because the interrupt request signal is used to clear the standby mode, if there is an interrupt source with the interrupt request flag set and the interrupt mask flag reset, the standby mode is immediately cleared if set. Thus, the STOP mode is reset to the HALT mode immediately after execution of the STOP instruction and the system returns to the operating mode as soon as the wait time set using the oscillation stabilization time select register (OSTS) has elapsed.

The operating states in the STOP mode are shown below.

Table 20-4: Operating States in STOP Mode (1/2)

STOP Mode Setting		when STOP Instruction is executed while CPU is operating on X1 Input Clock				when STOP Instruction is executed while CPU is operating on internal low-speed oscillator Clock	
		when internal low-speed oscillator oscillation continues		when internal low-speed oscillator oscillation stopped-Note 1			
Item		when subsystem clock used	when subsystem clock not used	when subsystem clock used	when subsystem clock not used	when subsystem clock used	when subsystem clock not used
System clock		Only X1 oscillator oscillation is stopped. Clock supply to the CPU is stopped.					
CPU		Operation stopped					
Port (latch)		Status before STOP mode was set is retained					
16-bit timer TM20		Operation stopped					
16-bit timer TMP0		Operation stopped					
16-bit timer TMP1		Operation stopped					
16-bit timer TMP2		Operation stopped					
8-bit timer/event counter 50		Operable only when TI50 is selected as the count clock					
8-bit timer/event counter 51		Operable only when TI51 is selected as the count clock					
Watch timer		Operable[Note 2]	Operation stopped	Operable[Note 2]	Operation stopped	Operable[Note 2]	Operation stopped
Watch-dog timer	Internal low-speed oscillator cannot be stopped[Note 3]	Operable		-		Operable	
	Internal low-speed oscillator can be stopped-Note 3	Operation stopped					

Notes: 1. When “Stopped by software” is selected for internal low-speed oscillator by an option byte and internal low-speed oscillator is stopped by software (for option bytes, see **Chapter 24 ”Mask Options / Option Bytes (µPD780822B Subseries)” on page 549**).

2. Operable when the subsystem clock is selected

3. “Internal low-speed oscillator cannot be stopped” or “Internal low-speed oscillator can be stopped by software” can be selected by an option byte.

Table 20-4: Operating States in STOP Mode (2/2)

STOP Mode Setting		when STOP Instruction is executed while CPU is operating on X1 Input Clock				when STOP Instruction is executed while CPU is operating on internal low-speed oscillator Clock	
		when internal low-speed oscillator oscillation continues		when internal low-speed oscillator oscillation stopped-Note 1			
Item		when subsystem clock used	when subsystem clock not used	when subsystem clock used	when subsystem clock not used	when subsystem clock used	when subsystem clock not used
A/D converter		Operation stopped					
Serial interface	UART6	Operable only when TM50 output is selected as the serial clock during TM50 operation					
	CSI10	Operable only when external $\overline{SCK10}$ is selected as the serial clock					
	CSI11	Operable only when external $\overline{SCK11}$ is selected as the serial clock					
Clock output/buzzer output controller		Operable[Note 2]	Operation stopped	Operable[Note 2]	Operation stopped	Operable[Note 2]	Operation stopped
CAN		Operation stopped[Note 3]					
Sound Generator		Operation stopped					
LCD C/D		Operation stopped					
Stepper Motor C/D		Operation stopped					
Clock monitor		Operation stopped					
External interrupt		Operable					

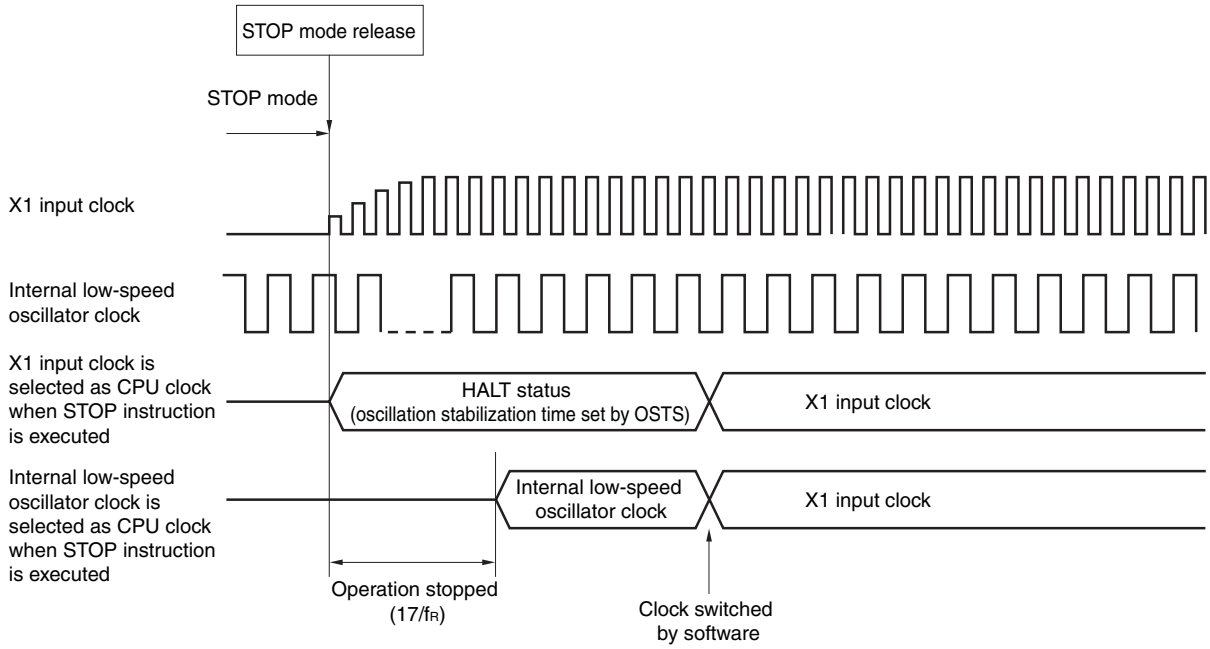
Notes: 1. When “Stopped by software” is selected for internal low-speed oscillator by an option byte and internal low-speed oscillator is stopped by software (for option bytes, see **Chapter 24 ”Mask Options / Option Bytes (µPD780822B Subseries)” on page 549**).

2. Operable when the subsystem clock is selected

3. DCAN has to be set in initialisation state, SLEEP mode or STOP mode before the CPU is switched into STOP mode. For details please refer to **15.17 ”Influence of the standby Function of the CAN Controller” on page 423**.

(2) STOP mode release

Figure 20-5: Operation Timing When STOP Mode Is Released



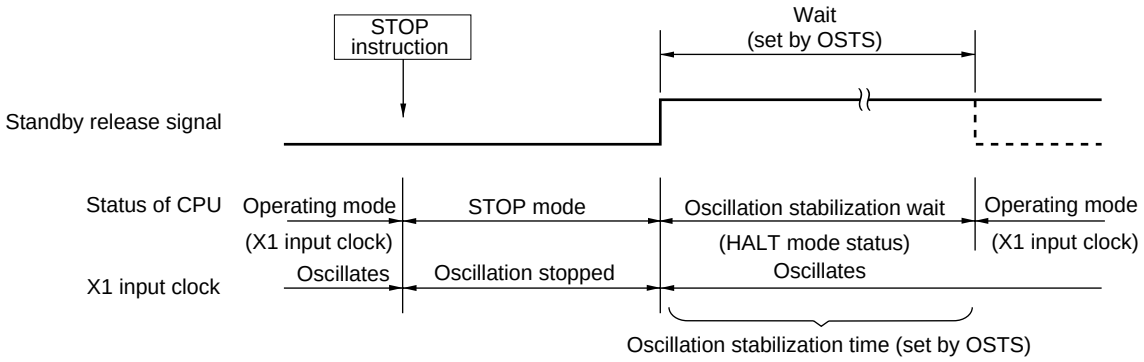
The STOP mode can be released by the following two sources.

(a) Release by unmasked interrupt request

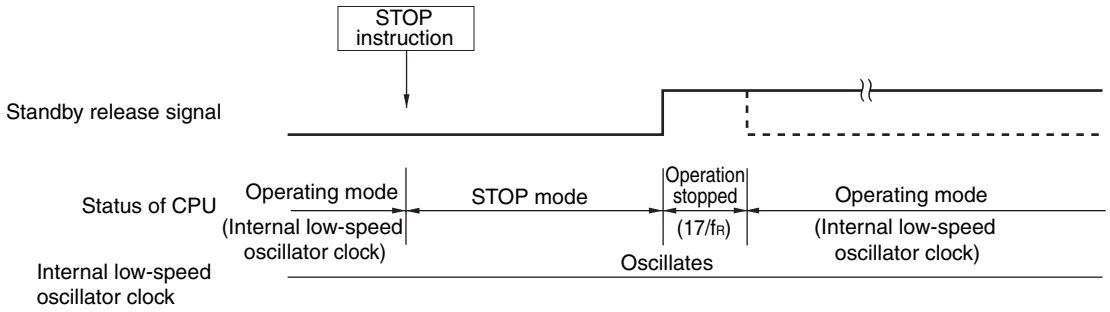
When an unmasked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

Figure 20-6: STOP Mode Release by Interrupt Request Generation

(a) When X1 input clock is used as CPU clock



(b) When internal low-speed oscillator clock is used as CPU clock



Remarks: 1. The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

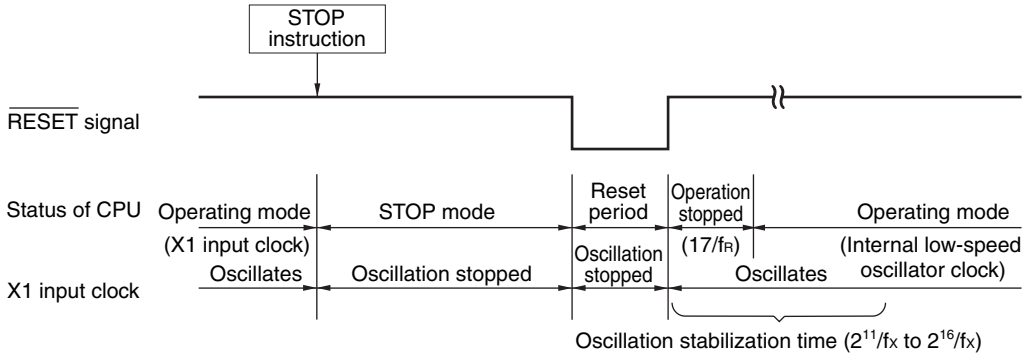
2. f_R: Internal low-speed oscillator clock oscillation frequency

(3) Release by $\overline{\text{RESET}}$ input

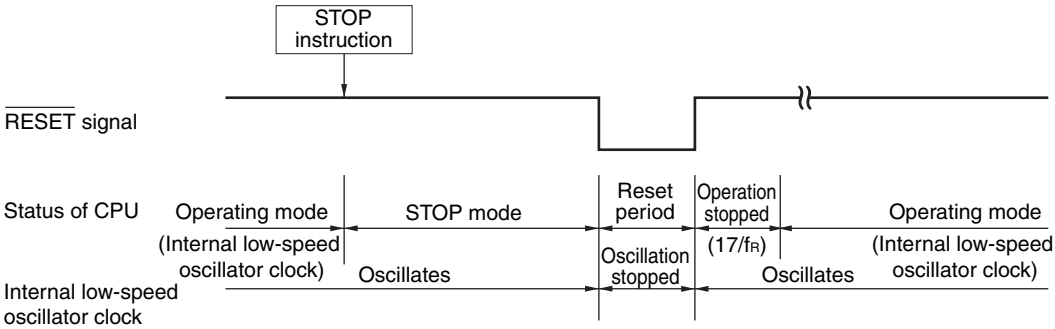
When the $\overline{\text{RESET}}$ signal is input, STOP mode is released and a reset operation is performed after the oscillation stabilization time has elapsed.

Figure 20-7: STOP Mode Release by $\overline{\text{RESET}}$ Input

(a) When X1 input clock is used as CPU clock



(b) When internal low-speed oscillator clock is used as CPU clock



Remarks: 1. f_X: X1 input clock oscillation frequency

2. f_R: Internal low-speed oscillator clock oscillation frequency

Table 20-5: Operation in Response to Interrupt Request in STOP Mode

Release Source	MK××	PR××	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt servicing execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt servicing execution
	1	×	×	×	STOP mode held
$\overline{\text{RESET}}$ input	-	-	×	×	Reset processing

Remark: ×: don't care

Chapter 21 Reset Function (µPD780822B Subseries)

The following three operations are available to generate a reset signal.

(1) External reset input via $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer program loop detection
(3) Internal reset by clock monitor X1 clock oscillation stop detection

External and internal resets have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H when the reset signal is input.
A reset is applied when a low level is input to the $\overline{\text{RESET}}$ pin, the watchdog timer overflows, X1 clock oscillation stop is detected by the clock monitor, and each item of hardware is set to the status shown in Table 21-1, “Hardware Status after Reset Acknowledgement,” on page 513. Each pin is high impedance during reset input or during the oscillation stabilization time just after reset release, except for P130, which is low-level output.
When a high level is input to the $\overline{\text{RESET}}$ pin, the reset is released and program execution starts using the internal low-speed oscillator clock after the CPU clock operation has stopped for $17/f_R$ (s). A reset generated by the watchdog timer and clock monitor sources is automatically released after the reset, and program execution starts using the internal low-speed oscillator clock after the CPU clock operation has stopped for $17/f_R$ (s) (please see Figures 21-2 to 21-4).

Cautions: 1. For an external reset, input a low level for 10 µs or more to the $\overline{\text{RESET}}$ pin.

2. During reset input, the X1 input clock and internal low-speed oscillator clock stop oscillating.

3. When the STOP mode is released by a reset, the STOP mode contents are held during reset input. However, the port pins become high-impedance, except for P130, which is set to low-level output.

Figure 21-1: Block Diagram of Reset Function

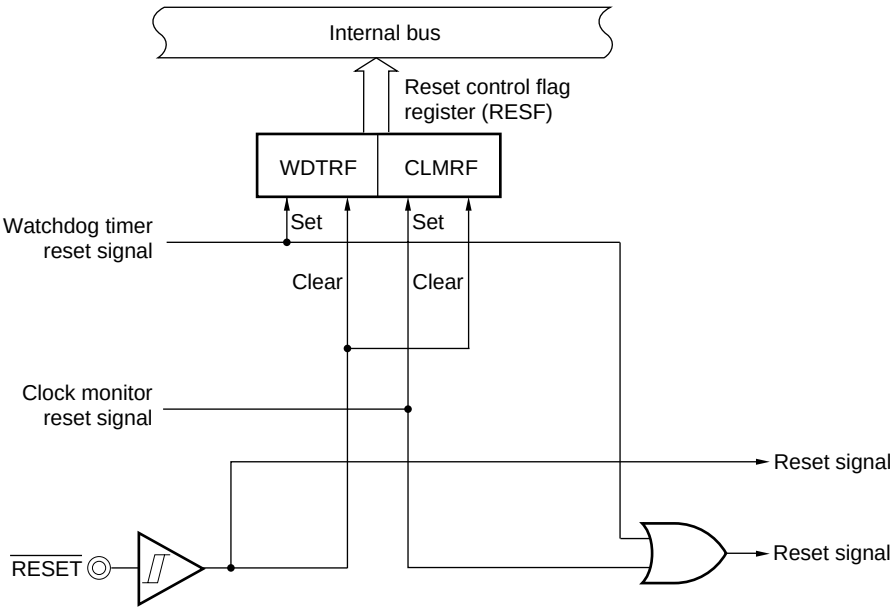
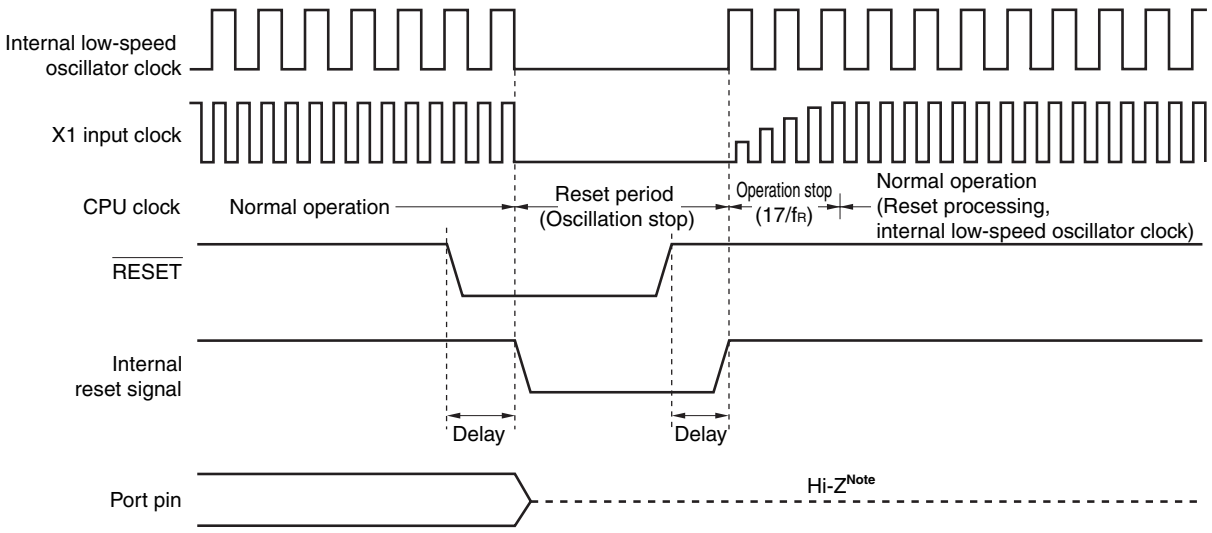
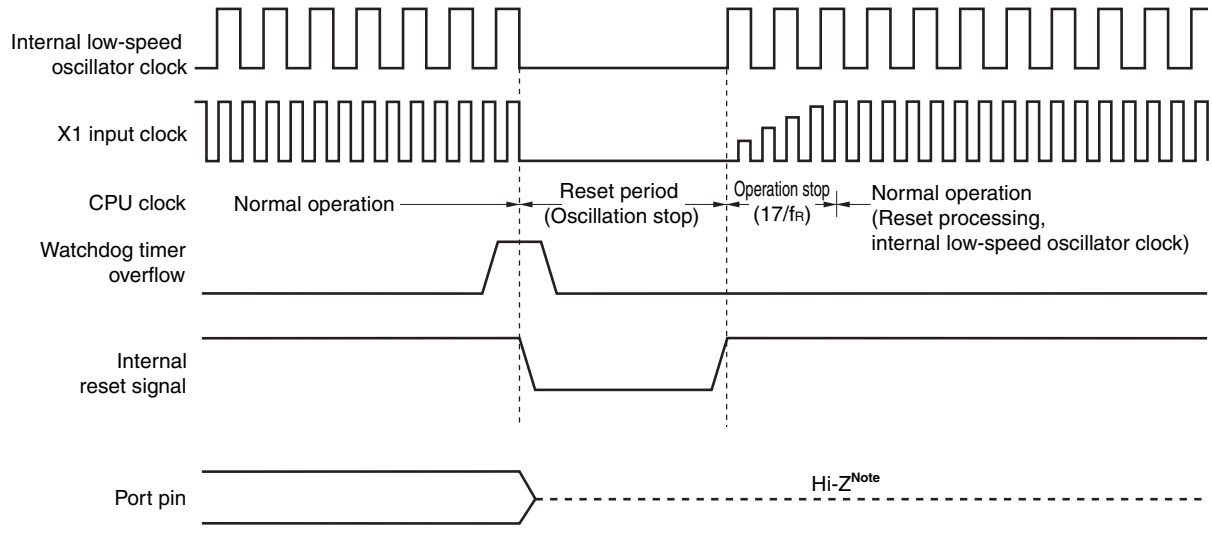


Figure 21-2: Timing of Reset by $\overline{\text{RESET}}$ Input



Note: The port pins become high impedance, except for P130, which is set to low-level output.

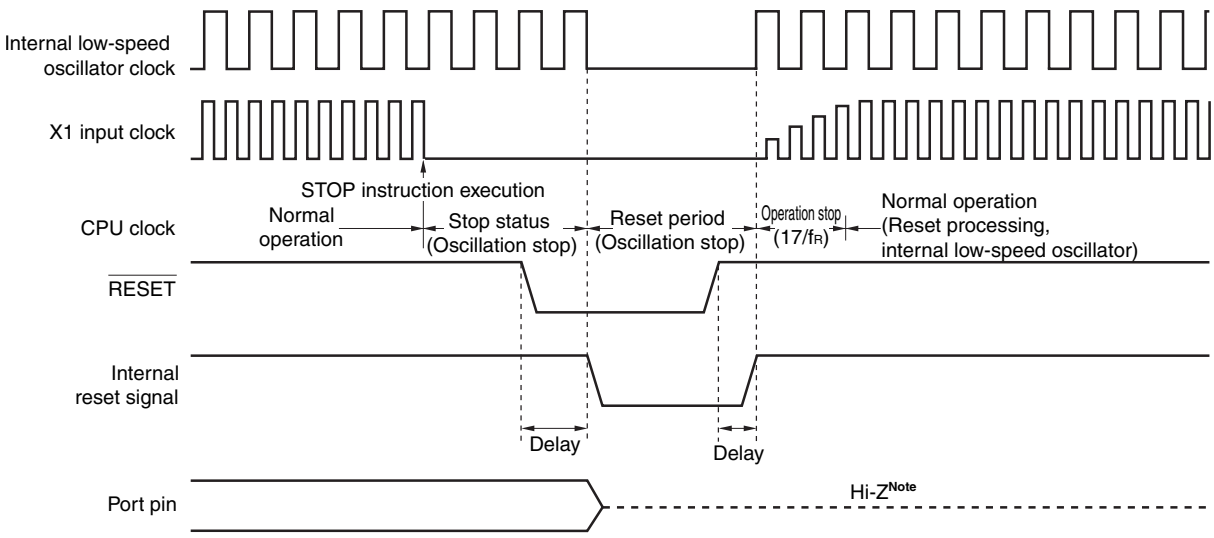
Figure 21-3: Timing of Reset Due to Watchdog Timer Overflow



Note: The port pins become high impedance, except for P130, which is set to low-level output.

Caution: A watchdog timer internal reset resets the watchdog timer.

Figure 21-4: Timing of Reset in STOP Mode by $\overline{RESET}$ Input



Note: The port pins become high impedance, except for P130, which is set to low-level output.

Table 21-1: Hardware Status after Reset Acknowledgement (1/4)

Hardware		Status after Reset Acknowledgement Note 1
Program counter (PC) Note 1		The contents of reset vector tables (0000H and 0001H) are set
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined Note 2
	General purpose registers	Undefined Note 2
	LCD Display Data Memory	Undefined Note 2
Port registers (P0 to P3, P7 to P15) (Output latches)		00H
Port mode register (PM0, PM1, PM3, PM7 to PM12, PM14, PM15)		FFH
Pull-up resistor option register (PU0, PU1, PU4, PU7 to PU12)		00H
Port function selection register (PF7 to PF11)		00H
Input switch control register (ISC)		00H
Internal memory size switching register (IMS)		CFH
Internal expansion RAM size switching register (IXS)		OCH
Bank selection register (B0)		00H
Processor clock control register (PCC)		00H

Notes: 1. During reset input or oscillation stabilization time wait, only the PC contents among the hardware states become undefined. All other hardware states remains unchanged after reset.

