NEC

Preliminary User's Manual

μPD780948, μPD78F0948
μPD780949, μPD78F0949

8-bit Single-Chip Microcontroller

Hardware

Document No. U12670EE1V0UM00 (2 nd edition)
Date Published April 1998

FIP is a trademark of NEC Corporation
EEPROM and IEBus are trademarks of NEC Corporation.
MS-DOS and MS-Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
PC/AT and PC DOS are trademarks of IBM Corp.
IBM-DOS, PC/AT and PC DOS are trademarks of International Business Machines Corporation.
HP9000 Series 300, HP9000 Series 700, and HP-UX are trademarks of Hewlett-Packard Company.
SPARCstation is a trademark of SPARC International, Inc.
Sun OS is a trademark of Sun Microsystems, Inc.
Ethernet is a trademark of Xerox Corp.
NEWS and NEWS-OS are trademarks of Sony Corporation.
OSF/Motif is a trade mark of OpenSoftware Foundation, Inc..
TRON is an abbreviation of The Realtime Operating system Nucleus.
ITRON is an abbreviation of Industrial TRON.

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

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

M4 94.11

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 800-366-9782
Fax: 800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.l.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Sao Paulo-SP, Brasil
Tel: 011-889-1680
Fax: 011-889-1689

Introduction

Readers

This manual has been prepared for user engineers who want to understand the functions of the μPD780949 subseries and design and develop its application systems and programs.

μPD780949 Subseries: μPD780949, μPD78F0949, μPD780948, μPD78F0948.

Purpose

This manual is intended for users to understand the functions described in the Organization below.

Organization

The μPD780949 subseries manual is separated into two parts: this manual and the instruction edition (common to the 78K/0 series).

μPD780949 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 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, already defined in the header file named sfrbit.h.
- To make sure the details of the registers when you know the register name.
  → Refer to **Appendix C.**

Chapter Organization

This manual devides the descriptions for the subseries into different chapters as shown below. Read only the chapters related to the device you use.

Chapter		μPD780948 Subseries	μPD78F0948 Subseries	μPD780949 Subseries	μPD78F0949 Subseries
Chapter 1	Outline (μPD780949 Subseries)	O	O	O	O
Chapter 2	Pin Function (μPD780949 Subseries)	O	O	O	O
Chapter 3	CPU Architecture	O	O	O	O
Chapter 4	EEPROM	—	—	O	O
Chapter 5	Port Functions	O	O	O	O
Chapter 6	Clock Generator	O	O	O	O
Chapter 7	16-Bit Timer/Counter	O	O	O	O
Chapter 8	16-Bit Timer 2 TM2	O	O	O	O
Chapter 9	8-Bit Timer/Event Counters 50, 51	O	O	O	O
Chapter 10	Watch Timer	O	O	O	O
Chapter 11	Watchdog Timer	O	O	O	O
Chapter 12	Clock Output Control Circuit	O	O	O	O
Chapter 13	A/D-Converter	O	O	O	O
Chapter 14	Serial Interface Outline	O	O	O	O
Chapter 15	Serial Interface Channel 30	O	O	O	O
Chapter 16	Serial Interface Channel 31	O	O	O	O
Chapter 17	Serial Interface UART	O	O	O	O
Chapter 18	CAN Controller	O	O	O	O
Chapter 19	LCD Controller/Driver	O	O	O	O
Chapter 20	Sound Generator	O	O	O	O
Chapter 21	Interrupt Functions	O	O	O	O
Chapter 22	External Device Expansions	O	O	O	O
Chapter 23	Standby Function	O	O	O	O
Chapter 24	Reset Function	O	O	O	O
Chapter 25	μPD78F0949, μPD78F0948	O	O	O	O
Chapter 26	Instruction Set	O	O	O	O
Appendix A	Development Tools	O	O	O	O
Appendix B	Embedded Software	O	O	O	O
Appendix C	Register	O	O	O	O
Appendix D	Revision History	O	O	O	O

Related Documents

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

- **Related documents for μPD780949 subseries**

Document name	Document No.	
	Japanese	English
μPD780949 Preliminary Product Information	Planned	U12273E
μPD78F0949 Preliminary Product Information	Planned	U12375E
μPD780949 Subseries User's Manual	Planned	This manual
78K/0 Series User's Manual-Instruction	IEU-849	IEU-1372
78K/0 Series Instruction Table	U10903J	—
78K/0 Series Instruction Set	U10904J	—
μPD780949 Subseries Special Function Register Table	—	—

- **Related documents for development tool (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	—
IE-78001-R-A		U10057J	U10057E
IE-780948-SL-EM1 IE-780948-SL-EM4		Planned	U12412E
EP-100GF-SL		Planned	TEMPR-2093
SM78K0 System Simulator Windows™ Base	Reference	U10181J	U10181E
SM78K0 Series System Simulator	External part user open Interface	U10092J	U10092E
ID78K0 Integrated Debugger EWS Base	Reference	U11151J	—
ID78K0 Integrated Debugger PC Base	Reference	U11539J	U11539E
ID78K0 Integrated Debugger Windows Base	Guide	U11649J	U11649E

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

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

- **Other Documents**

Document name	Document No.	
	Japanese	English
IC Package Manual	C10943X	
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.

Contents

Chapter 1 Outline (μPD780949 Subseries) 26

1.1 Features 26
1.2 Application 26
1.3 Ordering Information 26
1.4 Pin Configuration (Top View) 27
1.5 78K/0 Series Development 29
1.6 Block Diagram 30
1.7 Overview of Functions 31
1.8 Mask Options 32
1.9 Differences between Flash and Mask ROM version 32

Chapter 2 Pin Function (μPD780949 Subseries) 34

2.1 Pin Function List 34
2.2 Non-Port Pins 36
2.3 Description of Pin Functions 38
2.3.1 P00 to P07 (Port 0) 38
2.3.2 P10 to P17 (Port 1) 39
2.3.3 P20 to P26 (Port 2) 39
2.3.4 P30 to P34 (Port 3) 40
2.3.5 P40 to P47 (Port 4) 40
2.3.6 P50 to P57 (Port 5) 41
2.3.7 P64, P65 and P67 (Port 6) 41
2.3.8 P70 to P72 (Port 7) 41
2.3.9 P120 to P127 (Port 12) 42
2.3.10 P130 to P136 (Port 13) 42
2.3.11 P140 to P147 (Port 14) 42
2.3.12 CTxD 42
2.3.13 CRxD 42
2.3.14 COM0 to COM3 43
2.3.15 V_{LC0} to V_{LC2} 43
2.3.16 AV_{DD} 43
2.3.17 $\overline{AV_{SS}}$ 43
2.3.18 RESET 43
2.3.19 X1 and X2 43
2.3.20 CL1 and CL2 43
2.3.21 V_{DD} 43
2.3.22 V_{SS} 43
2.3.23 V_{PP} (μPD78F0948, μPD78F0949 only) 43
2.3.24 IC (Mask ROM version only) 44
2.4 Pin I/O Circuits and Recommended Connection of Unused Pins 45

Chapter 3 CPU Architecture 52
3.1 Memory Space 52
3.1.1 Internal program memory space 54
3.1.2 Internal data memory space 56
3.1.3 Special function register (SFR) area 56
3.1.4 External memory space 56
3.1.5 Data memory addressing 57
3.2 Processor Registers 59
3.2.1 Control registers 59
3.2.2 General registers 62
3.2.3 Special function register (SFR) 63
3.3 Instruction Address Addressing 68
3.3.1 Relative addressing 68
3.3.2 Immediate addressing 69
3.3.3 Table indirect addressing 70
3.3.4 Register addressing 71
3.4 Operand Address Adressing 72
3.4.1 Implied addressing 72
3.4.2 Register addressing 73
3.4.3 Direct addressing 74
3.4.4 Short direct addressing 75
3.4.5 Special function register (SFR) addressing 76
3.4.6 Register indirect addressing 77
3.4.7 Based addressing 78
3.4.8 Based indexed addressing 79
3.4.9 Stack addressing 79

Chapter 4 EEPROM 81

4.1 EEPROM Functions 81
4.2 EEPROM Configuration 82
4.3 EEPROM Control Register 83
4.4 EEPROM Reading 84
4.5 EEPROM Writing 85
4.6 EEPROM Control-Related Interrupt 85

Chapter 5 Port Functions 87

5.1 Port Functions 87
5.2 Port Configuration 90
5.2.1 Port 0 90
5.2.2 Port 1 92
5.2.3 Port 2 (μPD780949 Subseries) 93
5.2.4 Port 3 94
5.2.5 Port 4 95
5.2.6 Port 5 96
5.2.7 Port 6 97
5.2.8 Port 7 98
5.2.9 Port 12 99
5.2.10 Port 13 100
5.2.11 Port 14 101

5.3 Port Function Control Registers 102
5.4 Port Function Operations 107
5.4.1 Writing to input/output port 107
5.4.2 Reading to input/output port 107

Chapter 6 Clock Generator 109

6.1 Clock Generator Functions 109
6.2 Clock Generator Configuration 110
6.3 Clock Generator Control Register 111
6.4 System Clock Oscillator 112
6.4.1 Main system clock oscillator 112
6.4.2 Subsystem clock oscillator 113
6.4.3 When no subsystem clocks are used 115
6.5 Clock Generator Operations 116
6.5.1 Main system clock operations 117
6.5.2 Subsystem clock operations 118
6.6 Changing System Clock and CPU Clock Settings 119
6.6.1 Time required for switchover between system clock and CPU clock 119
6.6.2 System clock and CPU clock switching procedure 120

Chapter 7 16-Bit Timer/Event Counter 122

7.1 16-Bit Timer/Event Counter Function 122
7.2 16-Bit Timer/Event Counter Configuration 123
7.3 16-Bit Timer/Event Counter Control Register 127
7.4 16-Bit Timer/Event Counter Operations 134
7.4.1 Operation as interval timer (16 bits) 134
7.4.2 PPG output operation 136
7.4.3 Pulse width measurement 137
7.4.4 Operation as external event counter 144
7.4.5 Operation to output square wave 146
7.4.6 Operation to output one-shot pulse 148
7.5 16-Bit Timer/Event Counter Operating Precautions 153

Chapter 8 16-Bit Timer 2 TM2 157

8.1 16-Bit Timer 2 Functions 157
8.2 16-Bit Timer 2 Configuration 158
8.3 16-Bit Timer 2 Control Registers 159
8.4 16-Bit Timer 2 Operations 162
8.4.1 Pulse width measurement operations 162
8.5 16-Bit Timer 2 Precautions 165

Chapter 9 8-Bit Timer/Event Counters 50 and 51 167

9.1 8-Bit Timer/Event Counters 5 and 6 Functions 167
9.2 8-Bit Timer/Event Counters 50 and 51 Configuration 170
9.3 8-Bit Timer/Event Counters 50 and 51 Control Registers 173
9.4 8-Bit Timer/Event Counters 50 and 51 Operations 178
9.4.1 Interval timer operations 178
9.4.2 External event counter operation 182
9.4.3 Square-wave output 182
9.4.4 PWM output operations 185
9.5 Cautions on 8-Bit Timer/Event Counters 50 and 51 188

Chapter 10 Watch Timer 191

10.1 Watch Timer Functions 191
10.2 Watch Timer Configuration 192
10.3 Watch Timer Mode Register (WTM) 193
10.4 Watch Timer Operations 194
10.4.1 Watch timer operation 194
10.4.2 Interval timer operation 194

Chapter 11 Watchdog Timer 197

11.1 Watchdog Timer Functions 197
11.2 Watchdog Timer Configuration 198
11.3 Watchdog Timer Control Registers 199
11.4 Watchdog Timer Operations 201
11.4.1 Watchdog timer operation 201
11.4.2 Interval timer operation 202

Chapter 12 Clock Output Control Circuit 204

12.1 Clock Output Control Circuit Functions 204
12.2 Clock Output Control Circuit Configuration 205
12.3 Clock Output Function Control Registers 206

Chapter 13 A/D Converter 209

13.1 A/D Converter Functions 209
13.2 A/D Converter Configuration 210
13.3 A/D Converter Control Registers 212
13.4 A/D Converter Operations 215
13.4.1 Basic operations of A/D converter 215
13.4.2 Input voltage and conversion result 217
13.4.3 A/D converter operation mode 218
13.5 A/D Converter Precautions 220
13.6 Cautions on Emulation 223
13.6.1 D/A converter mode register (DAM0) 223

Chapter 14 Serial Interface Outline 225

14.1 Serial Interface Outline 225

Chapter 15 Serial Interface Channel 30 227

15.1 Serial Interface Channel 30 Functions 227
15.2 Serial Interface Channel 30 Configuration 228
15.3 List of SFRs (Special Function Registers) 228
15.4 Serial Interface Control Registers 229
15.5 Serial Interface Operations 230
15.5.1 Operation stop mode 230
15.5.2 Three-wire serial I/O mode 231

Chapter 16 Serial Interface Channel 31 234

16.1 Serial Interface Channel 31 Functions 234
16.2 Serial Interface Channel 31 Configuration 235
16.3 List of SFRs (Special Function Registers) 235
16.4 Serial Interface Control Registers 236
16.5 Serial Interface Operations 237
16.5.1 Operation stop mode 237
15.5.2 Two-wire serial I/O mode 238

Chapter 17 Serial Interface UART 242

17.1 Serial Interface UART Functions 242
17.2 Serial Interface UART Configuration 243
17.3 List of SFRs (Special Function Registers) 244
17.4 Serial Interface Control Registers 244
17.5 Serial Interface Operations 248
17.5.1 Operation stop mode 248
17.5.2 Asynchronous serial interface (UART) mode 248
17.6 Standby Function 260

Chapter 18 CAN Controller 262
18.1 Protocol 263
18.1.1 Protocol mode function 263
18.1.2 Message format 263
18.1.3 Data frame/remote frame 264
18.1.4 Error frame 269
18.1.5 Overload frame 270

18.2 Function 271
18.2.1 Bus priority decision 271
18.2.2 Bit stuffing 271
18.2.3 Multi master 271
18.2.4 Multi cast 271
18.2.5 Sleep mode/Stop function 271
18.2.6 Error control function 272
18.2.7 Baud rate control function 275
18.2.8 State shift chart 278
18.3 Outline Description 281
18.4 Connection with target system 281
18.5 CAN Module Configuration 282
18.6 Operation 283
18.6.1 Special function register for CAN-module 283
18.7 Message and Buffer Configuation 284
18.8 Transmit Buffer Structure 285
18.9 Transmit Message 286
18.10 Transmit Structure 290
18.11 Receive Message 291
18.12 Mask Function 297
18.13 CAN 301
18.13.1 Status register 301
18.13.2 CAN error status register 304
18.14 Baudrate Generation 310
18.15 Function Control 315
18.15.1 Transmit control 315
18.15.2 Receive control 317
18.15.3 Mask control 318
18.15.4 Special functions 320
18.16 Interrupt Information 322
18.16.1 Interrupt vectors 322
18.16.2 Transmit interrupt 322
18.16.3 Receive interrupt 322
18.16.4 Error interrupt 322
18.17 Standby Function 323
18.17.1 CPU halt mode 323
18.17.2 CPU stop mode 323
18.17.3 DCAN sleep mode 323
18.17.4 DCAN stop mode 323
18.18 Functional Description by Flowcharts 324
18.18.1 Initialization 324
18.18.2 Prepare transmit 325
18.18.3 Abort transmit 326
18.18.4 Handling by the DCAN 327
18.18.5 Receive event oriented 328
18.18.6 Receive task oriented 329

Chapter 19 LCD Controller/Driver 331

19.1 LCD Controller/Driver Functions 331
19.2 LCD Controller/Driver Configuration 332
19.3 LCD Controller/Driver Control Registers 334
19.4 LCD Controller/Driver Settings 336
19.5 LCD Display Data Memory 337
19.6 Common Signals and Segment Signals 338
19.7 Supplying LCD Drive Voltage V_{LC0}, V_{LC1}, and V_{LC2} 340
19.8 Display Mode 342
19.8.1 4-time-division display example 342
19.9 Cautions on Emulation 345
19.9.1 LCD timer control register (LCDTM) 345

Chapter 20 Sound Generator 347

20.1 Sound Generator Function 347
20.2 Sound Generator Configuration 348
20.3 Sound Generator Control Registers 349
20.4 Sound Generator Operations 354
20.4.1 To output basic cycle signal SGOF (without amplitude) 354
20.4.2 To output basic cycle signal SGO (with amplitude) 354

Chapter 21 Interrupt Functions 356

21.1 Interrupt Function Types 356
21.2 Interrupt Sources and Configuration 357
21.3 Interrupt Function Control Registers 360
21.4 Interrupt Servicing Operations 366
21.4.1 Non-maskable interrupt request acknowledge operation 366
21.4.2 Maskable interrupt request acknowledge operation 369
21.4.3 Software interrupt request acknowledge operation 371
21.4.4 Multiple interrupt servicing 372
21.4.5 Interrupt request reserve 375

Chapter 22 External Device Expansion 377

22.1 External Device Expansion Functions 377
22.2 External Device Expansion Function Control Register 380
22.3 External Device Expansion Function Timing 383
22.4 Example of Connection with Memory 388

Chapter 23 Standby Function 390

23.1 Standby Function and Configuration 390
23.1.1 Standby function 390
23.1.2 Standby function control register 391
23.2 Standby Function Operations 392
23.2.1 HALT mode 392
23.2.2 STOP mode 395

Chapter 24 Reset Function 399

24.1 Reset Function 399

Chapter 25 μPD78F0948, μPD78F0949 405

25.1 Memory Size Switching Register (IMS) 406
25.2 Internal expansion RAM Size Switching Register 407
25.3 Flash Memory Programming 408
25.3.1 Selection of transmission method 408
25.3.2 Initialization of the programming mode 408
25.3.3 Flash memory programming function 409
25.3.4 Flashpro connection 409
25.3.5 Flash programming precautions 410

Chapter 26 Instruction Set 412

26.1 Legends Used in Operation List 413
26.1.1 Operands identifiers and description mode 413
26.1.2 Description of "operation" column 414
26.1.3 Description of "flag operation" column 414
26.2 Operation List 415
26.3 Instructions Listed by Addressing Type 423

Appendix A Development Tools 428

A.1 Language Processing Software 429
A.2 Flash Memory Writing Tools 430
A.3 Debugging Tools 430
A.3.1 Hardware 430
A.3.2 Software 431
A.4 OS for IBM PC 433
A.5 Development Environment when Using IE-78001-R-A 434

Appendix B Embedded Software 436

B.1 Real-Time OS (2/2) 437
B.2 Fuzzy Inference Development Support System 439

Appendix C Register Index 441

C.1 Register Index (In Alphabetical Order with Respect to Register Names) 441
C.2 Register Index (In Alphabetical Order with Respect to Register Symbol) 444

Appendix D Revision History 448

Contents of Figures

Figure No.	Title	Page
1-1	Pin Configuration	27
1-2	Block Diagram	30
2-1	Connection of IC Pins	44
2-2	Pin Input/Output Circuits	48
3-1	Memory Map (μPD780949)	52
3-2	Memory Map (μPD78F0949)	53
3-3	Data Memory Addressing (μPD780949)	57
3-4	Data Memory Addressing (μPD78F0949)	58
3-5	Program Counter Configuration	59
3-6	Program Status Word Configuration	59
3-7	Stack Pointer Configuration	61
3-8	Data to be Saved to Stack Memory	61
3-9	Data to be Reset to Stack Memory	61
3-10	General Register Configuration	62
3-11	Relative Addressing	68
3-12	Immediate Addressing	69
3-13	Table Indirect Addressing	70
3-14	Register Addressing	71
3-15	Register Addressing	73
3-16	Short Direct Addressing	75
3-17	Special-Function Register (SFR) Addressing	76
3-18	Special-Function Register (SFR) Addressing	77
4-1	EEPROM Block Diagram	82
4-2	EEPROM Write Control Register (EEWC) Format	83
5-1	Port Types	87
5-2	P00 to P07 Configurations	91
5-3	P10 to P17 Configurations	92
5-4	P20 to P26 Configurations	93
5-5	P30 to P34 Configurations	94
5-6	P40 to P47 Configurations	95
5-7	P50 to P57 Configurations	96
5-8	P64, P65 and P67 Configurations	97
5-9	P70 to P77 Configurations	98
5-10	P120 to P127 Configurations	99
5-11	P130 to P137 Configurations	100
5-12	P140 to P147 Configurations	101
5-13	Port Mode Register Format	103
5-14	Pull-Up Resistor Option Register (PU0, PU4, PU7 and PU13) Format	104
5-15	Port Function Register (PF2, PF5, PF7, PF12 to PF14) Format	105
5-16	Memory Expansion Mode Register Format	106

Figure No.	Title	Page

6-1 Block Diagram of Clock Generator 110
6-2 Processor Clock Control Register Format 111
6-3 External Circuit of Main System Clock Oscillator 112
6-4 External Circuit of Subsystem Clock Oscillator 113
6-5 Examples of Oscillator with Bad Connection (3/3) 114
6-6 Main System Clock Stop Function (2/2) 117
6-7 System Clock and CPU Clock Switching 120

7-1 Block Diagram of 16-Bit Timer/Event Counter (TM0) 123
7-2 Format of 16-Bit Timer Mode Control Register (TMC0) 128
7-3 Format of Capture/Compare Control Register 0 (CRC0) 130
7-4 Format of 16-Bit Timer Output Control Register (TOC0) 131
7-5 Format of Prescaler Mode Register 0 (PRM0) 132
7-6 Port Mode Register 0 (PM0) Format 133
7-7 Control Register Settings When Timer 0 Operates as Interval Timer 134
7-8 Configuration of Interval Timer 135
7-9 Timing of Interval Timer Operation 135
7-10 Control Register Settings in PPG Output Operation 136
7-11 Control Register Settings for Pulse Width Measurement with Free Running Counter and One Capture Register 137
7-12 Configuration for Pulse Width Measurement with Free Running Counter 138
7-13 Timing of Pulse Width Measurement with Free Running Counter and One Capture Register (with both edges specified) 138
7-14 Control Register Settings for Measurement of Two Pulse Widths with Free Running Counter 139
7-15 CR01 Capture Operation with Rising Edge Specified 140
7-16 Timing of Pulse Width Measurement with Free Running Counter (with both edges specified) 140
7-17 Control Register Settings for Pulse Width Measurement with Free Running Counter and Two Capture Registers 141
7-18 Timing of Pulse Width Measurement with Free Running Counter and Two Capture Registers (with rising edge specified) 142
7-19 Control Register Settings for Pulse Width Measurement by Restarting 143
7-20 Timing of Pulse Width Measurement by Restarting (with rising edge specified) 144
7-21 Control Register Settings in External Event Counter Mode 145
7-22 Configuration of External Event Counter 145
7-23 Timing of External Event Counter Operation (with rising edge specified) 146
7-24 Set Contents of Control Registers in Square Wave Output Mode 147
7-25 Timing of Square Wave Output Operation 147
7-26 Control Register Settings for One-Shot Pulse Output with Software Trigger 149
7-27 Timing of One-Shot Pulse Output Operation with Software Trigger 150
7-28 Control Register Settings for One-Shot Pulse Output with External Trigger 151
7-29 Timing of One-Shot Pulse Output Operation with External Trigger (with rising edge specified) 152
7-30 Start Timing of 16-Bit Timer Register 153
7-31 Timing after Changing Compare Register during Timer Count Operation 153
7-32 Data Hold Timing of Capture Register 154
7-33 Operation Timing of OVF0 Flag 155

Figure No.	Title	Page
8-1	Timer 0 (TM0) Block Diagram	157
8-2	16-Bit Timer Mode Control Register (TMC2) Format	159
8-3	Capture Pulse Control Register (CRC2) Format	160
8-4	Prescaler Mode Register (PRM2) Format	161
8-5	Configuration Diagram for Pulse Width Measurement by Using the Free Running Counter	162
8-6	Timing of Pulse Width Measurement Operation by Using the Free Running Counter and One Capture Register (with Both Edges Specified)	162
8-7	CR2m Capture Operation with Rising Edge Specified	163
8-8	Timing of Pulse Width Measurement Operation by Free Running Counter (with Both Edges Specified)	164
8-9	16-Bit Timer Register Start Timing	165
8-10	Capture Register Data Retention Timing	165
9-1	8-Bit Timer/Event Counter 50 Block Diagram	170
9-2	8-Bit Timer/Event Counter 51 Block Diagram	171
9-3	Block Diagram of 8-Bit Timer/Event Counters 50 and 51 Output Control Circuit	172
9-4	Timer Clock Select Register 50 Format	173
9-5	Timer Clock Select Register 51 Format	174
9-6	8-Bit Timer Output Control Register Format	175
9-7	8-Bit Timer Output Control Register 51 Format	176
9-8	Port Mode Register 10 Format	177
9-9	8-Bit Timer Mode Control Register Settings for Interval Timer Operation	178
9-10	Interval Timer Operation Timings (3/3)	178
9-11	8-Bit Timer Mode Control Register Setting for External Event Counter Operation	182
9-12	External Event Counter Operation Timings (with Rising Edge Specified)	182
9-13	8-Bit Timer Mode Control Register Settings for Square-Wave Output Operation	183
9-14	Square-wave Output Operation Timing	183
9-15	8-Bit Timer Control Register Settings for PWM Output Operation	185
9-16	PWM Output Operation Timing (Active high setting)	186
9-17	PWM Output Operation Timings (CRn0 = 00H, active high setting)	186
9-18	PWM Output Operation Timings (CRn = FFH, active high setting)	187
9-19	PWM Output Operation Timings (CRn changing, active high setting)	187
9-20	8-bit Timer Registers 50 and 51 Start Timings	188
9-21	External Event Counter Operation Timings	188
9-22	Timings after Compare Register Change during Timer Count Operation	189
10-1	Block Diagram of Watch Timer	191
10-2	Watch Timer Mode Control Register (WTM) Format	193
10-3	Operation Timing of Watch Timer/Interval Timer	195
11-1	Watchdog Timer Block Diagram	198
11-2	Timer Clock Select Register 2 Format	199
11-3	Watchdog Timer Mode Register Format	200
12-1	Remote Controlled Output Application Example	204
12-2	Clock Output Control Circuit Block Diagram	205
12-3	Clock Output Selection Register Format	206
12-4	Port Mode Register 3 Format	207