2. When a reset is executed in the standby mode, the pre-reset status is held even after reset

3. These values vary depending on the reset source (please see Table 21-2, “RESF Status When Reset Request Is Generated,” on page 517

Table 21-1: Hardware Status after Reset Acknowledgement (2/4)

Hardware		Status after Reset Acknowledgement[Note 1]
Internal low-speed oscillator mode register (RCM)		00H
Main clock mode register (MCM)		00H
Main OSC control register (MOC)		00H
Oscillation stabilization time select register (OSTS)		05H
Oscillation stabilization time counter status register (OSTC)		00H
16-bit timer/event counter 2	Timer registers TM2 (TM2L, TM2H)	00H
	Capture control registers CR20, CR21, CR22 (CR20L, CR20H, CR21L, CR21H, CR22L, CR22H)	00H
	Prescaler mode register (PRM2)	00H
	Mode control register (TMC2)	00H
	Capture/compare control register (CRC2)	00H
16-bit timer/event counter P (TMP)	TMPn control registers (TP0CTL0, TP0CTL1, TP1CTL0, TP1CTL1, TP2CTL0, TP2CTL1)	00H
	TMpn input/output control registers (TP0IOC0, TP0IOC1, TP1IOC0, TP1IOC1, TP2IOC0, TP2IOC1)	00H
	TMPn option registers (TP0OPT0, TP1OPT0, TP2OPT0)	00H
	TMPn capture/compare registers (TP0CCR0, TP0CCR1, TP1CCR0, TP1CCR1, TP2CCR0, TP2CCR1)	0000H
	TMPn count registers (TP0CNT, TP1CNT, TP2CNT)	0000H
	TMPn capture/compare buffer registers (TP0CCB0, TP0CCB1, TP1CCB0, TP1CCB1, TP2CCB0, TP2CCB1)	0000H
8-bit timer/event counters 50, 51	Timer counters 50, 51 (TM50, TM51)	00H
	Compare registers 50, 51 (CR50, CR51)	00H
	Timer clock select registers (TCL50, TCL51)	00H
	Mode control registers (TMC50, TMC51)	00H
Watch timer	Operation mode register (WTM)	00H
Clock output/buzzer output controller	Clock output selection register (CKS)	00H
Watchdog timer	Mode register (WDTM)	67H
	Enable register (WDTE)	9AH

Notes: 1. During reset input or oscillation stabilization time wait, only the PC contents among the hardware states become undefined. All other hardware states remains unchanged after reset.

2. When a reset is executed in the standby mode, the pre-reset status is held even after reset

3. These values vary depending on the reset source (please see Table 21-2, “RESF Status When Reset Request Is Generated,” on page 517

Table 21-1: Hardware Status after Reset Acknowledgement (3/4)

Hardware		Status after Reset Acknowledgement[Note 1]
A/D converter	Mode register (ADM)	00H
	Conversion result register (ADCR)	undefined
	Analog input channel specification register (ADS)	00H
	Power-fail comparison mode register (PFM)	00H
	Power-fail threshold register (PFT)	00H
Serial Interface UART6	Receive buffer register 6 (RXB6)	FFH
	Transmit buffer register 6 (TXB6)	FFH
	Asynchronous serial interface operation mode register 6 (ASIM6)	01H
	Asynchronous serial interface reception error status register 6 (ASIS6)	00H
	Asynchronous serial interface transmission status register 6 (ASIF6)	00H
	Clock selection register 6 (CKSR6)	00H
	Baud rate generator control register (BRGC6)	FFH
	Asynchronous serial interface control register 6 (ASICL6)	16H
Serial interfaces CSI10, CSI11	Transmit buffer registers 10, 11 (SOTB10, SOTB11)	undefined
	Serial I/O shift registers 10, 11 (SIO10, SIO11)	undefined
	Serial operation mode registers 10, 11 (CSIM10, CSIM11)	00H
	Serial clock selection registers 10, 11 (CSIC10, CSIC11))	00H
Clock monitor	Mode register (CLM)	00H
Sound generator	Control register (SGCR)	00H
	Amplitude control (SGAM)	00H
	Buzzer control (SGBR)	00H
LCD-controller/driver	Mode register (LCDM)	00H
	Control register (LCDC)	80H

Notes: 1. During reset input or oscillation stabilization time wait, only the PC contents among the hardware states become undefined. All other hardware states remains unchanged after reset.

2. When a reset is executed in the standby mode, the pre-reset status is held even after reset

3. These values vary depending on the reset source (please see Table 21-2, “RESF Status When Reset Request Is Generated,” on page 517

Table 21-1: Hardware Status after Reset Acknowledgement (4/4)

Hardware		Status after Reset Acknowledgement[Note 1]
CAN	DCAN control register (DCANC)	00H
	CAN Control register (CANC)	01H
	Transmit control register (TCR)	00H
	Receive message register (RMES)	00H
	Redefinition register (REDEF)	00H
	Error status register (CANES)	00H
	Transmit error counter register (TEC)	00H
	Receive error counter register (REC)	00H
	Message count register (MCNT)	C0H
	Bit rate prescaler register (BRPRS)	00H
	Synchronous control register (SYNC0)	18H
	Synchronous control register (SYNC1)	0EH
	Mask control register (MASKC)	00H
Stepper Motor controller/driver	PWM timer control register (MCNTC)	00H
	Port mode control register (PMC)	00H
	8-bit compare register (MCMP10, MCMP11, MCMP20, MCMP21, MCMP30, MCMP31, MCMP40, MCMP41)	00H
	Compare control register (MCMPC1, MCMPC2, MCMPC3, MCMPC4)	00H
	Meter controller/driver clock switch register (SMSWI)	00H
Reset function	Reset Source flag register (RESF)	**Note 3**
Interrupt	Request flag registers IF0L, IF0H, IF1L, IF1H	00H
	Mask flag registers MK0L, MK0H, MK1L	FFH
	Mask flag register MK1H	DFH
	Priority specify flag registers PR0L, PR0H, PR1L, PR1H	FFH
	External interrupt rising edge register (EGP)	00H
	External interrupt falling edge register (EGN)	00H

Notes: 1. During reset input or oscillation stabilization time wait, only the PC contents among the hardware states become undefined. All other hardware states remains unchanged after reset.

2. When a reset is executed in the standby mode, the pre-reset status is held even after reset

3. These values vary depending on the reset source (please see Table 21-2, “RESF Status When Reset Request Is Generated,” on page 517

21.1 Register for Confirming Reset Source

The reset control flag register (RESF) is used to store which source has generated the reset request.
RESF can be read by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input and reading RESF clear RESF to 00H.

Figure 21-5: Reset Control Flag Register (RESF) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
RESF	0	0	0	WDTRF	0	0	CLMRF	0	FFACH	00H**Note**	R

WDTRF	Internal reset request by watchdog timer (WDT)
0	Internal reset request is not generated, or RESF is cleared.
1	Internal reset request is generated.

CLMRF	Internal reset request by clock monitor (CLM)
0	Internal reset request is not generated, or RESF is cleared.
1	Internal reset request is generated.

Note: The value after reset varies depending on the reset source.

Caution: Do not read data by an 1-bit memory manipulation instruction.

The status of RESF when a reset request is generated is shown in Table 21-2.

Table 21-2: RESF Status When Reset Request Is Generated

Flag \ Reset Source	RESET Input	Reset by WDT	Reset by CLM
WDTRF	Cleared (0)	Set (1)	Held
CLMRF		Held	Set (1)

[MEMO]

Chapter 22 Clock Monitor (µPD780822B Subseries)

22.1 Functions of Clock Monitor

The clock monitor samples the X1 input clock using the on-chip internal low-speed oscillator, and generates an internal reset signal when the X1 input clock is stopped.
When a reset signal is generated by the clock monitor, bit 1 (CLMRF) of the reset control flag register (RESF) is set to 1. For details of RESF, see **Chapter 21 "Reset Function (µPD780822B Subseries)" on page 511**.
The clock monitor automatically stops under the following conditions.

- Reset is released and during the oscillation stabilization time
- In STOP mode and during the oscillation stabilization time
- When the X1 input clock is stopped by software (MSTOP = 1 or MCC = 1) and during the oscillation stabilization time
- When the internal low-speed oscillator clock is stopped

Remark: MSTOP: Bit 7 of main OSC control register (MOC)
MCC: Bit 7 of processor clock control register (PCC)

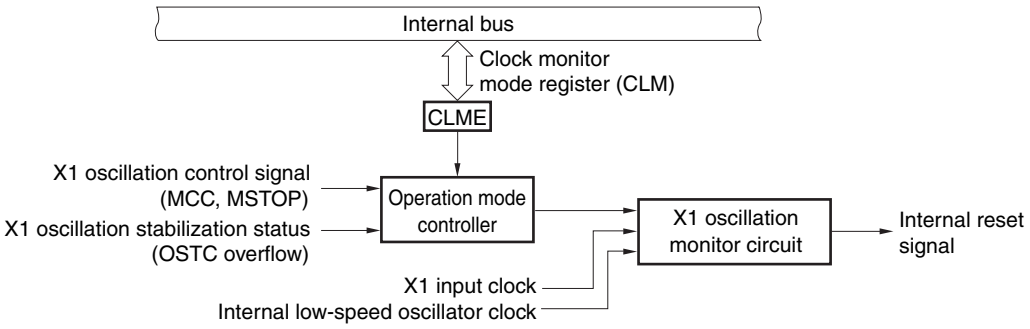
22.2 Configuration of Clock Monitor

Clock monitor consists of the following hardware.

Table 22-1: Configuration of Clock Monitor

Item	Configuration
Control register	Clock monitor mode register (CLM)

Figure 22-1: Block Diagram of Clock Monitor



Remark: MCC: Bit 7 of processor clock control register (PCC)
MSTOP: Bit 7 of main OSC control register (MOC)
OSTC: Oscillation stabilization time counter status register (OSTC)

22.3 Registers Controlling Clock Monitor

Clock monitor is controlled by the clock monitor mode register (CLM).

(1) Clock monitor mode register (CLM)

This register sets the operation mode of the clock monitor.
This register can be set by an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 22-2: Format of Clock Monitor Mode Register (CLM)

Symbol	7	6	5	4	3	2	1	<0>	Address	After Reset	R/W
CLM	0	0	0	0	0	0	0	CLME	FFA9H	00H	R/W

CLME	Enables/disables clock monitor operation
0	Disables clock monitor operation
1	Enables clock monitor operation

Cautions: 1. Once bit 0 (CLME) is set to 1, it cannot be cleared to 0 except by RESET input or the internal reset signal.

2. If the reset signal is generated by the clock monitor, CLME is cleared to 0 and bit 1 (CLMRF) of the reset control flag register (RESF) is set to 1.

22.4 Operation of Clock Monitor

This section explains the functions of the clock monitor. The monitor start and stop conditions are as follows.

(1) Monitor start condition

When bit 0 (CLME) of the clock monitor mode register (CLM) is set to operation enabled (1).

(2) Monitor stop condition

- Reset is released and during the oscillation stabilization time
- In STOP mode and during the oscillation stabilization time
- When the X1 input clock is stopped by software (MSTOP = 1 or MCC = 1) and during the oscillation stabilization time
- When the internal low-speed oscillator clock is stopped

Remark: MSTOP: Bit 7 of main OSC control register (MOC)
MCC: Bit 7 of processor clock control register (PCC)

Table 22-2: Operation Status of Clock Monitor (When CLME = 1)

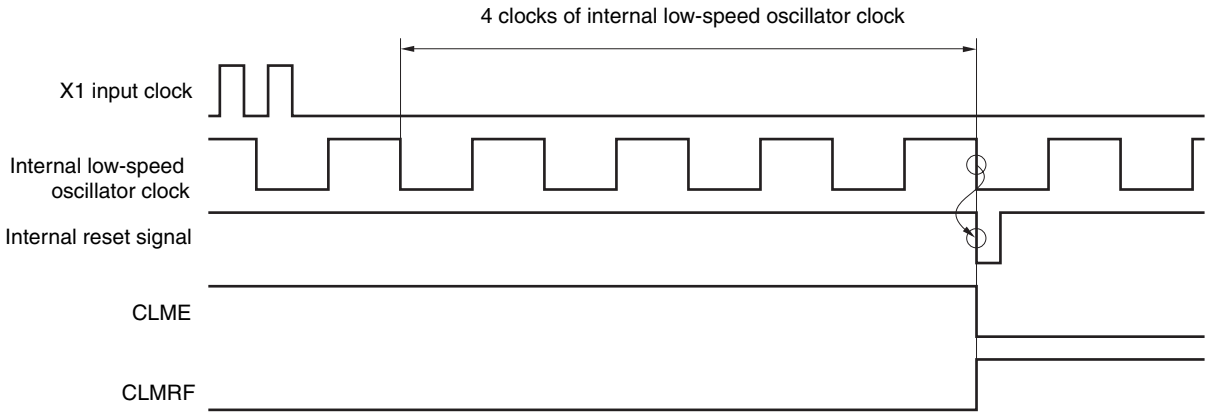
CPU Operation Clock	Operation Mode	X1 Input Clock Status	Internal Low-speed Oscillator Clock Status	Clock Monitor Status
X1 input clock	STOP mode	Stopped	Oscillating	Stopped
			Stopped**Note**	
	$\overline{\text{RESET}}$ input		Oscillating	
			Stopped**Note**	
	Normal operation mode HALT mode	Oscillating	Oscillating	Operating
			Stopped**Note**	Stopped
Internal low-speed oscillator clock	STOP mode	Stopped	Oscillating	Stopped
	$\overline{\text{RESET}}$ input			
	Normal operation mode HALT mode	Oscillating		Operating
		Stopped		Stopped

Note: The internal low-speed oscillator clock is stopped only when the “Internal low-speed oscillator can be stopped by software” is selected by an option byte. If “Internal low-speed oscillator cannot be stopped” is selected, the internal low-speed oscillator clock cannot be stopped.

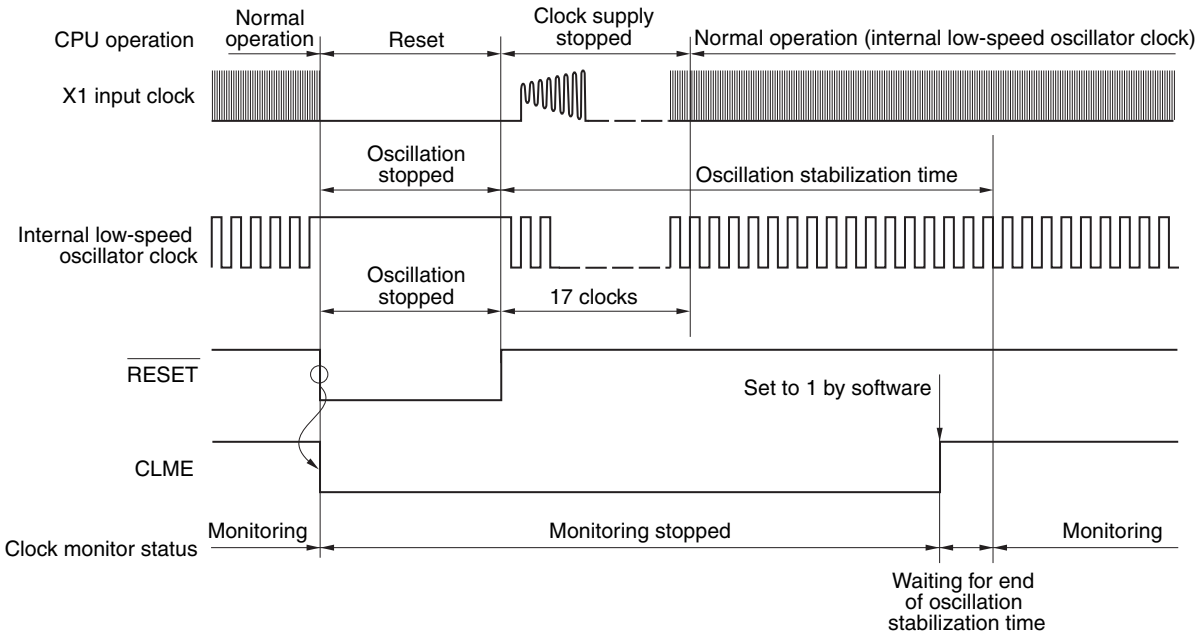
The clock monitor timing is as shown in Figure 22-3.

Figure 22-3: Timing of Clock Monitor (1/4)

(a) When internal reset is executed by oscillation stop of X1 input clock



(b) Clock monitor status after $\overline{\text{RESET}}$ input
(CLME = 1 is set after $\overline{\text{RESET}}$ input and during X1 input clock oscillation stabilization time)

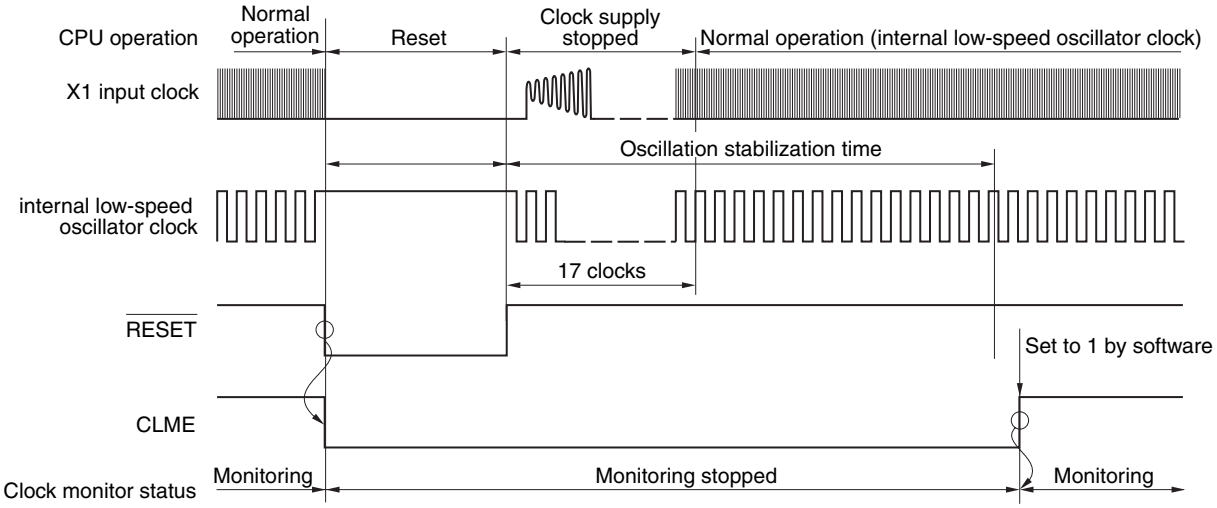


$\overline{\text{RESET}}$ input clears bit 0 (CLME) of the clock monitor mode register (CLM) to 0 and stops the clock monitor operation. Even if CLME is set to 1 by software during the oscillation stabilization time (reset value of OSTS register is 05H ($2^{16}/f_X$)) of the X1 input clock, monitoring is not performed until the oscillation stabilization time of the X1 input clock ends. Monitoring is automatically started at the end of the oscillation stabilization time.

Figure 22-3: Timing of Clock Monitor (2/4)

(c) Clock monitor status after $\overline{\text{RESET}}$ input

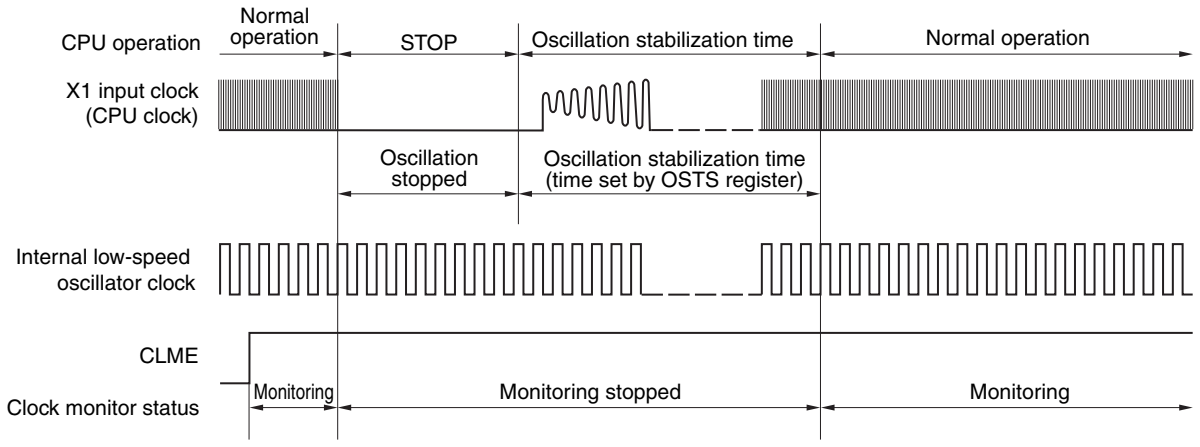
(CLME = 1 is set after $\overline{\text{RESET}}$ input and at the end of X1 input clock oscillation stabilization time)



$\overline{\text{RESET}}$ input clears bit 0 (CLME) of the clock monitor mode register (CLM) to 0 and stops the clock monitor operation. When CLME is set to 1 by software at the end of the oscillation stabilization time (reset value of OSTS register is 05H ($2^{16}/f_X$)) of the X1 input clock, monitoring is started.

(d) Clock monitor status after STOP mode is released

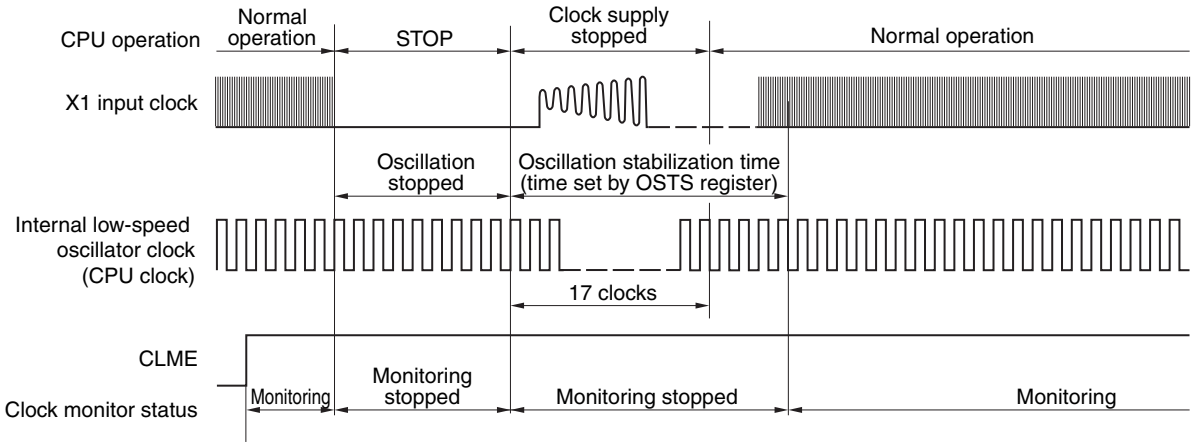
(CLME = 1 is set when CPU clock operates on X1 input clock and before entering STOP mode)



When bit 0 (CLME) of the clock monitor mode register (CLM) is set to 1 before entering STOP mode, monitoring automatically starts at the end of the X1 input clock oscillation stabilization time. Monitoring is stopped in STOP mode and during the oscillation stabilization time.

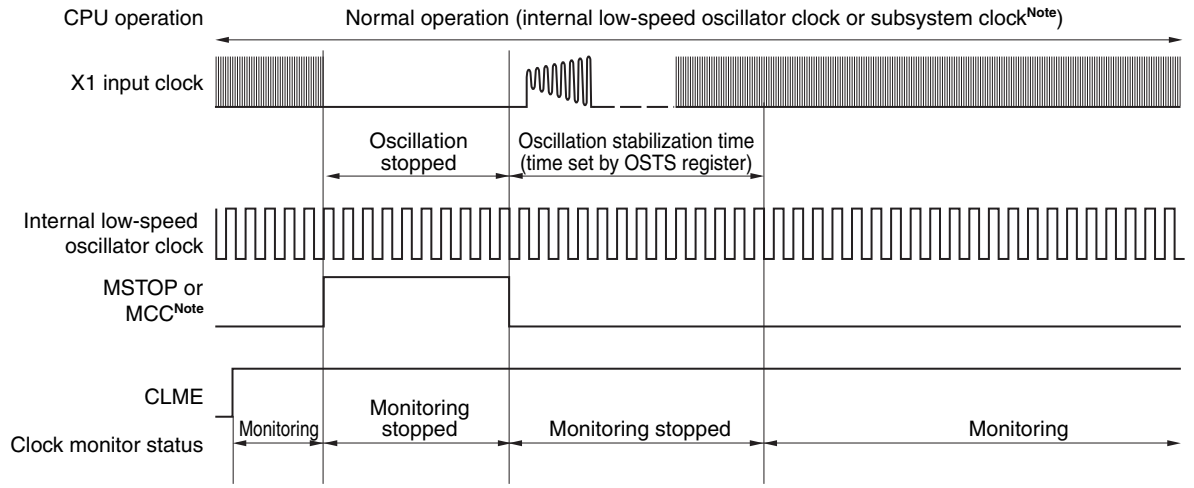
Figure 22-3: Timing of Clock Monitor (3/4)

(e) Clock monitor status after STOP mode is released
(CLME = 1 is set when CPU clock operates on internal low-speed oscillator clock and before entering STOP mode)



When bit 0 (CLME) of the clock monitor mode register (CLM) is set to 1 before entering STOP mode, monitoring automatically starts at the end of the X1 input clock oscillation stabilization time. Monitoring is stopped in STOP mode and during the oscillation stabilization time.

(f) Clock monitor status after X1 input clock oscillation is stopped by software



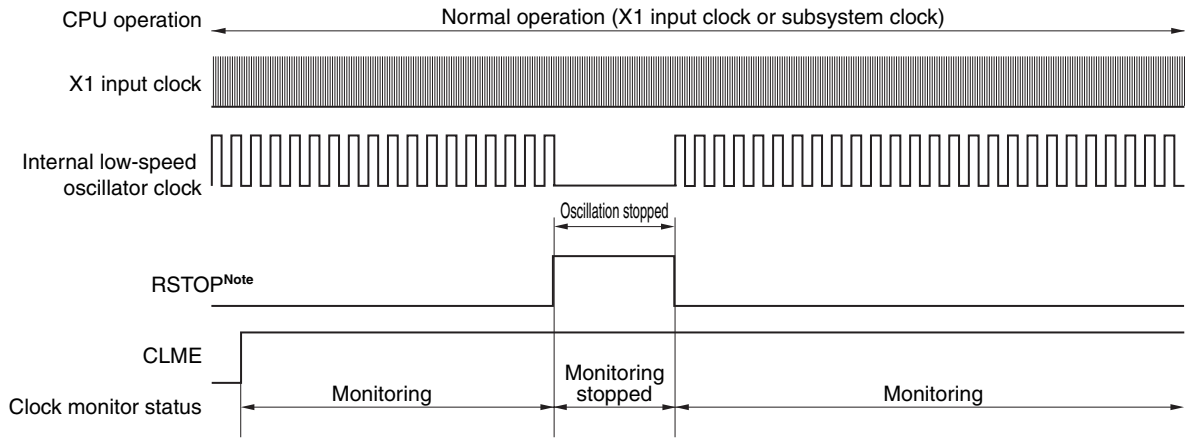
When bit 0 (CLME) of the clock monitor mode register (CLM) is set to 1 before or while oscillation of the X1 input clock is stopped, monitoring automatically starts at the end of the X1 input clock oscillation stabilization time. Monitoring is stopped when oscillation of the X1 input clock is stopped and during the oscillation stabilization time.

Note: The register that controls oscillation of the X1 input clock differs depending on the type of the clock supplied to the CPU.

- When CPU operates on internal low-speed oscillator clock:
  Controlled by bit 7 (MSTOP) of the main OSC control register (MOC)
- When CPU operates on subsystem clock:
  Controlled by bit 7 (MCC) of the processor clock control register (PCC)

Figure 22-3: Timing of Clock Monitor (4/4)

(g) Clock monitor status after internal low-speed oscillator clock oscillation is stopped by software



When bit 0 (CLME) of the clock monitor mode register (CLM) is set to 1 before or while oscillation of the internal low-speed oscillator clock is stopped, monitoring automatically starts after the internal low-speed oscillator clock is stopped. Monitoring is stopped when oscillation of the internal low-speed oscillator clock is stopped.

Note: If it is specified by an option byte that internal low-speed oscillator cannot be stopped, the setting of bit 0 (RSTOP) of the internal low-speed oscillator mode register (RCM) is invalid. To set RSTOP, be sure to confirm that bit 1 (MCS) of the main clock mode register (MCM) is 1.

[MEMO]

Chapter 23 Flash Memory (µPD780822B Subseries)

The flash memory version of the µPD780822B Subseries includes the µPD78F0822.
The µPD78F0822 replaces the internal mask ROM of the µPD780822B with flash memory to which a program can be written, erased, and overwritten while mounted on the PCB.
Table 23-1 lists the differences among the µPD78F0822 and Mask ROM Version.

Table 23-1: Differences among µPD78F0822 and Mask ROM Version

Item	µPD78F0822	Mask ROM Versions
Internal ROM configuration	Flash memory (single power supply)	Mask ROM
Internal ROM capacity	128 KB**Note** (8 KB for firmware)	µPD780821: 90 KB µPD780822B: 120 KB
Internal high-speed ROM capacity	1 KB**Note**	µPD780821: 1 KB µPD780822B: 1 KB
Internal expansion RAM capacity	4 KB**Note**	µPD780821: 4 KB µPD780822B: 4 KB
Pin 16	FLMD0 pin	IC pin
Pin 40	P17/TIO51/FLMD1 pin	P17/TIO51
Self-programming function	Available	None
Electrical specifications	Refer to **Chapter 26 ”Electrical Specifications (µPD780822B Subseries)” on page 565**	

Note: The same capacity as the mask ROM versions can be specified by means of the internal memory size switching register (IMS) and the internal expansion RAM size switching register (IXS).

Caution: There are differences in noise immunity and noise radiation between the flash memory and mask ROM versions. When pre-producing an application set with the flash memory version and then mass-producing it with the mask ROM version, be sure to conduct sufficient evaluations for the commercial samples (not engineering samples) of the mask ROM versions.

23.1 Internal Memory Size Switching Register

The internal memory capacity is selected by using the Internal memory size switching register (IMS) so that the same memory map as the mask ROM versions can be achieved by using the flash device.
IMS sets the size of the capacity of internal ROM and the capacity of internal high-speed RAM.
IMS is set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IMS to CFH.

Caution: When later on a mask version of the µPD780822B Subseries is selected, be sure to set the values of this mask device as specified in the Table 23-1 to IMS. Other settings are prohibited!

Figure 23-1: Internal Memory Size Switching Register (IMS) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IMS	RAM2	RAM1	RAM0	0	ROM3	ROM2	ROM1	ROM0	FFF0H	CFH	R/W

RAM2	RAM1	RAM0	Internal high-speed RAM capacity selection	
			Capacity	Address Area
1	1	0	1024 bytes	FB00H - FEFFH
Other than above			Setting prohibited	

ROM3	ROM2	ROM1	ROM0	Internal ROM capacity selection	
				Capacity	Address Area
1	1	0	0	48 KB	0000H - BFFFH
Other than above				Setting prohibited	

The value to be set after reset is the same for all µPD780822B Subseries products (please see Table 23-2).

Table 23-2: Internal Memory Size Switching Register Settings

Target Mask ROM Versions	IMS Setting
µPD780821	CCH
µPD780822B	CCH
µPD78F0822	CCH

23.2 Internal Expansion RAM Size Switching Register

This register is used to set the internal expansion RAM capacity via software.
This register is set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IXS to 0CH.

Cautions: 1. Because the initial value of IXS is 0CH, set IXS to 04H at initialization.

2. When later on a mask version of the µPD780822B Subseries is selected, be sure to set the values of this mask device as specified in the Table 23-3 to IMS. Other settings are prohibited!

Figure 23-2: Internal Expansion RAM Size Switching Register (IXS) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IXS	0	0	0	IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	0CH	R/W

IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	Internal expansion RAM capacity selection	
					Capacity	Address Area
0	0	1	0	0	4096 bytes	E800H - F7FFH
Other than above					Setting prohibited	

The value to be set after reset is the same for all µPD780822B Subseries products (please see Table 23-3).

Table 23-3: Internal Expansion RAM Size Switching Register Settings

Target Mask ROM Versions	IXS Setting
µPD780821	04H
µPD780822B	04H
µPD78F0822	04H

23.3 Writing with Flash Programmer

Data can be written to the flash memory on-board or off-board, by using a dedicated flash programmer.

(1) On-board programming

The contents of the flash memory can be rewritten after the µPD78F0822 has been mounted on the target system. The connectors that connect the dedicated flash programmer must be mounted on the target system.

(2) Off-board programming

Data can be written to the flash memory with a dedicated program adapter (FA series) before the µPD78F0822 is mounted on the target system.

Remark: The FA series is a product of Naito Densei Machida Mfg. Co., Ltd.

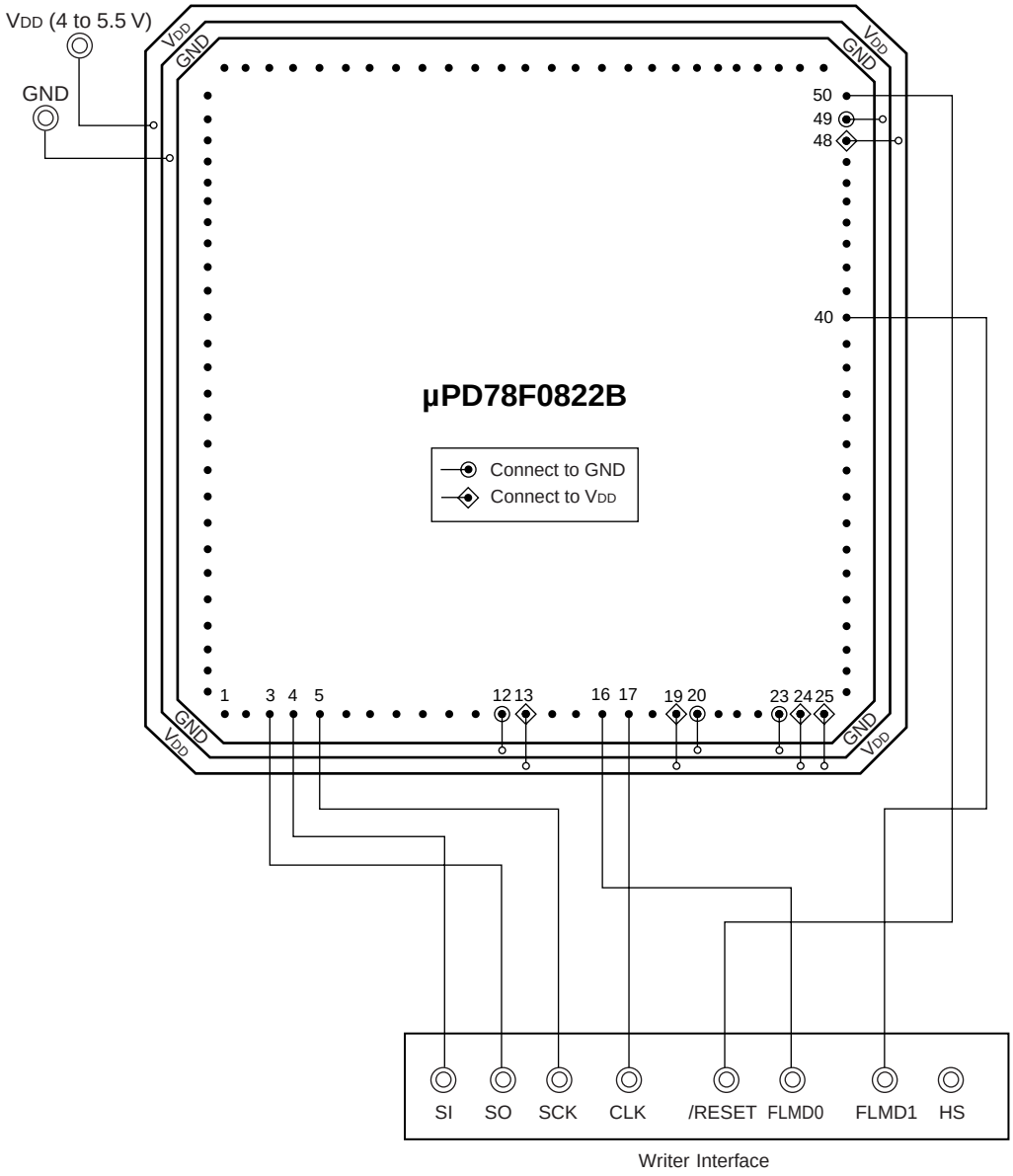
Table 23-4: Wiring Between µPD78F0822 and Dedicated Flash Programmer

Pin Configuration of Dedicated Flash Programmer			With CSI10		With CSI10 + HS		With UART6	
Signal Name	I/O	Pin Function	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.
SI/RXD	Input	Receive signal	SO10/S26/P102	4	SO10/S26/P102	4	TXD6/P13	47
SO/TXD	Output	Transmit signal	SI10/S25/P101	3	SI10/S25/P101	3	RXD6/P12	46
$\overline{SCK}$	Output	Transfer clock	$\overline{SCK10}$/S27/P103	5	$\overline{SCK10}$/S27/P103	5	not needed	not needed
CLK	Output	Clock to µPD780822B	X1	17	X1	17	X1	17
			X2**Note**	18	X2**Note**	18	X2**Note**	18
$\overline{RESET}$	Output	Reset signal	$\overline{RESET}$	50	$\overline{RESET}$	50	$\overline{RESET}$	50
FLMD0	Output	Mode signal	FLMD0	16	FLMD0	16	FLMD0	16
FLMD1	Output	Mode signal	FLMD1/P17/ TIO51	40	FLMD1/P17/ TIO51	40	FLMD1/TI50/ TO50/P17	40
H/S	Input	Handshake signal	not needed	not needed	P100/S24/ TIOP00	2	not needed	not needed
V_{DD}	I/O	V_{DD} voltage generation	V_{DD0}, V_{DD1}, V_{DD2}	19, 13, 48	V_{DD0}, V_{DD1}, V_{DD2}	19, 13, 48	V_{DD0}, V_{DD1}, V_{DD2}	19, 13, 48
			AV_{DD}	24	AV_{DD}	24	AV_{DD}	24
			AV_{REF0}	25	AV_{REF0}	25	AV_{REF0}	25
GND	-	Ground	V_{SS0}, V_{SS1}, V_{SS2}	20, 12, 49	V_{SS0}, V_{SS1}, V_{SS2}	20, 12, 49	V_{SS0}, V_{SS1}, V_{SS2}	20, 12, 49
			AV_{SS}	23	AV_{SS}	23	AV_{SS}	23

Note: When using the clock out of the flash programmer, connect CLK of the programmer to X1, and connect its inverse signal to X2.

Examples of the recommended connection when using the adapter for flash memory writing are shown below.

Figure 23-3: Example of wiring adapter for Flash Memory writing in 3-Wire Serial I/O (CSI10) Mode

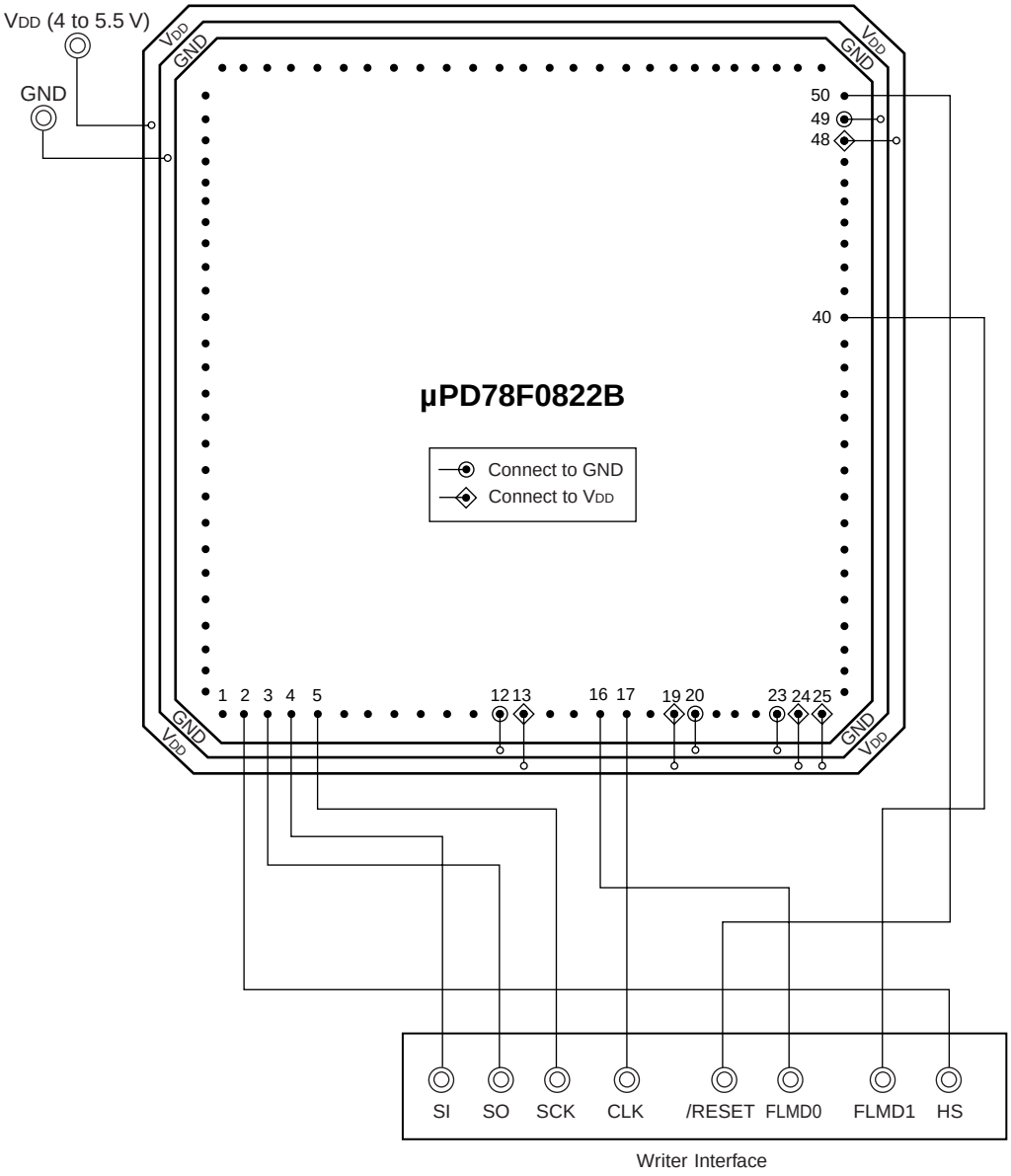


Supply the clock by preparing an oscillator on the board.

Remark: Handle the pins not described above in accordance with the specified handling of unused pins (refer to section **2.4 "Pin I/O Circuits and Recommended Connection of Unused Pins" on page 58**).
When connecting to V_{DD} via a resistor, use of a resistor of 1 KΩ to 10 KΩ is recommended.

Note: When using GF type package, FA-100GF-3BA-A adapter is needed.
When using GC type package, FA-100GC-8EU-A adapter is needed.

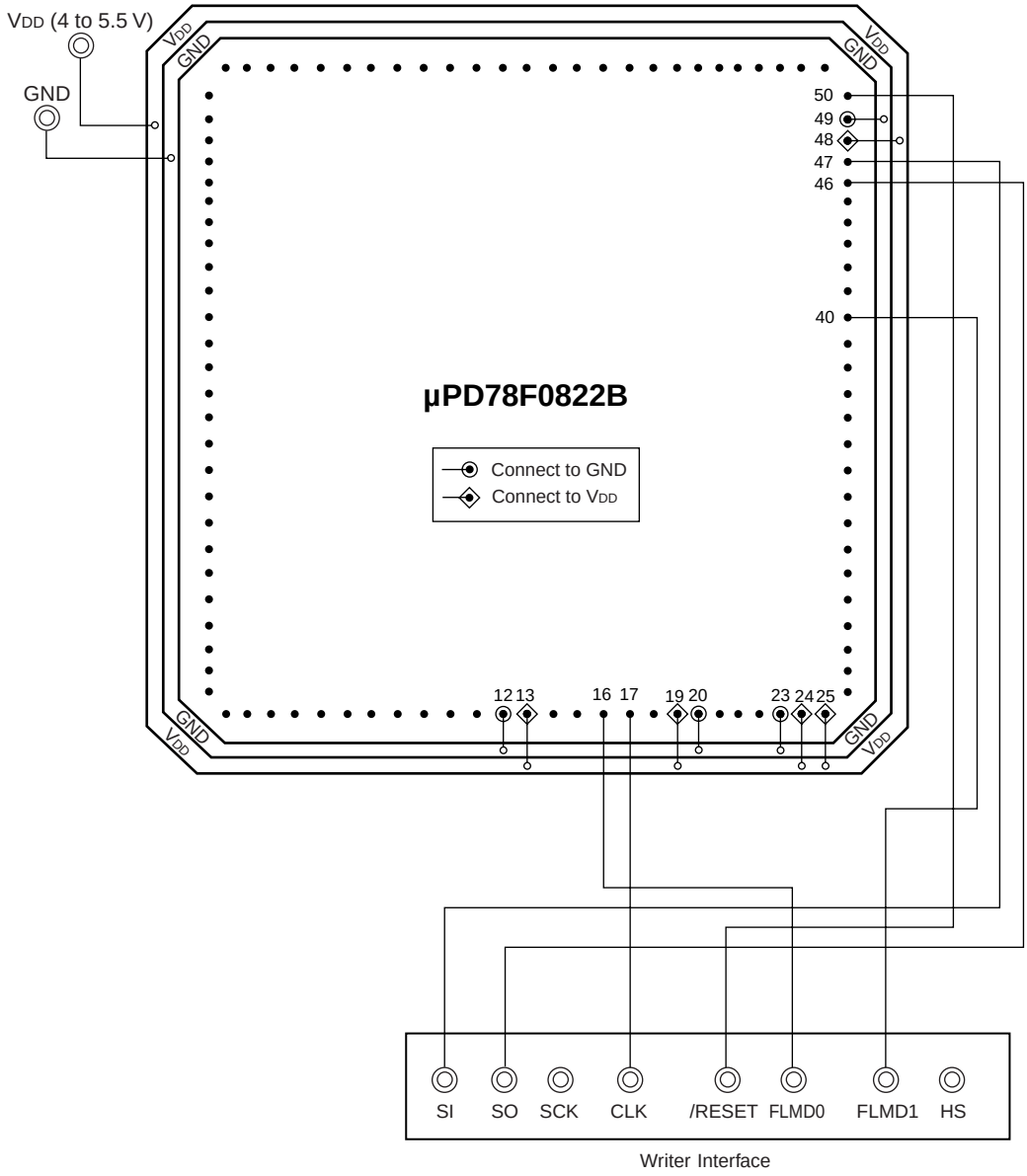
Figure 23-4: Example of wiring adapter for Flash Memory writing in 3-Wire Serial I/O (CSI10 + HS) Mode



Supply the clock by preparing an oscillator on the board.

Remark: Handle the pins not described above in accordance with the specified handling of unused pins (refer to section **2.4 "Pin I/O Circuits and Recommended Connection of Unused Pins" on page 58**).).
When connecting to V_{DD} via a resistor, use of a resistor of 1 KΩ to 10 KΩ is recommended.