Figure No.	Title	Page
13-1	A/D Converter Block Diagram	209
13-2	Power-Fail Detection Function Block Diagram	210
13-3	A/D Converter Mode Register (ADM1) Format	212
13-4	Analog Input Channel Specification Register (ADS1) Format	213
13-5	Power-Fail Compare Mode Register (PFM) Format	214
13-6	Power-fail compare threshold value register (PFT)	214
13-7	Basic Operation of 8-Bit A/D Converter	216
13-8	Relation between Analog Input Voltage and A/D Conversion Result	217
13-9	A/D Conversion	219
13-10	Example Method of Reducing Current Consumption in Standby Mode	220
13-11	Analog Input Pin Handling	221
13-12	A/D Conversion End Interrupt Request Generation Timing	222
13-13	D/A Converter Mode Register (DAM0) Format	223
15-1	Block Diagram of SIO30 Macro	227
15-2	Format of Serial Operation Mode Register 30 (CSIM30)	229
15-3	Format of Serial Operation Mode Register 30 (CSIM30)	230
15-4	Format of Serial Operation Mode Register 30 (CSIM30)	231
15-5	Timing of Three-wire Serial I/O Mode	232
16-1	Block Diagram of SIO3 Macro	234
16-2	Format of Serial Operation Mode Register 31 (CSIM31)	236
16-3	Format of Serial Operation Mode Register 31 (CSIM31)	237
16-4	Format of Serial Operation Mode Register 31 (CSIM31)	238
16-5	Timing of Three-wire Serial I/O Mode	239
16-6	2-Wire Mode Connection	240
17-1	Block Diagram of UART	242
17-2	Format of Asynchronous Serial Interface Mode Register (ASIM0)	245
17-3	Format of Asynchronous Serial Interface Status Register (ASIS0)	246
17-4	Format of Baud Rate Generator Control Register (BRGC0)	247
17-5	Register Settings	248
17-6	Asynchronous serial interface mode register (ASIM0)	249
17-7	Asynchronous serial interface status register (ASIS0)	250
17-8	Baud rate generator control register (BRGC0)	251
17-9	Error Tolerance (when k = 0), including Sampling Errors	254
17-10	Format of Transmit/Receive Data in Asynchronous Serial Interface	255
17-11	Timing of Asynchronous Serial Interface Transmit Completion Interrupt	257
17-12	Timing of Asynchronous Serial Interface Receive Completion Interrupt	258
17-13	Receive Error Timing	259
18-1	Data Frame	264
18-2	Remote Frame	264
18-3	Data Frame	265
18-4	Arbitration Field/Standard Format Mode	265
18-5	Arbitration Field/Expanded Format Mode	265
18-6	Control Field	266
18-7	Data Field	266

Figure No.	Title	Page
18-8	CRC Field	267
18-9	ACK Field	267
18-10	End of Frame	267
18-11	Interframe Space/Error Active	268
18-12	Interframe Space/Error Passive	268
18-13	Error Frame	269
18-14	Overload Frame	270
18-15	Nominal Bit Time (8 to 25 Time Quantum)	275
18-16	Adjusting Synchronization of the Data Bit	276
18-17	Bit Synchronization	277
18-18	Transmission State Shift Chart	278
18-19	Reception State Shift Chart	279
18-20	Error State Shift Chart	280
18-21	Structural Block Diagram	281
18-22	Connection to the CAN Bus	281
18-23	Transmit Message Definition Bits	287
18-24	Transmit Identifier	288
18-25	Transmit Data	289
18-26	Control Bits for Receive Identifier	292
18-27	Receive Status Bits	293
18-28	Receive Identifier	295
18-29	Receive Data	296
18-30	Identifier Compare with Mask	298
18-31	Control Bits for Mask Identifier	299
18-32	Mask Identifier	300
18-33	CAN Control Register	301
18-34	DCAN Support	302
18-35	Time Stamp Function	303
18-36	SOFOUT Toggle Function	303
18-37	Global Time System Function	303
18-38	CAN Error Status Register	304
18-39	Transmit Error Counter	307
18-40	Receive Error Counter	308
18-41	Message Count Register	309
18-42	Bit Rate Prescaler	310
18-43	Synchronization Control Register 0 and 1 (2/2)	311
18-44	Synchronization Control Register 1	313
18-45	Transmit Control Register	315
18-46	Receive Message Register	317
18-47	Mask Control Register	318
18-48	Redefinition Control Register	320
18-49	Initialization Flow Chart	324
18-50	Transmit Preparation	325
18-51	Transmit Abort, Software Flow	326
18-52	Handling of Semaphore Bits by DCAN-Module	327
18-53	Receive with Interrupt, Software Flow	328
18-54	Receive, Software Polling	329

Figure No. **Title** **Page**

19-1 LCD Controller/Driver Block Diagram 332
19-2 LCD Clock Select Circuit Block Diagram 333
19-3 LCD Display Mode Register (LCDM) Format 334
19-4 LCD Display Control Register (LCDC) Format 335
19-5 Relationship between LCD Display Data Memory Contents and Segment/Common Outputs L 337
19-6 Common Signal Waveform 339
19-7 Common Signal and Segment Signal Voltages and Phases 339
19-8 Example of Connection of LCD Drive Power Supply 341
19-9 4-Time-Division LCD Display Pattern and Electrode Connections 342
19-10 4-Time-Division LCD Panel Connection Example 343
19-11 4-Time-Division LCD Drive Waveform Examples (1/3 Bias Method) 344
19-12 LCD Timer Control Register (LCDTM) Format 345

20-1 Sound Generator Block Diagram 347
20-2 Concept of Each Signal 348
20-3 Sound Generator Control Register (SGCR) Format 350
20-4 Sound Generator Buzzer Control Register (SGBR) Format 352
20-5 Sound Generator Amplitude Register (SGAM) Format 353
20-6 Sound Generator Output Operation Timing 354
20-7 Sound Generator Output Operation Timing 354

21-1 Basic Configuration of Interrupt Function (2/2) 358
21-2 Interrupt Request Flag Register Format 361
21-3 Interrupt Mask Flag Register Format 362
21-4 Priority Specify Flag Register Format 363
21-5 Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register 364
21-6 Program Status Word Format 365
21-7 Flowchart from Non-Maskable Interrupt Generation to Acknowledge 367
21-8 Non-Maskable Interrupt Request Acknowledge Timing 367
21-9 Non-Maskable Interrupt Request Acknowledge Operation 368
21-10 Interrupt Request Acknowledge Processing Algorithm 370
21-11 Interrupt Request Acknowledge Timing (Minimum Time) 371
21-12 Interrupt Request Acknowledge Timing (Maximum Time) 371
21-13 Multiple Interrupt Example (2/2) 373
21-14 Interrupt Request Hold 375

22-1 Memory Map when Using External Device Expansion Function (2/2) 378
22-2 Memory Expansion Mode Register Format 380
22-3 Memory Expansion Wait Register Format 381
22-4 Memory Size Switching Register Format 382
22-5 Instruction Fetch from External Memory 384
22-6 External Memory Read Timing 385
22-7 External Memory Write Timing 386
22-8 External Memory Read Modify Write Timing 387
22-9 Connection Example of μPD780949 and Memory 388

Figure No.	Title	Page
23-1	Oscillation Stabilization Time Select Register Format	391
23-2	HALT Mode Clear upon Interrupt Generation	393
23-3	HALT Mode Release by RESET Input	394
23-4	STOP Mode Release by Interrupt Generation	396
23-5	Release by STOP Mode RESET Input	397
24-1	Block Diagram of Reset Function	399
24-2	Timing of Reset Input by RESET Input	400
24-3	Timing of Reset due to Watchdog Timer Overflow	400
24-4	Timing of Reset Input in STOP Mode by RESET Input	400
25-1	Memory Size Switching Register Format	406
25-2	Internal Expansion RAM Size Switching Register Format	407
25-3	Transmission Method Selection Format	408
25-4	Connection of Flashpro Using 3-Wire Serial I/O Method	409
25-5	Flashpro Connection Using UART Method	410
25-6	Flashpro Connection Using Pseudo 3-wire Serial I/O	410
A-1	Development Tool Configuration	428

Contents of Tables

Table No.	Title	Page
1-1	Mask Options of Mask ROM Versions	32
1-2	Differences between Flash and Mask ROM version	32
2-1	Pin Input/Output Types (2/2)	34
2-2	Non-Port Pins (2/2)	36
2-3	Types of Pin Input/Output Circuits (3/3)	45
3-1	Internal ROM Capacities	54
3-2	Vectored Interrupts	55
3-3	Special Function Register List (4/4)	64
3-4	Implied Addressing	72
3-5	Register Addressing	73
3-6	Direct Addressing	74
3-7	Short Direct Addressing	75
3-8	Special-Function Register (SFR) Addressing	76
3-9	Register Indirect Addressing	77
3-10	Based Addressing	78
3-11	Based Indexed Addressing	79
5-1	Pin Input/Output Types (2/2)	88
5-2	Port Configuration	90
6-1	Clock Generator Configuration	110
6-2	Maximum Time Required for CPU Clock Switchover	119
7-1	Configuration of 16-bit Timer/Event Counter (TM0)	123
7-2	Valid Edge of TI00 Pin and Valid Edge of Capture Trigger of Capture/Compare Register	125
7-3	Valid Edge of TI01 Pin and Valid Edge of Capture Trigger of Capture/Compare Register	125
8-1	Timer 2 Configuration	158
9-1	8-Bit Timer/Event Counter 50 Interval Times	168
9-2	8-Bit Timer/Event Counter 51 Interval Times	168
9-3	8-Bit Timer/Event Counter 50 Square-Wave Output Ranges	169
9-4	8-Bit Timer/Event Counter 50 Square-Wave Output Ranges	169
9-5	8-Bit Timer/Event Counters 50 and 51 Configurations	170
9-6	8-Bit Timer/Event Counters 50 Interval Times	181
9-7	8-Bit Timer/Event Counters 51 Interval Times	181
9-8	8-Bit Timer/Event Counters 50 Square-Wave Output Ranges	184
9-9	8-Bit Timer/Event Counters 51 Square-Wave Output Ranges	184
10-1	Interval Timer Interval Time	192
10-2	Watch Timer Configuration	192
10-3	Interval Timer Operation	194

Table No.	Title	Page
11-1	Watchdog Timer Inadvertent Program Overrun Detection Times	197
11-2	Interval Times	197
11-3	Watchdog Timer Configuration	198
11-4	Watchdog Timer Overrun Detection Time	201
11-5	Interval Timer Interval Time	202
12-1	Clock Output Control Circuit Configuration	207
13-1	A/D Converter Configuration	210
14-1	Differences between the Serial Interface Channels	225
15-1	Composition of SIO30	228
15-2	List of SFRs (Special Function Registers)	228
16-1	Composition of SIO31	235
16-2	List of SFRs (Special Function Registers)	235
17-1	Configuration of UART	243
17-2	List of SFRs (Special Function Registers)	244
17-3	Relation between 5-bit Counter's Source Clock and "n" Value	252
17-4	Relation between Main System Clock and Baud Rate	253
17-5	Causes of Receive Errors	259
18-1	Outline of the Function	262
18-2	Bit Number of the Identifier	265
18-3	RTR Setting	266
18-4	Mode Setting	266
18-5	Data Length Code Setting	266
18-6	Bit Length of the Intermission	268
18-7	Operation in the Error State	268
18-8	Definition of each Field	269
18-9	Definition of each Frame	270
18-10	Bus Priority Decision	271
18-11	Bit Stuffing	271
18-12	Error Types	272
18-13	Output Timing of the Error Frame	272
18-14	Types of Error State	273
18-15	Error Counter	274
18-16	Segment Name and Segment Length	275
18-17	CAN Configuration	282
18-18	SFR Definitions	283
18-19	SFR Bit Definitions	283
18-20	Message and Buffer Structure	284
18-21	Transmit Message Structure	286
18-22	Receive Message Structure	291
18-23	Mask Function	297
18-24	Possible Setup of the SOFOUT Function	302

Table No.	Title	Page
18-25	Transmission/Reception Flag	302
18-26	Possible Reactions of the DCAN	305
18-27	Mask Operation Buffers	319
18-28	Interrupt Sources	322
19-1	Maximum Number of Display Pixels	331
19-2	LCD Controller/Driver Configuration	332
19-3	COM Signals	338
19-4	LCD Drive Voltage	338
19-5	LCD Drive Voltage Supply	340
19-6	Selection and Non-Selection Voltages (COM0 to COM3)	342
20-1	Sound Generator Configuration	348
20-2	Maximum and Minimum Values of the Buzzer Output Frequency	352
21-1	Interrupt Source List	357
21-2	Various Flags Corresponding to Interrupt Request Sources	360
21-3	Times from Maskable Interrupt Request Generation to Interrupt Service	369
21-4	Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing	372
22-1	Pin Functions in External Memory Expansion Mode	377
22-2	State of Port 4 to Port 6 Pins in External Memory Expansion Mode	377
22-3	Values when the Memory Size Switching Register is Reset	382
23-1	HALT Mode Operating Status	392
23-2	Operation after HALT Mode Release	394
23-3	STOP Mode Operating Status	395
23-4	Operation after STOP Mode Release	397
24-1	Hardware Status after Reset (3/3)	401
25-1	Differences among μPD78F0948, μPD78F0949 and Mask ROM Versions	405
25-2	Values when the Memory Size Switching Register is Reset	406
25-3	Examples of internal Expansion RAM Size Switching Register Settings	407
25-4	Transmission Method List	408
25-5	Main Functions of Flash Memory Programming	407
26-1	Operand Identifiers and Description Methods	412

Chapter 1 Outline (μPD780949 Subseries)

1.1 Features

- Internal high capacity ROM and RAM

Item / Part Number	Program Memory (ROM)	Data Memory				Package
		Internal High-Speed RAM	LCD Display RAM	Internal Expansion RAM	EEPROM	
μPD780948	60K bytes	1024 bytes	40 bytes	992 bytes	—	100-pin plastic QFP (fine pitch)
μPD780949	60K bytes	1024 bytes	40 bytes	992 bytes	256 bytes	100-pin plastic QFP (fine pitch)
μPD78F0948	60K bytes	1024 bytes	40 bytes	992 bytes	—	100-pin plastic QFP (fine pitch)
μPD78F0949	60K bytes	1024 bytes	40 bytes	992 bytes	256 bytes	100-pin plastic QFP (fine pitch)

- External memory expansions space : 1K bytes
- Instruction execution time can be changed from high speed (0.25 μs) to ultra low speed
- I/O ports: 79 (N-ch open drain : 5)
- 8-bit resolution A/D converter : 8 channels
- Sound generator
- LCD-controller / driver
- CAN-Interface
- Serial interface : 3 channels
  - 2-wire mode : 1 channel
  - 3-wire mode : 1 channel
  - UART mode : 1 channel
- Timer : 6 channels
- Supply voltage : V_{DD} = 4.0 to 5.5 V

1.2 Application

Dashboard, climate controller, security unit etc.

1.3 Ordering Information

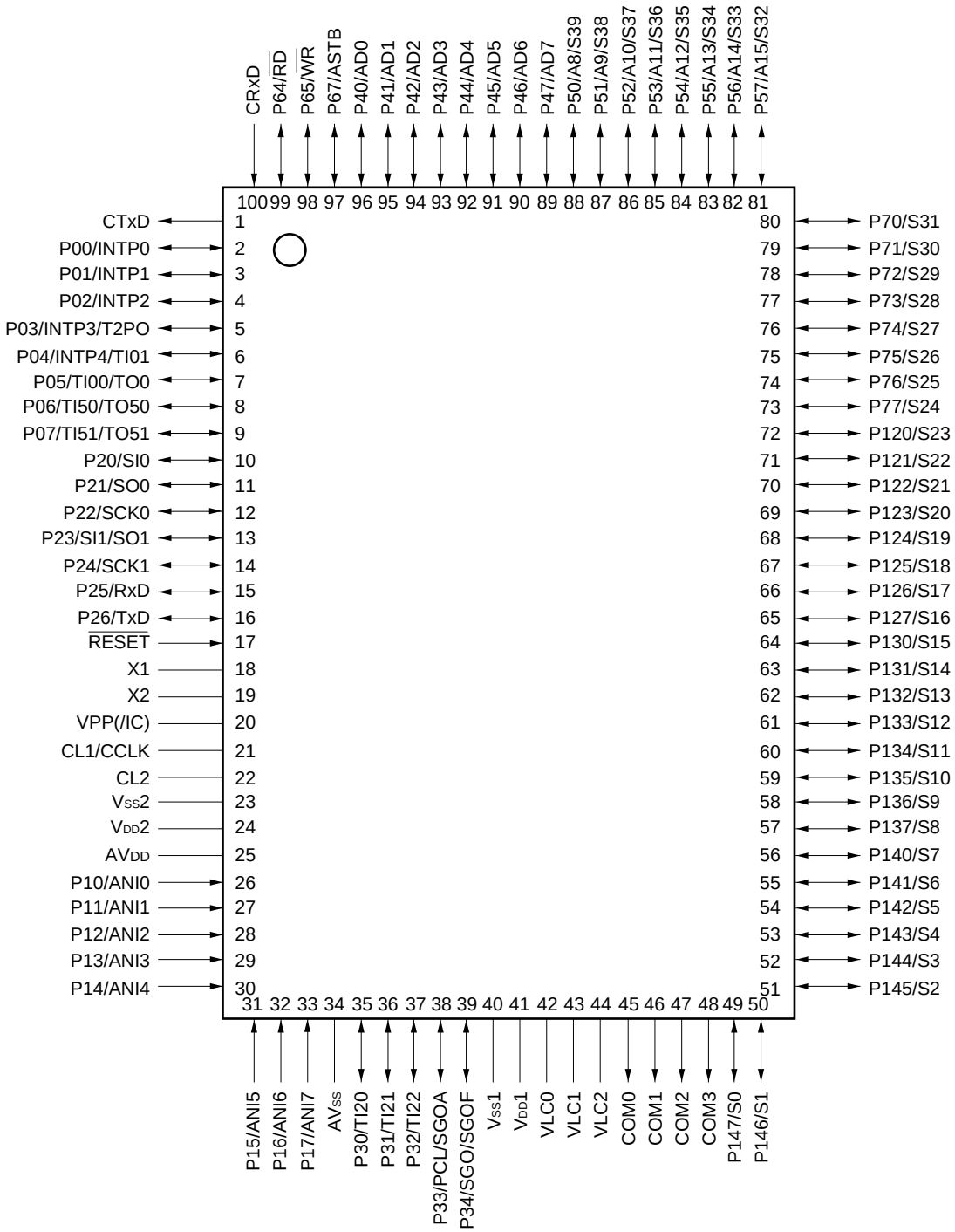
Part Number	Package
μPD780948GF(A)-3BA	100-pin plastic QFP (14 x 20 mm, resin thickness 2.7 mm)
μPD780949GF(A)-3BA	100-pin plastic QFP (14 x 20 mm, resin thickness 2.7 mm)
μPD780F948GF-3BA	100-pin plastic QFP (14 x 20 mm, resin thickness 2.7 mm)
μPD780F949GF-3BA	100-pin plastic QFP (14 x 20 mm, resin thickness 2.7 mm)

1.4 Pin Configuration (Top View)

⇑100-pin plastic QFP (14 x 20 mm)

μPD780948GF(A)-XXX-3BA
μPD780949GF(A)-XXX-3BA
μPD78F0948GF-XXX-3BA
μPD78F0949GF-XXX-3BA

Figure 1-1: Pin Configuration



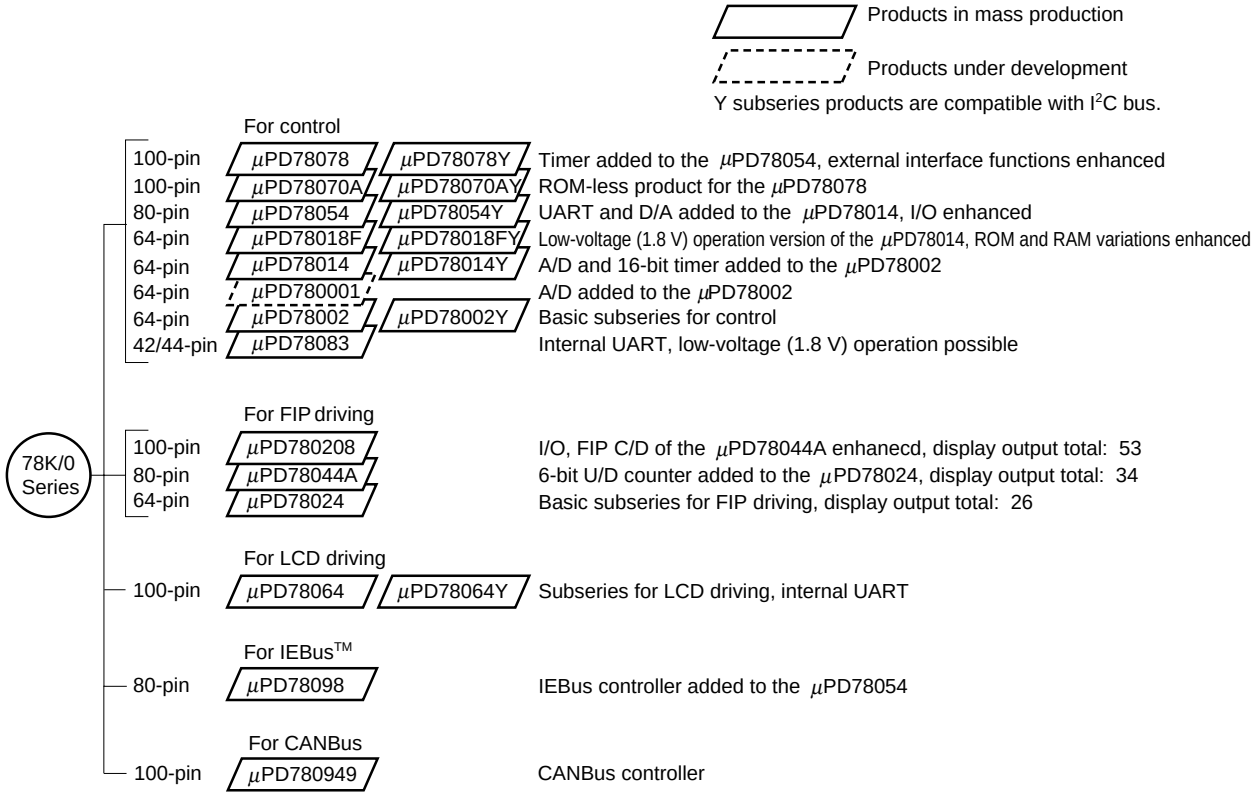
Cautions: 1. Connect IC (internally connected) pin directly to V_{SS}.
2. AV_{DD} pin should be connected to V_{DD}.
3. AV_{SS} pin should be connected to V_{SS}.

Pin Identifications

P00 to P07	: Port0	RxD	: Receive Data
P10 to P17	: Port1	TxD	: Transmit Data
P20 to P26	: Port2	SGO	: Sound Generator Output
P30 to P34	: Port3	SGOA	: Sound Generator Amplitude
P40 to P47	: Port4	SGOF	: Sound Generator Frequency
P50 to P57	: Port5	PCL	: Programmable Clock Output
P64, P65, P67	: Port6	AD0 to AD7	: Address / Data Bus
P70 to P77	: Port7	A8 to A15	: Address Bus
P120 to P127	: Port12	RD	: Read Strobe
P130 to P137	: Port13	WR	: Write Strobe
P140 to P147	: Port14	ASTB	: Address Strobe
INTP0 to INTP4	: Interrupt from Peripherals	S0 to S39	: Segment Output
TI00, TI01, TI50, TI51	: Timer Input	COM0 to COM3	: Common Output
TI20 to TI22	: Timer Input	X1, X2	: Crystal (Main System Clock)
TO0 , TO51, TO52	: Timer Output	CL1, CL2	: RC (Subsystem Clock)
T2PO	: Timer Output	RESET	: Reset
CRxD	: CAN Receive Data	ANI0 to ANI7	: Analog Input
CTxD	: CAN Transmit Data	AV_{SS}	: Analog Ground
CCLK	: CAN Clock	AV_{DD}	: Analog Reference Voltage
SI0	: Serial Input	V_{DD}	: Power Supply
SO0	: Serial Output	V_{PP}	: Programming Power supply
SIO1	: Serial Input / Output	V_{SS}	: Ground
SCK0, SCK1	: Serial Clock	IC	: Internally Connected

1.5 78K/0 Series Development

These products are a further development in the 78K/0 Series. The designations appearing inside the boxes are subseries names.