Figure 23-5: Example of Wiring Adapter for Flash Memory Writing in UART (UART6) Mode



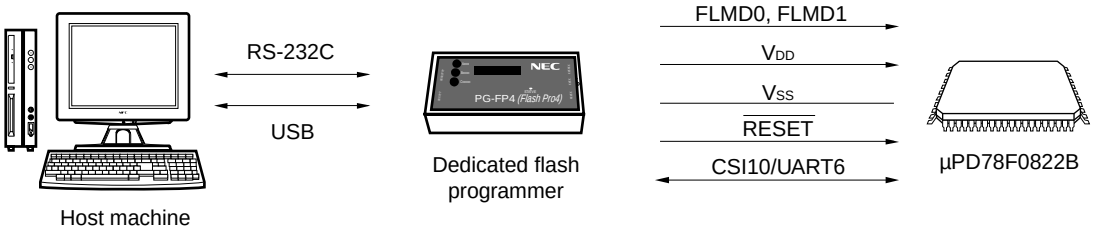
Supply the clock by preparing an oscillator on the board.

Remark: Handle the pins not described above in accordance with the specified handling of unused pins (refer to section **2.4 "Pin I/O Circuits and Recommended Connection of Unused Pins" on page 58**).).
When connecting to V_{DD} via a resistor, use of a resistor of 1 KΩ to 10 KΩ is recommended.

23.4 Programming Environment

The environment required for writing a program to the flash memory of the µPD78F0822 is illustrated below.

Figure 23-6: Environment for Writing Program to Flash Memory



A host machine that controls the dedicated flash programmer is necessary.
CSI10 or UART6 is used to interface between the dedicated flash programmer and the µPD78F0822 for manipulation such as writing and erasing. To write the flash memory off-board, a dedicated program adapter (FA series) is necessary.

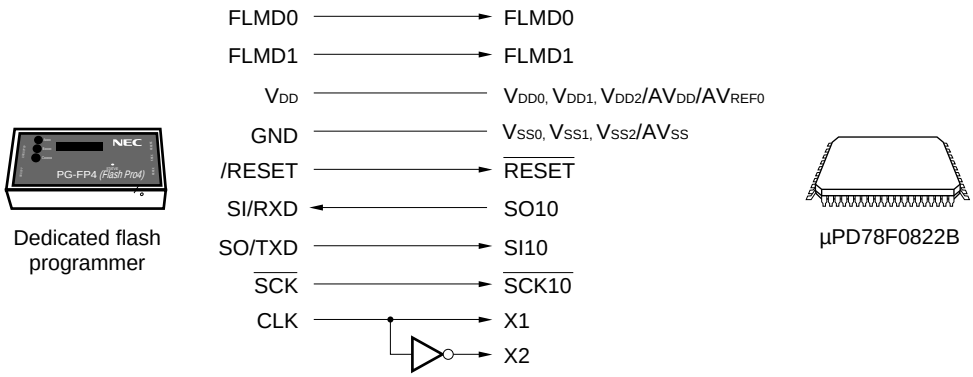
23.5 Communication Mode

Communication between the dedicated flash programmer and the µPD78F0822 is established by serial communication via CSI10 or UART6 of the µPD78F0822.

(1) CSI10

Transfer rate: 2.4 kHz to 2.5 MHz

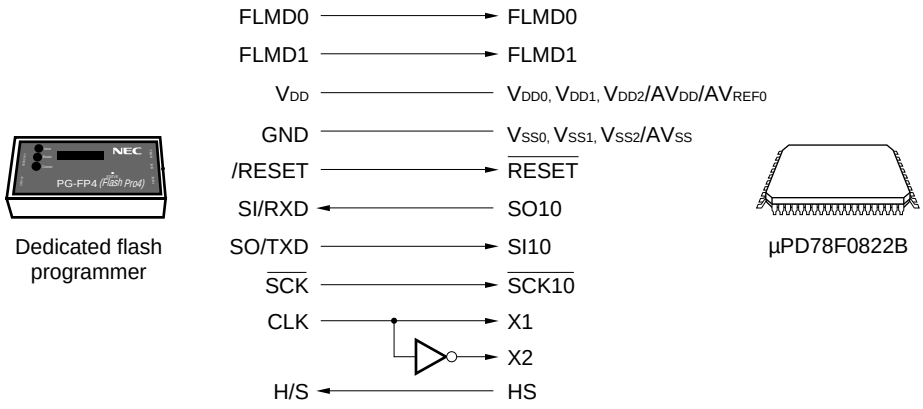
Figure 23-7: Communication with Dedicated Flash Programmer (CSI10)



(2) CSI communication mode supporting handshake

Transfer rate: 2.4 kHz to 2.5 MHz

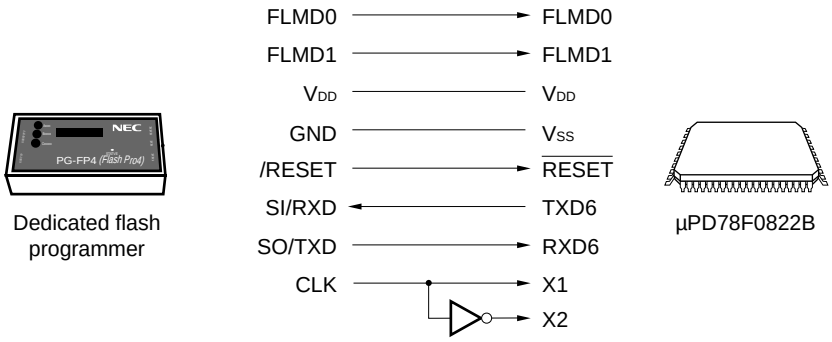
Figure 23-8: Communication with Dedicated Flash Programmer (CSI10 + HS)



(3) UART6

Transfer rate: 9600 to 153600 bps

Figure 23-9: Communication with Dedicated Flash Programmer (UART6)



If Flashpro IV is used as the dedicated flash programmer, Flashpro IV generates the following signal for the µPD78F0822. For details, refer to the Flashpro IV Manual.

Table 23-5: Pin Connection

Flashpro IV			µPD78F0822	Connection	
Signal Name	I/O	Pin Function	Pin Name	CSI10	UART6
FLMD0	Output	Mode signal	FLMD0	◎	◎
FLMD1	Output	Mode signal	FLMD1	◎	◎
V_{DD}	I/O	V_{DD} voltage generation	V_{DD0}, V_{DD1}, V_{DD2}, AV_{DD}, AV_{REF0}	◎	◎
GND	-	Ground	V_{SS0}, V_{SS1}, V_{SS2}, AV_{SS}	◎	◎
CLK	Output	Clock output to µPD78F0822	X1, X2**Note**	O	O
/RESET	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎
SI/RXD	Input	Receive signal	SO10/TXD6	◎	◎
SO/TXD	Output	Transmit signal	SI10/RXD6	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCK10}}$	◎	×
H/S	Input	Handshake signal	HS	Δ	×

Note: For off-board writing only: connect the clock output of the flash programmer to X1 and its inverse signal to X2.

Remark: ◎: Be sure to connect the pin.
O:The pin does not have to be connected if the signal is generated on the target board.
×: The pin does not have to be connected.
Δ: In handshake mode

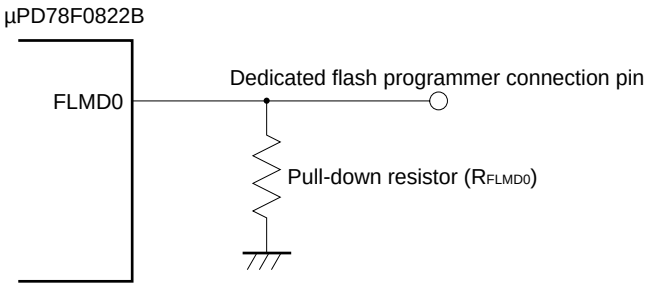
23.6 Processing of Pins on Board

To write the flash memory on-board, connectors that connect the dedicated flash programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.
When the flash memory programming mode is set, all the pins not used for programming the flash memory are in the same status as immediately after reset. Therefore, if the external device does not recognize the state immediately after reset, the pins must be processed as described below.

23.6.1 FLMD0 pin

In the normal operation mode, 0 V is input to the FLMD0 pin. In the flash memory programming mode, the V_{DD} write voltage is supplied to the FLMD0 pin. The following shows an example of the connection of the FLMD0 pin.

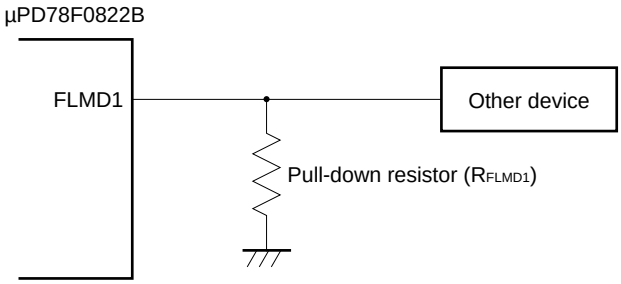
Figure 23-10: FLMD0 Pin Connection Example



23.6.2 FLMD1 pin

When 0 V is input to the FLMD0 pin, the FLMD1 pin does not function. When V_{DD} is supplied to the FLMD0 pin, the flash memory programming mode is entered, so 0 V must be input to the FLMD1 pin. An FLMD1 pin connection example is shown below.

Figure 23-11: FLMD1 Pin Connection Example



Caution: If the V_{DD} signal is input to the FLMD1 pin from another device during on-board writing and immediately after reset, isolate this signal.

23.6.3 Serial interface pins

The pins used by each serial interface are listed below.

Table 23-6: Pins Used by Each Serial Interface

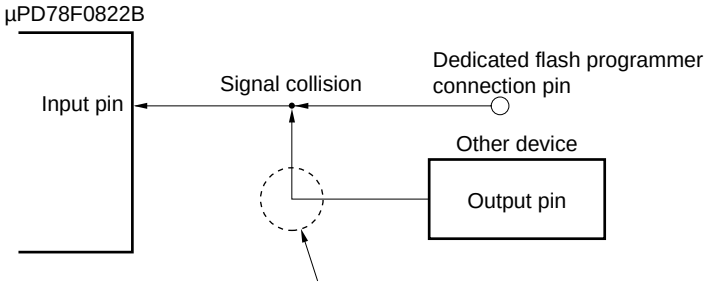
Serial Interface	Pins Used
CSI10	SO10, SI10, $\overline{\text{SCK10}}$
CSI10 + HS	SO10, SI10, $\overline{\text{SCK10}}$, HS/S24/TIOP00
UART6	TXD6, RXD6

To connect the dedicated flash programmer to the pins of a serial interface that is connected to another device on the board, care must be exercised so that signals do not collide or that the other device does not malfunction.

(1) Signal collision

If the dedicated flash programmer (output) is connected to a pin (input) of a serial interface connected to another device (output), signal collision takes place. To avoid this collision, either isolate the connection with the other device, or make the other device go into an output high-impedance state.

Figure 23-12: Signal Collision (Input Pin of Serial Interface)

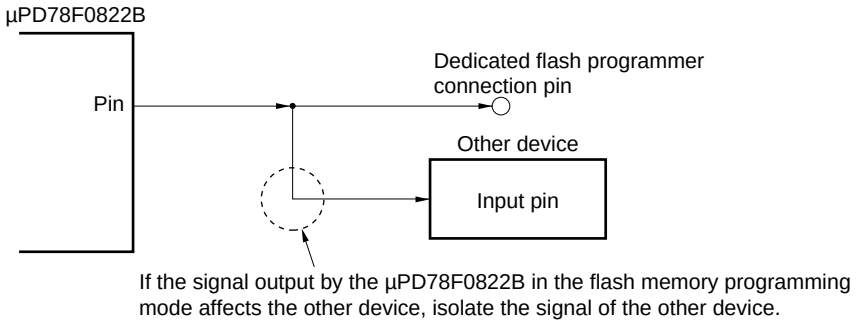


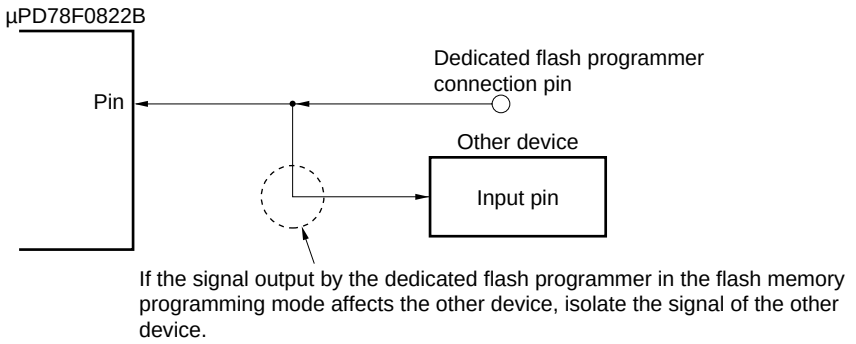
In the flash memory programming mode, the signal output by the device collides with the signal sent from the dedicated flash programmer. Therefore, isolate the signal of the other device.

(2) Malfunction of other device

If the dedicated flash programmer (output or input) is connected to a pin (input or output) of a serial interface connected to another device (input), a signal may be output to the other device, causing the device to malfunction. To avoid this malfunction, isolate the connection with the other device.

Figure 23-13: Malfunction of Other Device

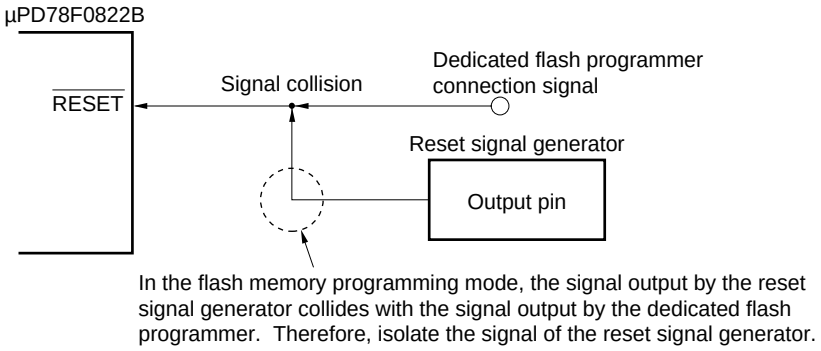




23.6.4 $\overline{\text{RESET}}$ pin

If the reset signal of the dedicated flash programmer is connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on the board, signal collision takes place. To prevent this collision, isolate the connection with the reset signal generator.
If the reset signal is input from the user system while the flash memory programming mode is set, the flash memory will not be correctly programmed. Do not input any signal other than the reset signal of the dedicated flash programmer.

Figure 23-14: Signal Collision ($\overline{\text{RESET}}$ Pin)



23.6.5 Port pins

When the flash memory programming mode is set, all the pins not used for flash memory programming enter the same status as that immediately after reset. If external devices connected to the ports do not recognize the port status immediately after reset, the port pin must be connected to V_{DD} or V_{SS} via a resistor.

23.6.6 Other signal pins

Connect X1 and X2 in the same status as in the normal operation mode when using the on-board clock. To input the operating clock from the programmer, however, connect the clock out of the programmer to X1, and its inverse signal to X2.

23.6.7 Power supply

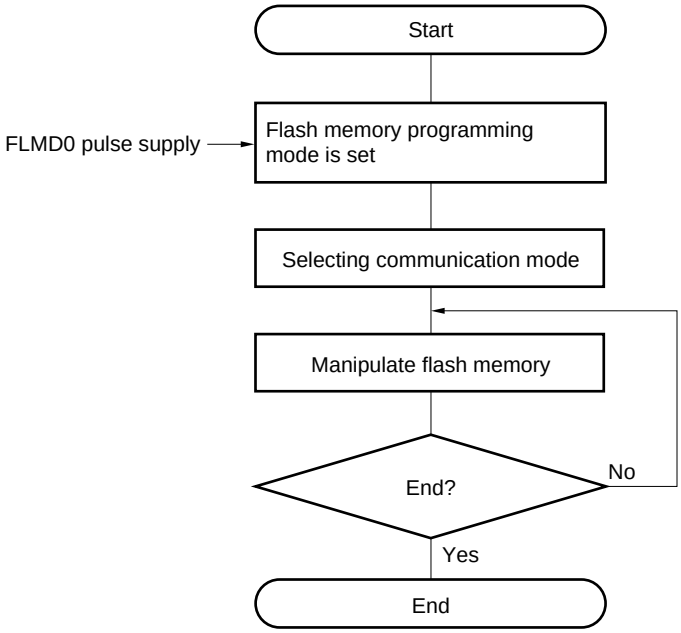
To use the supply voltage output of the flash programmer, connect the V_{DD} pins to V_{DD} of the flash programmer, and the V_{SS} pins to V_{SS} of the flash programmer.
To use the on-board supply voltage, connect in compliance with the normal operation mode.
Supply the same other power supplies (AV_{DD}, AV_{SS} and AV_{REF}) as those in the normal operation mode.

23.7 Programming Method

23.7.1 Controlling flash memory

The following figure illustrates the procedure to manipulate the flash memory.

Figure 23-15: Flash Memory Manipulation Procedure



23.7.2 Flash memory programming mode

To rewrite the contents of the flash memory by using the dedicated flash programmer, set the µPD78F0822 in the flash memory programming mode. To set the mode, set the FLMD0 pin to V_{DD} and clear the reset signal.
Change the mode by using a jumper when writing the flash memory on-board.

Figure 23-16: Flash Memory Programming Mode

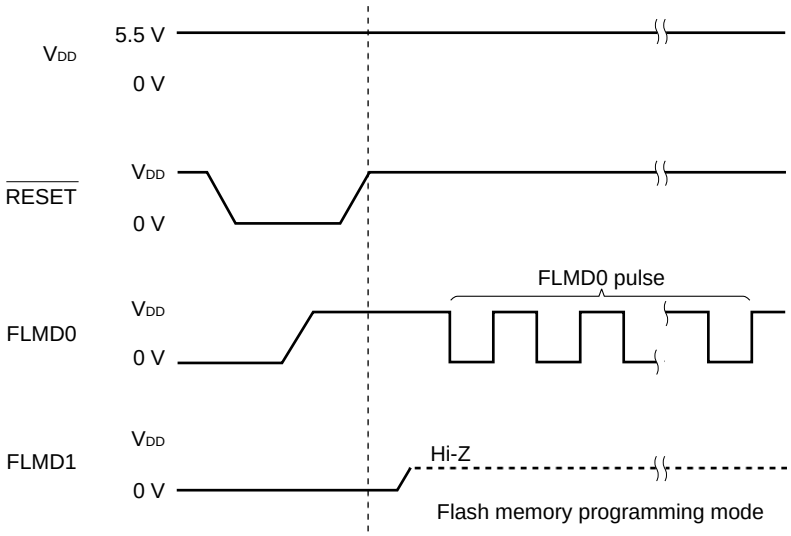


Table 23-7: Relationship of Operation Mode of FLMD0 and FLMD1 Pins

FLMD0	FLMD1	Operation Mode
0	×	Normal operation mode
V_{DD}	0	Flash memory programming mode
V_{DD}	V_{DD}	Setting prohibited

23.7.3 Selecting communication mode

In the µPD78F0822 a communication mode is selected by inputting pulses (up to 11 pulses) to the FLMD0 pin after the dedicated flash memory programming mode is entered. These FLMD0 pulses are generated by the flash programmer.
The following table shows the relationship between the number of pulses and communication modes.

Table 23-8: Communication Modes

Communication Mode	Standard Setting[Note 1]					Pins Used	Number of FLMD0 Pulses
	Port	Speed	On Target	Frequency	Multiply Rate		
UART (UART6)	UART-ch0	9600 bps to 153600 bps[Notes 2, 3]	Optional	2 MHz to 16 MHz	1.0	TXD6, RXD6	0
3-wire serial I/O (CSI10)	SIO-ch0	2.4 kHz to 2.5 MHz[Note 2]				SO10, SI10, $\overline{\text{SCK10}}$	8
3-wire serial I/O with handshake (CSI10 + HS)	SIO-H/S	2.4 kHz to 2.5 MHz[Note 2]				SO10, SI10, $\overline{\text{SCK10}}$ HS/S24/TIOP00	11

Notes: 1. Selection items for Standard settings on Flashpro IV.

2. The possible setting range differs depending on the voltage. For details, refer to the electrical specifications chapter.

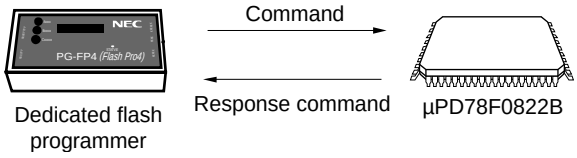
3. Because factors other than the baud rate error, such as the signal waveform slew, also affect UART communication, thoroughly evaluate the slew as well as the baud rate error.

Caution: When UART6 is selected, the receive clock is calculated based on the reset command sent from the dedicated flash programmer after the FLMD0 pulse has been received.

23.7.4 Communication commands

The µPD78F0822 communicates with the dedicated flash programmer by using commands. The signals sent from the flash programmer to the µPD78F0822 are called commands, and the commands sent from the µPD78F0822 to the dedicated flash programmer are called response commands.

Figure 23-17: Communication Commands



The flash memory control commands of the µPD78F0822 are listed in the table below. All these commands are issued from the programmer and the µPD78F0822 perform processing corresponding to the respective commands.

Table 23-9: Flash Memory Control Commands

Classification	Command Name	Function
Verify	Batch verify command	Compares the contents of the entire memory with the input data.
Erase	Batch erase command	Erases the contents of the entire memory.
Blank check	Batch blank check command	Checks the erasure status of the entire memory.
Data write	High-speed write command	Writes data by specifying the write address and number of bytes to be written, and executes a verify check.
	Successive write command	Writes data from the address following that of the high-speed write command executed immediately before, and executes a verify check.
System setting, control	Status read command	Obtains the operation status
	Oscillation frequency setting command	Sets the oscillation frequency
	Erase time setting command	Sets the erase time for batch erase
	Write time setting command	Sets the write time for writing data
	Baud rate setting command	Sets the baud rate when UART is used
	Silicon signature command	Reads the silicon signature information
	Reset command	Escapes from each status

The µPD78F0822 return a response command for the command issued by the dedicated flash programmer. The response commands sent from the µPD78F0822 are listed below.

Table 23-10: Response Commands

Command Name	Function
ACK	Acknowledges command/data.
NAK	Acknowledges illegal command/data.

23.8 Flash Memory Programming by Self-Programming

The µPD780822B supports a self-programming function that can be used to rewrite the flash memory via a user program. Because this function allows a user application to rewrite the flash memory by using the 78K0/Kx1+ selfprogramming library, it can be used to upgrade the program in the field.
If an interrupt occurs during self-programming, self-programming can be temporarily stopped and interrupt servicing can be executed. To execute interrupt servicing, restore the normal operation mode after self-programming has been stopped, and execute the EI instruction. After the self-programming mode is later restored, self-programming can be resumed.

Remark: For details of the self-programming function and the 78K0/Kx1+ self-programming library, refer to a separate document to be published (document name: 78K0/Kx1+ Application Note, release schedule: Pending).