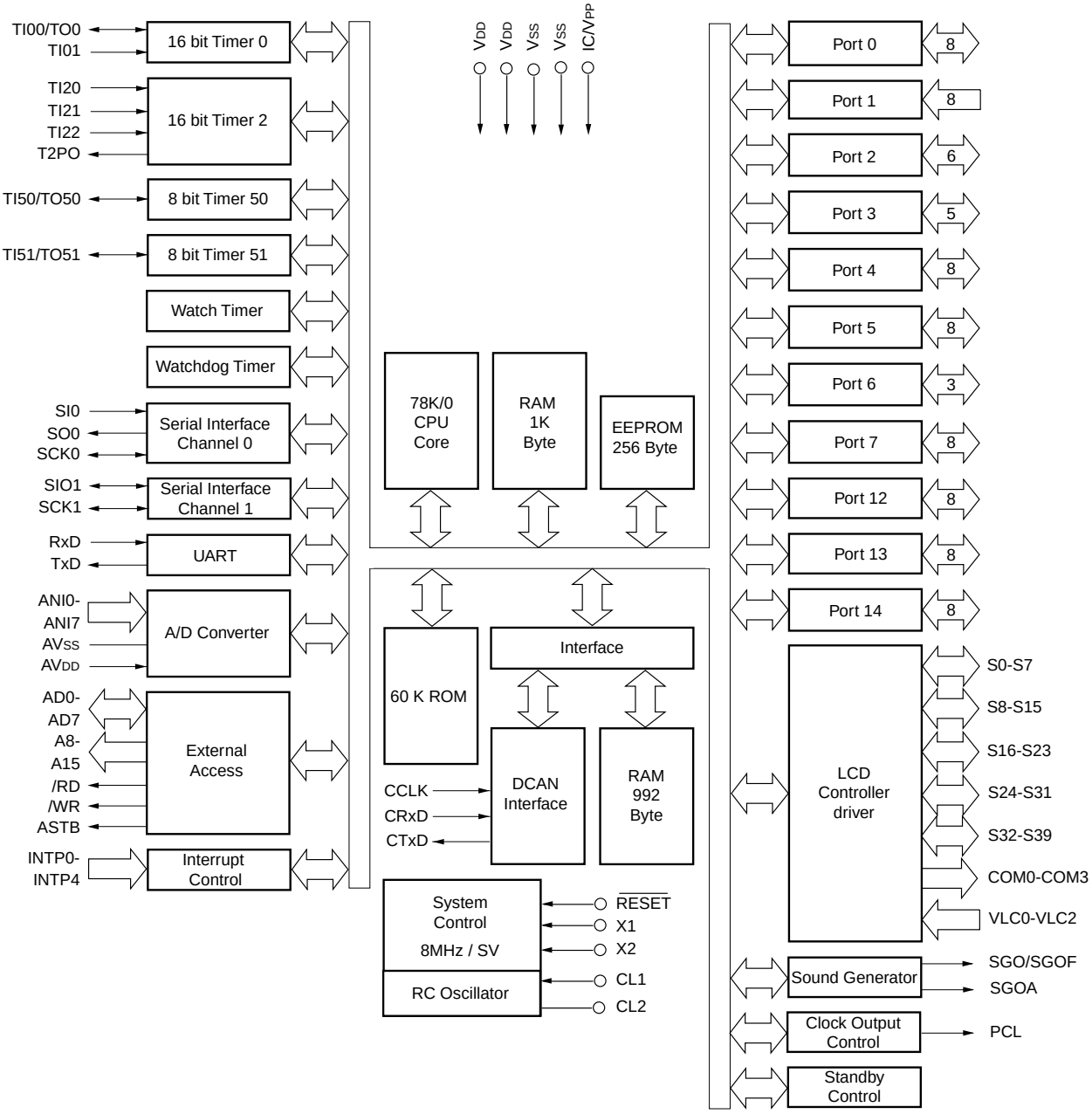


Major functional differences among the subseries

Subseries Name \ Function		ROM Capacity	Timer				8-bit A/D	8-bit D/A	Serial Interface	V_{DD} I/O	MIN. Value	External Expansion
			8-bit	16-bit	Watch	WDT						
For Control	μPD78078	32 K-60 K	4ch	1ch	1ch	1ch	8ch	2ch	3ch (UART: 1ch)	88	1.8 V	O
	μPD78070A	—								61	2.7 V	
	μPD78054	16 K-60 K	2ch							69	2.0 V	
	μPD78018F	8 K-60 K						—	2ch	53	1.8 V	
	μPD78014	8 K-32 K									2.7 V	
	μPD780001	8 K		—	—				1ch	39		—
	μPD78002	8 K-16 K			1ch		—			53		O
	μPD78083				—		8ch		1ch (UART: 1ch)	33	1.8 V	—
For FIP driving	μPD780208	32 K-60 K	2ch	1ch	1ch	1ch	8ch	—	2ch	74	2.7 V	—
	μPD78044A	16 K-40 K								68		
	μPD78024	24 K-32 K								54		
For LCD driving	μPD78064	16 K-32 K	2ch	1ch	1ch	1ch	8ch	—	2ch (UART: 1ch)	57	2.0 V	—
For IEBus	μPD78098	32 K-60 K	2ch	1ch	1ch	1ch	8ch	2ch	3ch (UART: 1ch)	69	2.7 V	O
For CANBus	μPD780949	60 K	2ch	2ch	1ch	1ch	8ch	—	3ch (UART: 1ch)	79	4.0 V	O

1.6 Block Diagram

Figure 1-2: Block Diagram



Remark: The internal ROM and RAM capacity depends on the product.
The EEPROM is only available in the μPD78F0949 and not in the μPD78F0948.

1.7 Overview of Functions

Item \ Part Number		μPD780948	μPD780949	μPD78F0948	μPD78F0949
Internal memory	ROM	60 Kbytes			
	Internal high-speed RAM	1024 bytes			
	LCD Display RAM	40 bytes			
	Internal Expansion RAM	992 bytes			
	EEPROM	—	256 bytes	—	256 bytes
Memory space		64 Kbytes			
General registers		8 bits x 32 registers (8 bits x 8 registers x 4 banks)			
Instruction cycle		On-chip instruction execution time selective function			
	When main system clock selected	0,25 μs/0,5 μs/1 μs/2 μs/4 μs (at 8 MHz)			
	When subsystem clock selected	122 μs (at 32.768 kHz)			
Instruction set		• 16-bit operation • Multiplication/division (8 bits x 8 bits, 16 bits – 8 bits) • Bit manipulation (set, reset, test, boolean operation) • BCD adjustment, etc.			
I/O ports		Total : 79 • CMOS input : 8 • CMOS I/O : 71			
A/D converter		• 8 bit resolution x 8 channels			
Serial Interface		• 3-wire mode : 1 channel • 2-wire mode : 1 channel • UART mode : 1 channel			
Timer		• 16 bit timer / event counter : 2 channels • 8 bit timer / event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel			
Timer output		3 (16-bit RWM output x 1, 8-bit RWM output x 2)			
Clock output		62,5 kHz, 125 kHz, 250 kHz, 500 kHz, 1 MHz, 2 MHz, 4 MHz, 8 MHz (at main system clock of 8.0 MHz)			
CAN		1 channel			
Vectored interrupts	Maskable interrupts	Internal : 22 External : 5			
	Non-maskable interrupts	Internal : 1			
	Software interrupts	Internal : 1			
Supply voltage		V_{DD} = 4,0 V to 5,5 V			
Package		100-pin plastic QFP (14 mm x 20 mm)			

1.8 Mask Options

The mask ROM version (μPD780948, μPD780949) provides LCD split resistor which allows user to specify whether to connect LCD split resistor externally.

The mask options provided in the μPD780949 subseries are shown in Table 1-1.

Table 1-1: Mask Options of Mask ROM Versions

Pin Names	Mask Options
V_{LC0}, V_{LC1}, V_{LC2}	LCD-split resistor can be specified internally

1.9 Differences between Flash and Mask ROM version

The differences between the two versions are shown in the table below. Differences of the electrical specification are given in the data sheet.

Table 1-2: Differences between Flash and Mask ROM version

	Flash Version	Mask ROM Version
ROM	Flash EEPROM	Mask ROM
LCD Split Resistor	None	Mask Option
V_{PP} Pin	Yes	None (IC pin)

[Memo]

Chapter 2 Pin Function (μPD780949 Subseries)

2.1 Pin Function List

Normal Operating Mode Pins / Pin Input/Output Types

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

Input / Output	Pin Name	Function	Alternate Function	After Reset
Input / Output	P00	Port 0 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	INTP0	Input
	P01		INTP1	Input
	P02		INTP2	Input
	P03		INTP3/T2P0	Input
	P04		INTP4/TI01	Input
	P05		TI00/TO0	Input
	P06		TI50/TO50	Input
	P07		TI51/TO51	Input
Input	P10-P17	Port 1 8 bit input port Input mode can be specified bit-wise	ANI0-ANI7	Input
Input / Output	P20	Port 2 7 bit input/output port Input / output mode can be specified bit-wise	SI0	Input
	P21		SO0	Input
	P22		/SCK0	Input
	P23		SI/SO1	Input
	P24		/SCK1	Input
	P25		RxD	Input
	P26		TxD	Input
Input/ Output	P40-P47	Port 4 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	AD0-AD7	Input
Input/ Output	P50-P57	Port 5 8 bit input / output port Input / output mode can be specified bit-wise This port can be used in External Memory Expansion Mode with the 4, 6 or 8 bit address by setting the Memory Expansion Mode Register Not for external memory expansion used ports can be used either for LCD or port function	A8/S39-A15/S32	Input
Input / Output	P64	Port 6 3 bit input / output port input / output mode can be specified bit-wise	/RD	Input
	P65		/WR	Input
	P67		ASTB	Input

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

Input / Output	Pin Name	Function	Alternate Function	After Reset
Input/ Output	P70-P77	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 This port can be used as a segment signal output port or an I/O port in 1 bit units by setting port function	S31-S24	Input
Input/ Output	P120-P127	Port 12 8 bit input / output port Input / output mode can be specified bit-wise This port can be used as a segment signal output port or an I/O port in 8 bit units by setting LCD control register	S23-S16	Input
Input/ Output	P130-P137	Port 13 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 This port can be used as a segment signal output port or an I/O port in 8 bit units by setting LCD control register	S15-S8	Input
Input/ Output	P140-P147	Port 14 8 bit input / output port Input / output mode can be specified bit-wise This port can be used as a segment signal output port or an I/O port in 8 bit units by setting LCD control register	S7-S0	Input

2.2 Non-Port Pins

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

Pin Name	I/O	Function	After Reset	Alternate Function Pin
INTP0	Input	External interrupts with specifiable valid edges (rising edge, falling edge, both rising and falling edges)	Input	P00
INTP1				P01
INTP2				P02
INTP3				P03/T2P0
INTP4				P04/TI01
SI0	Input	Serial interface serial data input	Input	P20
SI1				P23/SO1
SO0	Output	Serial interface serial data output	Input	P21
SO1				P23/SI1
$\overline{SCK0}$	Input/ Output	Serial interface serial clock input / output	Input	P22
$\overline{SCK1}$				P24
RxD	Input	Asynchronous serial interface data input	Input	P25
TxD	Output	Asynchronous serial interface data output	Input	P26
CRxD	Input	CAN serial data input	Input	-
CTxD	Output	CAN serial data output	Output	-
CCLK	Input	CAN serial clock input	-	CL1
TI00	Input	External count clock input to 16-bit timer (TM0)	Input	P05/TO0
Ti01				P04/INTP4
TI20		Capture trigger input		P30
TI21		Capture trigger input		P31
TI22		Capture trigger input		P32
TI50		External count clock input to 8-bit timer (TM50)		P06/TO50
TI51		External count clock input to 8-bit timer (TM51)		P07/TO51
TO0	Output	16-bit timer output	Input	P05/TI00
T2P0		16-bit timer output		P03/INTP3
TO50		8-bit timer output (also used for PWM output)		P06/TI50
TO51		8-bit timer output (also used for PWM output)		P07/TI51
PCL	Output	Clock output (for main system clock trimming)	Input	P33/SGOA
AD0 to AD7	Input/ Output	Low-order address/data bus at external memory expansion	Input	P40 to P47
A8 to A15	Output	High-order address/data bus at external memory expansion	Input	P50 to P57 S39 to S32
$\overline{RD}$	Output	Strobe signal output for read operation from external memory	Input	P64
$\overline{WR}$		Strobe signal output for read operation from external memory		P65
ASTB	-	Strobe output externally latching address information output to ports 4, 5 to access external memory	Input	P67
S0 to S7	Output	Segment signal output of LCD controller / driver	Input	P147 to P140
S8 to S15				P137 to P130
S16 to S23				P127 to P120
S24 to S31				P77 to P70
S32 to S39				P57 to P50 A15 to A8

Table 2-2: Non-Port Pins (2/2)

Pin Name	I/O	Function	After Reset	Alternate Function Pin
COM0-COM3	Output	Common signal output of LCD controller/driver	Output	-
$V_{LC}0$ to $V_{LC}2$	-	LCD drive voltage	-	-
SGO	Output	Sound generator output	Input	P34/SGOF
SGOA	Output	Sound generator amplitude output	Input	P33/PCL
SGOF	Output	Sound generator frequency output	Input	P34/SGO
ANI0 to ANI7	Input	A/D Converter analog input	Input	P10 – P17
AV_{DD}	-	A/D Converter reference voltage input and power supply	-	-
AV_{SS}	-	A/D Converter ground potential. Connect to V_{SS}.	-	-
$\overline{\text{RESET}}$	Input	System reset input	-	-
X1	-	Crystal connection for main system clock	-	-
X2	-		-	-
CL1	Input	RC connection for subsystem clock	-	CCLK
CL2	-		-	-
$V_{DD}1$, $V_{DD}2$	-	Positive power supply	-	-
$V_{SS}1$, $V_{SS}2$	-	Ground potential	-	-
V_{PP}	-	High voltage supply for flash programming (only flash version)	-	IC
IC	-	Internal connection. Connect directly to V_{SS} (only MaskROM version)	-	V_{PP}

2.3 Description of Pin Functions

2.3.1 P00 to P07 (Port 0)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as an external interrupt input, an external count clock input to the timer, a capture trigger signal input, a timer signal output.

The following operating modes can be specified bit-wise.

(1) Port mode

P00 to P07 function as input/output ports. P00 to P07 can be specified for input or output ports bitwise with a port mode register 0. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistors bitwise in the pull-up resistor option register 0.

(2) Control mode

In this mode, these ports function as an external interrupt input, an external count clock input to the timer, and a timer signal output.

(a) INTP0 to INTP4

INTP0 to INTP4 are external interrupt input pins which can specify valid edges (rising edge, falling edge, and both rising and falling edges). INTP4 becomes a 16-bit timer/event counter capture trigger signal input pin with a valid edge input.

(b) TI00

Pin for external count clock input to 16-bit timer/event counter and pin for capture trigger signal input to the 16-bit timer/event counter capture register (CR01).

(c) TI01

Pin for capture trigger signal input to capture register of 16-bit timer/event counter (CR00).

(d) TI50

Pin for external count clock input to 8-bit timer/event counter.

(e) TI51

Pin for external count clock input to 8-bit timer/event counter.

(f) TO0

Pin for output of the 16-bit timer/event counter.

(g) TO50

Pin for output of the 8-bit timer/event counter.

(h) TO51

Pin for output of the 8-bit timer/event counter.

(i) T2PO

Pin for output of the 16-bit timer (TM2).

2.3.2 P10 to P17 (Port 1)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as an A/D converter analog input.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 8-bit input ports.

(2) Control mode

These ports function as A/D converter analog input pins (ANI0 to ANI7).

2.3.3 P20 to P26 (Port 2)

These are 7-bit input/output ports. Besides serving as input/output ports, they function as data input/output to/from the serial interface, clock input/output.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 7-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 2. P20 to P24 are selectable as N-ch open drain or as CMOS output.

(2) Control mode

These ports function as serial interface data input/output, clock input/output.

(a) SI0, SI1, SO0, SO1

Serial interface serial data input/output pins

(b) SCK0 and SCK1

Serial interface serial clock input/output pins

(c) RxD, TxD

Asynchronous serial interface data input/output pins

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

2.3.4 P30 to P34 (Port 3)

These are 5-bit input/output ports. Beside serving as input/output ports, they function as timer input, clock ouput and sound generator ouput.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 5-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 3.

(2) Control mode

These ports function as timer input, clock output, and sound generator output.

(a) TI20, TI21 and TI22

Pin for external capture trigger input to the 16-bit timer/capture registers of TM2.

(c) PCL

Clock output pin.

(d) SGO, SGOA and SGOF

Pins for separate or composed signal ouput of the sound generator.

2.3.5 P40 to P47 (Port 4)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as an address/data bus.

The following operating mode can be specified in 8-bit units.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified in 8-bit units for input or output ports by using the memory expansion mode register. When they are used as input ports, pull-up resistors can be connected bitwise by defining the pull-up resistor option register 4.

(2) Control mode

These ports function as low-order address/data bus pins (AD0 to AD7) in external memory expansion mode. When pins are used as an address/data bus, the pull-up resistor is automatically disabled.

2.3.6 P50 to P57 (Port 5)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as an address bus and LCD controller/driver.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input/output ports with port mode register 5.

(2) Control mode

These ports function as high-order address bus pins (A8 to A15) in external memory expansion mode or as segment signal ouput pins (S32 to S39) of LCD controller/driver output.

2.3.7 P64, P65 and P67 (Port 6)

These are 3-bit input/output ports. Besides serving as input/output ports, they are used for control in external memory expansion mode.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 3-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 6.

(2) Control mode

These ports function as control signal output pins ($\overline{RD}$, $\overline{WR}$, $\overline{ASTB}$) in external memory expansion mode, therefore a pin has to be used as a control signal output.

2.3.8 P70 to P77 (Port 7)

This is a 8-bit input/output port. In addition to its use as an input/output port, it also segment sognal output functions of the LCD controller/driver.

The following operating modes can be specified bit-wise.

(1) Port mode

Port 7 functions as a 8-bit input/output port. Bit-wise specification as an input port or output port is possible by means of port mode register 7. When used as input ports, pull-up resistors can be connected by defining the pull-up resistor option register 7.

(2) Control mode

Port 7 functions as segment signal output pins (S24 to S31) of LCD controller/driver.

2.3.9 P120 to P127 (Port 12)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as segment signal output pins of LCD controller/driver.

The following operating modes can be specified bit-wise or byte-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 12.

(2) Control mode

These ports function as segment output signal pins (S16 to S23) of LCD controller/driver.

2.3.10 P130 to P136 (Port 13)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as segment signal output pins of LCD controller/driver.

The following operating modes can be specified bit-wise or byte-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 13. When used as input ports, pull-up resistors can be connected by defining the pull-up resistor option register 7.

(2) Control mode

These ports function as segment output signal pins (S8 to S15) of LCD controller/driver.

2.3.11 P140 to P147 (Port 14)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as segment signal output pins of LCD controller/driver.

The following operating modes can be specified bit-wise or byte-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 14.

(2) Control mode

These ports function as segment output signal pins (S0 to S7) of LCD controller/driver.

2.3.12 CTxD

These pin functions as CAN-controller transmit output.

2.3.13 CRxD

These pin functions as CAN-controller receive input.

2.3.14 COM0 to COM3

These are LCD controller/driver common signal output pins. They output common signals under the following condition:

- 4-time-devision is performed in 1/3 bias mode

2.3.15 V_{LC0} to V_{LC2}

These are LCD drive voltage pins. In the MaskROM product, a split resistor for LCD drive voltage generation can be incorporated by a mask option, without connecting external split resistors.

2.3.16 AV_{DD}

A/D converter reference voltage input pin and the power supply for the A/D-converter.
When A/D converter is not used, connect this pin to V_{DD}.

2.3.17 AV_{SS}

This is a ground voltage pin of A/D converter. Always use the same voltage as that of the V_{SS} pin even when A/D converter is not used.

2.3.18 $\overline{\text{RESET}}$

This is a low-level active system reset input pin.

2.3.19 X1 and X2

Crystal resonator connect pins for main system clock oscillation. For external clock supply, input it to X1.

2.3.20 CL1 and CL2

Crystal resonator connect pins for subsystem clock oscillation.
For external clock supply, input it to CL1 and its inverted signal to CL2. For CAN-clock, input it to CL1 and leave CL2 open.

2.3.21 V_{DD}

Positive power supply pin

2.3.22 V_{SS}

Ground potential pin

2.3.23 V_{PP} (μPD78F0948, μPD78F0949 only)

High-voltage apply pin for FLASH programming mode setting. Connect directly to V_{SS} in normal operating mode.

2.3.24 IC (Mask ROM version only)

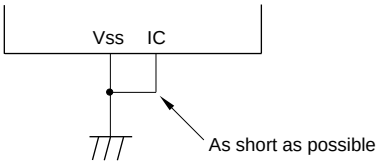
The IC (Internally Connected) pin is provided to set the test mode to check the μPD78F0948, μPD78F0949 at delivery.

Connect it directly to the VSS with the shortest possible wire in the normal operating mode.

When a voltage difference is produced between the IC pin and VSS pin because the wiring between those two pins is too long or an external noise is input to the IC pin, the user's program may not run normally.

Figure 2-1: Connection of IC Pins

• Connect IC pins to Vss 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, see table.

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

Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
P00/INTP0	8-A	I/O	Connect to Vdd or Vss via a resistor individually
P01/INTP1			
P02/INTP2			
P03/INPT3/T2P0			
P04/INTP4/TI01			
P05/TI00/TO0			
P06/TI50/TO50			
P07/TI51/TO51			
P10/ANI0	11-B	I	Connect to Vdd or Vss via a resistor individually
P11/ANI1			
P12/ANI2			
P13/ANI3			
P14/ANI4			
P15/ANI5			
P16/ANI6			
P17/ANI7			
P20/SI0	10	I/O	Connect to Vdd or Vss via a resistor individually
P21/SO0			
P22/$\overline{\text{SCK0}}$			
P23/SI1/SOA			
P24/$\overline{\text{SCK1}}$			
P25/RxD	8		
P26/TxD	5		
P30/TI20	8	I/O	Connect to Vdd or Vss via a resistor individually
P31/TI21			
P32/TI22			
P33/PCL/SGOA	5		
P34/SGO/SGOF			
P40/AD0	5-A	I/O	Connect to Vdd or Vss via a resistor individually
P41/AD1			
P42/AD2			
P43/AD3			
P44/AD4			
P45/AD5			
P46/AD6			
P47/AD7			

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

Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
P50/A8/S39	17	I/O	Connect to Vdd or Vss via a resistor individually
P51/A9/S38			
P52/A10/S37			
P53/A11/S36			
P54/A12/S35			
P55/A13/S34			
P56/A14/S33			
P57/A15/S32			
P64/$\overline{RD}$	5-A	I/O	Connect to Vdd or Vss via a resistor individually
P65/$\overline{WR}$			
P67/ASTB			
P70/S31	17-B	I/O	Connect to Vdd or Vss via a resistor individually
P71/S30			
P72/S29			
P73/S28			
P74/S27			
P75/S26			
P76/S25			
P77/S24			
P120/S23	17-C	I/O	Connect to Vdd or Vss via a resistor individually
P121/S22			
P122/S21			
P123/S20			
P124/S19			
P125/S18			
P126/S17			
P127/S16			
P130/S15	17-A	I/O	Connect to Vdd or Vss via a resistor individually
P131/S14			
P132/S13			
P133/S12			
P134/S11			
P135/S10			
P136/S9			
P137/S8			
P140/S7	17-A	I/O	Connect to Vdd or Vss via a resistor individually
P141/S6			
P142/S5			
P143/S4			
P144/S3			
P145/S2			
P146/S1			
P147/S0			

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

Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
COM0 – COM3	18	O	Leave open
$V_{LC}0$ – $V_{LC}2$	-	-	
CRxD	1	I	Connect to Vdd or Vss via a resistor individually
CTxD	2	O	Leave open
CL1/CCLK	-	-	Connect to Vdd or Vss via a resistor individually
C2	-	-	Leave open
$\overline{\text{RESET}}$	1	I	-
AV_{DD}	-	-	Connect to V_{DD}
AV_{SS}	-	-	Connect to V_{SS}
IC	-	-	Connect directly to V_{SS}
V_{PP}			

Figure 2-2: Pin Input/Output Circuits (1/3)

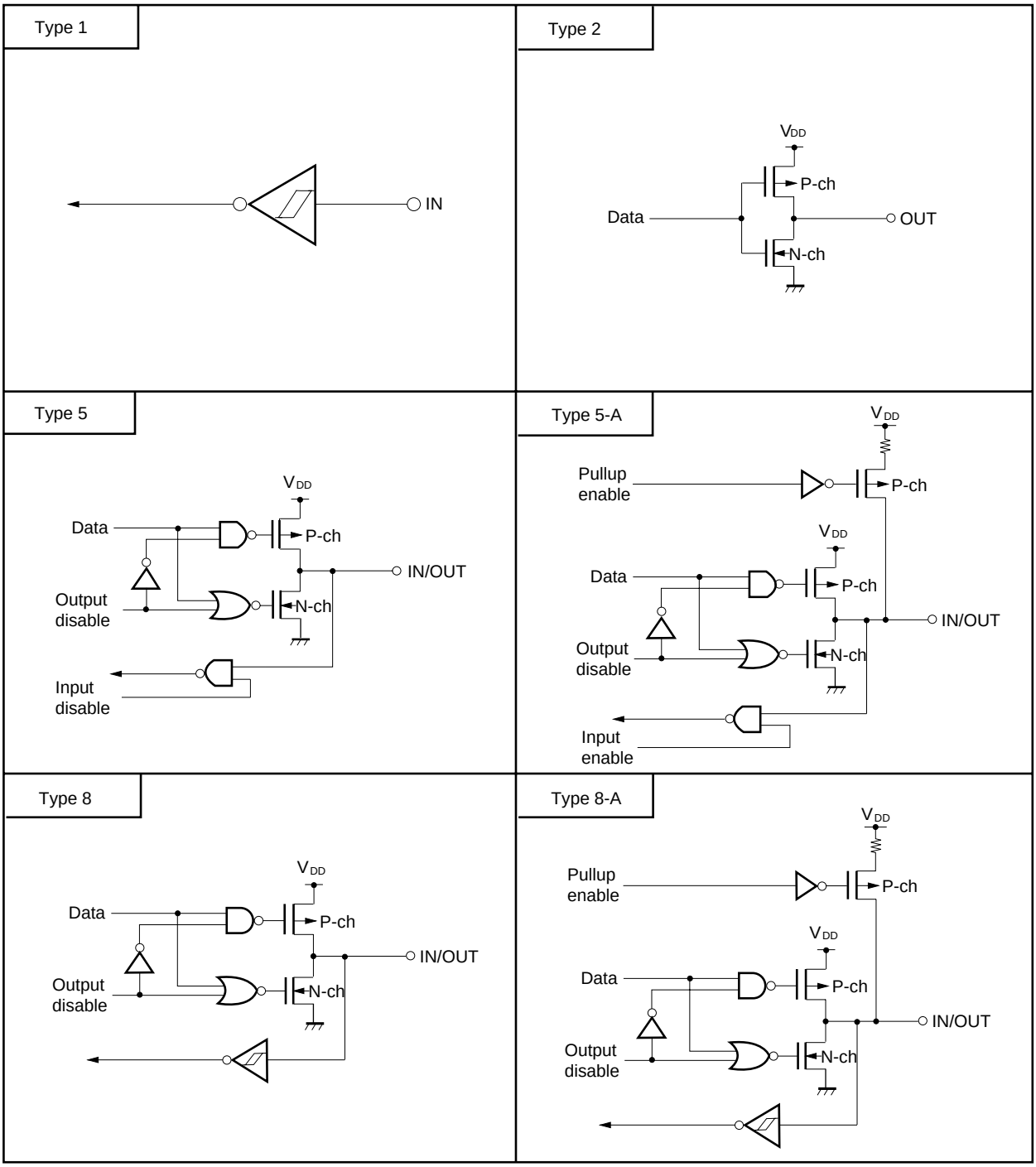


Figure 2-2: Pin Input/Output Circuits (2/3)

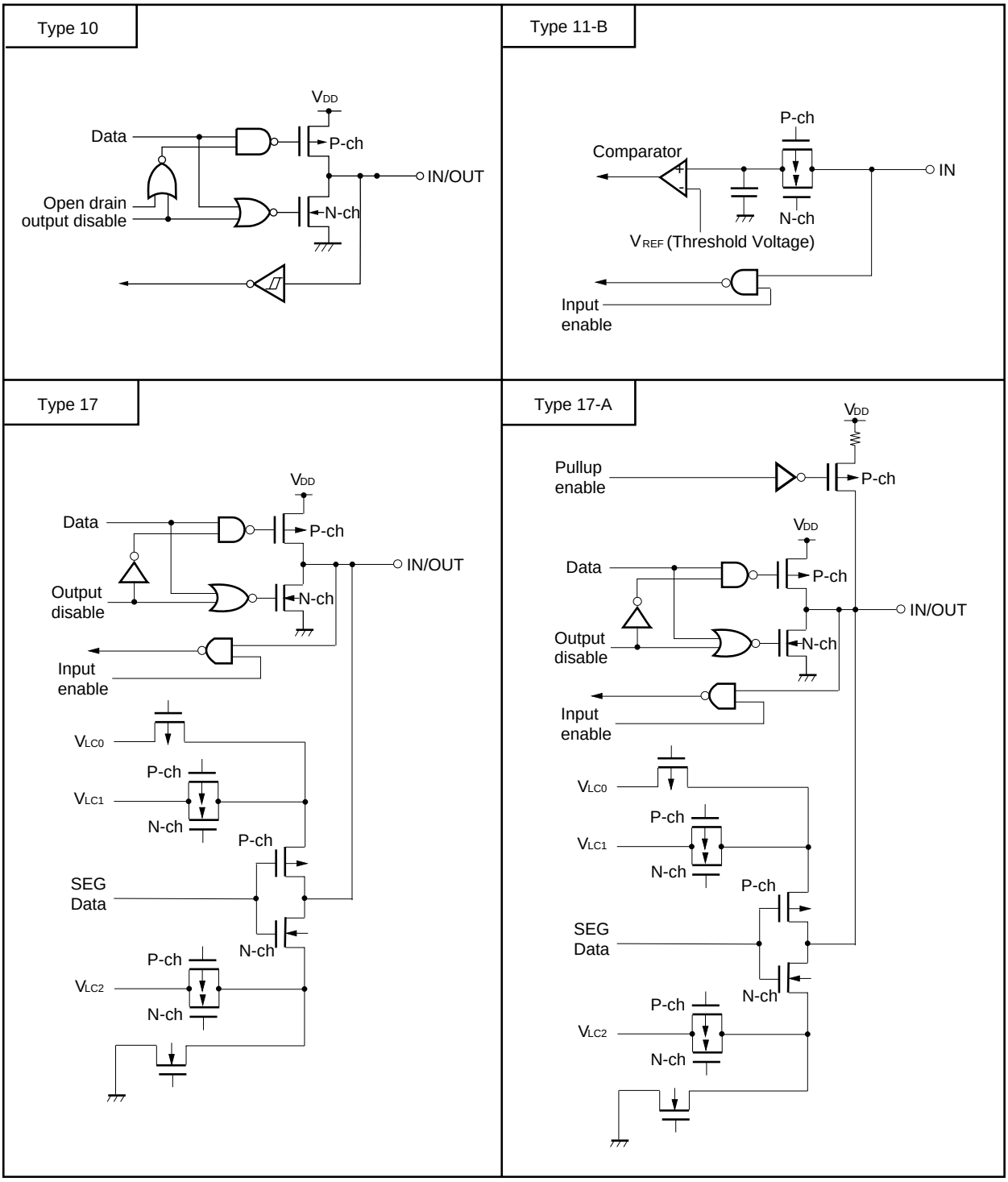
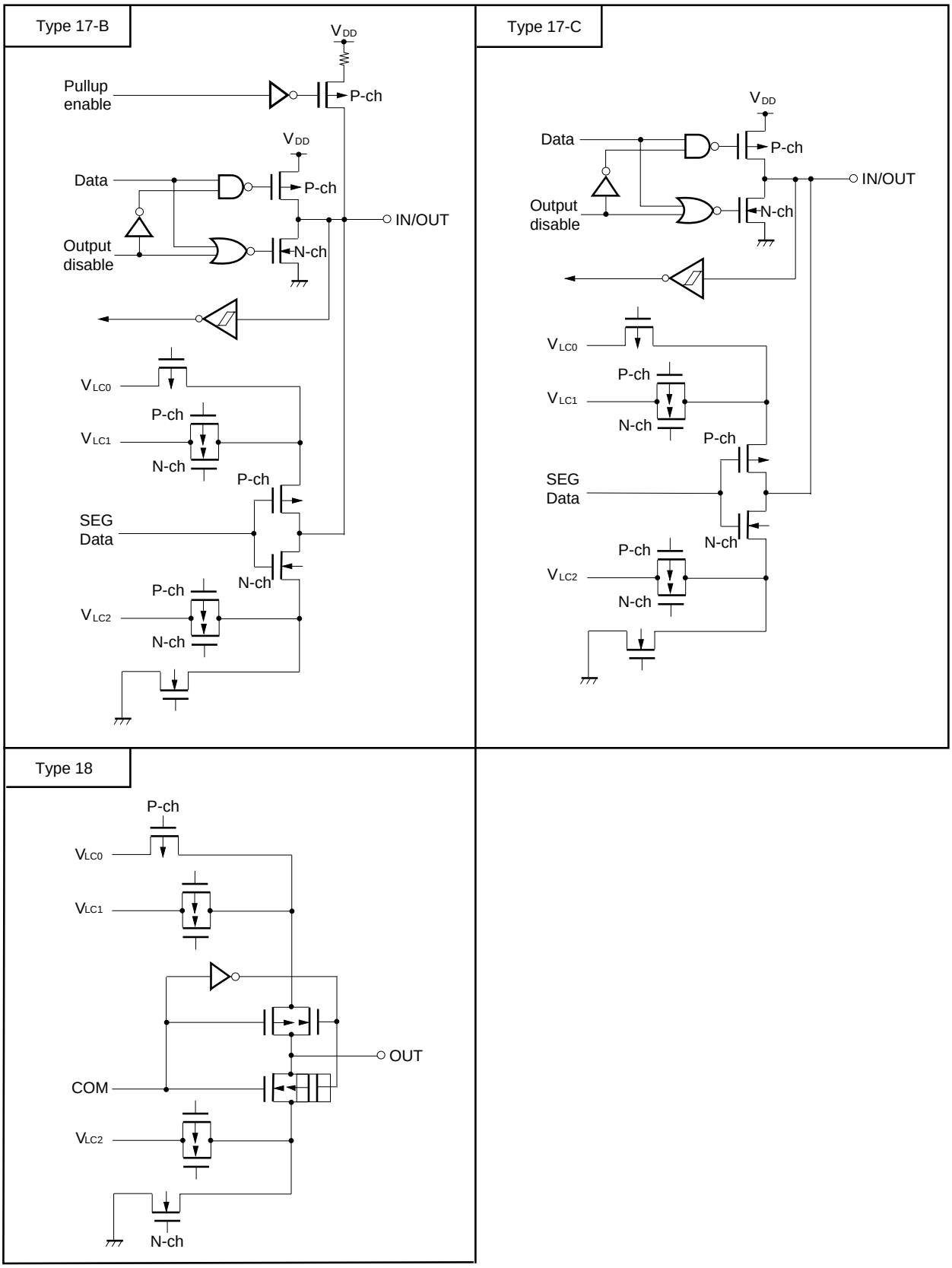


Figure 2-2: Pin Input/Output Circuits (3/3)



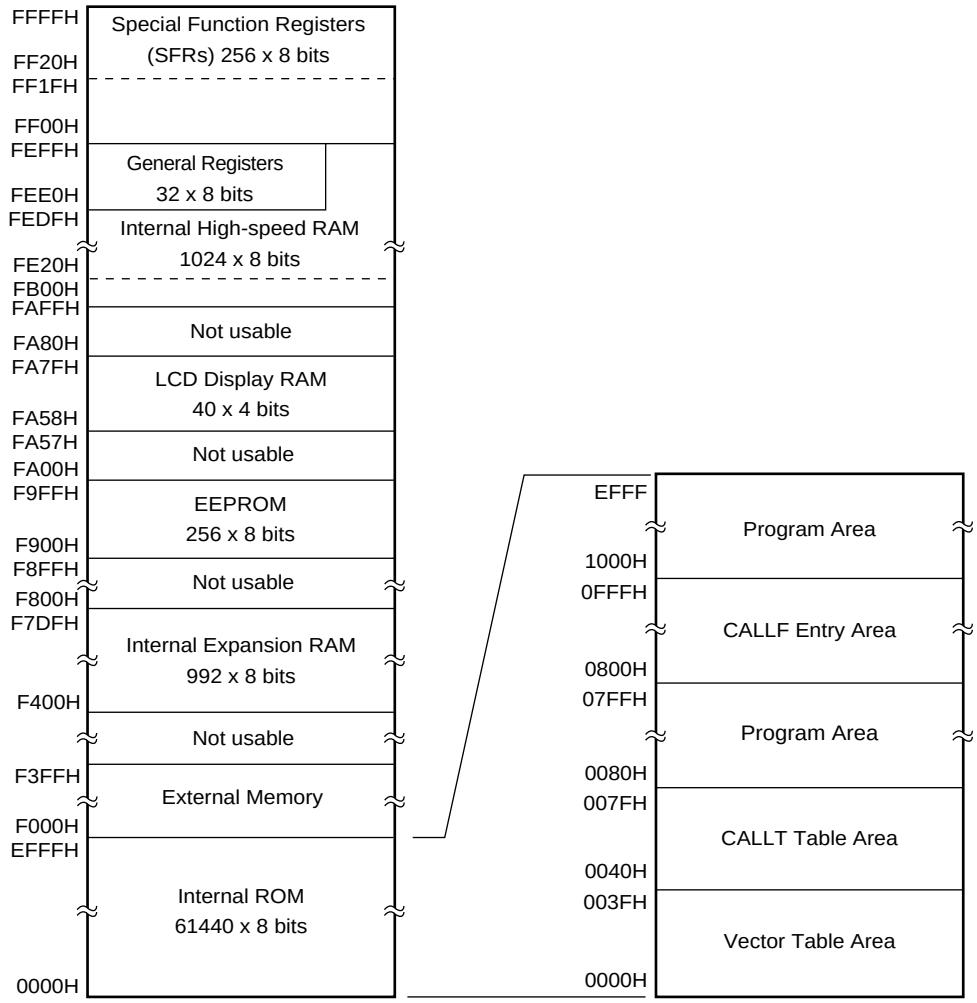
[Memo]

Chapter 3 CPU Architecture

3.1 Memory Space

The memory map of the μPD780949 is shown in Figure 3-1.

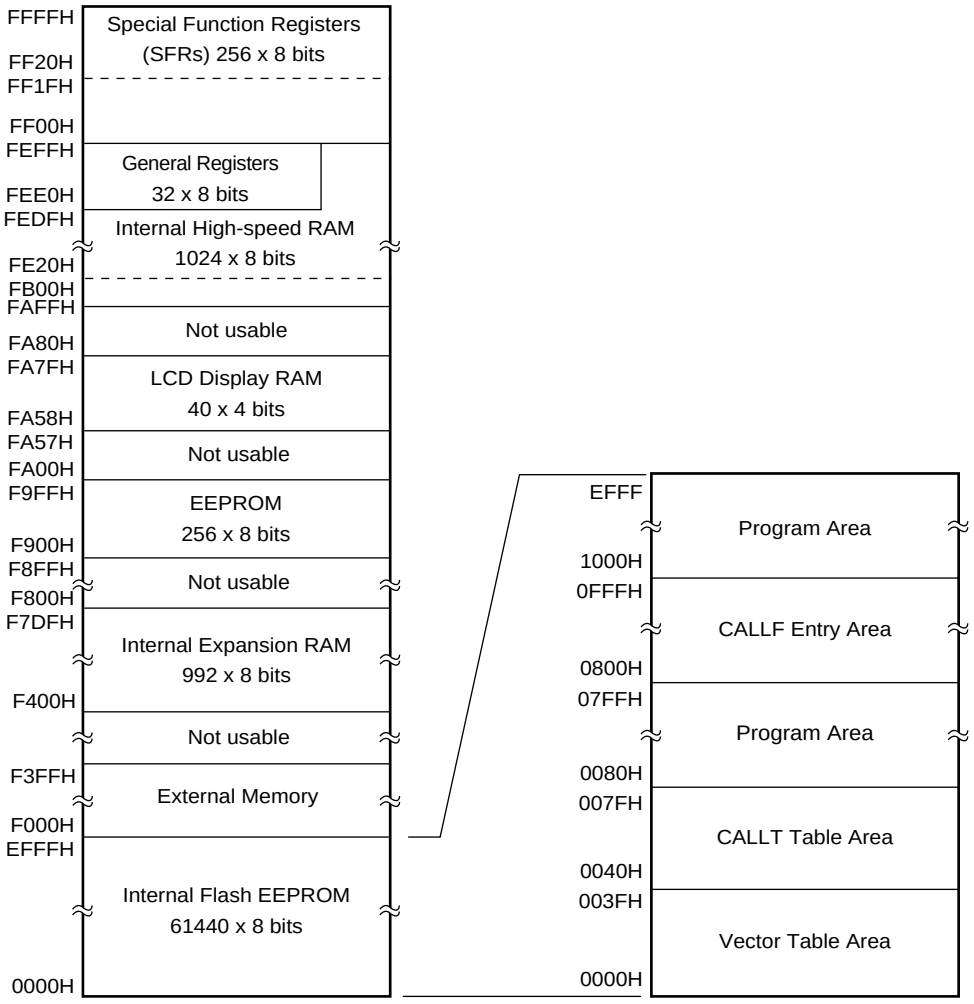
Figure 3-1: Memory Map (μPD780949)



Note: The EEPROM is only valid in the μPD780949 and not in the μPD780948.

The memory map of the μPD78F0949 is shown in Figure 3-2.

Figure 3-2: Memory Map (μPD78F0949)



Note: The EEPROM is only valid in the μPD78F0949 and not in the μPD78F0948.

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 μPD780949 subseries have various size of internal ROMs or Flash EPROM as shown below.

Table 3-1: Internal ROM Capacities

Part Number	Internal ROM	
	Type	Capacity
μPD780949	Mask ROM	61440 x 8-bits
μPD780948	Mask ROM	61440 x 8-bits
μPD78F0949	Flash	61440 x 8-bits
μPD78F0948	Flash	61440 x 8-bits

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

(1) Vector table area

The 64-byte area 0000H to 003FH is reserved as a vector table area. The $\overline{\text{RESET}}$ input and program start addresses for branch upon generation of each interrupt request 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
0004H	INWDT
0006H	INTAD
0008H	INTOVF
000AH	INTTM20
000CH	INTTM21
000EH	INTTM22
0010H	INTP0
0012H	INTP1
0014H	INTP2
0016H	INTP3
0018H	INTP4
001AH	INTCE
001CH	INTCR
001EH	INTCT0
0020H	INTCT1
0022H	INTCSI0
0024H	INTCSI1
0026H	INTSER
0028H	INTSR
002AH	INTST
002CH	INTTM00
002EH	INTTM01
0030H	INTTM50
0032H	INTTM51
0034H	INTWE
0036H	INTWTI
0038H	INTWT
003EH	BRK

(2) CALLT instruction table area

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

(3) CALLF instruction entry area

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

3.1.2 Internal data memory space

The μPD780949 subseries units incorporate the following RAMs.

(1) Internal high-speed RAM

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

(2) LCD-Display RAM

Buffer RAM is allocated to the 40 x 4 bits area from FA58H to FA7FH. LCD-Display RAM can also be used as normal RAM.

(3) Internal expansion RAM

Internal expansion RAM is allocated to the 992-byte area from F400H to F7DFH.

3.1.3 Special function register (SFR) area

An on-chip peripheral hardware special function register (SFR) is allocated in the area FF00H to FFFFH. (Refer to **Table 3-3**).

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

3.1.4 External memory space

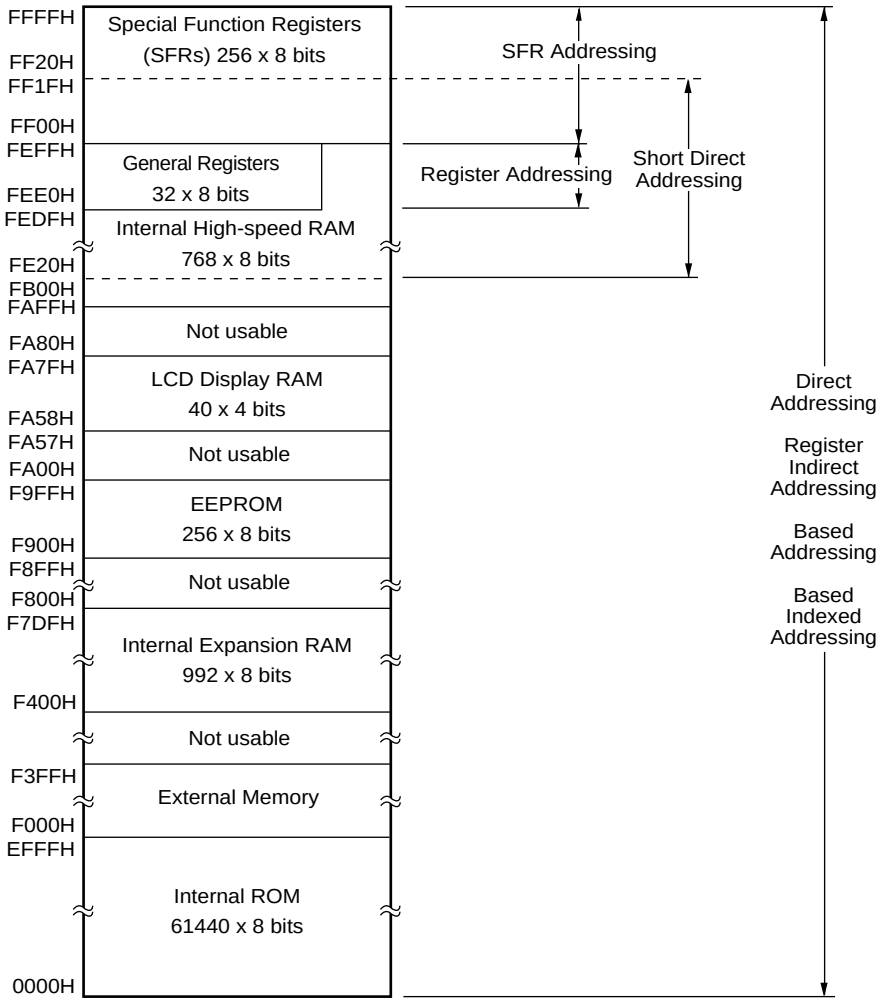
The external memory space is accessible by setting the memory expansion mode register. External memory space can store program, table data, etc. and allocate peripheral devices.

3.1.5 Data memory addressing

The μPD780949 subseries is provided with a varity 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 64K-byte space (0000H to FFFFH). Figures 3-3 and 3-4 show the data memory addressing modes.

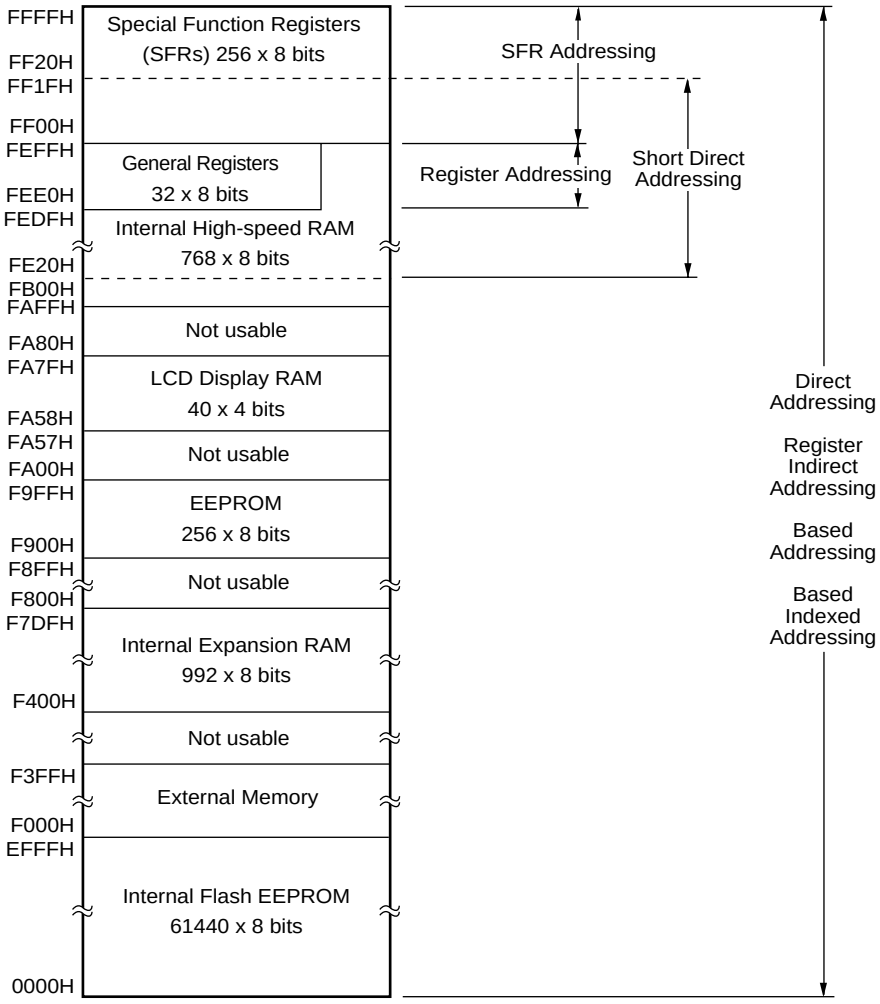
For details of addressing, refer to **3.4 Operand Address Addressing**.

Figure 3-3: Data Memory Addressing (μPD780949)



Note: The EEPROM is only valid in the μPD780949 and not in the μPD780948.

Figure 3-4: Data Memory Addressing (μPD78F0949)



Note: The EEPROM is only valid in the μPD78F0949 and not in the μPD78F0948.

3.2 Processor Registers

The μPD780949 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.

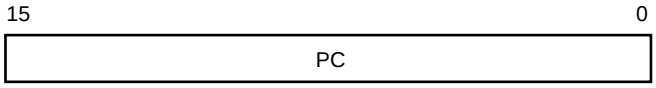
(1) Program counter (PC)

The program counter is a 16-bit register which holds the address information of the next program 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 3-5: 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 stacked upon interrupt request generation or PUSH PSW instruction execution and are automatically reset upon execution of the RETB, RETI and POP PSW instructions.

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

Figure 3-6: Program Status Word Configuration

7							0
IE	Z	RBS1	AC	RBS0	0	ISP	CY

(a) Interrupt enable flag (IE)

This flag controls the interrupt request acknowledge operations of the CPU.

When 0, the IE is set to interrupt disabled (DI) status. All interrupts except non-maskable interrupt are disabled.

When 1, the IE is set to interrupt enabled (EI) status and interrupt request acknowledge is controlled with an in-service priority flag (ISP), an interrupt mask flag for various interrupt sources, 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 operation result 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.

(e) In-service priority flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts. When 0, acknowledgment of the vectored interrupt request specified to low-order priority with the priority specify flag registers (PR0L, PR0H, and PR1L) is disabled. Whether an actual interrupt request is acknowledged or not is controlled with the interrupt enable flag (IE).

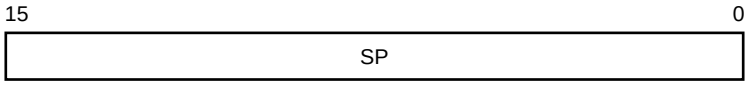
(f) Carry flag (CY)

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 to hold the start address of the memory stack area. Only the internal high-speed RAM area can be set as the stack area.

Figure 3-7: Stack Pointer Configuration



The SP is decremented ahead of write (save) to the stack memory and is incremented after read (reset) from the stack memory.

Each stack operation saves/resets data as shown in Figures 3-8 and 3-9.

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

Figure 3-8: Data to be Saved to Stack Memory

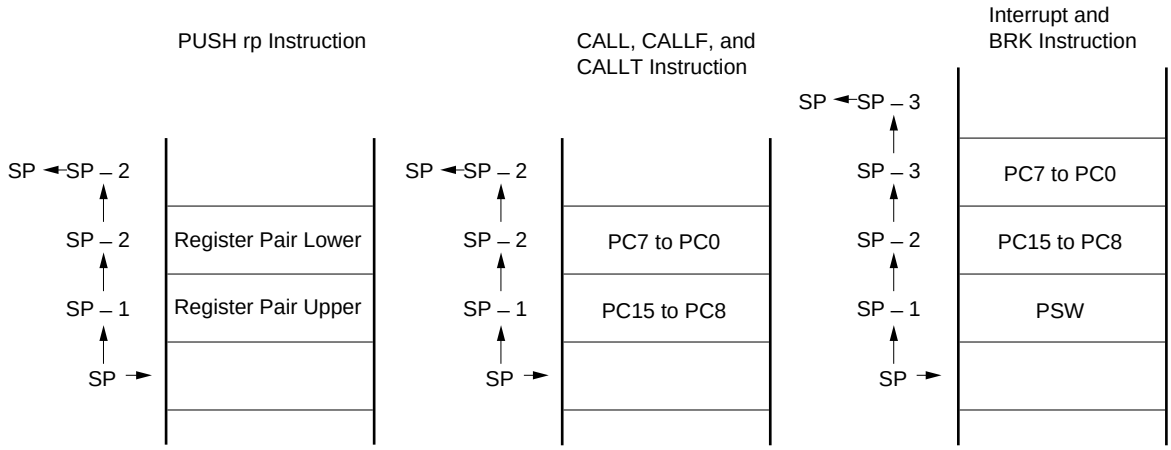
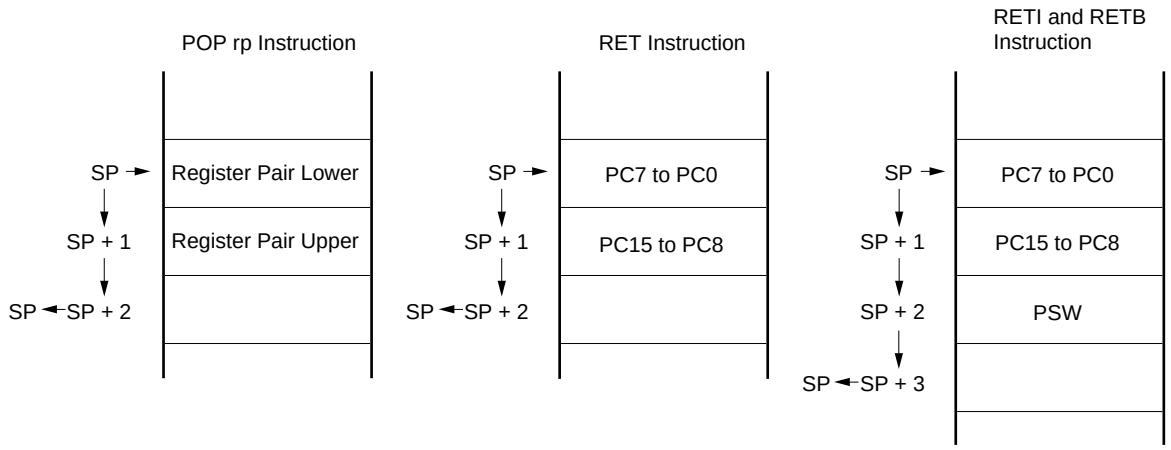


Figure 3-9: Data to be Reset to Stack Memory



3.2.2 General registers

A general register is mapped at particular addresses (FEE0H to FEFFH) of the data memory. It consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

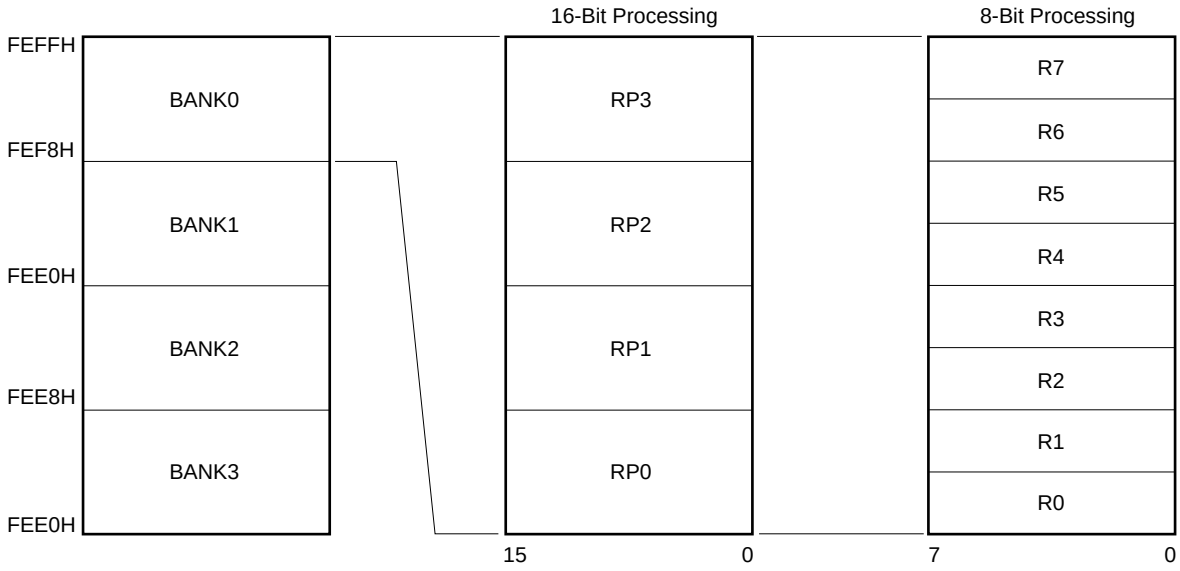
Each register can also be used as an 8-bit register. Two 8-bit registers can be used in pairs as a 16-bit register (AX, BC, DE, and HL).