Cautions:

1. **The self-programming function cannot be used when the CPU operates with the subsystem clock.**
2. **Input a high level to the FLMD0 pin during self-programming.**
3. **Be sure to execute the DI instruction before starting self-programming.**
4. **Self-programming is executed with the X1 input clock.**
5. **Locate the entry program for self-programming in the common area of 0000H to 7FFFH.**
6. **For the flash memory size of 120 KB, specify a flash real address, instead of a CPU address, as a flash write/erase address.**

Figure 23-18: Operation Mode and Memory Map for Self-Programming (μPD78F0822)

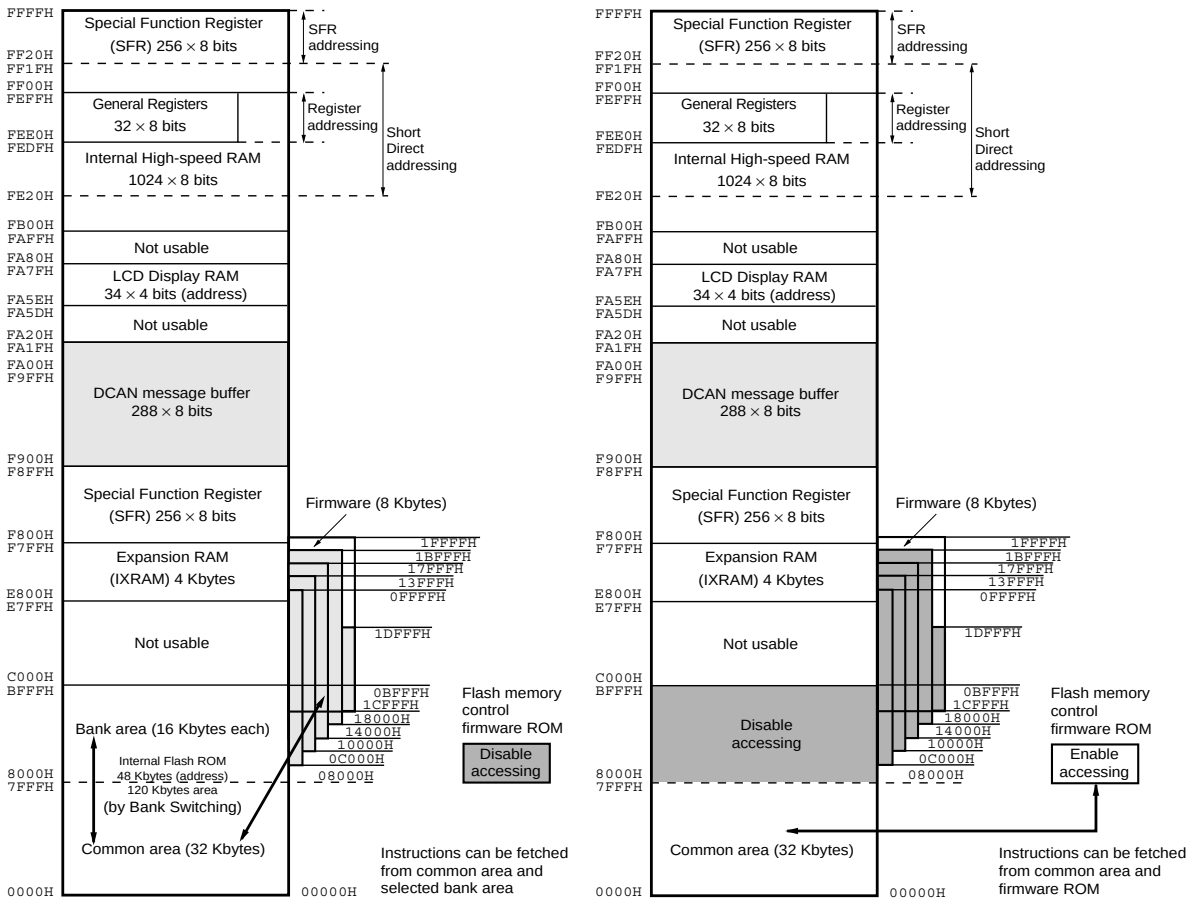
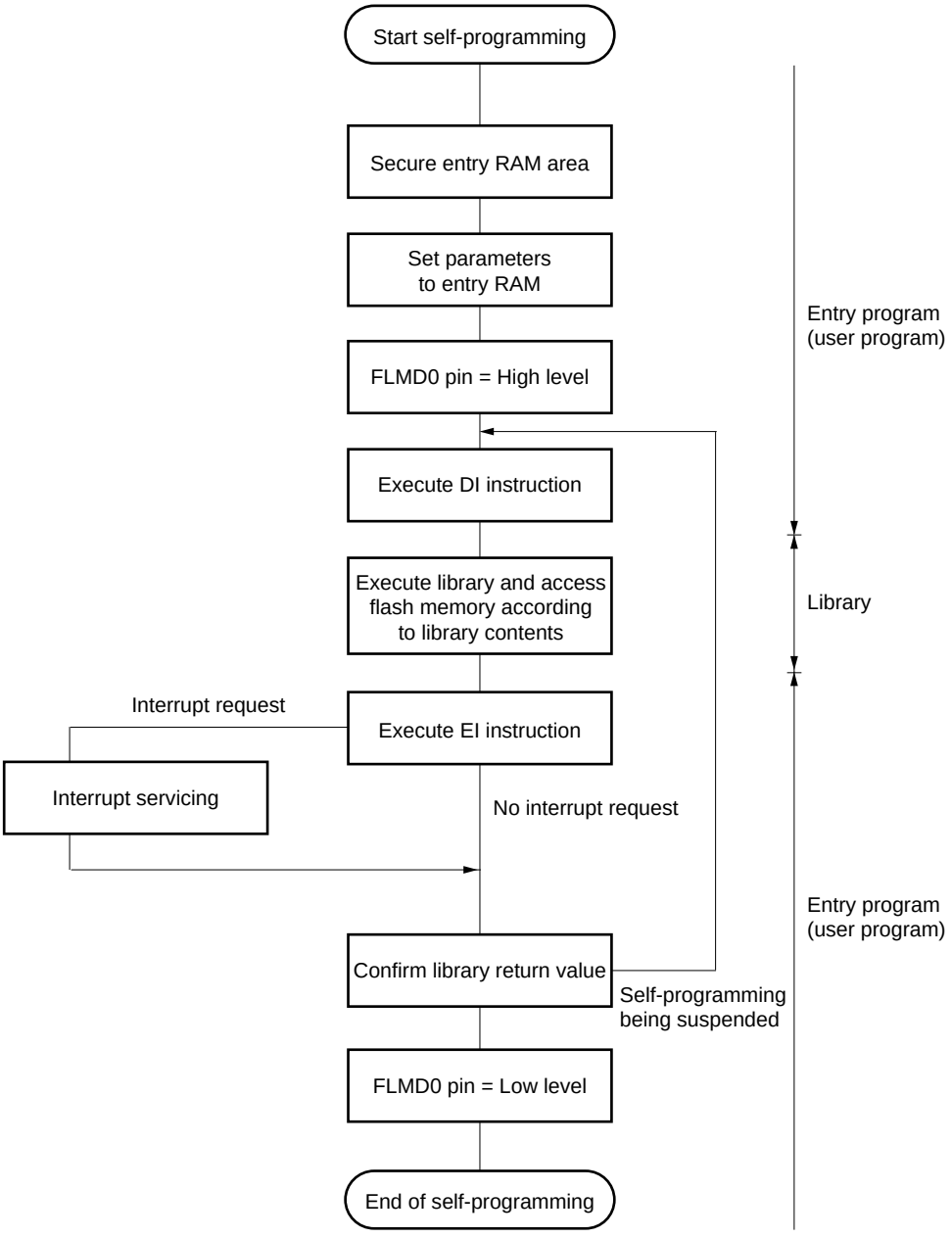


Table 23-11: Correspondence among Bank Numbers, CPU Addresses, and Flash Real Addresses for μPD78F0822

Bank No.	CPU Address	Real Address of Flash Memory
-	0000H to 7FFFH (common area)	00000H to 07FFFH
0	8000H to BFFFH	08000H to 0BFFFH
1		0C000H to 0FFFFH
2		10000H to 13FFFH
3		14000H to 17FFFH
4		18000H to 1BFFFH
5		1C000H to 1DFFFH
6 or more	Setting prohibited	

The procedure of self-programming is illustrated below.

Figure 23-19: Self-Programming Procedure



23.9 Boot Swap Function

The µPD78F0822 has a boot swap function.
Even if a momentary power failure occurs for some reason while the boot area is being rewritten by self-programming and the program in the boot area is lost, the boot swap function can execute the program correctly after re-application of power, reset, and start.
The boot program area of the µPD78F0822 is as follows: 0000H to 0FFFH are boot cluster 0, and 1000H to 1FFFH are boot cluster 1 (fixed in 4 KB units). Boot clusters 0 and 1 are swapped during boot swapping.

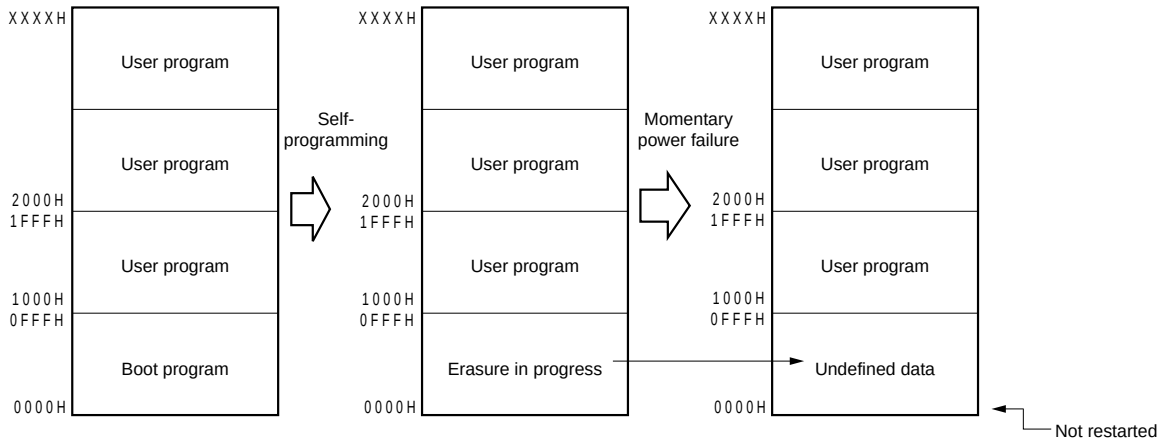
23.9.1 Outline of boot swap function

Before erasing the boot program area by self-programming, write a new boot program to the block to be swapped, and also set the boot flag[Note]. Even if a momentary power failure occurs, the address is swapped when the system is reset and started next time. Consequently, the above area to be swapped is used as a boot area, and the program is executed correctly. Figure 26-18 shows an image of the boot swap function.

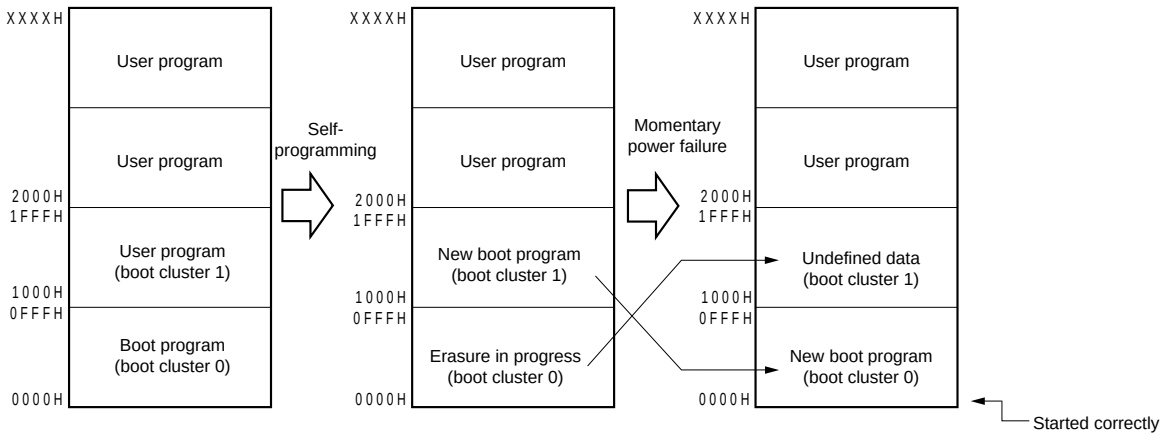
Note: The boot flag is controlled by the flash memory control firmware of the µPD78F0822.

Figure 23-20: Image of Boot Swap Function

(a) If boot swap is not supported



(b) If boot swap is supported



Chapter 24 Mask Options / Option Bytes (µPD780822B Subseries)

24.1 Mask Options (Mask ROM Version)

Mask ROM versions have the following mask option:

- On-chip internal low-speed oscillator clock oscillation
  - Cannot be stopped[Note]
  - Can be stopped by software

Note: If “on-chip internal low-speed oscillator clock oscillation cannot be stopped” is selected, the source clock of the watchdog timer is fixed to the internal low-speed oscillator, and it cannot be changed.

24.2 Option Bytes (Flash Memory Version)

In the flash memory versions, the functions equivalent to the mask options of the mask ROM versions can be realized by setting using option bytes.
Option bytes are prepared at address 0080H in the flash memory.
When using flash memory version products, be sure to set the mask option information to the option bytes.

Figure 24-1: Allocation of Option Bytes (Flash Memory Version)

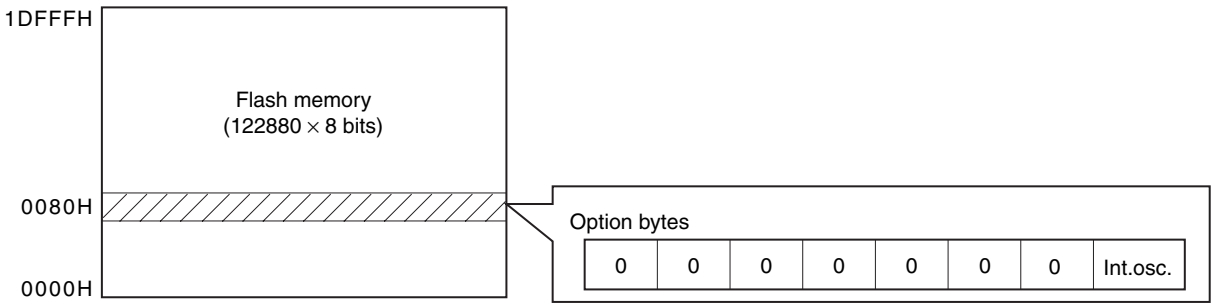


Figure 24-2: Format of Option Bytes (Int. osc.) (Flash Memory Version)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
Int. osc.	0	0	0	0	0	0	0	Int. osc.	0080H	00H	R/W

Int. osc.	Internal low-speed oscillator clock oscillation
0	Can be stopped by software
1	Cannot be stopped

Remark: An example of software coding for setting the option bytes is shown below.

```
OPT       CSEG    AT 0080H
OPTION:   DB      01H          ; Set to option byte
                                 internal low-speed oscillator
                                 cannot be stopped)
```

Chapter 25 Instruction Set (µPD780822B Subseries)

This chapter describes each instruction set of the µPD780822B Subseries as list table.
For details of its operation and operation code, refer to the separate document **“78K/0 series USER’S MANUAL - Instruction (U12326E)**.”

25.1 Legends Used in Operation List

25.1.1 Operand identifiers and description methods

Operands are described in “Operand” column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more description methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $ and [ ] are key words and must be described as they are. Each symbol has the following meaning.

- # : Immediate data specification
- ! : Absolute address specification
- $: Relative address specification
- [ ] : Indirect address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, $, and [ ] symbols.
For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used for description.

Table 25-1: Operand Identifiers and Description Methods

Identifier	Description Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special-function register symbol**Note**
sfrp	Special-function register symbol (16-bit manipulatable register even addresses only)**Note**
saddr	FE20H-FF1FH Immediate data or labels
saddrp	FE20H-FF1FH Immediate data or labels (even address only)
addr16	0000H-FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H-0FFFH Immediate data or labels
addr5	0040H-007FH Immediate data or labels (even address only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

Note: Addresses from FFD0H to FFDFH cannot be accessed with these operands.

Remark: For special-function register symbols, refer to **Table 3-3, “Special Function Register List,” on page 78**.

25.1.2 Description of “operation” column

A : A register; 8-bit accumulator
X : X register
B : B register
C : C register
D : D register
E : E register
H : H register
L : L register
AX : AX register pair; 16-bit accumulator
BC : BC register pair
DE : DE register pair
HL : HL register pair
PC : Program counter
SP : Stack pointer
PSW : Program status word
CY : Carry flag
AC : Auxiliary carry flag
Z : Zero flag
RBS : Register bank select flag
IE : Interrupt request enable flag
NMIS : Non-maskable interrupt servicing flag
( ) : Memory contents indicated by address or register contents in parentheses
X_H, X_L : Higher 8 bits and lower 8 bits of 16-bit register
: Logical product (AND)
: Logical sum (OR)
: Exclusive logical sum (exclusive OR)
——: Inverted data
addr16 : 16-bit immediate data or label
jdisp8 : Signed 8-bit data (displacement value)

25.1.3 Description of “flag operation” column

(Blank) : Not affected
0 : Cleared to 0
1 : Set to 1
X : Set/cleared according to the result
R : Previously saved value is restored

25.2 Operation List

Table 25-2: Operation List (1/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	r, #byte	2	4	-	r← byte			
		saddr, #byte	3	6	7	(saddr) ← byte			
		sfr, #byte	3	-	7	str ← byte			
		A, r Note 3	1	2	-	A ← r			
		r, A Note 3	1	2	-	r ← A			
		A, saddr	2	4	5	A ←(saddr)			
		saddr, A	2	4	5	(saddr) ← A			
		A, sfr	2	-	5	A ← sfr			
		sfr, A	2	-	5	sfr ← A			
		A, !addr16	3	8	9 + n	A ← (addr16)			
		!addr16, A	3	8	9 + m	(addr16) ← A			
		PSW, #byte	3	-	7	PSW ← byte	×	×	×
		A, PSW	2	-	5	A ← PSW			
		PSW, A	2	-	5	PSW ← A	×	×	×
		A, [DE]	1	4	5 + n	A ← (DE)			
		[DE], A	1	4	5 + m	(DE) ← A			
		A, [HL]	1	4	5 + n	A ← (HL)			
		[HL], A	1	4	5 + m	(HL) ← A			
		A, [HL + byte]	2	8	9 + n	A ← (HL + byte)			
		[HL + byte], A	2	8	9 + m	(HL + byte) ← A			
		A, [HL + B]	1	6	7 + n	A ← (HL + B)			
		[HL + B], A	1	6	7 + m	(HL + B) ← A			
		A, [HL + C]	1	6	7 + n	A ← HL + C)			
		[HL + C], A	1	6	7 + m	(HL + C) ← A			
	XCH	A, r Note 3	1	2	-	A ↔ r			
		A, saddr	2	4	6	A ↔ (saddr)			
		A, sfr	2	-	6	A ↔ (sfr)			
		A, !addr16	3	8	10+n+m	A ↔ (addr16)			
		A, [DE]	1	4	6+n+m	A ↔ (DE)			
		A, [HL]	1	4	6+n+m	A ↔ (HL)			
		A, [HL + byte]	2	8	10+n+m	A ↔ (HL + byte)			
		A, [HL + B]	2	8	10+n+m	A ↔ (HL + B)			
		A, [HL + C]	2	8	10+n+m	A ↔ (HL + C)			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (2/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	MOVW	rp, #word	3	6	-	rp ← word			
		saddrp, #word	4	8	10	(saddrp) ← word			
		sfrp, #word	4	-	10	sfrp ← word			
		AX, saddrp	2	6	8	AX ← (saddrp)			
		saddrp, AX	2	6	8	(saddrp) ← AX			
		AX, sfrp	2	-	8	AX ← sfrp			
		sfrp, AX	2	-	8	sfrp ← AX			
		AX, rp Note 4	1	4	-	AX ← rp			
		rp, AX Note 4	1	4	-	rp ← AX			
		AX, !addr16	3	10	12 + 2n	AX ← (addr16)			
		!addr16, AX	3	10	12 + 2m	(addr16) ← AX			
	XCHW	AX, rp Note 4	1	4	-	AX × rp			
8-bit operation	ADD	A, #byte	2	4	-	A, CY ← A + byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	×	×	×
		A, r Note 3	2	4	-	A, CY ← A + r	×	×	×
		r, A	2	4	-	r, CY ← r + A	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A + (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C)	×	×	×
	ADDC	A, #byte	2	4	-	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r Note 3	2	4	-	A, CY ← A + r + CY	×	×	×
		r, A	2	4	-	r, CY ← r + A + CY	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16) + CY	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A + (HL) + CY	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte) + CY	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B) + CY	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C) + CY	×	×	×

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (3/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	SUB	A, #byte	2	4	-	A, CY ← A - byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) - byte	×	×	×
		A, r **Note 3**	2	4	-	A, CY ← A - r	×	×	×
		r, A	2	4	-	r, CY ← r - A	×	×	×
		A, saddr	2	4	5	A, CY ← A - (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A - (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A - (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A - (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A - (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A - (HL + C)	×	×	×
	SUBC	A, #byte	2	4	-	A, CY ← A - byte - CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) - byte - CY	×	×	×
		A, r **Note 3**	2	4	-	A, CY ← A - r - CY	×	×	×
		r, A	2	4	-	r, CY ← r - A - CY	×	×	×
		A, saddr	2	4	5	A, CY ← A - (saddr) - CY	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A - (addr16) - CY	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A - (HL) - CY	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A - (HL + byte) - CY	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A - (HL + B) - CY	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A - (HL + C) - CY	×	×	×
	AND	A, #byte	2	4	-	A ← A ∧ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	×		
		A, r **Note 3**	2	4	-	A ← A ∧ r	×		
		r, A	2	4	-	r ← r ∧ A	×		
		A, saddr	2	4	5	A ← A ∧ (saddr)	×		
		A, !addr16	3	8	9 + n	A ← A ∧ (addr16)	×		
		A, [HL]	1	4	5 + n	A ← A ∧ (HL)	×		
		A, [HL + byte]	2	8	9 + n	A ← A ∧ (HL + byte)	×		
		A, [HL + B]	2	8	9 + n	A ← A ∧ (HL + B)	×		
		A, [HL + C]	2	8	9 + n	A ← A ∧ (HL + C)	×		