They can be described in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

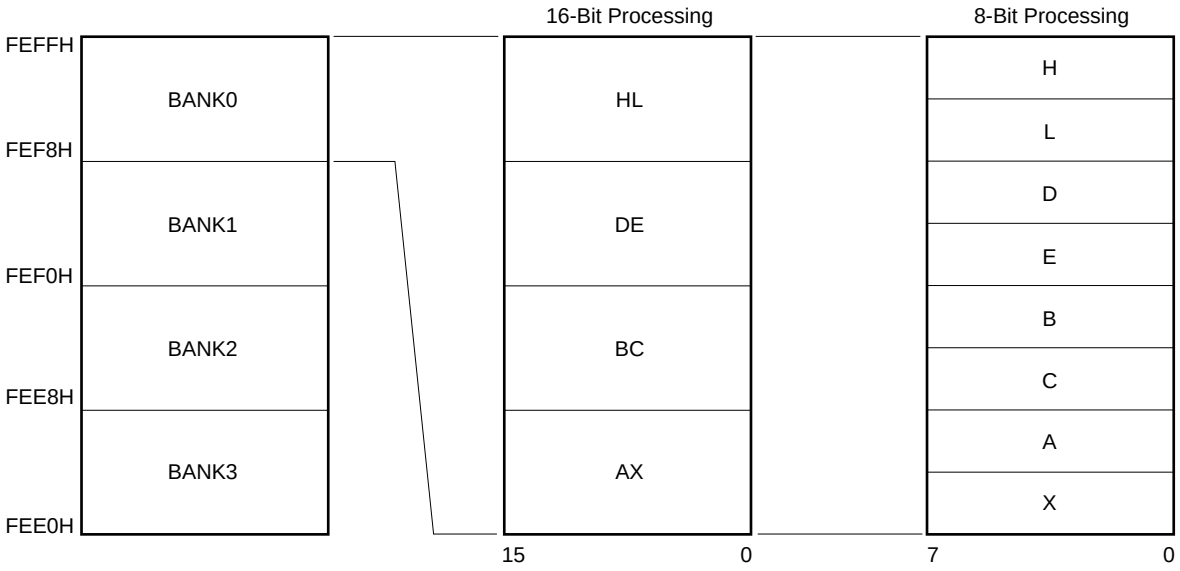
Register banks to be used for instruction execution are set with the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interruption for each bank.

Figure 3-10: General Register Configuration

(a) Absolute Name



(b) Function Name



3.2.3 Special function register (SFR)

Unlike a general register, each special function register has special functions.

It is allocated in the FF00H to FFFFH area.

The special function registers can be manipulated in a similar way as the general registers, by using operation, transfer, or bit-manipulate instructions. The special function registers are read from and written to in specified manipulation bit units (1, 8, and/or 16) depending on the register type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation
  Describe the symbol reserved with assembler for the 1-bit manipulation instruction operand (sfr.bit).
  This manipulation can also be specified with an address.
- 8-bit manipulation
  Describe the symbol reserved with assembler for the 8-bit manipulation instruction operand (sfr).
  This manipulation can also be specified with an address.
- 16-bit manipulation
  Describe the symbol reserved with assembler for the 16-bit manipulation instruction operand (sfrp).
  When addressing an address, describe an even address.

Table 3-3 gives a list of special function registers. The meaning of items in the table is as follows.

- Symbol
  The assembler software RA78K0 translates these symbols into corresponding addresses where the special function registers are allocated. These symbols should be used as instruction operands in the case of programming.
- 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 read out. A write to this register is ignored.
  W : A value can be written to the register. Reading values from the register is impossible.
- Manipulation
  The register can be manipulated in bit units.
- 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 (1/4)

Address	SFR Name	Symbol		R/W	Manipulatable Bit Unit			After Reset
					1 bit	8 bits	16 bits	
FF00H	Port 0	P0		R/W	○	○	—	00H
FF01H	Port 1	P1		R	○	○	—	00H
FF02H	Port 2	P2		R/W	○	○	—	00H
FF03H	Port 3	P3		R/W	○	○	—	00H
FF04H	Port 4	P4		R/W	○	○	—	00H
FF05H	Port 5	P5		R/W	○	○	—	00H
FF06H	Port 6	P6		R/W	○	○	—	00H
FF07H	Port 7	P7		R/W	○	○	—	00H
FF0CH	Port 12	P12		R/W	○	○	—	00H
FF0DH	Port 13	P13		R/W	○	○	—	00H
FF0EH	Port 14	P14		R/W	○	○	—	00H
FF10H	16bit timer/counter register 0	TM0	TM0L	R	—	—	○	00H
FF11H			TM0H	R	—	—	○	00H
FF12H		TM50		R	—	○	—	00H
FF13H		TM51		R	—	○	—	00H
FF14H	16bit capture/compare register 00	CR00	CR00L	R/W	—	—	○	00H
FF15H			CR00H	R/W	—	—	○	00H
FF16H	16bit capture/compare register 01	CR01	CR01L	R/W	—	—	○	00H
FF17H			CR01H	R/W	—	—	○	00H
FF18H	Compare register 50	CR50		R/W	—	○	—	00H
FF19H	Compare register 51	CR51		R/W	—	○	—	00H
FF1BH	A/D conversion result register	ADCR1		R	—	○	—	00H
FF1FH	Serial I/O shift register 30	SIO30		R/W	—	○	—	00H
FF20H	Port mode register 0	PM0		R/W	○	○	—	FFH
FF22H	Port mode register 2	PM2		R/W	○	○	—	FFH
FF23H	Port mode register 3	PM3		R/W	○	○	—	FFH
FF24H	Port mode register 4	PM4		R/W	○	○	—	FFH
FF25H	Port mode register 5	PM5		R/W	○	○	—	FFH
FF26H	Port mode register 6	PM6		R/W	○	○	—	FFH
FF27H	Port mode register 7	PM7		R/W	○	○	—	FFH
FF2CH	Port mode register 12	PM12		R/W	○	○	—	FFH
FF2DH	Port mode register 13	PM13		R/W	○	○	—	FFH
FF2EH	Port mode register 14	PM14		R/W	○	○	—	FFH
FF30H	Pull-up res. option register 0	PU0		R/W	○	○	—	00H
FF34H	Pull-up res. option register 4	PU4		R/W	○	○	—	00H
FF37H	Pull-up res. option register 7	PU7		R/W	○	○	—	00H
FF3DH	Pull-up res. option register 13	PU13		R/W	○	○	—	00H

Table 3-3: Special Function Register List (2/4)

Address	SFR Name	Symbol		R/W	Manipulatable Bit Unit			After Reset
					1 bit	8 bits	16 bits	
FF40H	Clock output select register	CKS		R/W	○	○	—	00H
FF41H	Watch timer mode register	WTM		R/W	○	○	—	00H
FF42H	Watch dog timer clock sel. register	WDCS		R/W	○	○	—	00H
FF47H	Memory expansion mode register	MEM		R/W	○	○	—	00H
FF48H	Ext. INT rising edge enable register	EGP		R/W	○	○	—	00H
FF49H	Ext. INT falling edge enable register	EGN		R/W	○	○	—	00H
FF4AH	LCD timer mode ctrl. register [Note]	LCDTM		R/W				
FF52H	Port function register 2	PF2		R/W	○	○	—	00H
FF55H	Port function register 5	PF5		R/W	○	○	—	00H
FF57H	Port function register 7	PF7		R/W	○	○	—	00H
FF5CH	Port function register 12	PF12		R/W	○	○	—	00H
FF5DH	Port function register 13	PF13		R/W	○	○	—	00H
FF5EH	Port function register 14	PF14		R/W	○	○	—	00H
FF60H	16-bit timer mode control register 0	TMC0		R/W	○	○	—	00H
FF61H	Prescaler mode register 0	PRM0		R/W	—	○	—	00H
FF62H	Capture/compare control register 0	CRC0		R/W	—	○	—	00H
FF63H	16-bit timer output control register 0	TOC0		R/W	○	○	—	00H
FF65H	16-bit timer mode control register 2	TMC2		R/W	○	○	—	00H
FF66H	Prescaler mode register 2	PRM2		R/W	—	○	—	00H
FF67H	Capture/compare control register 2	CRC2		R/W	—	○	—	00H
FF68H	16-bit timer/counter register 2	TM2	TM2L	R	—	—	○	00H
FF69H			TM2H	R	—	—	○	00H
FF6AH	16-bit capture register 20	CR20	CR20L	R	—	—	○	00H
FF6BH			CR20H	R	—	—	○	00H
FF6CH	16-bit capture register 21	CR21	CR21L	R	—	—	○	00H
FF6DH			CR21H	R	—	—	○	00H
FF6EH	16-bit capture register 22	CR22	CR22L	R	—	—	○	00H
FF6FH			CR22H	R	—	—	○	00H
FF70H	8-bit timer mode control register 50	TMC50		R/W	○	○	—	04H
FF71H	Timer clock selection register 50	TCL50		R/W	—	○	—	00H
FF74H	8-bit timer mode control register 51	TMC51		R/W	○	○	—	04H
FF75H	Timer clock selection register 51	TCL51		R/W	—	○	—	00H
FF80H	EEPROM write control register	EEWC		R/W	○	○	—	00H

Note: Only emulator has his register (D78P0308).

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

Address	SFR Name	Symbol	R/W	Manipulatable Bit Unit			After Reset
				1 bit	8 bits	16 bits	
FF90H	LCD display mode register	LCDM	R/W	○	○	—	00H
FF92H	LCD display control register	LCDC	R/W	○	○	—	00H
FF98H	A/D converter mode register 1	ADM1	R/W	○	○	—	00H
FF99H	Analog channel select register 1	ADS1	R/W		○	—	00H
FF9AH	Power fail comparator mode reg.	PFM	R/W	○	○	—	00H
FF9BH	Power fail comp. threshold reg.	PFT	R/W		○	—	00H
FF9CH	D/A conv. channel 0 mode reg. Note	DAM0	R/W	○	○	—	00H
FFA0H	UART operation mode register	ASIM0	R/W	○	○	—	00H
FFA1H	UART receive status register	ASIS0	R/W	—	○	—	00H
FFA2H	Baud rate generator control register	BRGC0	R/W	—	○	—	00H
FFA3H	Transmit shift register	TXS0	W	—	○	—	FFH
	Receive buffer register	RXB0	R	—	○	—	FFH
FFA8H	Serial mode register 0	CSIM30	R/W	○	○	—	00H
FFAAH	Serial mode register 1	CSIM31	R/W	○	○	—	00H
FFABH	Serial I/O shift register 31	SIO31	R/W	—	○	—	00H
FFB0H	CAN control register	CANC	R/W	○	○	—	01H
FFB1H	Transmit control register	TCR	R/W	—	○	—	00H
FFB2H	Received message register	RMES		—	○	—	00H
FFB3H	Redefinition control register	REDEF	R/W	○	○	—	00H
FFB4H	CAN error status register	CANES	R/W	—	○	—	00H
FFB5H	Transmit error counter	TEC	R	—	○	—	00H
FFB6H	Receive error counter	REC	R	—	○	—	00H
FFB7H	Message count register	MCNT	R	—	○	—	00H
FFB8H	Bit rate prescaler	BRPRS	R/W	○	○	—	3FH
FFB9H	Synchronous control register 0	SYNC0	R/W	—	○	—	18H
FFBAH	Synchronous control register 1	SYNC1	R/W	—	○	—	0EH
FFBBH	Mask control register	MASKC	R/W	○	○	—	00H
FFC0H	Sound generator control register	SGCR	R/W	○	○	—	04H
FFC1H	Sound gen. ampitude ctrl. register	SGAM	R/W	—	○	—	00H
FFC2H	Sound gen. buzzer control register	SGBR	R/W	—	○	—	00H

Note: This register is needed for the emulation of power fail detect (PFD) function.

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

Address	SFR Name	Symbol		R/W	Manipulatable Bit Unit			After Reset
					1 bit	8 bits	16 bits	
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 1L		MK0H	R/W	○	○		FFH
FFE6H	Interrupt mask flag register 1L	MK1	MK1L	R/W	○	○	○	FFH
FFE7H	Interrupt mask flag register 1H		MK1H	R/W	○	○		FFH
FFE8H	Priority order special flag 0L	PR0	PR0L	R/W	○	○	○	FFH
FFE9H	Priority order special flag 0H		PR0H	R/W	○	○		FFH
FFEAH	Priority order special flag 1L	PR1	PR1L	R/W	○	○	○	FFH
FFEBH	Priority order special flag 1H		PR1H	R/W	○	○		FFH
FFF0H	Memory size switching register	IMS		R/W	—	○	—	CFH
FFF4H	Internal exp. RAM size switch. reg.	IXS		R/W	—	○	—	0CH
FFF8H	Memory expansion wait register	MM		R/W	○	○	—	10H
FFF9H	Watch dog timer mode register	WDTM		R/W	○	○	—	00H
FFFAH	Oscillation stabilization time register	OSTS		R/W	—	○	—	04H
FFFBH	Processor clock control register	PCC		R/W	○	○	—	04H

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 each time another 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 (U12326EJ3V0UM00)**.

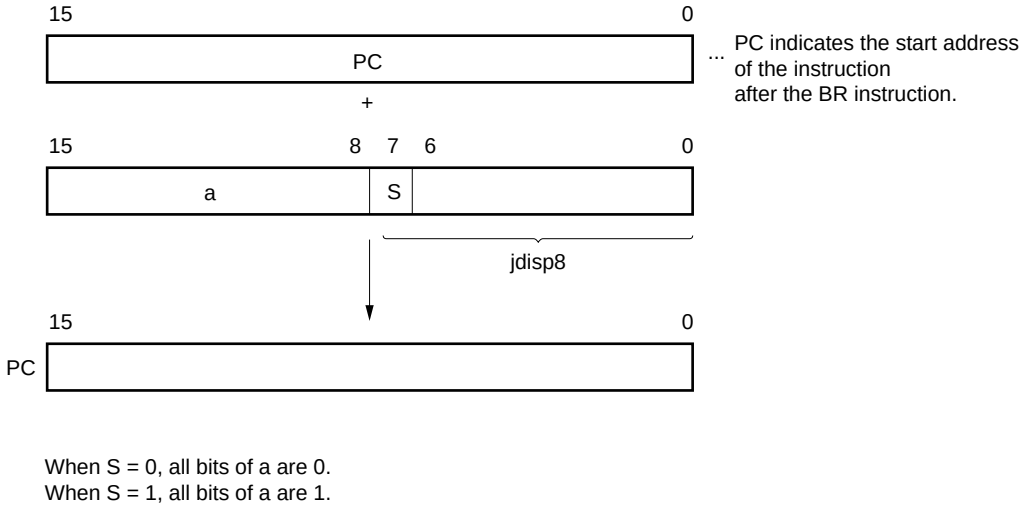
3.3.1 Relative addressing

The value obtained by adding 8-bit immediate data (displacement value: jdisp8) of an instruction code to the start address of the following instruction is transferred to the program counter (PC) and branched.

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 3-11: 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

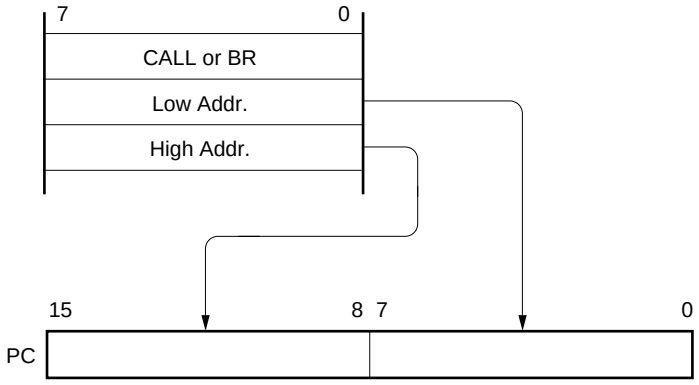
Immediate data in the instruction word is transferred to the program counter (PC) and branched.

This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed.

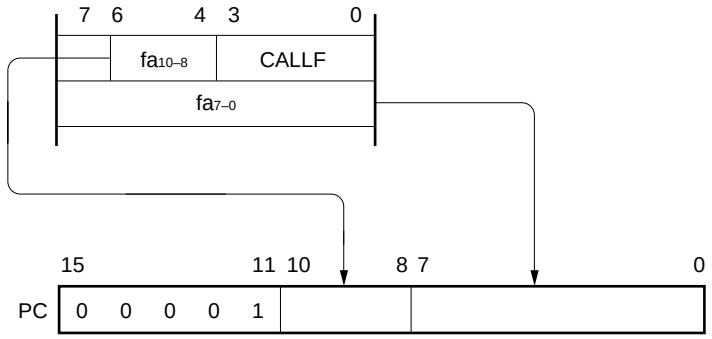
CALL !addr16 and BR !addr16 instructions can branch to all the memory space.

CALLF !addr11 instruction branches to the area from 0800H to 0FFFH.

Figure 3-12: Immediate Addressing



In the case of CALL !addr16 and BR !addr16 instructions



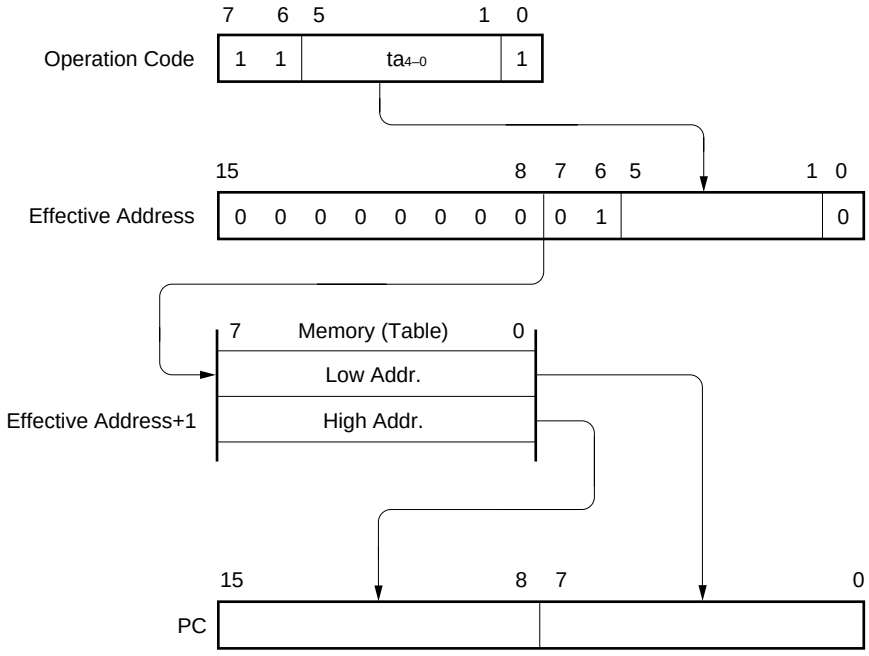
In the case of CALLF !addr11 instruction

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.

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 3-13: Table Indirect Addressing

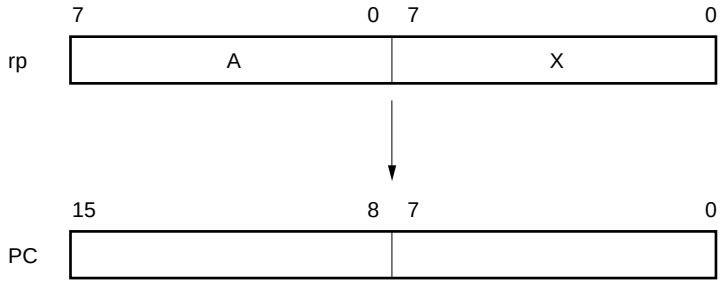


3.3.4 Register addressing

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 3-14: Register Addressing



3.4 Operand Address 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.

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 x 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

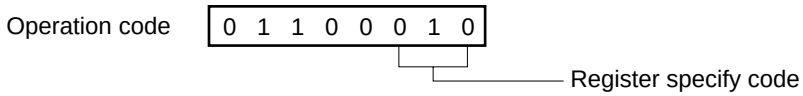
Table 3-5: Register Addressing

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

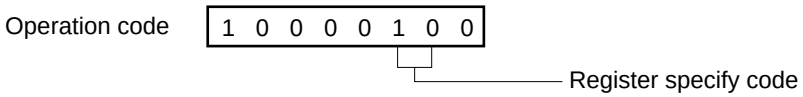
'r' and 'rp' can be described with function names (X, A, C, B, E, D, L, H, AX, BC, DE and HL) as well as absolute names (R0 to R7 and RP0 to RP3).

Description example

Figure 3-15: Register Addressing



MOV A, C; when selecting C register as r



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.

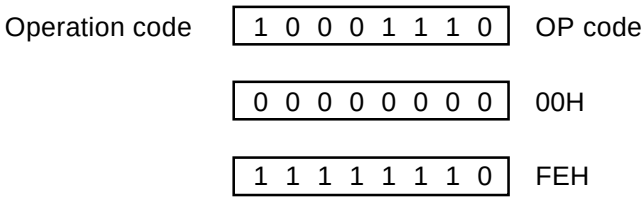
Operand format

Table 3-6: Direct Addressing

Identifier	Description
addr16	Label or 16-bit immediate data

Description example

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



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 a special function register (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 3-16 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)

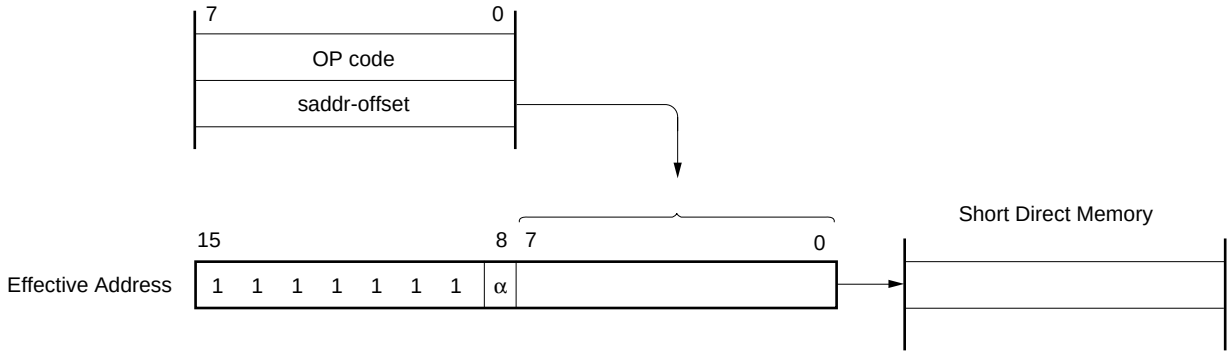
Description example

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

Operation code	0 0 0 1 0 0 0 1	OP code
	0 0 1 1 0 0 0 0	30H (saddr-offset)
	0 1 0 1 0 0 0 0	50H (immediate data)

Illustration

Figure 3-16: Short Direct Addressing



When 8-bit immediate data is 20H to FFH, α = 0

When 8-bit immediate data is 00H to 1FH, α = 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)

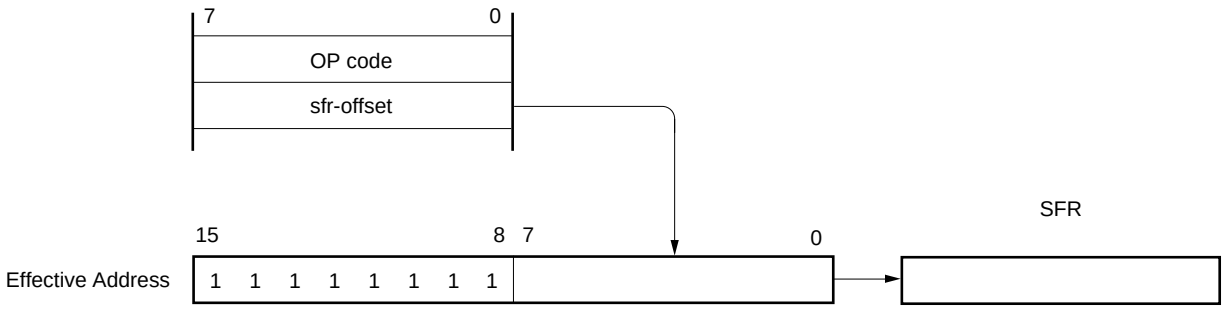
Description example

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

Operation code	1 1 1 1 0 1 1 0	OP code
	0 0 1 0 0 0 0 0	20H (sfr-offset)

Illustration

Figure 3-17: Special-Function Register (SFR) Addressing



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]

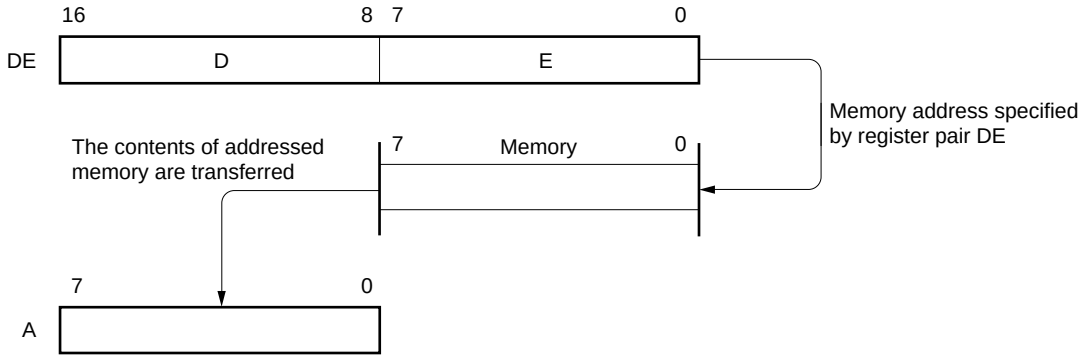
Description example

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

Operation code	1 0 0 0 0 1 0 1

Illustration

Figure 3-18: Special-Function Register (SFR) Addressing



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]

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]

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.