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (4/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	OR	A, #byte	2	4	-	A ← A ∨ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	×		
		A, r Note 3	2	4	-	A ← A ∨ r	×		
		r, A	2	4	-	r ← r ∨ A	×		
		A, saddr	2	4	5	A ← A ∨ (saddr)	×		
		A, !addr16	3	8	9 + n	A ← A ∨ (addr16)	×		
		A, [HL]	1	4	5 + n	A ← A ∨ (HL)	×		
		A, [HL + byte]	2	8	9 + n	A ← A ∨ (HL + byte)	×		
		A, [HL + B]	2	8	9 + n	A ← A ∨ (HL + B)	×		
		A, [HL + C]	2	8	9 + n	A ← A ∨ (HL + C)	×		
	XOR	A, #byte	2	4	-	A ← A ∨ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	×		
		A, r Note 3	2	4	-	A ← A ∨ r	×		
		r, A	2	4	-	r ← r ∨ A	×		
		A, saddr	2	4	5	A ← A ∨ (saddr)	×		
		A, !addr16	3	8	9 + n	A ← A ∨ (addr16)	×		
		A, [HL]	1	4	5 + n	A ← A ∨ (HL)	×		
		A, [HL + byte]	2	8	9 + n	A ← A ∨ (HL + byte)	×		
		A, [HL + B]	2	8	9 + n	A ← A ∨ (HL + B)	×		
		A, [HL + C]	2	8	9 + n	A ← A ∨ (HL + C)	×		
	CMP	A, #byte	2	4	-	A - byte	×	×	×
		saddr, #byte	3	6	8	(saddr) - byte	×	×	×
		A, r Note 3	2	4	-	AA - r	×	×	×
		r, A	2	4	-	r - A	×	×	×
		A, saddr	2	4	5	A - (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A - (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A - (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A - (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A - (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A - (HL + C)	×	×	×
16-bit operation	ADDW	AX, #word	3	6	-	AX, CY ← AX + word	×	×	×
	SUBW	AX, #word	3	6	-	AX, CY ← AX - word	×	×	×
	CMPW	AX, #word	3	6	-	AX – word	×	×	×
Multiply/ divide	MULU	X	2	16	-	AX ← A x X			
	DIVUW	C	2	25	-	AX (Quotient), C (Remainder) ← AX ÷ C			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (5/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Increment/ decrement	INC	r	1	2	-	r ← r + 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) + 1	×	×	
	DEC	r	1	2	-	r ← r – 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) – 1	×	×	
	INCW	rp	1	4	-	rp ← rp + 1			
	DECW	rp	1	4	-	rp ← rp - 1			
Rotate	ROR	A, 1	1	2	-	(CY, $A_7 \leftarrow A_0$, $A_m - 1 \leftarrow A_m$) x 1 time			×
	ROL	A, 1	1	2	-	(CY, $A_0 \leftarrow A_7$, $A_m + 1 \leftarrow A_m$) x 1 time			×
	RORC	A, 1	1	2	-	(CY ← A_0, A_7 ← CY, $A_m - 1 \leftarrow A_m$) x 1 time			×
	ROLC	A, 1	1	2	-	(CY ← A_7, A_0 ← CY, $A_m + 1 \leftarrow A_m$) x 1 time			×
	ROR4	[HL]	2	10	12+n+m	$A_{3-0} \leftarrow (HL)_{3-0}$, $(HL)_{7-4} \leftarrow A_{3-0}$, $(HL)_{3-0} \leftarrow (HL)_{7-4}$			
	ROL4	[HL]				$A_{3-0} \leftarrow (HL)_{7-4}$, $(HL)_{3-0} \leftarrow A_{3-0}$, $(HL)_{7-4} \leftarrow (HL)_{3-0}$			
BCD adjust	ADJBA		2	4	-	Decimal Adjust Accumulator after Addition	×	×	×
	ADJBS		2	4	-	Decimal Adjust Accumulator after Subtract	×	×	×
Bit manipulate	MOV1	CY, saddr.bit	3	6	7	CY ← saddr.bit)			×
		CY, sfr.bit	3	-	7	CY ← sfr.bit			×
		CY, A.bit	2	4	-	CY ← A.bit			×
		CY, PSW.bit	3	-	7	CY ← PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY ← (HL).bit			×
		saddr.bit, CY	3	6	8	(saddr.bit) ← CY			
		sfr.bit, CY	3	-	8	sfr.bit ← CY			
		A.bit, CY	2	4	-	A.bit ← CY			
		PSW.bit, CY	3	-	8	PSW.bit ← CY	×	×	
		[HL].bit, CY	2	6	8+n+m	(HL).bit ← CY			
	AND1	CY, saddr.bit	3	6	7	CY ← CY ∧ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY ← CY ∧ sfr.bit			×
		CY, A.bit	2	4	-	CY ← CY ∧ A.bit			×
		CY, PSW.bit	3	-	7	CY ← CY ∧ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∧ (HL).bit			×

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (6/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulate	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY ← CY ∨ sfr.bit			×
		CY, A.bit	2	4	-	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	-	7	CY ← CY ∨ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∨ (HL).bit			×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ∨ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY ← CY ∨ sfr.bit			×
		CY, A.bit	2	4	-	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	-	7	CY ← CY ∨ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∨ (HL).bit			×
	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1			
		sfr.bit	3	-	8	sfr.bit ← 1			
		A.bit	2	4	-	A.bit ← 1			
		PSW.bit	2	-	6	PSW.bit ← 1	×	×	×
		[HL].bit	2	6	8+n+m	(HL).bit ← 1			
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0			
		sfr.bit	3	-	8	sfr.bit ← 0			
		A.bit	2	4	-	A.bit ← 0			
		PSW.bit	2	-	6	PSW.bit ← 0	×	×	×
		[HL].bit	2	6	8+n+m	(HL).bit ← 0			
	SET1	CY	1	2	-	CY ← 1			1
	CLR1	CY	1	2	-	CY ← 0			0
	NOT1	CY	1	2	-	CY ← $\overline{CY}$			×

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (7/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Call/return	CALL	!addr16	3	7	-	$(SP-1) \leftarrow (PC+3)_H$, $(SP-2) \leftarrow (PC+3)_L$, PC ←addr16, SP ←SP – 2			
	CALLF	!addr11	2	5	-	$(SP-1) \leftarrow (PC+2)_H$, $(SP-2) \leftarrow (PC+2)_L$, $PC_{15-11} \leftarrow 00001$, $PC_{10-0} \leftarrow$ addr11, SP ← SP – 2			
	CALLT	[addr5]	1	6	-	$(SP-1) \leftarrow (PC+1)_H$, $(SP-2) \leftarrow (PC+1)_L$, $PC_H \leftarrow$ (00000000, addr5 + 1), $PC_L \leftarrow$ (00000000, addr5), SP ← SP – 2			
	BRK		1	6	-	$(SP-1) \leftarrow PSW$, $(SP-2) \leftarrow (PC+1)_H$, $(SP-3) \leftarrow (PC+1)_L$, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0			
	RET		1	6	-	$PC_H \leftarrow (SP+1)$, $PC_L \leftarrow (SP)$, SP ← SP + 2			
	RETI		1	6	-	$PC_H \leftarrow (SP+1)$, $PC_L \leftarrow (SP)$, PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R	R	R
	RETB		1	6	-	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R	R	R
Stack manipulate	PUSH	PSW	1	2	-	$(SP-1) \leftarrow PSW$, SP ← SP – 1			
		rp	1	4	-	$(SP-1) \leftarrow rp_H$, $(SP-2) \leftarrow rp_L$, SP ← SP – 2			
	POP	PSW	1	2	-	PSW ← (SP), SP ← SP + 1	R	R	R
		rp	1	4	-	$rp_H \leftarrow (SP+1)$, $rp_L \leftarrow (SP)$, SP ← SP + 2			
	MOVW	SP, #word	4	-	10	SP ← word			
		SP, AX	2	-	8	SP ← AX			
		AX, SP	2	-	8	AX ← SP			
Unconditional branch	BR	!addr16	3	6	-	PC ← addr16			
		$addr16	2	6	-	PC ← PC + 2 + jdisp8			
		AX	2	8	-	$PC_H \leftarrow A$, PCL ← X			
Conditional branch	BC	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if CY = 1			
	BNC	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if CY = 0			
	BZ	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if Z = 1			
	BNZ	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if Z = 0			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 25-2: Operation List (8/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Conditional branch	BT	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if(saddr.bit) = 1			
		sfr.bit, $addr16	4	-	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1			
		A.bit, $addr16	3	8	-	PC ← PC + 3 + jdisp8 if A.bit = 1			
		PSW.bit, $addr16	3	-	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1			
		[HL].bit, $addr16	3	10	11 + n	PC ← PC + 3 + jdisp8 if (HL).bit = 1			
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if(saddr.bit) = 0			
		sfr.bit, $addr16	4	-	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0			
		A.bit, $addr16	3	8	-	PC ← PC + 3 + jdisp8 if A.bit = 0			
		PSW.bit, $addr16	4	-	11	PC ← PC + 4 + jdisp8 if PSW. bit = 0			
		[HL].bit, $addr16	3	10	11 + n	PC ← PC + 3 + jdisp8 if (HL).bit = 0			
	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if(saddr.bit) = 1 then reset(saddr.bit)			
		sfr.bit, $addr16	4	-	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit			
		A.bit, $addr16	3	8	-	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit			
		PSW.bit, $addr16	4	-	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	×	×	×
		[HL].bit, $addr16	3	10	12+n+m	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit			
	DBNZ	B, $addr16	2	6	-	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0			
		C, $addr16	2	6	-	C ← C –1, then PC ← PC + 2 + jdisp8 if C ≠ 0			
		saddr. $addr16	3	8	10	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if(saddr) ≠ 0			
CPU control	SEL	RBn	2	4	-	RBS1, 0 ← n			
	NOP		1	2	-	No Operation			
	EI		2	-	6	IE ← 1(Enable Interrupt)			
	DI		2	-	6	IE ← 0(Disable Interrupt)			
	HALT		2	6	-	Set HALT Mode			
	STOP		2	6	-	Set STOP Mode			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

25.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

Table 25-3: 8-bit instructions

2nd Operand / 1st Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte] [HL + B] [HL + C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP										DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL + byte] [HL + B] [HL + C]		MOV											
X													MULU
C													DIVUW

Note: Except r = A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

Table 25-4: 16-bit instructions

2nd Operand / 1st Operand	#word	AX	rp[Note]	sfrp	saddrp	!addr16	sp	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
sp	MOVW	MOVW						

Note: Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

Table 25-5: Bit manipulation instructions

2nd Operand / 1st Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call/instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

Table 25-6: Call/instructions/branch instructions

2nd Operand / 1st Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

[MEMO]

Chapter 26 Electrical Specifications (µPD780822B Subseries)

26.1 Absolute Maximum Ratings

(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = 25°C)

(1/2)

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}		-0.3 to +6.5	V
	V_{SS}		-0.3 to +0.3	
	AV_{DD}/ AV_{REF0}	$AV_{DD} = V_{DD}$	-0.3 to V_{DD0}+0.3**Note**	
	AV_{SS}		-0.3 to +0.3	
	SMV_{DD}	$SMV_{DD} = V_{DD}$, V_{DD} = 5 V ± 10%	-0.3 to +6.5	
	SMV_{SS}		-0.3 to +0.3	
Input voltage	V_{I1}	P00 to P03, P10 to P17, P20 to P27, P30 to P33, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P140 to P147, P150 to P157, X1, X2, XT1, XT2, $\overline{\text{RESET}}$	-0.3 to V_{DD1}+0.3**Note**	
Output voltage	V_O		-0.3 to V_{DD1} + 0.3**Note**	
Analog input voltage	V_{AN}		AV_{SS} -0.3 to AV_{REF0}+0.3**Note** and -0.3 to V_{DD1}+0.3**Note**	
Output current, high	I_{OH}	P33	-20	mA
		P140 to P147 per pin	-35	
		P150 to P157 per pin	-35	
		P140 to P147 total	-80	
		P150 to P157 total	-80	
		Per pin except P33, P140 to P147, P150 to P157	-10	
		P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130 total	-30	

Note: Must be 6.5 V or lower.

Caution: Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

Remark: Unless otherwise specified, the characteristics of alternate-function pins are the same as those of port pins.

(2/2)

Parameter	Symbol	Conditions		Rating	Unit
Output current, low	I_{OL}	P33		20	mA
		P140 to P147 per pin		35	
		P150 to P157 per pin		35	
		P140 to P147 total		80	
		P150 to P157 total		80	
		Per pin except P33, P140 to P147, P150 to P157		10	
		P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130 total		30	
Maximum current		Total through V_{DD}, SMV_{DD} and/or V_{SS}/SMV_{SS}	µPD780821B(A) µPD780822B(A) µPD78F0822B(A)	200	
Operating ambient temperature	T_A	In normal operation mode		-40 to +85	°C
Storage temperature	T_{STG}	µPD780821B/µPD780822B		-65 to +150	
		µPD78F0822B		-40 to +125	

Caution: Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

Remark: Unless otherwise specified, the characteristics of alternate-function pins are the same as those of port pins.

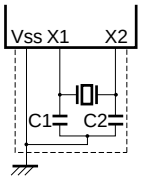
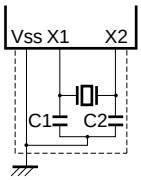
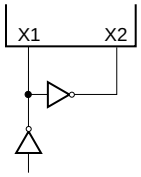
26.2 Capacitance

(1) μPD780821B(A), μPD780822B(A), μPD78F0822B(A) (T_A = 25°C, V_{DD} = V_{SS} = 0 V)

Parameter	Symbol	Function		Min.	Typ.	Max.	Unit
Input Capacitance	C_{IN}	f = 1MHz Other than measured pins: 0 V				15	pF
			P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130 per pin			15	pF
I/O Capacitance	C_{IO}		P33, P140 to P147, P150 to P157			30	pF

26.3 Main System Clock Oscillation Circuit Characteristics

(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended Circuit	Parameter	Conditions	Min.	Typ.	Max.	Unit
Ceramic resonator		Oscillator frequency (f_{XP}) **Note**	V_{DD} = 4.5 to 5.5 V	4.0		16.0	MHz
			V_{DD} = 4.0 to 4.5 V	4.0		8.0	
Crystal resonator		Oscillator frequency (f_{XP}) **Note**	V_{DD} = 4.5 to 5.5 V	4.0		16.0	MHz
			V_{DD} = 4.0 to 4.5 V	4.0		8.0	
External clock		X1 input frequency (f_{XP}) **Note**	V_{DD} = 4.5 to 5.5 V	4.0		16.0	MHz
			V_{DD} = 4.0 to 4.5 V	4.0		8.0	
		X1 input high-/low-level width (t_{XPH}, t_{XPL})	V_{DD} = 4.5 to 5.5 V	31.25		125.0	ns
			V_{DD} = 4.0 to 4.5 V	62.5		125.0	

Note: Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.

Cautions: 1. When using the X1 oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

2. Since the CPU is started by the internal low-speed oscillator after reset is released, check the oscillation stabilization time of the X1 input clock using the oscillation stabilization time status register (OSTC). Determine the oscillation stabilization time of the OSTC register and oscillation stabilization time select register (OSTS) after sufficiently evaluating the oscillation stabilization time with the resonator to be used.

Remark: For the resonator selection and oscillator constant, customers are requested to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

26.4 Internal Low-speed Oscillator Oscillation Circuit Characteristics

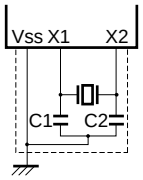
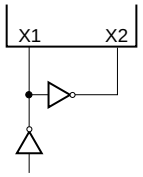
(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Parameter	Conditions	Min.	Typ.	Max.	Unit
Internal low-speed oscillator	Oscillator frequency (f_R)		120	240	480	kHz

26.5 Subsystem Clock Oscillation Circuit Characteristics

(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A)

(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended Circuit	Parameter	Conditions	Min.	Typ.	Max.	Unit
Crystal resonator		Oscillator frequency (f_{XT}) **Note**		32	32.768	35	kHz
External clock		XT1 input frequency (f_{XT}) **Note**		32		38.5	kHz
		XT1 input high-/low-level width (t_{XTH}, t_{XTL})		12		17	µs

Note: Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.

Cautions: 1. When using the subsystem clock oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

2. The subsystem clock oscillator is designed as a low-amplitude circuit for reducing power consumption, and is more prone to malfunction due to noise than the X1 oscillator. Particular care is therefore required with the wiring method when the subsystem clock is used.

Remark: For the resonator selection and oscillator constant, customers are requested to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

26.6 DC Characteristics

(1) μPD780821B(A), μPD780822B(A), μPD78F0822B(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

(1/2)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Output current, high	I_{OH}	P33	4.5 V ≤ SMV_{DD} ≤ 5.5 V			-15	mA
		P140 to P147 per pin	4.5 V ≤ SMV_{DD} ≤ 5.5 V			-27	
		P150 to P157 per pin				-27	
		P140 to P147 total				-70	
		P150 to P157 total				-70	
		Per pin except P33, P140 to P147, P150 to P157				-5	
		P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130 total				-25	
Output current, low	I_{OL}	P33	4.5 V ≤ SMV_{DD} ≤ 5.5 V			15	
		P140 to P147 per pin	4.5 V ≤ SMV_{DD} ≤ 5.5 V			27	
		P150 to P157 per pin				27	
		P140 to P147 total				70	
		P150 to P157 total				70	
		Per pin except P33, P140 to P147, P150 to P157				5	
		P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130 total				25	
Input voltage, high	V_{IH1}	P00 to P03, P11, P12, P14 to P17, P30 to P33, P120		0.8 V_{DD}		V_{DD}	V
		P10, P13, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113		0.7 V_{DD}		V_{DD}	
	V_{IH2}	$\overline{RESET}$		0.8 V_{DD}		V_{DD}	
	V_{IH3}	P20 to P27**Note**		0.7 AV_{REF0}		AV_{REF0}	
	V_{IH4}	X1, X2, XT1, XT2		V_{DD}-0.5		V_{DD}	
Input voltage, low	V_{IL1}	P00 to P03, P11, P12, P14 to P17, P30 to P33, P120		0		0.2 V_{DD}	V
		P10, P13, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113		0		0.3 V_{DD}	
	V_{IL2}	$\overline{RESET}$		0		0.2 V_{DD}	
	V_{IL3}	P20 to P27**Note**		0		0.3 AV_{REF0}	
	V_{IL4}	X1, X2, XT1, XT2		0		0.4	

Note: When used as A/D converter analog input pins and AV_{REF0} = V_{DD0}

(2/2)

Parameter	Symbol	Conditions			Min.	Typ.	Max.	Unit
Output voltage, high	V_{OH1}	P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130	I_{OH} = -1mA		V_{DD}-1.0			V
	V_{OH2}	P140 to P147, P150 to P157	I_{OH} = -27mA (T_A = 85°C)	4.5 V ≤ SMV_{DD} ≤ 5.5 V	V_{DD}-0.5			
			I_{OH} = -30mA (T_A = 25°C)					
			I_{OH} = -40mA (T_A = -40°C)					
	V_{OH3}	P33	I_{OH} = -15mA	4.5 V ≤ SMV_{DD} ≤ 5.5 V	V_{DD}-0.7		V_{DD}	
Output voltage, low	V_{OL1}	P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130	I_{OL} = 1.6mA				0.4	V
	V_{OL2}	P140 to P147, P150 to P157	I_{OL} = 27mA (T_A = 85°C)	4.5 V ≤ SMV_{DD} ≤ 5.5 V			0.5	
			I_{OL} = 30mA (T_A = 25°C)					
			I_{OL} = 40mA (T_A = -40°C)					
	V_{OL3}	P33	I_{OL} = 15mA	4.5 V ≤ SMV_{DD} ≤ 5.5 V			0.7	
Input leakage current, high	I_{L1H1}	V_I = V_{DD}	P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130, $\overline{RESET}$				3	μA
		V_I = AV_{REF0}	P20 to P27					
	I_{L1H2}	V_I = V_{DD}	X1, X2, XT1, XT2**Note**				20	
Input leakage current, low	I_{L1L1}	V_I = 0 V	P00 to P03, P10 to P17, P30 to P32, P70 to P77, P80 to P87, P90 to P97, P100 to P105, P110 to P113, P120, P130, $\overline{RESET}$				-3	μA
		V_I = AV_{SS0}	P20 to P27					
	I_{L1L2}	V_I = 0 V	X1, X2, XT1, XT2**Note**				-20	
Output leakage current, high	I_{LOH}	V_O = V_{DD}	P140 to P147, P150 to P157				3	μA
Output leakage current, low	I_{LOL}	V_O = 0 V	P140 to P147, P150 to P157				-3	μA
Software pull-up resistor	R_L	V_I = 0 V			10	30	100	kΩ
FLMD0 supply voltage	V_{FLMD}	In normal operation mode			0		0.2 V_{DD}	V

Note: The inverse level of X1 is input to X2 and the inverse level of XT1 is input to XT2

µPD78F0822B(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 V to 5.0 V)

Parameter	Symbol	Conditions			Min.	Typ.	Max.	Unit
Supply current**Note 1**	I_{DD1}**Note 2**	X1 crystal oscillation operating mode	f_X = 16 MHz**Note 3** 4.5 V to 5.5 V	A/D converter is stopped		30.5	45.0	mA
				A/D converter is operating**Note 4**		31.5	46.0	
			f_X = 8 MHz**Note 3** 4.0 V to 5.5 V	A/D converter is stopped		17.0	25.5	mA
				A/D converter is operating**Note 4**		18.0	27.0	
	I_{DD2}	X1 crystal oscillation HALT mode	f_X = 16 MHz**Note 3** 4.5 V to 5.5 V	Peripheral functions are stopped		3.0	6.0	mA
				Peripheral functions are operating		18.0	27.0	
			f_X = 8 MHz**Note 3** 4.0 V to 5.5 V	Peripheral functions are stopped		2.0	4.0	mA
				Peripheral functions are operating		10.0	15.0	
	I_{DD3}	Internal low-speed oscillator operating mode**Note 5**		Peripheral functions are stopped		1.5	4.5	mA
	I_{DD4}	32.768 kHz crystal oscillation operating mode**Note 4**				100	200	µA
	I_{DD5}	32.768 kHz crystal oscillation HALT mode**Note 4**				40	80	µA
	I_{DD6}	STOP mode **Note 6**		Internal low-speed oscillator off		3.5	35	µA
				Internal low-speed oscillator on		20	65	

Notes: 1. Total current flowing through the internal power supply (V_{DD}). Peripheral operation current is included (however, the current that flows through the pull-up resistors of ports is not included).

2. I_{DD1} includes the peripheral operation current.

3. When PCC = 00H.

4. Total current flowing through the V_{DD} and AV_{REF} pins.

5. When X1 oscillation is stopped.

6. XT1 = 0 V.

Remark: Typical values are with respect to T_A = 25°C.

μPD780821B(A), μPD780822B(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 V to 5.5 V)

Parameter	Symbol	Conditions			Min.	Typ.	Max.	Unit
Supply current**Note 1**	I_{DD1}**Note 2**	X1 crystal oscillation operating mode	f_X = 16 MHz**Note 3** 4.5 V to 5.5 V	A/D converter is stopped		25.0	40.0	mA
				A/D converter is operating**Note 4**		26.0	41.0	
			f_X = 8 MHz**Note 3** 4.0 V to 5.5 V	A/D converter is stopped		12.5	20.0	mA
				A/D converter is operating**Note 4**		13.5	21.0	
	I_{DD2}	X1 crystal oscillation HALT mode	f_X = 16 MHz**Note 3** 4.5 V to 5.5 V	Peripheral functions are stopped		3.0	6.0	mA
				Peripheral functions are operating		18.0	27.0	
			f_X = 8 MHz**Note 3** 4.0 V to 5.5 V	Peripheral functions are stopped		2.0	4.0	mA
				Peripheral functions are operating		10.0	15.0	
	I_{DD3}	Internal low-speed oscillator operating mode**Note 5**		Peripheral functions are stopped		1.5	4.5	mA
	I_{DD4}	32.768 kHz crystal oscillation operating mode**Note 4**				60	120	μA
	I_{DD5}	32.768 kHz crystal oscillation HALT mode**Note 4**				40	80	μA
	I_{DD6}	STOP mode **Note 6**		Internal low-speed oscillator off		3.5	35	μA
				Internal low-speed oscillator on		20	65	

Notes: 1. Total current flowing through the internal power supply (V_{DD}). Peripheral operation current is included (however, the current that flows through the pull-up resistors of ports is not included).

2. I_{DD1} includes the peripheral operation current.

3. When PCC = 00H.

4. Total current flowing through the V_{DD} and AV_{REF} pins.

5. When X1 oscillation is stopped.

6. XT1 = 0 V.

Remark: Typical values are with respect to T_A = 25°C.

μPD780821B(A), μPD780822B(A), μPD78F0822B(A)

LCD C/D 1/3 Bias Method

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
LCD drive voltage	V_{LCD}			3.0		V_{DD}	V
LCD split resistor	R_{LCD}			5	15	45	kΩ
LCD output voltage deviation[Note] (common)	V_{ODC}	$I_O = \pm 5\mu A$	$3.0\ V \le V_{LCD} \le V_{DD}$ $V_{LCD0} = V_{LCD}$ $V_{LCD1} = V_{LCD} \times 2/3$ $V_{LCD2} = V_{LCD1} \times 1/3$	0		±0.2	V
LCD output voltage deviation[Note] (segment)	V_{ODS}	$I_O = \pm 1\mu A$				±0.2	V

Note: The voltage deviation is the difference from the output voltage corresponding to the ideal value of the segment and common outputs (V_{LCD}).