Description example

In the case of PUSH DE

Operation code	1	0	1	1	0	1	0	1

[Memo]

Chapter 4 EEPROM

4.1 EEPROM Functions

The μPD780949 Subseries contain on-chip 256 x 8-bit EEPROM (Electrically Erasable PROM) in addition to internal high-speed RAM, as data memory.

EEPROM differs from ordinary RAM in that its contents are saved even after power is cut off. Moreover, unlike EPROM, its contents can be erased electrically without using UV light. For this reason, it is suitable for applications where the setting values of the odometer and trip meter on the dash board are saved, etc.

EEPROM can be manipulated with 8-bit memory manipulation instructions.

The EEPROM contained on-chip in the μPD780949 Subseries has the following features.

(1) Written contents are saved even when the power is cut off.
(2) Can be manipulated with 8-bit memory manipulation instructions in the same way as ordinary RAM.
(3) Erasure and writing is performed in the time set with EWCS0 and EWCS1 (EEPROM write control register (EEWC) bits 4 and 5) (see Figure 4-2). Therefore, the write time control software load is reduced. Moreover, during writing, instructions other than instructions related to EEPROM writing and reading can also be executed.
 • Rewrite frequency per 1 chip : 100.000 times

Caution: The values shown above are target values. These values are subject to change without notice, so please contact your NEC sales representative for the latest value before designing.

(4) When write is completed, interrupt request signal (INTWE) is issued.
(5) The write enabled/disabled status can be checked with EWST (EEPROM write control register (EEWC) bit 1).
(6) No code fetch is possible.
(7) No write operation is available in subsystem clock mode.

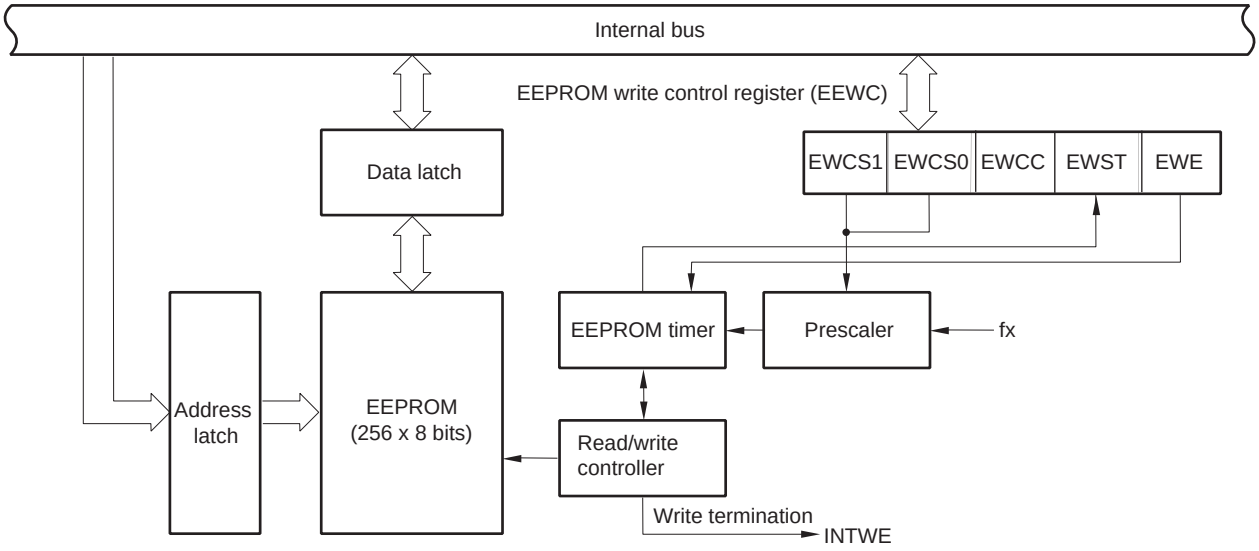
Note: The EEPROM is only available in the μPD780949, μPD78F0949 and not in the μPD780948, μPD78F0948.

4.2 EEPROM Configuration

EEPROM is composed of EEPROM itself and a control area.

The control area consists of the EEPROM write control register (EEWC) that controls EEPROM writing, and an area that generates an interrupt request signal (INTWE) upon detecting write termination.

Figure 4-1: EEPROM Block Diagram



4.3 EEPROM Control Register

EEPROM is controlled with the EEPROM write control register (EEWC).
EEWC is set with either a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets EEWC to 00H.

Figure 4-2: EEPROM Write Control Register (EEWC) Format

Symbol	7	6	5	4	3	2	1	0
EEWC	0	0	EWCS1	EWCS0	0	EWCC	EWST	EWE

EWCS1	EWCS0	EEPROM Write Time
0	1	$(2^{15} + 8 \times 2^{7})/fx$ (at 4 MHz)
1	0	$(2^{16} + 8 \times 2^{8})/fx$ (at 8 MHz)
1	1	$(2^{17} + 8x\ 2^{9})/fx$ (at 8 MHz)
Other than above		Setting prohibited

EWCC	EEPROM Operation Control
0	Operating mode
1	EEPROM stop

EWST	EEPROM Write Status
0	Not currently writing to EEPROM (EEPROM write/read is enabled. However, when EWE = 0, write is disabled)
1	Currently writing to EEPROM (EEPROM write/read is disabled)

EWE	EEPROM Write Operation Control
0	EEPROM write disabled
1	EEPROM write enabled

Cautions:
1. **When EWE is cleared (to 0) during EEPROM writing, writing is immediately interrupted. Data that was being written becomes undefined. Be sure to clear EWE before stopping the main system clock during the write period.**
2. **After EWCC is cleared (to 0), set a wait time of 20 μs or more by software to read EEPROM contents.**
3. **Be sure to check that EWST is 0 before performing EEPROM access.**
4. **Bits 3, 6, and 7 must be set to 0.**

4.4 EEPROM Reading

Reading of EEPROM data is performed with the following procedure.

<1> Check that EWST (EEPROM write control register (EEWC) bit 1) is 0 (EEPROM writing is not in progress).
<2> Execute read instruction.

Cautions: 1. Before reading, be sure to check that EWST is 0. If an EEPROM read instruction is executed during EEPROM write, read values are undefined.
2. If reading EEPROM contents immediately after changing EWCC (EEPROM write control register (EEWC) bit 2) from 1 to 0, set a wait time of at least 20 μs by software. If no wait time is set, the correct values cannot be read.

4.5 EEPROM Writing

Data writing to EEPROM is performed with the following procedure.

<1> Check that EWST (EEPROM write control register (EEWC) bit 1) is 0 (EEPROM writing is not in progress).
<2> Set the write time with EWCS0 and EWCS1 (EEWC bits 4 and 5).
<3> Set EWE (EEWC bit 0) to 1 (EEPROM writing enabled).
<4> Execute write instruction.

If performing several write operations in succession, perform the next write operation after the current write operation has been completed. The following methods can be used for write termination and time control.

(1) Method using write termination interrupt request (INTWE)
After writing 1 data, wait for generation of write termination interrupt request while processing other than write is performed. When write termination interrupt request is generated, start next write operation.

(2) Method using write status flag (EWST)
Poll EWST (EEPROM write control register (EEWC) bit 1), and wait for EWST to become 0. When EWST becomes 0, start the next write operation.

4.6 EEPROM Control-Related Interrupt

EEPROM write termination interrupt request (INTWE) is generated from EEPROM.
INTWE is an interrupt request generated upon termination of EEPROM writing.
This interrupt request is generated when the time set with EWCS0 and EWCS1 (EEPROM write control register (EEWC) bits 4 and 5) has elapsed.
When this interrupt request is generated, data writing to EEPROM is terminated, indicating that writing of the next data is enabled.

[Memo]

Chapter 5 Port Functions

5.1 Port Functions

The μPD780949 subseries units incorporate eight input ports and eighty-six input/output ports. Figure 5-1 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 5-1: Port Types

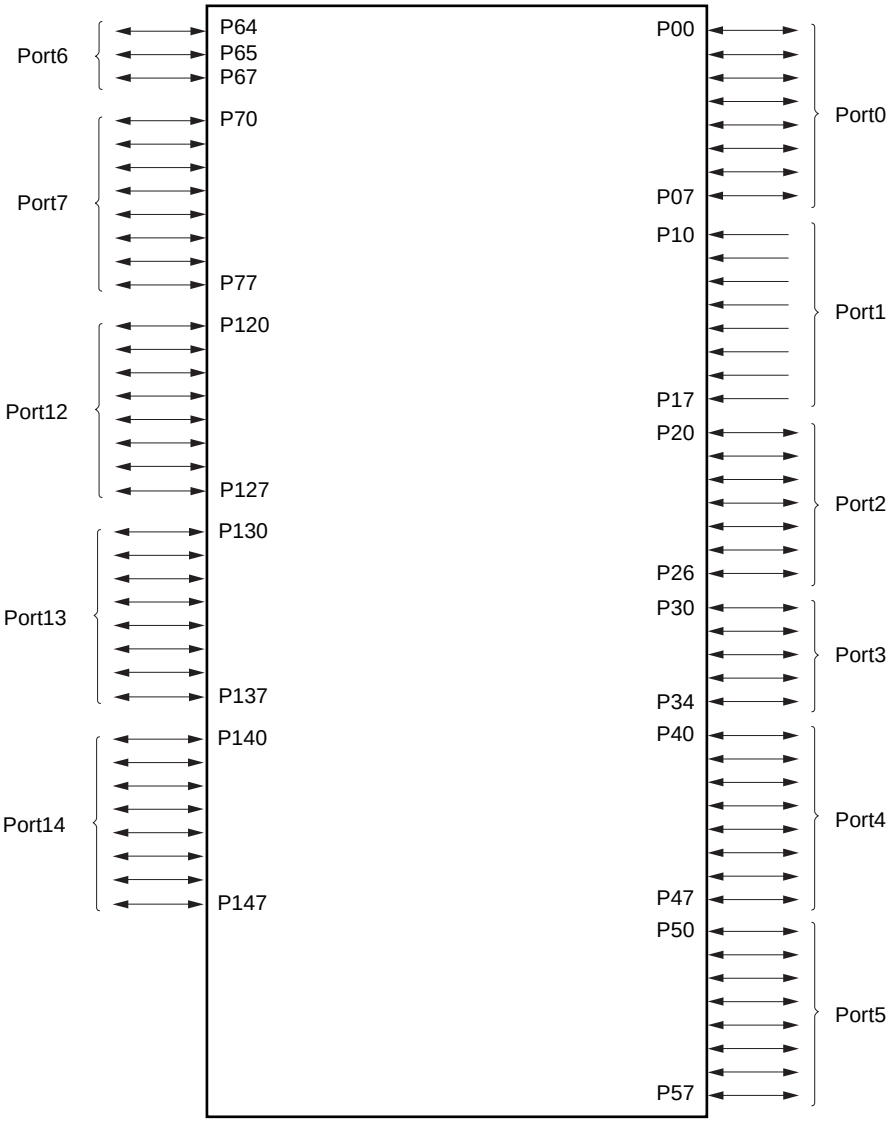


Table 5-1: Pin Input/Output Types (1/2)

Input / Output	Pin Name	Function	Alternate Function	After Reset
Input / Output	P00	Port 0 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	INTP0	Input
	P01		INTP1	Input
	P02		INTP2	Input
	P03		INTP3/T2P0	Input
	P04		INTP4/TI01	Input
	P05		TI00/TO0	Input
	P06		TI50/TO50	Input
	P07		TI51/TO51	Input
Input	P10-P17	Port 1 8 bit input port Input mode can be specified bit-wise	ANI0-ANI7	Input
Input / Output	P20	Port 2 7 bit input/output port Input / output mode can be specified bit-wise	SI0	Input
	P21		SO0	Input
	P22		/SCK0	Input
	P23		SI/SO1	Input
	P24		/SCK1	Input
	P25		RxD	Input
	P26		TxD	Input
Input/ Output	P40-P47	Port 4 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	AD0-AD7	Input
Input/ Output	P50-P57	Port 5 8 bit input / output port Input / output mode can be specified bit-wise This port can be used in External Memory Expansion Mode with the 4, 6 or 8 bit address by setting the Memory Expansion Mode Register Not for external memory expansion used ports can be used either for LCD or port function	A8/S39-A15/S32	Input
Input / Output	P64	Port 6 3 bit input / output port input / output mode can be specified bit-wise	/RD	Input
	P65		/WR	Input
	P67		ASTB	Input

Table 5-1: Pin Input/Output Types (2/2)

Input / Output	Pin Name	Function	Alternate Function	After Reset
Input/ Output	P70-P77	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 This port can be used as a segment signal output port or an I/O port in 1 bit units by setting port function	S31-S24	Input
Input/ Output	P120-P127	Port 12 8 bit input / output port Input / output mode can be specified bit-wise This port can be used as a segment signal output port or an I/O port in 8 bit units by setting LCD control register	S23-S16	Input
Input/ Output	P130-P137	Port 13 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 This port can be used as a segment signal output port or an I/O port in 8 bit units by setting LCD control register	S15-S8	Input
Input/ Output	P140-P147	Port 14 8 bit input / output port Input / output mode can be specified bit-wise This port can be used as a segment signal output port or an I/O port in 8 bit units by setting LCD control register	S7-S0	Input

5.2 Port Configuration

A port consists of the following hardware:

Table 5-2: Port Configuration

Item	Configuration
Control register	Port mode register (PMm: m = 0, 2 to 7, 12, 13, 14) Pull-up resistor option register (PUm: m = 0, 4, 7, 13) Port function register (PFm: m = 2, 5, 7, 12, 13, 14) Memory expansion mode register (MEM) Note
Port	Total: 79 ports
Pull-up resistor	Mask ROM versions ..... Total: 79 pins (software-specifiable for 32 pins) μPD78F048..... Total: 79 pins

Note: Memory expansion mode register specifies port 4, port 5 and port 6 as port mode.

5.2.1 Port 0

Port 0 is an 8-bit input/output port with output latch. P00 to P07 pins can specify the input mode/output mode in 1-bit units with the port mode register 0 (PM0). When P00 to P07 pins are used as input ports, a pull-up resistor can be connected to them bitwise with a pull-up resistor option register (PUO).

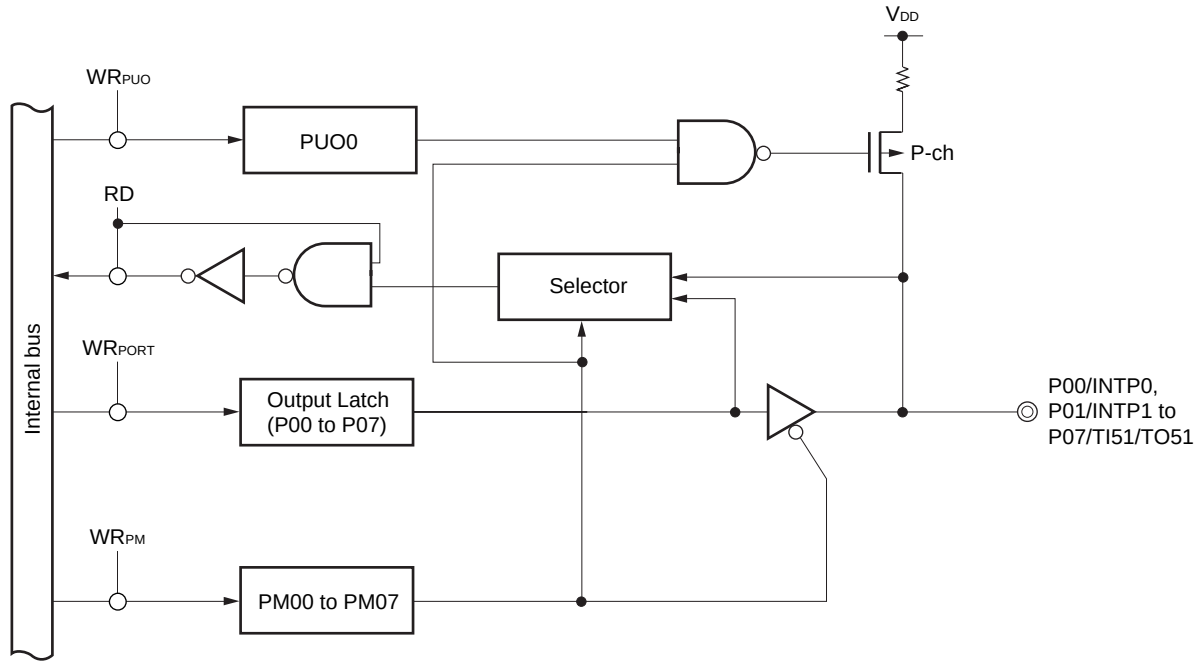
Dual-functions include external interrupt request input, external count clock input to the timer and timer output.

RESET input sets port 0 to input mode.

Figure 5-2 shows block diagram of port 0.

Caution: Because port 0 also serves for external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. Thus, when the output mode is used, set the interrupt mask flag to 1.

Figure 5-2: P00 to P07 Configurations



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 0 read signal
WR : Port 0 write signal

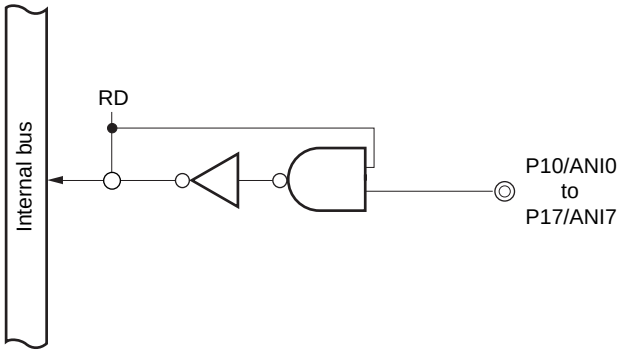
5.2.2 Port 1

Port 1 is an 8-bit input only port.

Dual-functions include an A/D converter analog input.

Figure 5-3 shows a block diagram of port 1.

Figure 5-3: P10 to P17 Configurations



RD : Port 1 read signal

5.2.3 Port 2

Port 2 is an 7-bit input/output port with output latch. P20 to P26 pins can specify the input mode/output mode in 1-bit units with the port mode register 2 (PM2).

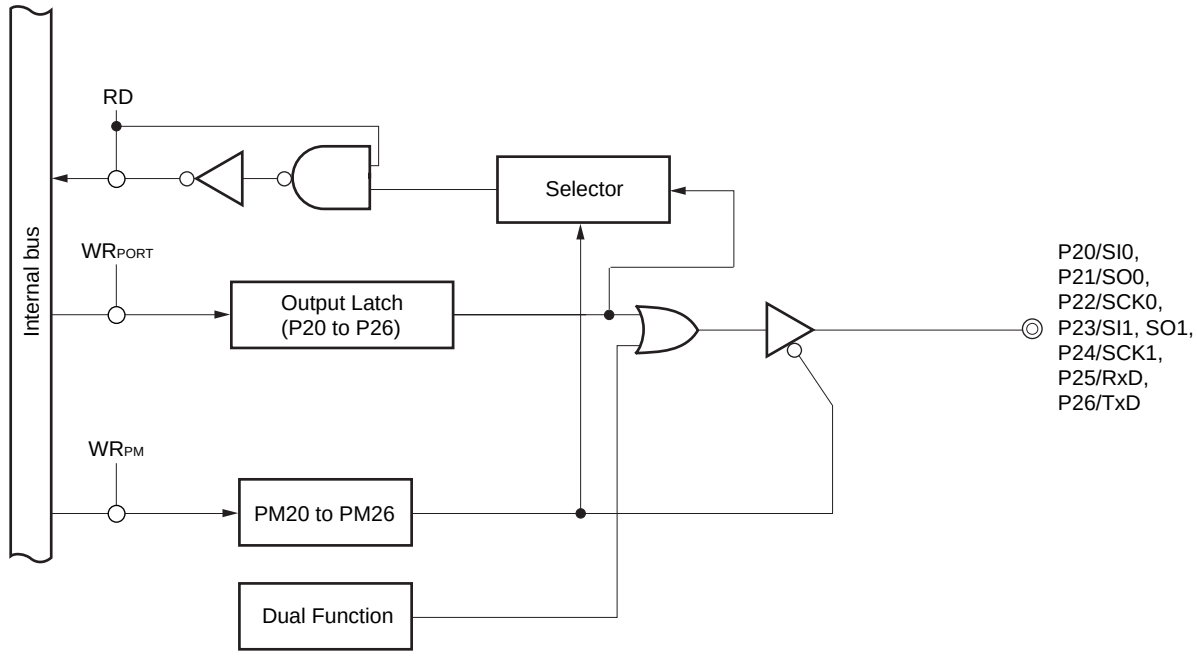
Dual-functions include serial interface data input/output, clock input/output. When P20 to P24 pins are used as output ports, the output buffer is selectable between CMOS-type or N-channel open drain.

$\overline{\text{RESET}}$ input sets port 2 to input mode.

Figure 5-4 shows a block diagram of port 2.

Caution: When used as a serial interface, set the input/output and output latch according to its functions. For the setting method, refer to the Serial Operating Mode Register Format and Serial Operating Mode Register Format.

Figure 5-4: P20 to P26 Configurations



PM : Port mode register
RD : Port 2 read signal
WR : Port 2 write signal

5.2.4 Port 3

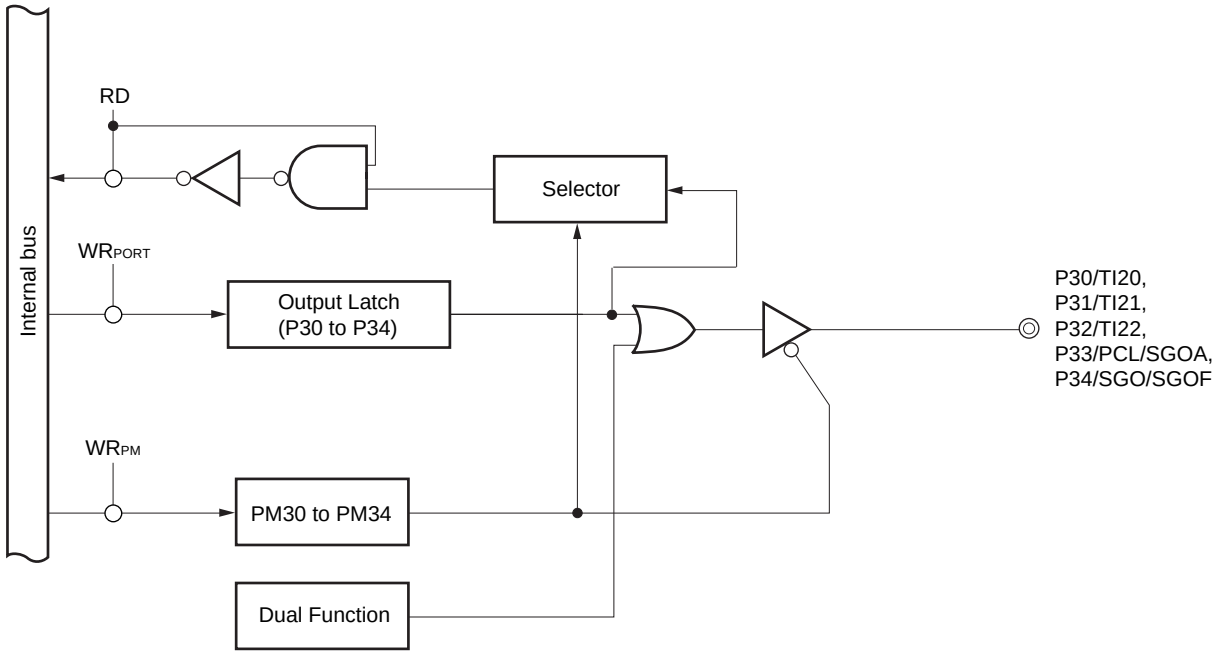
Port 3 is an 5-bit input/output port with output latch. P30 to P34 pins can specify the input mode/output mode in 1-bit units with the port mode register 3 (PM3).

Dual-functions include timer input, clock output and sound generator output.

$\overline{\text{RESET}}$ input sets port 3 to input mode.

Figure 5-5 shows a block diagram of port 3.

Figure 5-5: P30 to P34 Configurations



PM : Port mode register
RD : Port 3 read signal
WR : Port 3 write signal

5.2.5 Port 4

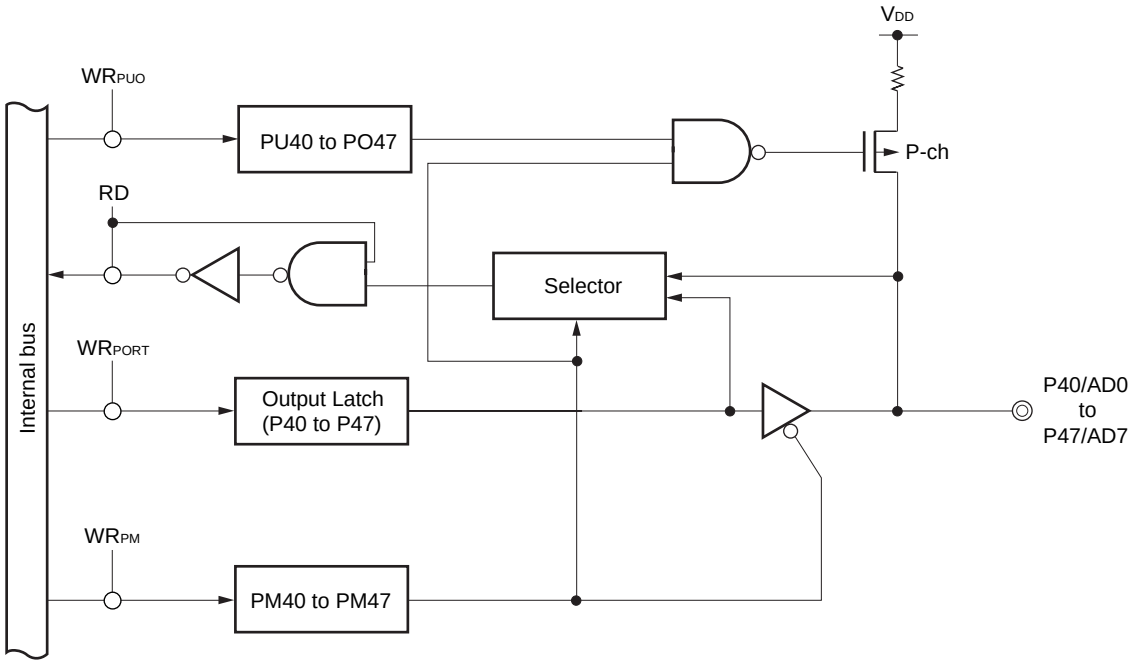
Port 4 is an 8-bit input/output port with output latch. P40 to P47 pins can specify the input mode/output mode in 8-bit units with the memory expansion mode register (MM). When P40 to P47 pins are used as input ports, an on-chip pull-up resistor can be connected to them bitwise with the pull-up resistor option register (PU4).

Dual-functions include address/data bus function in external memory expansion mode.

$\overline{\text{RESET}}$ input sets port 4 to input mode.

Figure 5-6 shows a block diagram of port 4.

Figure 5-6: P40 to P47 Configurations



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 4 read signal
WR : Port 4 write signal

5.2.6 Port 5

Port 5 is an 8-bit input/output port with output latch. P50 to P57 pins can specify the input mode/output mode in 1-bit units with the port mode register 5 (PM5).

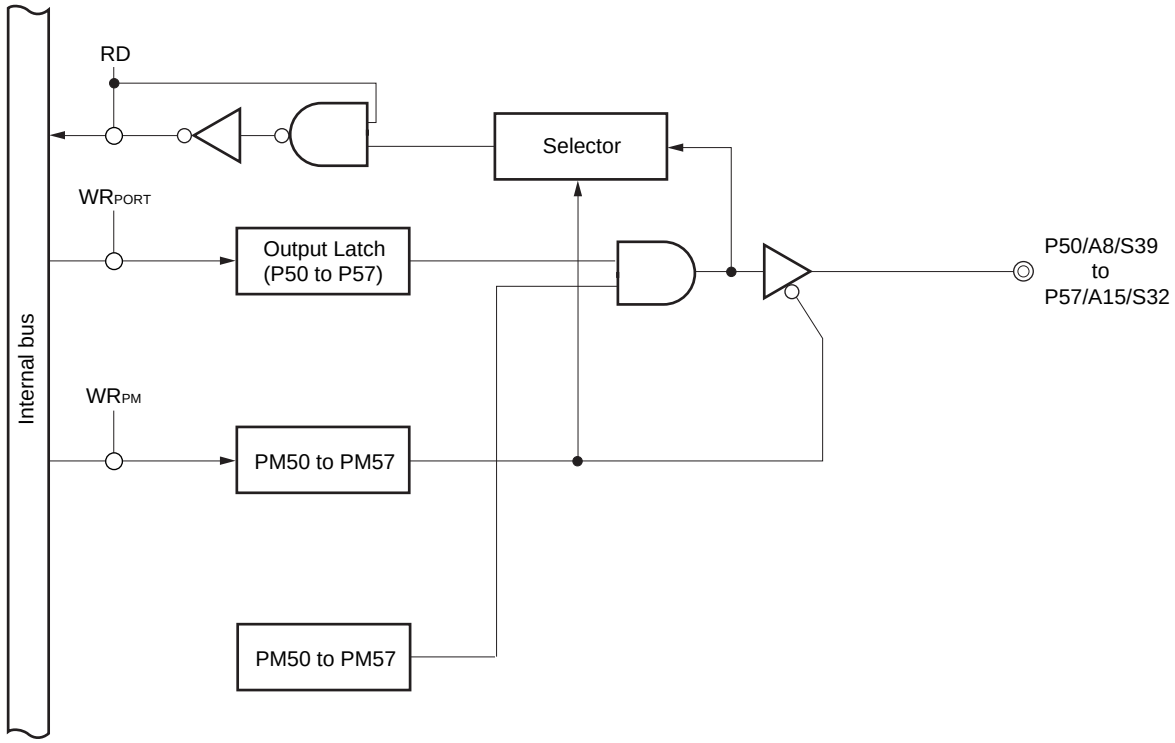
Dual-functions include address bus function in external memory expansion mode and segment signal outputs of LCD controller driver.

RESET input sets port 5 to input mode.

Figure 5-7 shows a block diagram of port 5.

Caution: See port 7 with change of PE7 to PF5.

Figure 5-7: P50 to P57 Configurations



PM : Port mode register
RD : Port 5 read signal
WR : Port 5 write signal

5.2.7 Port 6

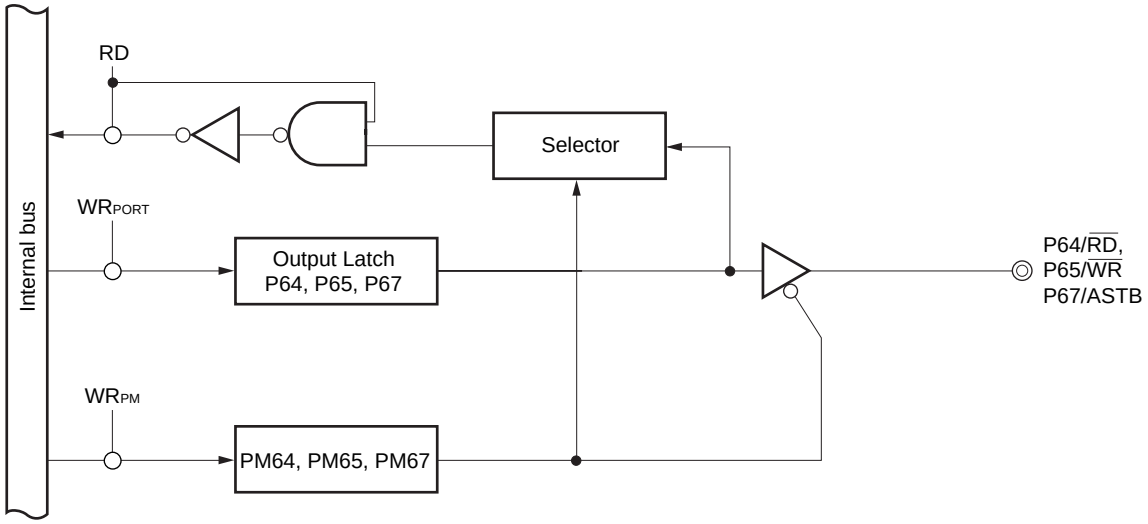
Port 6 is a 3-bit input/output port with output latch. P64, P65, P67 pins can specify the input mode/output mode in 1-bit units with the port mode register 6 (PM6).

Dual-functions include the control signal output function in external memory expansion mode.

$\overline{\text{RESET}}$ input sets port 6 to input mode.

Figure 5-8 shows block diagrams of port 6.

Figure 5-8: P64, P65 and P67 Configurations



PM : Port mode register
RD : Port 6 read signal
WR : Port 6 write signal

5.2.8 Port 7

This is a 8-bit input/output port with output latches. Input mode/output mode can be specified in 1-bit units with a port mode register 7. When pins P70 to P77 are used as input port pins, an on-chip pull-up resistor can be connected bitwise with a pull-up resistor option register (PU7).

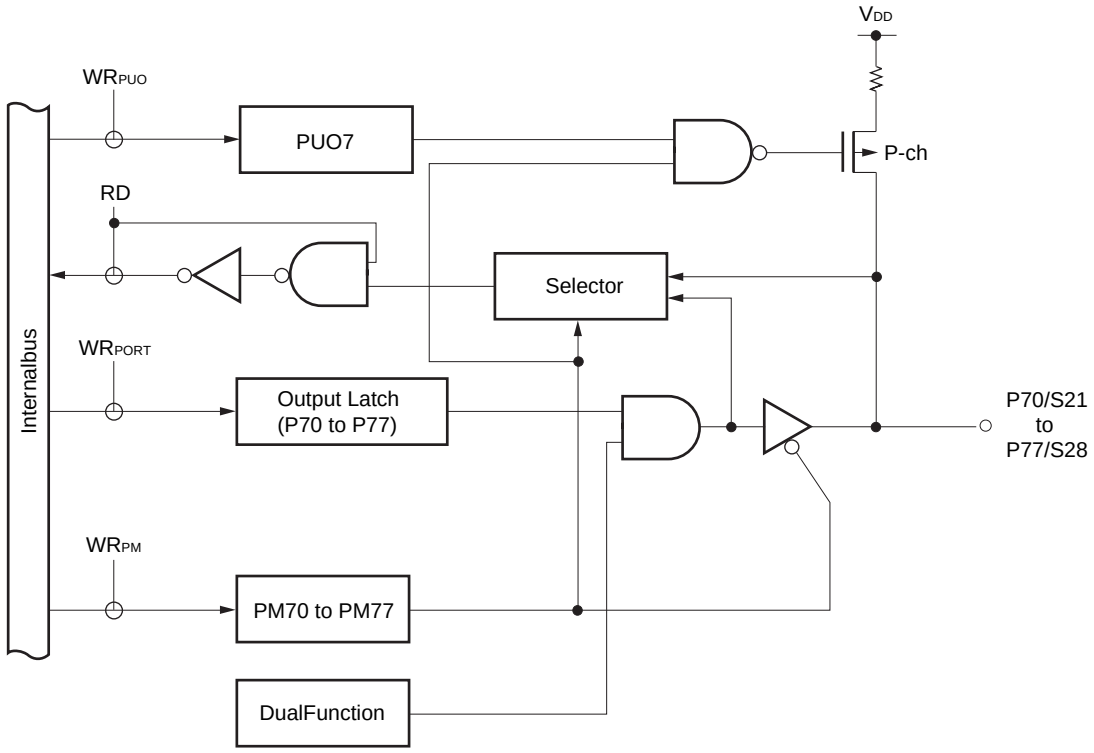
Dual-functions include segment signal output of LCD controller/driver.

$\overline{\text{RESET}}$ input sets the input mode.

Port 7 block diagram is shown in Figure 5-9.

Caution: When used as segment lines, set the port function PF7 according to its functions.

Figure 5-9: P70 to P77 Configurations



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 7 read signal
WR : Port 7 write signal

5.2.9 Port 12

This is an 8-bit input/output port with output latches. Input mode/output mode can be specified in 1-bit units with the port mode register 12.

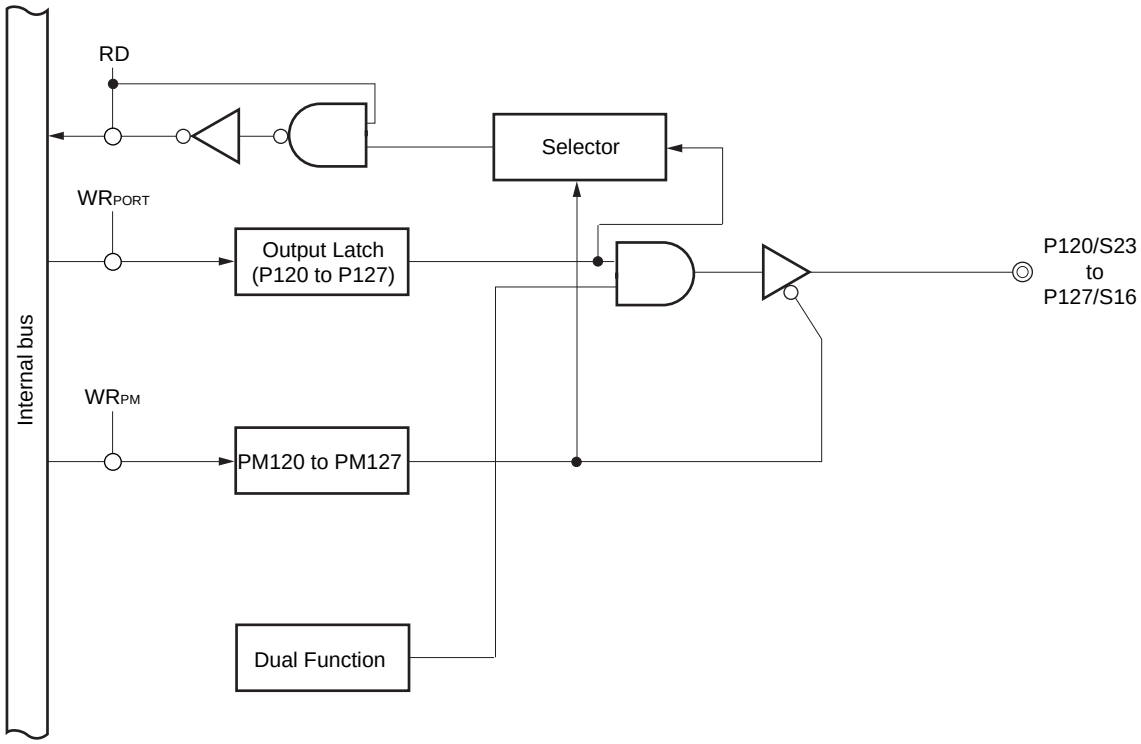
These pins are dual function pin and serve as segment signal output of LCD controller/driver.

$\overline{\text{RESET}}$ input sets the input mode.

The port 12 block diagram is shown in Figure 5-10.

Caution: When used as segment lines, set the port function (PF12) according to its function.

Figure 5-10: P120 to P127 Configurations



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 12 read signal
WR : Port 12 write signal

5.2.10 Port 13

This is a 8-bit input/output port with output latches. Input mode/output mode can be specified in 1-bit units with the port mode register 13. When pins P130 and P137 are used as input port pins, an on-chip pull-up resistor can be connected bitwise with a pull-up resistor option register (PU13).

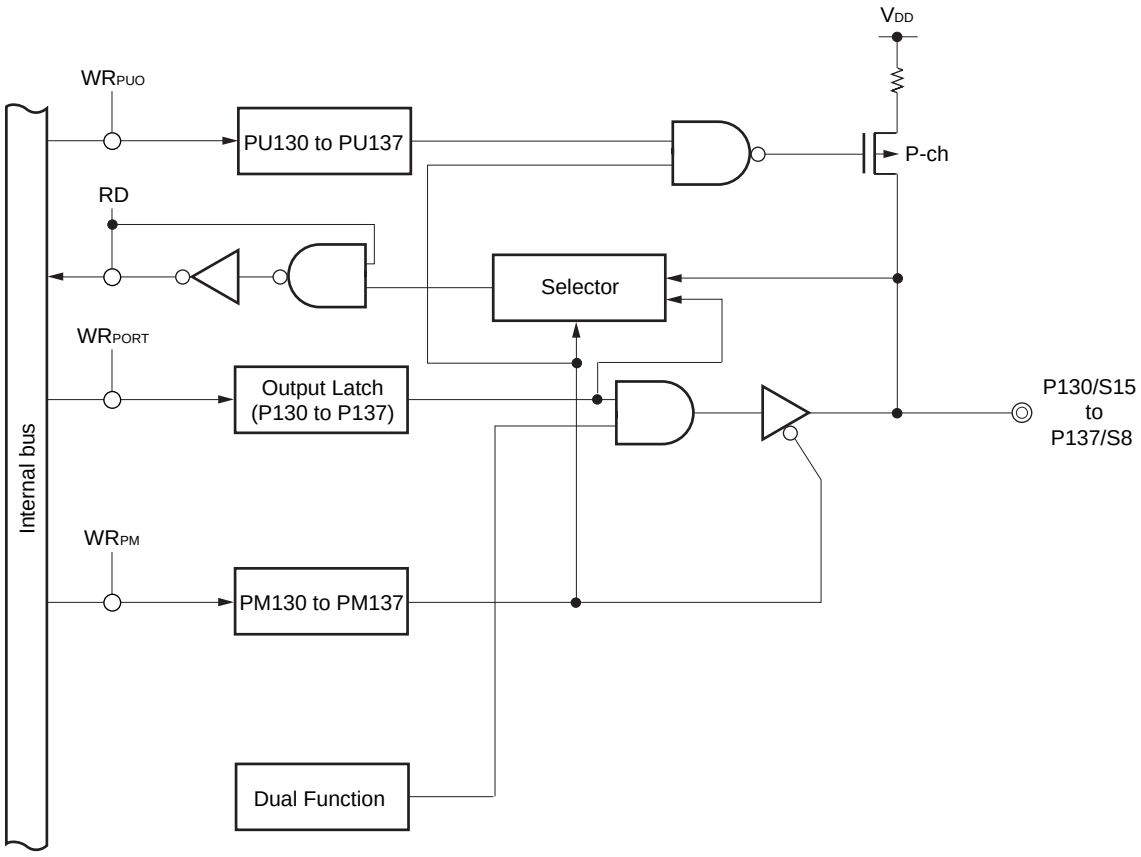
These pins are dual function pin and serve as segment signal output of LCD controller/driver.

$\overline{\text{RESET}}$ input sets the input mode.

The port 13 block diagram is shown in Figure 5-11.

Caution: See port 12 with change to PF12.

Figure 5-11: P130 to P137 Configurations



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 13 read signal
WR : Port 13 write signal

5.2.11 Port 14

This is a 8-bit input/output port with output latches. Input mode/output mode can be specified in 1-bit units with the port mode register 14.

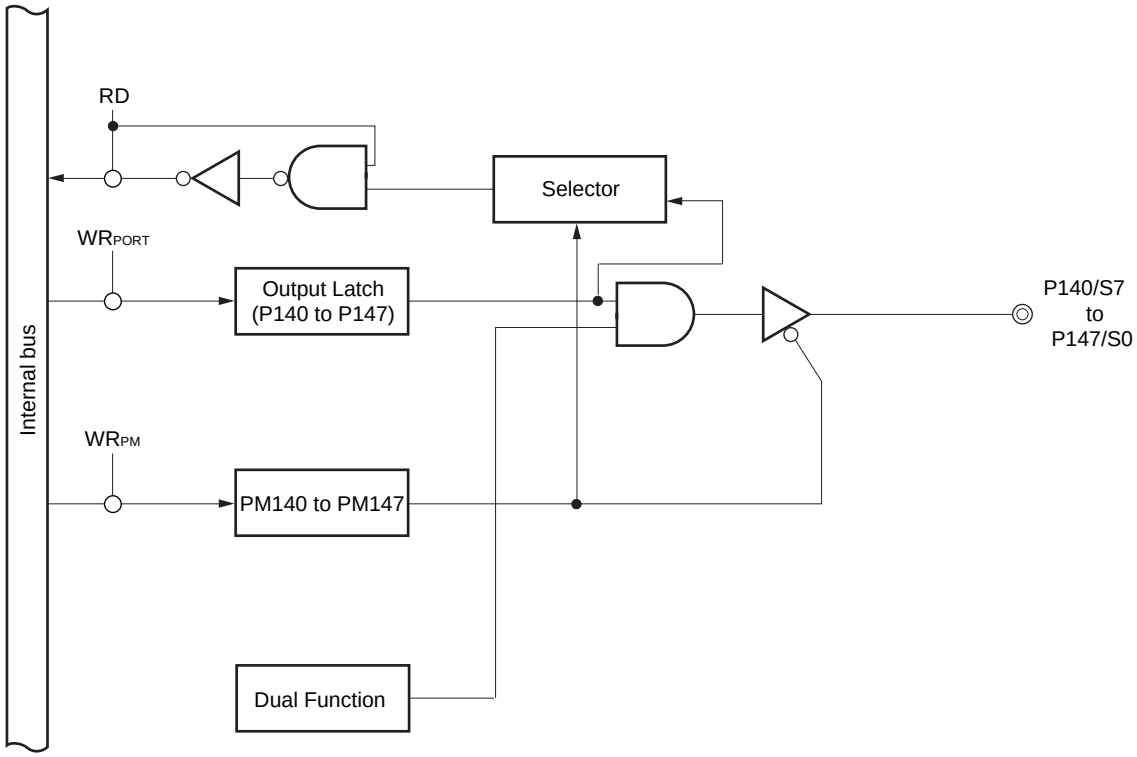
These pins are dual function pin and serve as segment signal output of LCD controller/driver.

$\overline{\text{RESET}}$ input sets the input mode.

The port 14 block diagram is shown in Figure 5-12.

Caution: See port 7 with change of PF7 to PF14.

Figure 5-12: P140 to P147 Configurations



PM : Port mode register
RD : Port 14 read signal
WR : Port 14 write signal

5.3 Port Function Control Registers

The following four types of registers control the ports.

- Port mode registers (PM0 to PM7, PM12, PM13, PM14)
- Pull-up resistor option register (PUm : m = 0, 4, 7, 13)
- Port function registers (PFm : m = 2, 5, 7, 14)
- Memory expansion mode register (MEM)

(1) Port mode registers (PM0 to PM7, PM12, PM13, PM14)

These registers are used to set port input/output in 1-bit units.

PM0 to PM7, PM12, PM13 and PM14 are independently set with a 1-bit or 8-bit memory manipulation instruction

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

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

Cautions: 1. Pins P10 to P17 are input-only pins.

2. As port 0 has an alternate function as external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

3. The memory expansion mode register specifies P40 to P47, P50 to P57 and P64, P65 and P67 as port pins.

Figure 5-13: Port Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM2	1	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	1	1	1	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM6	PM67	1	PM65	PM64	1	1	1	1	FF26H	FFH	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	FF27H	FFH	R/W
PM12	PM127	PM126	PM125	PM124	PM123	PM122	PM121	PM120	FF2CH	FFH	R/W
PM13	PM137	PM136	PM135	PM134	PM133	PM132	PM131	PM130	FF2DH	FFH	R/W
PM14	PM147	PM146	PM145	PM144	PM143	PM142	PM141	PM140	FF2EH	FFH	R/W

PMmn	PMmn Pin Input/Output Mode Selection (m = 0 - 7, 12, 13, 14; n = 0 – 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option register

This register is used to set whether to use an internal pull-up resistor at each port or not. A pull-up resistor is internally used at bits which are set to the input mode at a port where on-chip pull-up resistor use has been specified with PU0, PU4, PU7 and PU13. No on-chip pull-up resistors can be used to the bits set to the output mode, irrespective of PU0, PU4, PU7 and PU13 setting.

PU0, PU4, PU7 and PU13 are set with a 1-bit or 8-bit memory manipulation instruction.

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

Caution: When ports P0, P4, P7, and P13 pins are used as dual-function pins, an on-chip pull-up resistor cannot be used even if 1 is set in PUm (m = 0, 4, 7, or 13).

Figure 5-14: Pull-Up Resistor Option Register (PU0, PU4, PU7 and PU13) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PU0	PU07	PU06	PU05	PU04	PU03	PU02	PU01	PU00	FF30H	00H	R/W
PU4	PU47	PU46	PU45	PU44	PU43	PU42	PU41	PU40	FF34H	00H	R/W
PU7	PU77	PU76	PU75	PU74	PU73	PU72	PU71	PU70	FF37H	00H	R/W
PU13	PU137	PU136	PU135	PU134	PU133	PU132	PU131	PU130	FF3DH	00H	R/W

PMmn	Pmn Pin Internal Pull-up Resistor Selection (m = 0, 4, 7, 13; n = 0 - 7)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

3) Port function register (PF2, PF5, PF7, PF12 to PF14)

This register is used to set the output buffer of port 2 (P20 to P24) and the LCD segment function of ports 5, 7, 12, 13, and 14.

PF2, PF5 and PF7 are set with an 1-bit or 8-bit manipulation instruction. PF12 to PF14 are set with an 8-bit manipulation instruction.

$\overline{\text{RESET}}$ input set this registors to 00H.

Figure 5-15: Port Function Register (PF2, PF5, PF7, PF12 to PF14) Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
PF2	0	0	0	PF24	PF23	PF22	PF21	PF20	FF52H	00H	R/W

PM2n	P2n Port Function Selection (n = 0 - 4)
0	Push pull output buffer
1	N-channel open drain output buffer

PF5	PF57	PF56	PF55	PF54	PF53	PF52	PF51	PF50	FF55H	00H	R/W
PF7	PF77	PF76	PF75	PF74	PF73	PF72	PF71	PF70	FF57H	00H	R/W
PF12	PF127	PF126	PF125	PF124	PF123	PF122	PF121	PF120	FF5CH	00H	R/W
PF13	PF137	PF136	PF135	PF134	PF133	PF132	PF131	PF130	FF5DH	00H	R/W
PF14	PF147	PF146	PF145	PF144	PF143	PF142	PF141	PF140	FF5EH	00H	R/W

PMmn	PFmn Port Function Selection (m = 5, 7, 12, 13, 14)
0	Port function
1	LCD segment function

Caution: For PF12 to PF14 it is only allowed to set 00h or FFh.

(4) Memory expansion mode register (MEM)

This register is used to set input/output of port 4, 5 and 6.

MM is set with a 1-bit or 8-bit memory manipulation instruction.

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

Figure 5-16: Memory Expansion Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
MEM	0	0	0	0	0	MM2	MM1	MM0	FF47H	00H	R/W

MM2	MM1	MM0	Single-chip/Memory Expansion Mode Selection		P40-P47, P50-P57, P64, P65, P67 Pin State				
					P40-P47	P50-P53	P54, P55	P56, P57	P64-P67
0	0	0	Single-chip mode		Port mode	Port mode			
0	1	1	Memory expansion mode	256-byte mode	AD0-AD7	Port mode			P64=RD P65=WR P67=ASTB
1	0	0		4K-byte mode		A8-A11	Port mode		
1	0	1		16K-byte mode			A12, A13	Port mode	
1	1	1		Full address mode [Note]				A14, A15	
Other than above			Setting prohibited						

Note: The full address mode allows external expansion for all areas of the 64-Kbyte address space, except the internal ROM, RAM, SFR, and use-prohibited areas.

5.4 Port Function Operations

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

5.4.1 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 pin.

Once data is written to the output latch, it is retained until data is written to the output latch again.

(2) Input 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. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined except for the manipulated bit.

5.4.2 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

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

5.4.3 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 are undefined, even for bits other than the manipulated bit.

[Memo]

Chapter 6 Clock Generator

6.1 Clock Generator Functions

The clock generator generates the clock to be supplied to the CPU and peripheral hardware. The following two types of system clock oscillators are available.