26.7 AC Characteristics

26.7.1 Basic Operation

(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

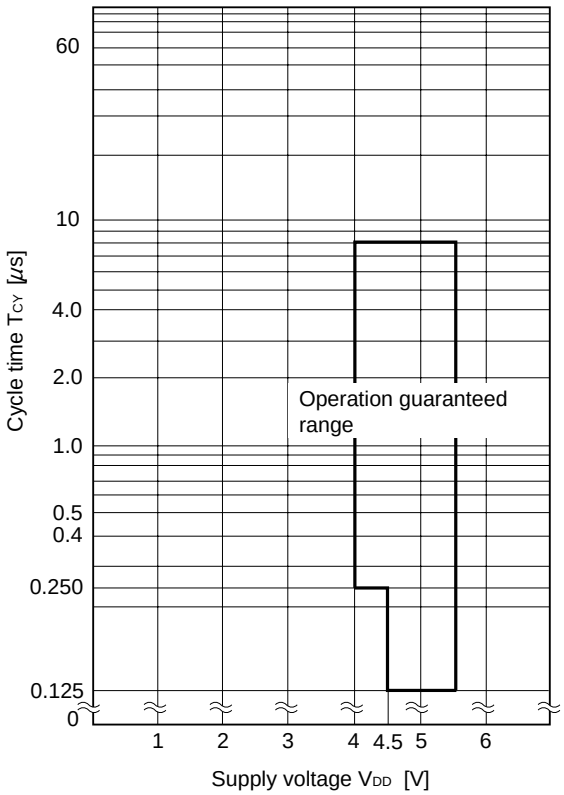
Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Cycle time**Note 1**	T_{CY}	X1 input clock operation	V_{DD} = 4.5 V to 5.5 V	0.125		8	µs
				0.250		8	
		Internal low-speed oscillator clock operation		4.17	8.33	16.67	
		Subsystem clock operation		114	122	125	
TI50, TI51 input frequency	f_{TI5}			0		10	MHz
TI50, TI51 input high/low level width	t_{TIH5} t_{TIL5}			50			ns
TI20, TI21, TI22 input high/low level width	t_{TIH2} t_{TIL2}			$3/f_{SMP2}$ **Note 2**			µs
Interrupt input high/low level width	T_{INTH} T_{INTL}	INTP0 - INTP2		1			
$\overline{\text{RESET}}$ low level width	t_{RSL}			10			

Notes: 1. The cycle time equals to the minimum instruction execution time.

- For example: 1 NOP instruction corresponds to 2 CPU clock cycles (f_{CPU}) selected by the processor clock control register (PCC) and the main clock mode register (MCM).

2. f_{SMP2} (sampling clock) = $f_X/8$, $f_X/16$, $f_X/32$, $f_X/64$

T_{CY} vs V_{DD}


60
10
4.0
2.0
1.0
0.5
0.4
0.250
0.125
0
Cycle time T_{CY} [µs]
Operation guaranteed range
1
2
3
4
4.5
5
6
Supply voltage V_{DD} [V]

26.7.2 Serial interface

(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

(a) Serial interface CSI (SIO10, SIO11)

Master mode, $\overline{SCK10}$, $\overline{SCK11}$ Internal clock output

Parameter	Symbol	Conditions	Min.	Max.	Unit
$\overline{SCK1}$ cycle time	t_{KCY1}		200		ns
$\overline{SCK1}$ high/low level width	t_{KH1} t_{KL1}		t_{KCY1}/2-10		
SI10; SI11 setup time (to $\overline{SCK10}$, $\overline{SCK11}$) ↑	t_{SIK1}		30		
SI10; SI11 hold time (from $\overline{SCK10}$, $\overline{SCK11}$) ↑	t_{KSI1}		30		
SO10; SO11 output delay time (to $\overline{SCK10}$, $\overline{SCK11}$) ↓	t_{KSO1}	C = 100pF**Note**		30	

Note: C is the load capacitance of the $\overline{SCK10}$, $\overline{SCK11}$ and SO10, SO11 output lines.

Slave mode, $\overline{SCK10}$, $\overline{SCK11}$ External clock input

Parameter	Symbol	Conditions	Min.	Max.	Unit
$\overline{SCK1}$ cycle time	t_{KCY2}		400		ns
$\overline{SCK1}$ high/low level width	t_{KH2} t_{KL2}		t_{KCY2}/2		
SI10; SI11 setup time (to $\overline{SCK10}$, $\overline{SCK11}$) ↑	t_{SIK2}		80		
SI10; SI11 hold time (from $\overline{SCK10}$, $\overline{SCK11}$) ↑	t_{KSI2}		50		
SO10; SO11 output delay time (to $\overline{SCK10}$, $\overline{SCK11}$) ↓	t_{KSO2}	C = 100pF**Note**		120	

Note: C is the load capacitance of the SO10, SO11 output lines.

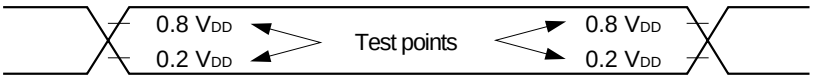
(b) Serial interface UART (UART6, dedicated baud rate generator output)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Transfer rate					312.5	kbps

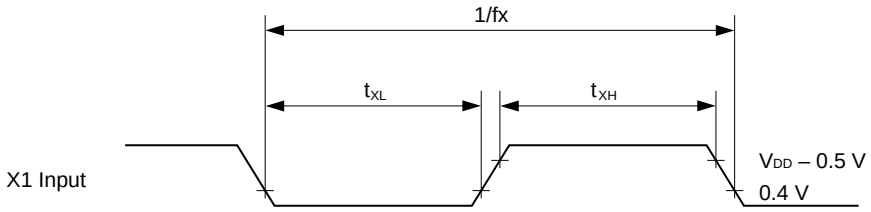
(c) Serial interface CAN

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Transfer rate					1	Mbps

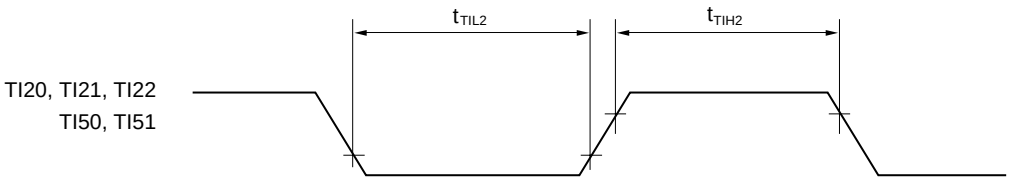
AC Timing Test Points (excluding X1 input)



Clock Timing

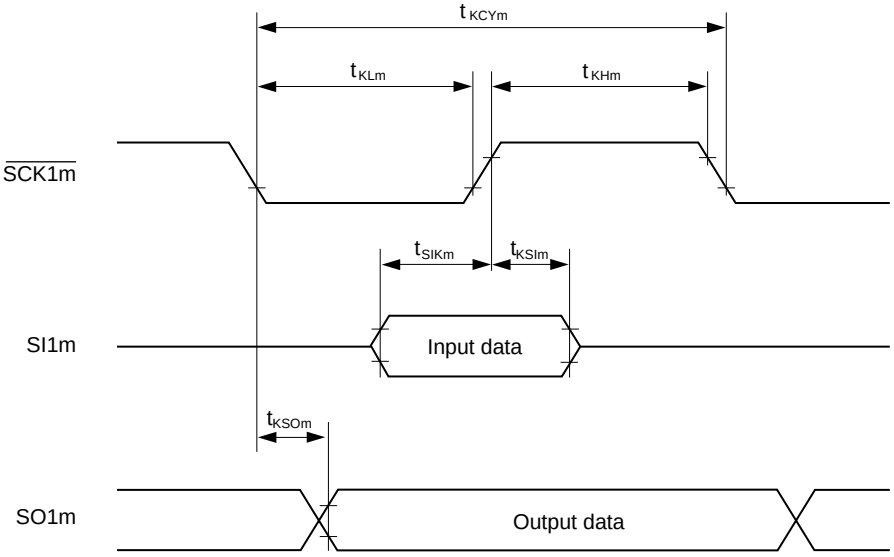


TI Timing



Remark: n = 2, 5

TI Timing



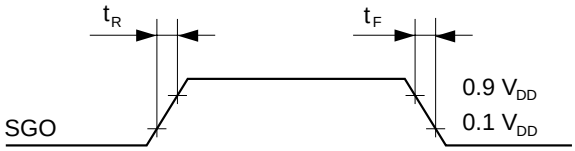
Remark: m = 0, 1

26.7.3 Sound Generator Characteristics

(1) μPD780821B(A), μPD780822B(A), μPD78F0822B(A) (T_A = -40°C to 85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Sound generator input frequency	f_{SG1}	V_{DD} = 4.5 V to 5.5 V			16	MHz
					8	
SGO output rise time	f_R	C = 100pF			200	ns
SGO output fall time	f_F	C = 100pF			200	ns

Sound Generator Output Timing



26.7.4 Meter Controller/Driver Characteristics

(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = -40°C to 85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Meter Controller/Driver input frequency	f_{MC} **Note 1**	V_{DD} = 4.5 V to 5.5 V			16	MHz
					8	
PWM output rise time	f_R	C = 100pF **Note 2**	80		200	ns
SGO output rise time	f_F	C = 100pF **Note 2**	80		200	ns
Symmetry performance **Note 3**	ΔHSPmn	I_{OH} = -27 mA ΔHSPmn = | V_{OH}[$(SMmn)_{max}$ - $(SMmn)_{min}$] |			50	mV
	ΔHSPmn	I_{OH} = 27 mA ΔHSPmn = | V_{OH}[(SMmn)max - (SMmn)min] |			50	mV

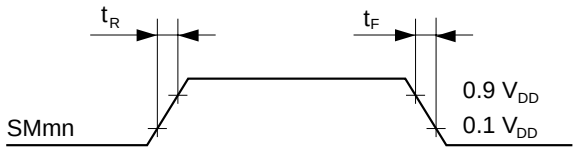
Notes: 1. Source clock of the free-running counter.

2. C is the load capacitance of the PWM output line.

3. Indicates the dispersion of 16 PWM output voltages.

Remark: m = 1 to 4, n = 1 to 4

Meter Controller/Driver Output Timing



26.7.5 A/D Converter Characteristics

(1) μPD780821B(A), μPD780822B(A), μPD78F0822B(A)

(T_A = -40°C to 85°C, V_{DD} = 4.0 to 5.5 V, 4.0 V ≤ AV_{REF0} ≤ AV_{DD} ≤ V_{DD}, V_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution			10	10	10	bit
Overall error[Note 1, 2]				±0.2	±0.4	%FSR
Conversion time	t_{CONV}		14		100	μs
Zero scale error[Note 1, 2]					±0.4	%FSR
Full-scale error[Note 1, 2]					±0.4	%FSR
Integral non-linearity error[Note 1]					±2.5	LSB
Differential non-linearity error[Note 1]					±1.5	LSB
Analog input voltage	V_{AIN}		AV_{SS}		AV_{REF0}	mV

Notes: 1. Excludes quantization error (±1/2 LSB).

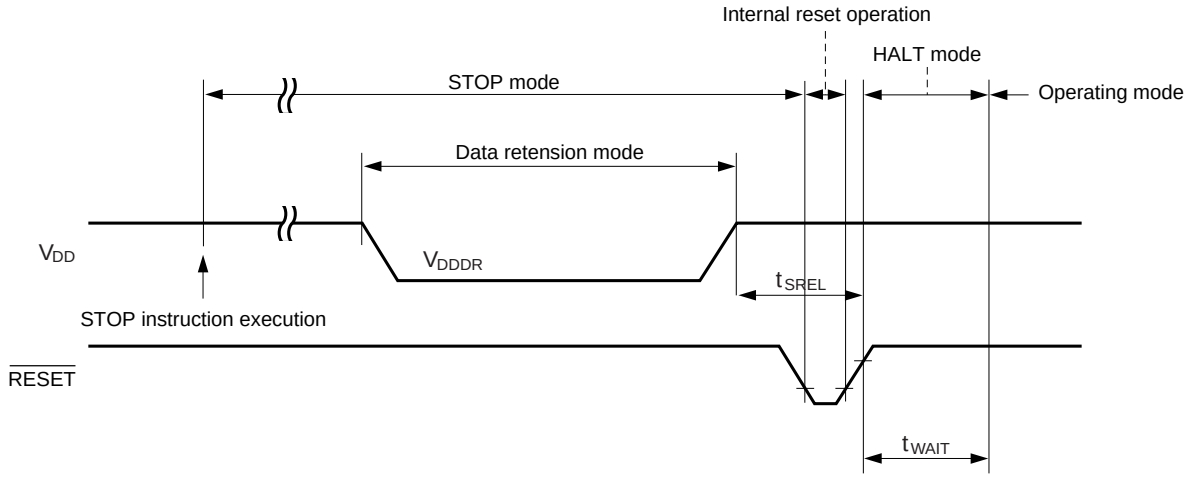
2. This value is indicated as a ratio (%FSR) to the full-scale value.

26.7.6 Data Memory Stop Mode Low Supply Voltage Data Retention Characteristics

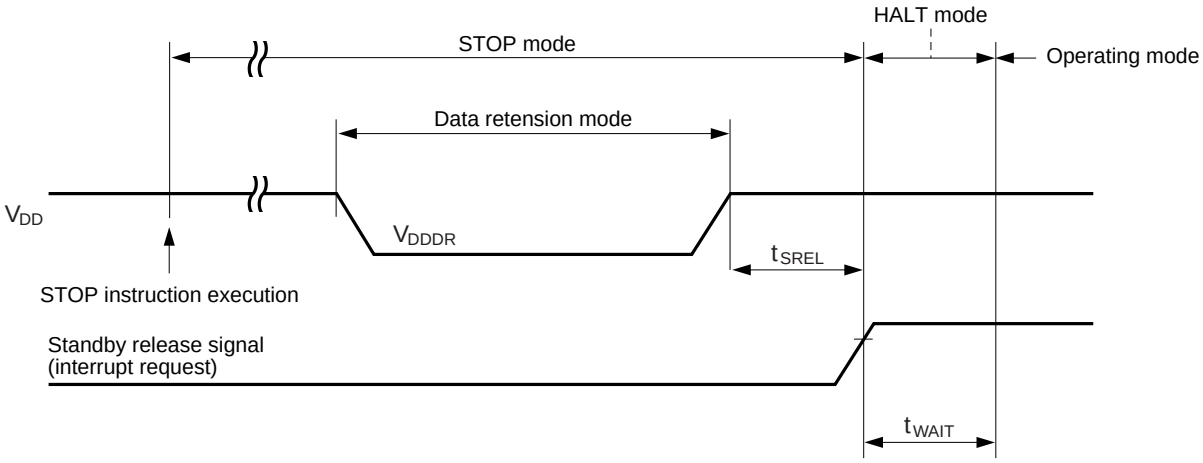
(1) µPD780821B(A), µPD780822B(A), µPD78F0822B(A) (T_A = -40°C to 85°C)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Data retention power supply voltage	V_{DDDR}		2.5		5.5	V
Release signal set time	t_{SREL}		0			µs

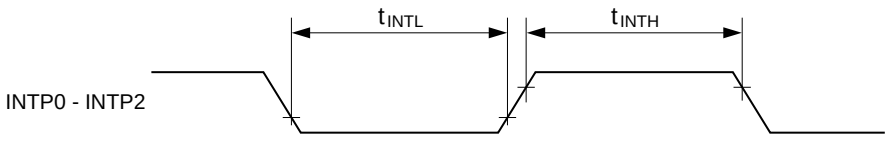
Data Retention Timing (STOP mode release by $\overline{\text{RESET}}$)



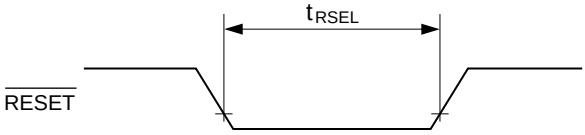
Data Retention Timing (Standby release signal: STOP mode release by interrupt signal)



Interrupt Input Timing



$\overline{\text{RESET}}$ Input Timing



26.8 Flash Memory Programming Characteristics

µPD78F0822B(A)

(T_A = 10°C to 65°C, 4.0 ≤ V_{DD} ≤ 5.5 V, 4.0 V ≤ AV_{DD} = AV_{REF0} ≤ V_{DD}, V_{SS} = AV_{SS} = 0 V)

(1) Basic Characteristics

Parameter		Symbol	Conditions	Min.	Typ.	Max.	Unit
V_{DD} supply current		I_{DD}	f_X = 16 MHz, V_{DD} = 5.5 V			70.5	mA
Step erase time	sector unit	T_{ERASS}			10.0		ms
Erase time**Note 1**	sector unit	T_{ERASA}			0.01	2.55	s
Programming time**Note 2**		T_{WRWA}			50	500	µs
Number of rewrites per chip		C_{ERWR}	1 erase + 1 write after erase = 1 rewrite**Note 3**	100			times

Notes: 1. The erase verify time is not included.

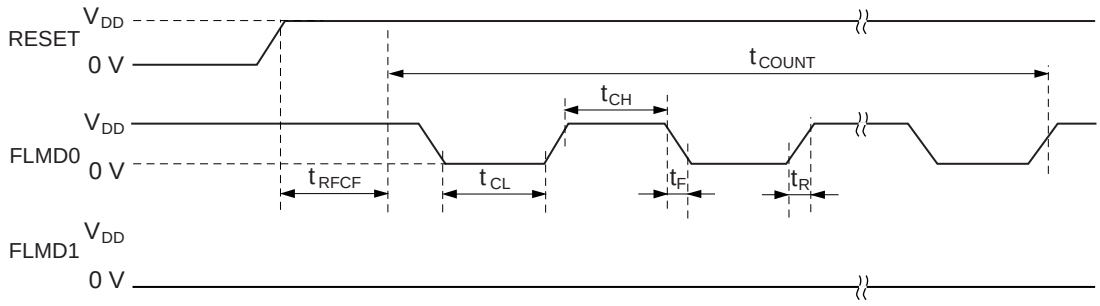
2. After programming a sector in word units.

3. When a product is first written after shipment, “erase → write” and “write only” are both taken as one rewrite.

(2) Serial Write Operation Characteristics

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Time from $\overline{RESET}\uparrow$ to FLMD0 count start	T_{RFCF}		$2^{19}/f_X + \alpha$			µs
Count execution time	T_{COUNT}				10	ms
FLMD0 counter high-/low-level width	T_{CH}/T_{CL}		10		100	µs

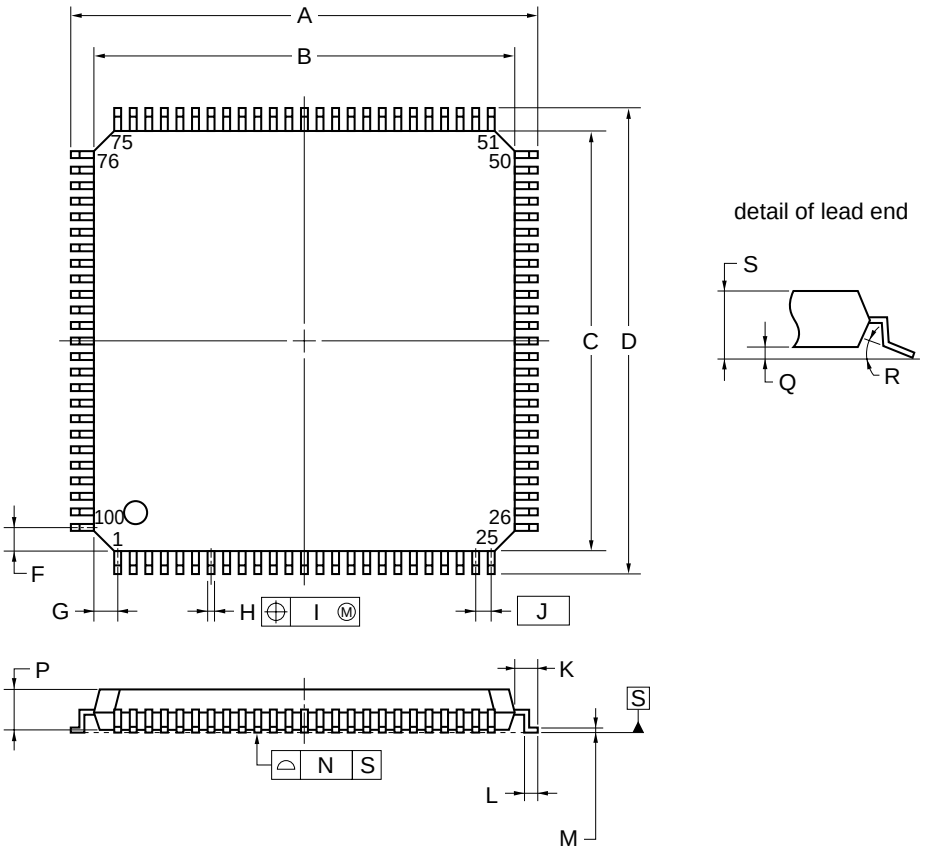
Serial write operation



[MEMO]

Chapter 27 Package Drawing (µPD780822B Subseries)

100-PIN PLASTIC LQFP (FINE PITCH) (14x14)



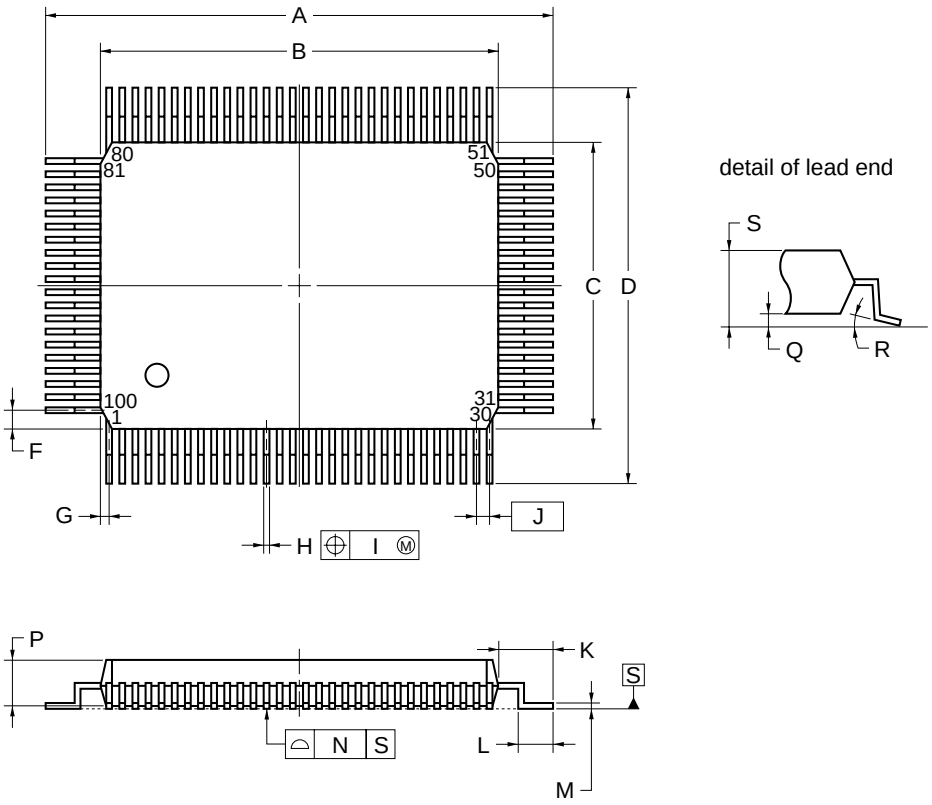
NOTE

Each lead centerline is located within 0.08 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	16.00±0.20
B	14.00±0.20
C	14.00±0.20
D	16.00±0.20
F	1.00
G	1.00
H	$0.22^{+0.05}_{-0.04}$
I	0.08
J	0.50 (T.P.)
K	1.00±0.20
L	0.50±0.20
M	$0.17^{+0.03}_{-0.07}$
N	0.08
P	1.40±0.05
Q	0.10±0.05
R	$3^{\circ}{}^{+7^{\circ}}_{-3^{\circ}}$
S	1.60 MAX.

S100GC-50-8EU, 8EA-2

100-PIN PLASTIC QFP (14x20)



NOTE

Each lead centerline is located within 0.15 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	23.6±0.4
B	20.0±0.2
C	14.0±0.2
D	17.6±0.4
F	0.8
G	0.6
H	0.30±0.10
I	0.15
J	0.65 (T.P.)
K	1.8±0.2
L	0.8±0.2
M	$0.15^{+0.10}_{-0.05}$
N	0.10
P	2.7±0.1
Q	0.1±0.1
R	5°±5°
S	3.0 MAX.

P100GF-65-3BA1-4

Chapter 28 Recommended Soldering Conditions (µPD780822B Subseries)

The µPD780822B Subseries should be soldered and mounted under the conditions in the table below. For detail of recommended soldering conditions, refer to the information document **Semiconductor Device Mounting Technology Manual (IEI-1207)**.
For soldering methods and conditions other than those recommended below, consult our sales personnel.

- µPD780821BGC(A)-XXX-8EA : 100-pin plastic QFP (14 × 14 mm)
- µPD780821BGF(A)-XXX-3BA : 100-pin plastic QFP (14 × 20 mm)
- µPD780822BGC(A)-XXX-8EA : 100-pin plastic QFP (14 × 14 mm)
- µPD780822BGF(A)-XXX-3BA : 100-pin plastic QFP (14 × 20 mm)
- µPD78F0822BGC(A)-8EA : 100-pin plastic QFP (14 × 14 mm)
- µPD78F0822BGF(A)-3BA : 100-pin plastic QFP (14 × 20 mm)

Surface Mounting Type Soldering Conditions

Soldering Method	Soldering conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C. Duration: 30 sec max. (at 210°C or above). Number of times: twice max. <Precautions> (1) The second reflow should be started after the first reflow device temperature has returned to the ordinary state. (2) Flux washing must not be performed by the use of water after the first reflow.	IR35-107-2
VP	Package peak temperature: 215°C. Duration: 40 sec max. (at 210°C or above). Number of times: twice max. <Precautions> (1) The second reflow should be started after the first reflow device temperature has returned to the ordinary state. (2) Flux washing must not be performed by the use of water after the first reflow.	VP15-107-2
Wave soldering	Soldering bath temperature: 260°C max. Duration: 10 sec max. Number of times: once. Preheating temperature: 120°C max. (package surface temperature)	WS60-107-1
Pin part heating	Pin temperature: 300°C max. Duration: 3 sec max. (per device side)	-

Caution: Use of more than one soldering method should be avoided (except in the case of pin part heating).

Chapter 29 Cautions For Wait (μPD780822B Subseries)

29.1 Cautions for Wait

This product has two internal system buses.
One is a CPU bus and the other is a peripheral bus that interfaces with the low-speed peripheral hardware.
Because the clock of the CPU bus and the clock of the peripheral bus are asynchronous, unexpected illegal data may be passed if an access to the CPU conflicts with an access to the peripheral hardware.
When accessing the peripheral hardware that may cause a conflict, therefore, the CPU repeatedly executes processing, until the correct data is passed.
As a result, the CPU does not start the next instruction processing but waits. If this happens, the number of execution clocks of an instruction increases by the number of wait clocks (for the number of wait clocks, see **Table 29-1**). This must be noted when real-time processing is performed.

29.2 Peripheral Hardware That Generates Wait

Table 29-1 shows the registers that generate a wait request when an access from the CPU is made, and the number of wait clocks of the CPU. Table 29-2 shows RAM accesses that generate a wait request, and the number of wait clocks of the CPU.

Table 29-1: Registers that Generate Wait and Number of CPU Wait Clocks (1/2)

<table>
<tr><th>Peripheral Hardware</th><th>Register</th><th>Access</th><th>Number of Wait Clocks</th></tr>
<tr><td>Watchdog timer</td><td>WDTM</td><td>Write</td><td>3 clocks (fixed)</td></tr>
<tr><td>Serial interface UART6</td><td>ASIS6</td><td>Read</td><td>1 clock (fixed)</td></tr>
<tr><td rowspan="6">A/D converter</td><td>ADM</td><td>Write</td><td rowspan="4">1 to 5 clocksNote 1 (when ADM.5 flag = “1”)
1 to 9 clocksNote 1 (when ADM.5 flag = “0”)</td></tr>
<tr><td>ADS</td><td>Write</td></tr>
<tr><td>PFM</td><td>Write</td></tr>
<tr><td>PFT</td><td>Write</td></tr>
<tr><td>ADCR</td><td>Read</td><td>1 to 5 clocks (when ADM.5 flag = “1”)
1 to 9 clocks (when ADM.5 flag = “0”)</td></tr>
<tr><td colspan="3"><Calculating maximum number of wait clocks>
{(1/f_{MACRO}) × 2/(1/f_{CPU})} + 1
• The result after the decimal point is truncated if it is less than t_{CPUL} after it has been multiplied by (1/f_{CPU}), and is rounded up if it exceeds t_{CPUL}.
f_{MACRO}: Macro operating frequency
(when bit 5 (FR2) of ADM = “1”: f_X/2, when bit 5 (FR2) of ADM = “0”: $f_X/2^2$)
f_{CPU}: CPU clock frequency
t_{CPUL}: Low-level width of CPU clock</td></tr>
<tr><td rowspan="5">CAN Controller</td><td>TEC, REC</td><td>Read</td><td>1 (fixed)</td></tr>
<tr><td colspan="3"><Calculating maximum number of wait clocks>
{(1/f_{MACRO}) × (1/f_{CPU})} + 1, round decimals to the next whole number</td></tr>
<tr><td>All message buffers</td><td>Read/Write</td><td>1 (fixed)</td></tr>
<tr><td colspan="3"><Calculating maximum number of wait clocks>
{(1/f_{MACRO}) × 3 / (1/f_{CPU})} + 1
• The result after the decimal point is truncated if it is less than t_{CPUL} after it has been multiplied by (1/f_{CPU}), and is rounded up if it exceeds t_{CPUL}.
f_{MACRO}: Macro operating frequency
(when bit 5 (FR2) of ADM = “1”: f_X/2, when bit 5 (FR2) of ADM = “0”: $f_X/2^2$)
f_{CPU}: CPU clock frequency
t_{CPUL}: Low-level width of CPU clock</td></tr>
</table>

Table 29-1: Registers that Generate Wait and Number of CPU Wait Clocks (2/2)

Peripheral Hardware	Register	Access	Number of Wait Clocks
TmPn (n = 0, 1, 2)	TPnCNT	Read	1 (fixed)
	<Calculating maximum number of wait clocks> $\{(1/f_{MACRO}) \times 3 / (1/f_{CPU})\} + 1$ • The result after the decimal point is truncated if it is less than t_{CPUL} after it has been multiplied by $(1/f_{CPU})$, and is rounded up if it exceeds t_{CPUL}. f_{MACRO}: Macro operating frequency (when bit 5 (FR2) of ADM = "1": $f_X/2$, when bit 5 (FR2) of ADM = "0": $f_X/2^2$) f_{CPU}: CPU clock frequency t_{CPUL}: Low-level width of CPU clock		
	TPnCCR0**Note 2**, TPnCCR1**Note 2**	Read	1 (fixed)
	<Calculating maximum number of wait clocks> $\{(1/f_{MACRO}) \times 3 / (1/f_{CPU})\} + 1$ • The result after the decimal point is truncated if it is less than t_{CPUL} after it has been multiplied by $(1/f_{CPU})$, and is rounded up if it exceeds t_{CPUL}. f_{MACRO}: Macro operating frequency (when bit 5 (FR2) of ADM = "1": $f_X/2$, when bit 5 (FR2) of ADM = "0": $f_X/2^2$) f_{CPU}: CPU clock frequency t_{CPUL}: Low-level width of CPU clock		
	TPnCCR0**Note 3**, TPnCCR1**Note 3**	Write	0 (fixed)
	<Calculating maximum number of wait clocks> $\{(1/f_{MACRO}) \times 5 / (1/f_{CPU})\} + 1$ • The result after the decimal point is truncated if it is less than t_{CPUL} after it has been multiplied by $(1/f_{CPU})$, and is rounded up if it exceeds t_{CPUL}. f_{MACRO}: Macro operating frequency (when bit 5 (FR2) of ADM = "1": $f_X/2$, when bit 5 (FR2) of ADM = "0": $f_X/2^2$) f_{CPU}: CPU clock frequency t_{CPUL}: Low-level width of CPU clock		

Notes: 1. No wait cycle is generated for the CPU if the number of wait clocks calculated by the above expression is 1.

2. on Capture mode

3. on Compare mode

Caution: When the CPU is operating on the subsystem clock and the X1 input clock is stopped (MCC = 1), do not access the registers listed above using an access method in which a wait request is issued.

Remark: The clock is the CPU clock (f_{CPU}).

29.3 Example of Wait Occurrence

<1> Watchdog timer

<On execution of MOV WDTM, A>
Number of execution clocks: 8
(5 clocks when data is written to a register that does not issue a wait (MOV sfr, A).)
<On execution of MOV WDTM, #byte>
Number of execution clocks: 10
(7 clocks when data is written to a register that does not issue a wait (MOV sfr, #byte).)

<2> Serial interface UART6

<On execution of MOV A, ASIS6>
Number of execution clocks: 6
(5 clocks when data is read from a register that does not issue a wait (MOV A, sfr).)

<3> A/D converter

<On execution of MOV ADM, A; MOV ADS, A; or MOV A, ADCR>

- When f_X = 10 MHz, t_{CPUL} = 50 ns

Table 29-2: Number of Wait Clocks and Number of Execution Clocks on Occurrence of Wait (A/D Converter)

Value of Bit 5 (FR2) of ADM Register	f_{CPU}	Number of Wait Clocks	Number of Execution Clocks
0	f_X	9 clocks	14 clocks
	$f_X/2$	5 clocks	10 clocks
	$f_X/2^2$	3 clocks	8 clocks
	$f_X/2^3$	2 clocks	7 clocks
	$f_X/2^4$	0 clocks (1 clock**Note**)	5 clocks (6 clocks**Note**)
1	f_X	5 clocks	10 clocks
	$f_X/2$	3 clocks	8 clocks
	$f_X/2^2$	2 clocks	7 clocks
	$f_X/2^3$	0 clocks (1 clock**Note**)	5 clocks (6 clocks**Note**)
	$f_X/2^4$	0 clocks (1 clock**Note**)	5 clocks (6 clocks**Note**)

Note: On execution of MOV A, ADCR

Remark: The clock is the CPU clock (f_{CPU}).
f_X: X1 input clock frequency
t_{CPUL}: Low-level width of CPU clock

Appendix A Development Tools

The following development tools are available for the development of systems that employ the µPD780822B Subseries.
Figure A-1 shows the development tool configuration.

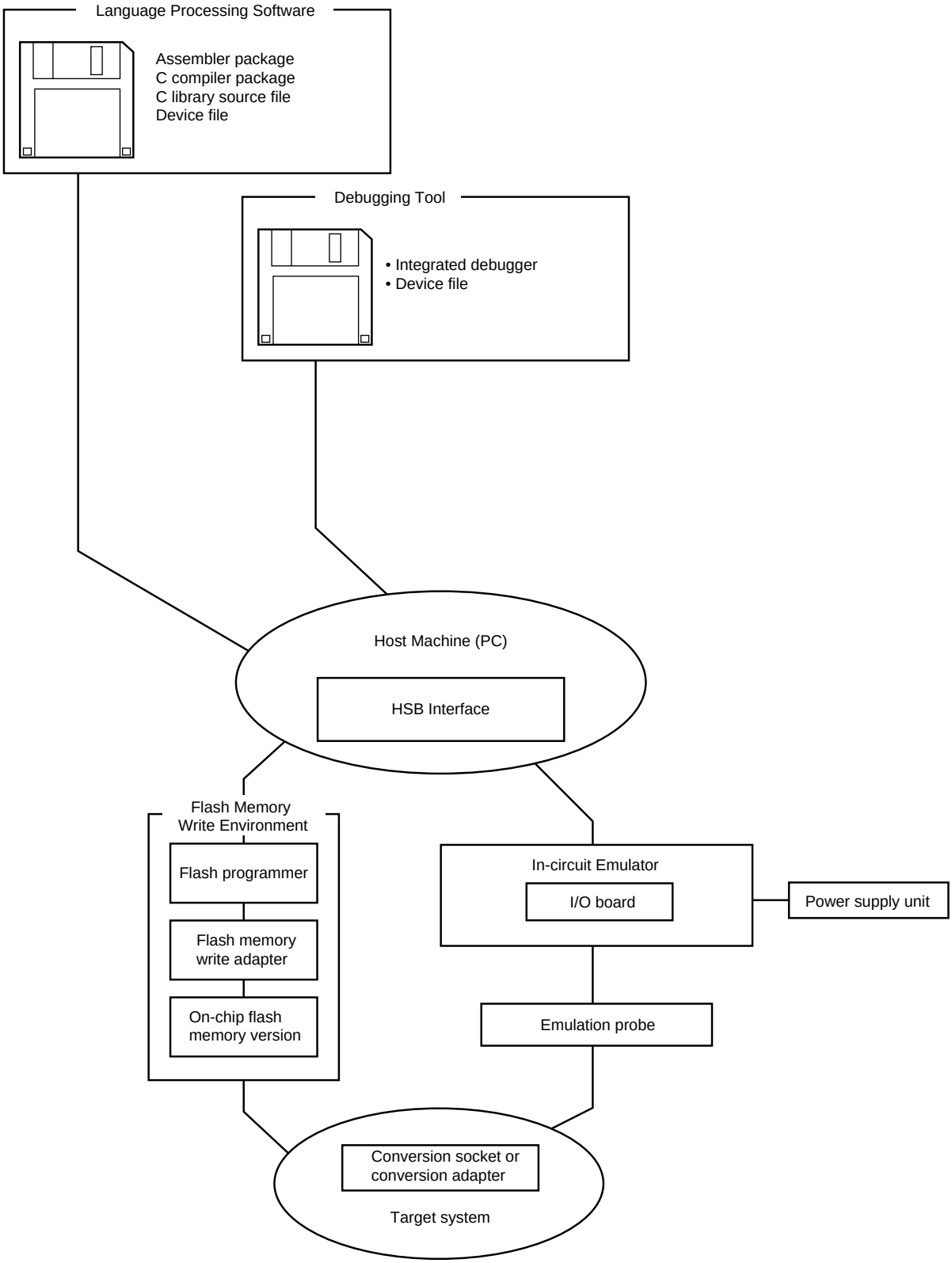
- Support for PC98-NX series

Unless otherwise specified, products compatible with IBM PC/ATTM computers are compatible with PC98-NX series computers. When using PC98-NX series computers, refer to the explanation for IBM PC/AT computers.

- Windows (Unless otherwise specified, “Windows” means the following OS).
- Windows 95/98
- Windows NT Version 4.0
- Windows 2000
- Windows XP

Figure A-1: Development Tool Configuration

(a) When using the in-circuit emulator QB-780822



Remark: Items in broken line boxes differ according to the development environment. See A.3.1 Hardware.

A.1 Language Processing Software

IAR Software

EW78000 Embedded Workbench	C-Compiler, Assembler, and higher package for 78K0 Series. C-CPY Debugger

A.2 Flash Memory Writing Tools

FlashMASTER Flashpro III (part number: FL-PR3, PG-FP3) Flashpro IV (part number: PG-FP4) Flash Programmer	Flash programmer dedicated to microcontrollers with on-chip flash memory.
FA-100GC-8EU-A FA-100GF-3BA	Flash memory writing adapter

A.3 Debugging Tools

A.3.1 Hardware

(1) When using the In-Circuit Emulator QB-780822

QB-780822 In-circuit Emulator	The in-circuit emulator serves to debug hardware and software when developing application systems using a µPD780822B Subseries product. It corresponds to integrated debugger (ID78K0-QB). This emulator should be used in combination with power supply unit, emulation probe, and interface adapter which is required to connect this emulator to the host machine.
EB-Power FW 75550/15 Power Supply Unit	This adapter is used for supplying power from a receptable of 100 V to 240 V AC
QB-144-EP-01S Emulation Probe	This probe is used to connect the in-circuit emulator to a target system and is designed for use with 100-pin plastic QFP.
QB-100GC-EA-02T QB-100GC-YQ-01T QB-100GC-NQ-01T QB-100GC-HQ-01T QB-100GC-YS-01T QB-100GC-SA-01T QB-100GF-EA-02T QB-100GF-YQ-01T QB-100GF-NQ-01T QB-100GF-HQ-01T QB-100GF-YS-01T QB-100GF-SA-01T	These conversion adapters connect the QB-144-EP-01S to a target system board designed for a 100-pin plastic QFP.

(2) Socket Details

QB-100GC-EA-02T QB-100GF-EA-02T	Exchange adapter
QB-100GC-NQ-01T QB-100GF-NQ-01T	Target connector
QB-100GC-YQ-01T QB-100GF-YQ-01T	ICE connector
QB-100GC-HQ-01T QB-100GF-HQ-01T	Mount adapter
QB-100GC-YS-01T QB-100GF-YS-01T	Space adapter
QB-144CA-01	Check pin adapter

A.3.2 Software

ID78K0-QB Integrated Debugger (supporting In-Circuit Emulator QB-780822)	This debugger is a control program to debug 78K/0 Series microcontrollers. It adopts a graphical user interface, which is equivalent visually and operationally to Windows or OSF/Motif™. It also has an enhanced debugging function for C language programs, and thus trace results can be displayed on screen in C-language level by using the windows integration function which links a trace result with its source program, disassembled display, and memory display. In addition, by incorporating function modules such as task debugger and system performance analyzer, the efficiency of debugging programs, which run on real-time OSs can be improved. It should be used in combination with the optional device file.

Appendix B Index

Numerics
16-bit timer mode control register (TMC2) 159
16-bit timer register (TM2) 157
8-bit timer mode control register 50 (TMC50) 218
8-bit timer mode control register 51 (TMC51) 219
8-bit timer registers 50 and 51 (TM50, TM51) 215

A
A/D conversion result register (ADCR) 274
A/D converter mode register (ADM) 271
Analog input channel specification register (ADS) 273
Asynchronous serial interface control register 6 (ASICL6) 326
Asynchronous serial interface operation mode register 6 (ASIM6) 320
Asynchronous serial interface reception error status register 6 (ASIS6) 322
Asynchronous serial interface transmission status register 6 (ASIF6) 323

B
Baud rate generator control register 6 (BRGC6) 325
Bit Rate Prescaler Register (BRPRS) 405

C
CAN control register (CANC) 395
CAN Receive Error Counter (REC) 402
CAN Transmit Error Counter (TEC) 402
Capture pulse control register (CRC2) 160
Capture register 20 (CR20) 157
Capture register 21 (CR21) 158
Capture register 22 (CR22) 158
Clock monitor mode register (CLM) 518
Clock output selection register (CKS) 262
Clock selection register 6 (CKSR6) 324
Compare control register (MCMPCn) 462
Compare control register n (MCMPCn) 460
Compare register 50 and 51 (CR50, CR51) 215
Compare registers nm (MCMPnm) 459

D
DCAN control register (DCANC) 394
DCAN Error Status Register (CANES) 399

E
External interrupt falling edge enable register (EGN) 479
External interrupt rising edge enable register (EGP) 479

F
Free running up counter (SMCNT) 459

G
General registers 74

I
Input switch control register (ISC) 132, 158, 183, 328
Internal memory size switching register (IMS) 526
Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H) 477

L
LCD display control register (LCDC) 434
LCD display mode register (LCDM) 433

M
Main clock mode register (MCM) 128
Main OSC control register (MOC) 129
Mask control register (MASKC) 414
Mask Identifier Control Register (MCON) 392
Message Count Register (MCNT) 403
Meter controller/driver clock register (SMSWI) 465

O
Option Bytes (Internal low-speed oscillator) 548
Oscillation stabilization time counter status register (OSTC) 130, 495
Oscillation stabilization time select register (OSTS) 131, 496

P
Port function register (PF7 to PF11) 119
Port function register 11 (PF11) 221
Port mode control register (PMC) 463
Port mode register 1 (PM1) 220, 328
Port mode register 11 (PM11) 220
Port mode register 3 (PM3) 264
Port mode registers (PM0, PM1, PM3, PM7 to PM12, PM14 and PM15) 116
Port mode registers 10 and 11 (PM10, PM11) 302
Power-fail comparison mode register (PFM) 275
Power-fail comparison threshold register (PFT) 275
Prescaler mode register (PRM2) 161
Priority specify flag registers (PR0L, PR0H, PR1L, PR1H) 478
Program counter (PC) 71
Program status word (PSW) 71, 480
Pull-up resistor option register (PU0, PU1, PU3, and PU7 to PU12) 118

R
Receive buffer register 6 (RXB6) 319
Receive Identifier (IDREC0 to IDREC4) 388
Receive message register (RMES) 413
Receive shift register 6 (RXS6) 319
Redefinition control register (REDEF) 417
Reset Control Flag Register (RESF) 515

S
Serial clock selection register 1n (CSIC1n) 299
Serial I/O shift register 1n (SIO1n) 295
Serial operation mode register 1n (CSIM1n) 296
Sound generator amplitude register (SGAM) 451
Sound generator buzzer control register (SGBR) 451
Sound generator control register (SGCR) 449
Special function register (SFR) 75, 89
Stack pointer (SP) 73
Synchronization Control Registers (SYNC0 and SYNC1) 407

T
Timer clock select register 50 (TCL50) 216
Timer clock select register 51 (TCL51) 217
Timer mode control register (MCNTC) 461
TMPn capture/compare register 0 (TPnCCR0) 174

TMPn capture/compare register 1 (TPnCCR1) 175
TMPn control register 0 (TPnCTL0) 177
TMPn counter read buffer register (TPnCNT) 176
TMPn I/O control register 0 (TPnIOC0) 180
TMPn I/O control register 1 (TPnIOC1) 181
TMPn option register 0 (TPnOPT0) 182
TMPn timer control register 1 (TPnCTL1) 179
Transmit buffer register 1n (SOTB1n) 295
Transmit buffer register 6 (TXB6) 319
Transmit control register (TCR) 411
Transmit Identifier (IDTX0 to IDTX4) 381
Transmit shift register 6 (TXS6) 319

W

Watch Timer Mode Register (WTM) 238
Watchdog timer mode register (WDTM) 249

Appendix C Revision History

The following shows the revision history up to present. Application portions signifies the chapter of each edition.

(1/2)

Edition No.	Major items revised	Revised Sections
EE2V0	Section 12.7 “Cautions on Emulation performed with IE-78K0-NS-A Emulator and IE-780822-NS-EM1 Emulation Board” updated	12.7, p.287
	Remark 2 in “Figure 7-20: Timing of Basic Operation in PWM Mode” modified	7.5.4, p.198, p.199
	Caution in “Figure 9-4: Operation Timing of Watch Timer/Interval Timer” modified	9.3.2, p.242
	Section “9.4 Cautions for Watch Timer” added	9.4, p.243
	Figure 9-5: Example of Generation of Watch Timer Interrupt Request (INTWT) (When Interrupt Period = 0.5 s)” added	9.4, p.243
	Figure 14-10: “Asynchronous Serial Interface Control Register 6 (ASICL6) Format” modified	14.3.6, p.324
	Section 14.4.2 (2) (h) “SBF transmission” changed, Figure 14-23 “Example of Setting Procedure of SBF Transmission (Flowchart)” deleted	14.4.2 (2) (h), p.341

(2/2)

Edition No.	Major items revised	Revised Sections