(1) Main system clock oscillator

This circuit oscillates at frequencies of 4 to 8.5 MHz. Oscillation can be stopped by executing the STOP instruction or setting the processor clock control register.

(2) Subsystem clock oscillator

The circuit oscillates at a typical frequency of 40 kHz. Oscillation cannot be stopped.

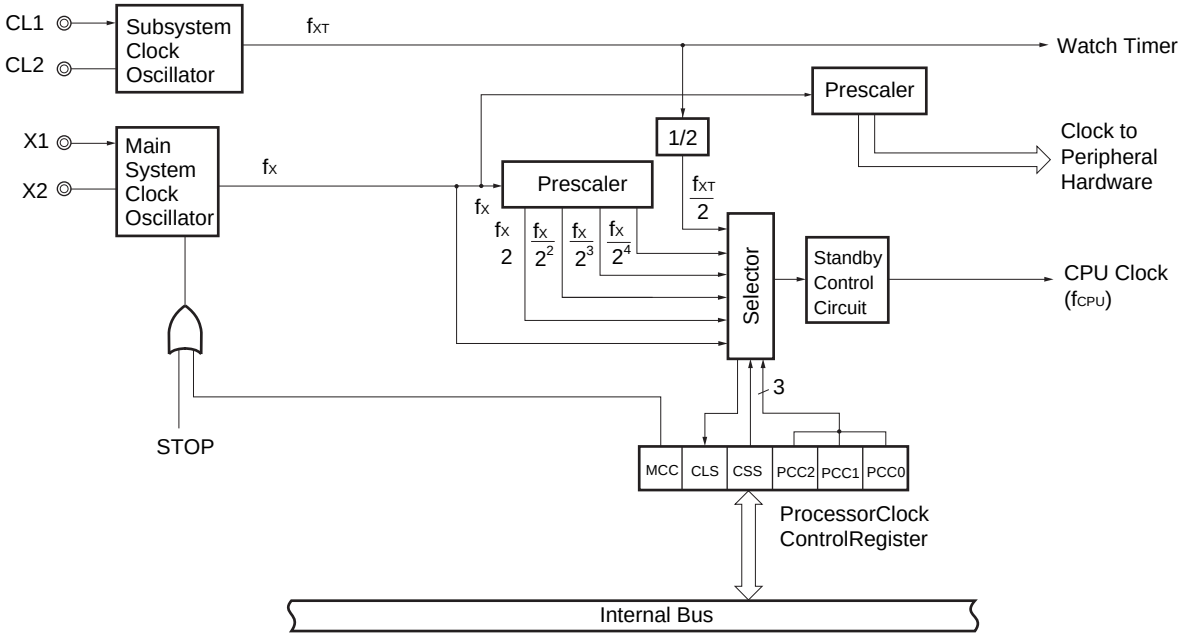
6.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 6-1: Clock Generator Configuration

Item	Configuration
Control register	Processor clock control register (PCC)
Oscillator	Main system clock oscillator Subsystem clock oscillator

Figure 6-1: Block Diagram of Clock Generator



6.3 Clock Generator Control Register

The clock generator is controlled by the processor clock control register (PCC).

(1) Processor clock control register (PCC)

The PCC selects a CPU clock and the division ratio, determines whether to make the main system clock oscillator operate or stop.

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

$\overline{\text{RESET}}$ input sets the PCC to 04H.

Figure 6-2: Processor Clock Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PCC	MCC	0	CLS	CSS	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W Note 1

R/W

CSS	PCC2	PCC1	PCC0	CPU Clock Selection (f_{CPU})
0	0	0	0	fx (0.25 μs)
	0	0	1	fx/2 (0.5 μs)
	0	1	0	$f_x/2^2$ (1 μs)
	0	1	1	$f_x/2^3$ (2 μs)
	1	0	0	$f_x/2^4$ (4 μs)
1	0	0	0	f_{XT}/2 (122 μs)
	0	0	1	
	0	1	0	
	0	1	1	
	1	0	0	
Other than above				Setting prohibited

R

CLS	CPU Clock Status
0	Main system clock
1	Subsystem clock

R/W

MCC	Main System Clock Oscillation Control
0	Oscillation possible
1	Oscillation stopped

Notes: **1.** Bit 5 is a read-only bit.

2. When the CPU is operating on the subsystem clock, MCC should be used to stop the main system clock oscillation. A STOP instruction should not be used.

Cautions: **1. Bit 3 must be set to 0.**

2. When external clock input is used MCC should not be set, because the X2 pin is connected to V_{DD} via a resistor.

Remarks: **1.** f_X : Main system clock oscillation frequency

2. f_{XT} : Subsystem clock oscillation frequency

3. Figures in parentheses indicate minimum instruction execution time: $2f_{CPU}$ when operating at f_X = 8.0 MHz or f_{XT} = 32.768 kHz.

6.4 System Clock Oscillator

6.4.1 Main system clock oscillator

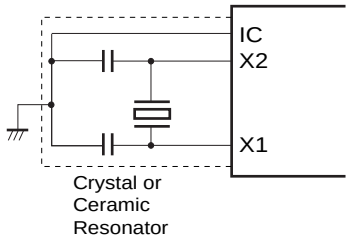
The main system clock oscillator oscillates with a crystal resonator or a ceramic resonator (standard: 8.0 MHz) connected to the X1 and X2 pins.

External clocks can be input to the main system clock oscillator. In this case, the clock signal to the X1 pin and an inversed phase clock signal to the X2 pin.

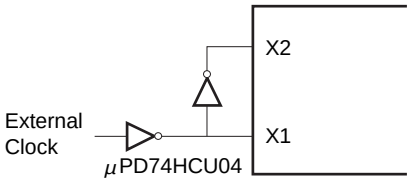
Figure 6-3 shows an external circuit of the main system clock oscillator.

Figure 6-3: External Circuit of Main System Clock Oscillator

(a) Crystal and ceramic oscillation



(b) External clock



Caution: Do not execute the STOP instruction and do not set MCC [bit 7 of processor clock control register (PCC)] to 1 if an external clock is input. This is because when the STOP instruction or MCC is set to 1, the main system clock operation stops and the X2 pin is connected to V_{DD1} via a pull-up resistor.

6.4.2 Subsystem clock oscillator

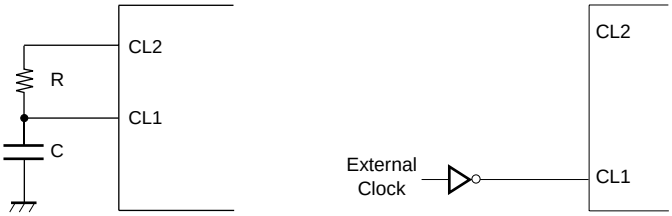
The subsystem clock oscillator oscillates with a RC-resonator (standard: 40kHz) connected to the CL1 and CL2 pins.

External clocks can be input to the subsystem clock oscillator. In this case, input a clock signal to the CL1 pin and open the CL2 pin.

Figure 6-4 shows an external circuit of the subsystem clock oscillator.

Figure 6-4: External Circuit of Subsystem Clock Oscillator

(a) RC oscillation **(b) External clock**



Cautions: 1. When an external clock is used for CAN, the CPU operation and the watch timer operation with subsystem clock are prohibited.

The setting of the CSS-bit (PCC-register) and the WTM 7-bit (WTM-register) to 1 is prohibited.

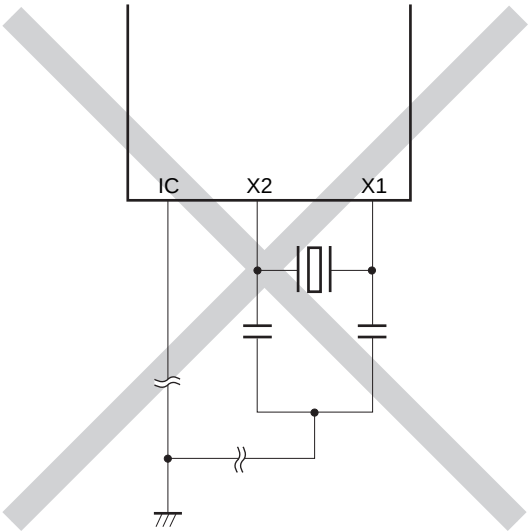
2. When using a main system clock oscillator and a subsystem clock oscillator, carry out wiring in the broken-line area in Figures 6-3 and 6-4 as follows to prevent any effects from wiring capacities.

- **Minimize the wiring length.**
- **Do not allow wiring to intersect with other signal conductors. Do not allow wiring to come near abruptly changing high current.**
- **Set the potential of the grounding position of the oscillator capacitor to that of V_{SS}. Do not ground to any ground pattern where high current is present.**
- **Do not fetch signals from the oscillator.**

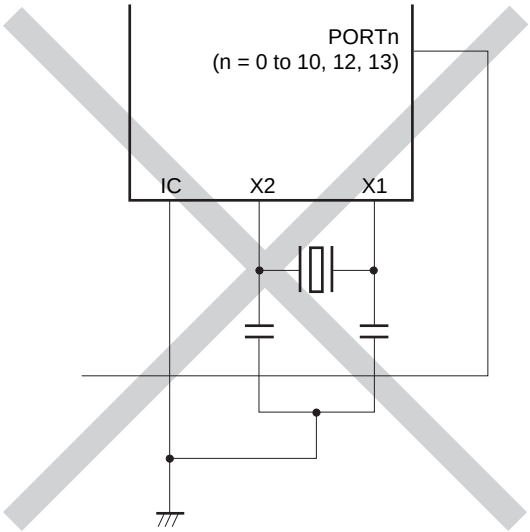
Take special note of the fact that the subsystem clock oscillator is a circuit with low-level amplification so that current consumption is maintained at low levels.
Figure 6-5 shows examples of oscillator having bad connection.

Figure 6-5: Examples of Oscillator with Bad Connection (1/3)

(a) Wiring of connection circuits is too long



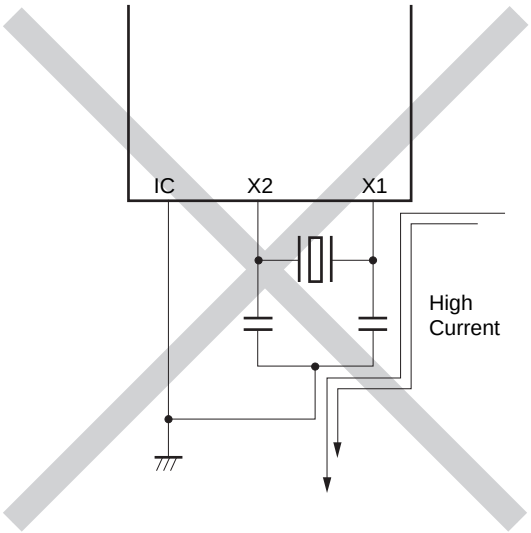
(b) A signal line crosses over oscillation circuit lines



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

Figure 6-5: Examples of Oscillator with Bad Connection (2/3)

(c) Changing high current is too near a signal conductor



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

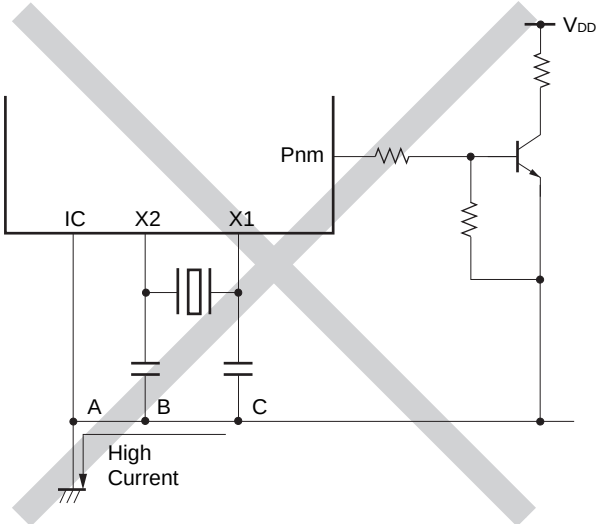
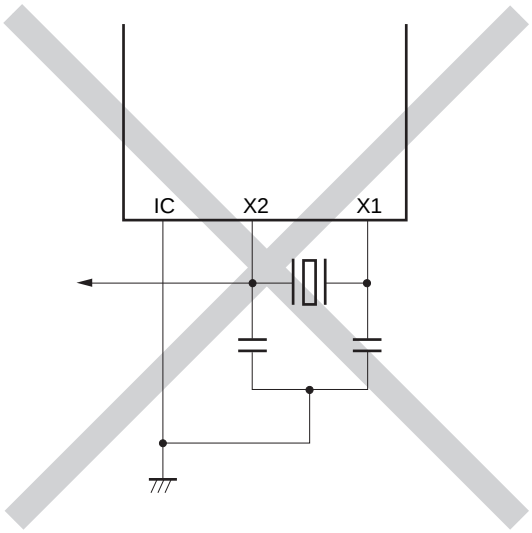
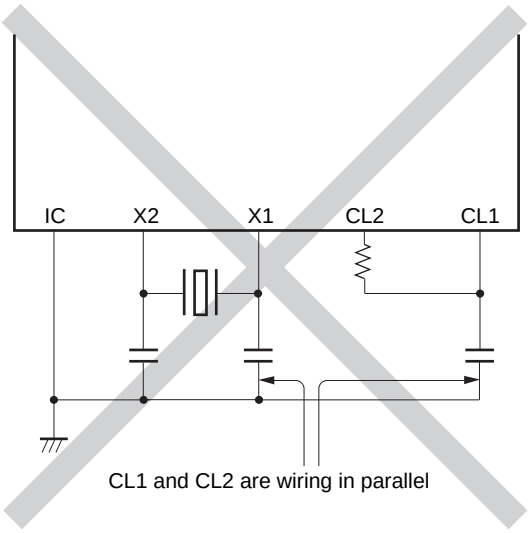


Figure 6-5: Examples of Oscillator with Bad Connection (3/3)

(e) Signals are fetched



(f) Signal conductors of the main and sub-system clock are parallel and near each other



Remark: When using a subsystem clock, replace X1 and X2 with CL1 and CL2, respectively.

Caution: In Figure 6-5 (f), CL1 and X1 are wired in parallel. Thus, the cross-talk noise of X1 may increase with CL1, resulting in malfunctioning. To prevent that from occurring, it is recommended to wire CL1 and X1 so that they are not in parallel, and to connect the IC pin between CL1 and X1 directly to V_{SS}.

6.4.3 When no subsystem clocks are used

If it is not necessary to use subsystem clocks for low power consumption operations and clock operations, connect the CL1 and CL2 pins as follows.

CL1: Connect to V_{DD} or GND
CL2: Open

6.5 Clock Generator Operations

The clock generator generates the following various types of clocks and controls the CPU operating mode including the standby mode.

- Main system clock f_X
- Subsystem clock f_{XT}
- CPU clock f_{CPU}
- Clock to peripheral hardware

The following clock generator functions and operations are determined with the processor clock control register (PCC).

(a) Upon generation of $\overline{\text{RESET}}$ signal, the lowest speed mode of the main system clock (4 μs when operated at 8.0 MHz) is selected (PCC = 04H). Main system clock oscillation stops while low level is applied to $\overline{\text{RESET}}$ pin.

(b) With the main system clock selected, one of the five CPU clock stages (f_X, $f_X/2$, $f_X/2^2$, $f_X/2^3$ or $f_X/2^4$) can be selected by setting the PCC.

(c) With the main system clock selected, two standby modes, the STOP and HALT modes, are available.

(d) The PCC can be used to select the subsystem clock and to operate the system with low current consumption (122 μs when operated at 32.768 kHz).

(e) With the subsystem clock selected, main system clock oscillation can be stopped with the PCC. The HALT mode can be used. However, the STOP mode cannot be used. (Subsystem clock oscillation cannot be stopped.)

6.5.1 Main system clock operations

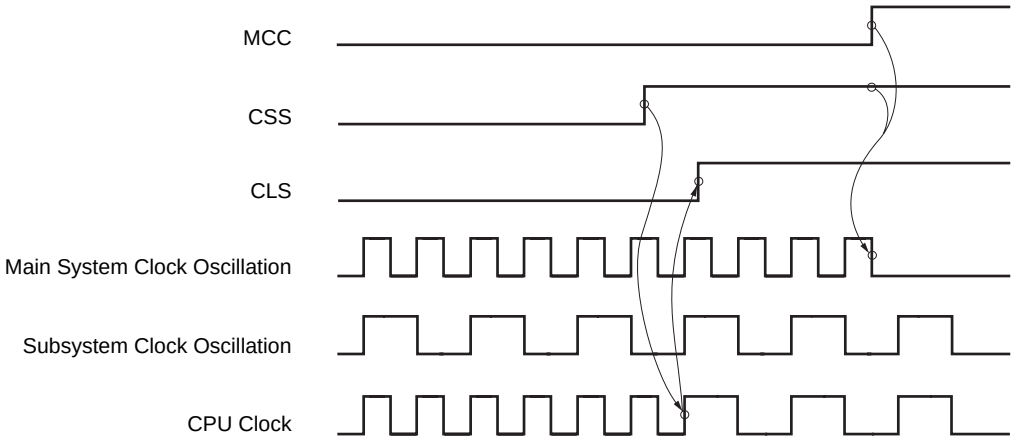
When operated with the main system clock (with bit 5 (CLS) of the processor clock control register (PCC) set to 0), the following operations are carried out by PCC setting.

(a) Because the operation guarantee instruction execution speed depends on the power supply voltage, the instruction execution time can be changed by bits 0 to 2 (PCC0 to PCC2) of the PCC.

(b) If bit 7 (MCC) of the PCC is set to 1 when operated with the main system clock, the main system clock oscillation does not stop. When bit 4 (CSS) of the PCC is set to 1 and the operation is switched to subsystem clock operation (CLS = 1) after that, the main system clock oscillation stops (see Figure 6-6).

Figure 6-6: Main System Clock Stop Function (1/2)

(a) Operation when MCC is set after setting CSS with main system clock operation



(b) Operation when MCC is set in case of main system clock operation

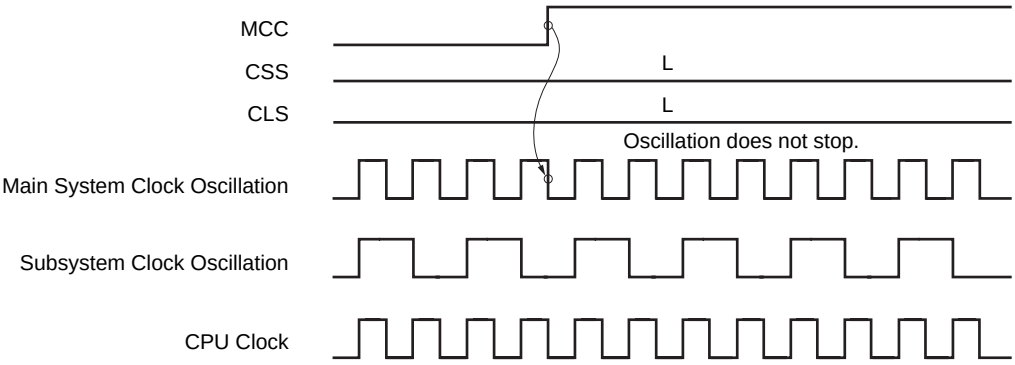
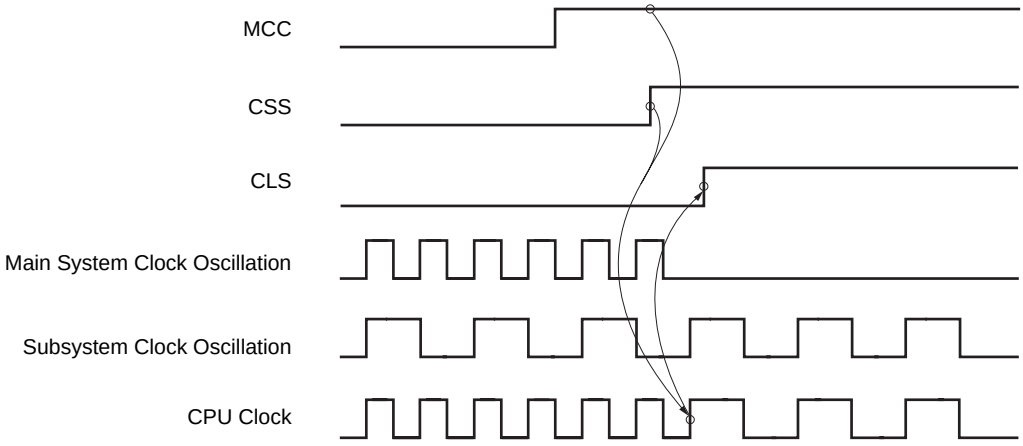


Figure 6-6: Main System Clock Stop Function (2/2)

(c) Operation when CSS is set after setting MCC with main system clock operation



6.5.2 Subsystem clock operations

When operated with the subsystem clock (with bit 5 (CLS) of the processor clock control register (PCC) set to 1), the following operations are carried out.

(a) The instruction execution time remains constant (122 μs when operated at 32.768 kHz) irrespective of bits 0 to 2 (PCC0 to PCC2) of the PCC.

(b) Watchdog timer counting stops.

Caution: Do not execute the STOP instruction while the subsystem clock is in operation.

6.6 Changing System Clock and CPU Clock Settings

6.6.1 Time required for switchover between system clock and CPU clock

The system clock and CPU clock can be switched over by means of bit 0 to bit 2 (PCC0 to PCC2) and bit 4 (CSS) of the processor clock control register (PCC).

The actual switchover operation is not performed directly after writing to the PCC, but operation continues on the pre-switchover clock for several instructions (see Table 6-2).

Determination as to whether the system is operating on the main system clock or the subsystem clock is performed by bit 5 (CLS) of the PCC register.

Table 6-2: Maximum Time Required for CPU Clock Switchover

Set Values after Switchover					Set Values before Switchover																							
MCS	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	X	X	X
X	0	0	0	0					8 instructions				4 instructions				2 instructions				1 instruction				1 instruction			
		0	0	1	16 instructions								4 instructions				2 instructions				1 instruction				1 instruction			
		0	1	0	16 instructions				8 instructions								2 instructions				1 instruction				1 instruction			
		0	1	1	16 instructions				8 instructions				4 instructions								1 instruction				1 instruction			
		1	0	0	16 instructions				8 instructions				4 instructions				2 instructions								1 instruction			
1	1	X	X	X	$f_X/2f_{XT}$ instruction (77 instructions)				$f_X/4f_{XT}$ instruction (39 instructions)				$f_X/8f_{XT}$ instruction (20 instructions)				$f_X/16f_{XT}$ instruction (10 instructions)				$f_X/32f_{XT}$ instruction (5 instructions)							
0					$f_X/4f_{XT}$ instruction (39 instructions)				$f_X/8f_{XT}$ instruction (20 instructions)				$f_X/16f_{XT}$ instruction (10 instructions)				$f_X/32f_{XT}$ instruction (5 instructions)				$f_X/64f_{XT}$ instruction (3 instructions)							

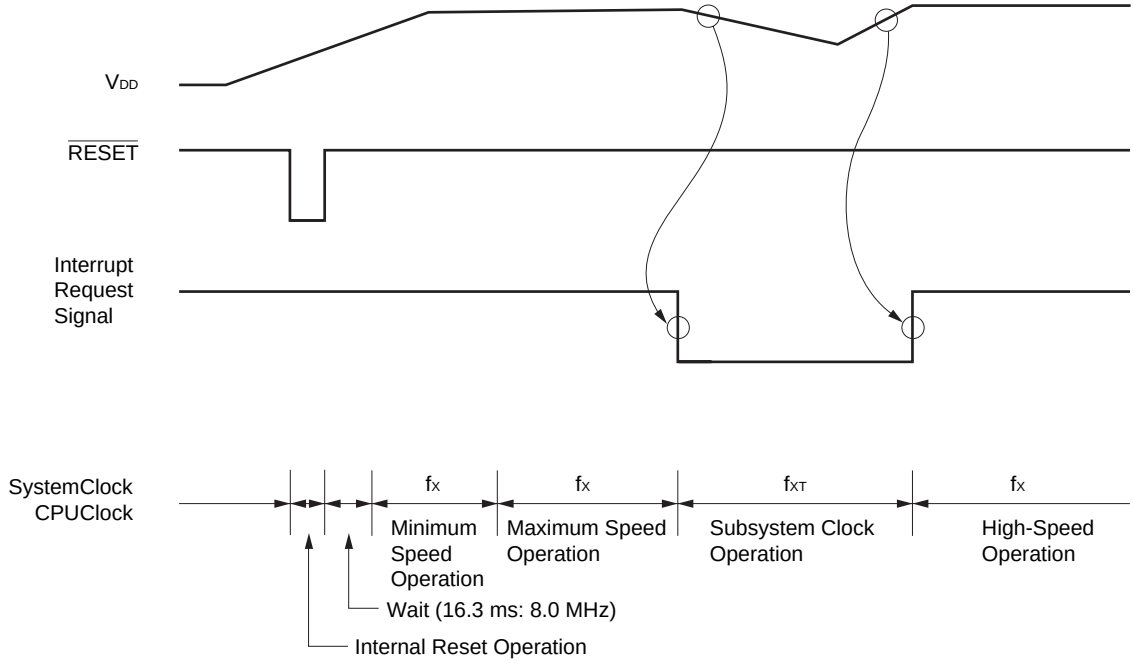
Caution: Selection of the CPU clock cycle scaling factor (PCC0 to PCC2) and switchover from the main system clock to the subsystem clock (changing CSS from 0 to 1) should not be performed simultaneously. Simultaneous setting is possible, however, for selection of the CPU clock cycle scaling factor (PCC0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

Remarks: 1. One instruction is the minimum instruction execution time with the pre-switchover CPU clock.

6.6.2 System clock and CPU clock switching procedure

This section describes switching procedure between system clock and CPU clock.

Figure 6-7: System Clock and CPU Clock Switching



(1) The CPU is reset by setting the $\overline{\text{RESET}}$ signal to low level after power-on. After that, when reset is released by setting the $\overline{\text{RESET}}$ signal to high level, main system clock starts oscillation. At this time, oscillation stabilization time ($2^{17}/f_X$) is secured automatically.
After that, the CPU starts executing the instruction at the minimum speed of the main system clock (4 μs when operated at 8.0 MHz).

(2) After the lapse of a sufficient time for the V_{DD} voltage to increase to enable operation at maximum speeds, the processor clock control register (PCC) is rewritten and the maximum-speed operation is carried out.

(3) Upon detection of a decrease of the V_{DD} voltage due to an interrupt request signal, the main system clock is switched to the subsystem clock (which must be in an oscillation stable state).

(4) Upon detection of V_{DD} voltage reset due to an interrupt request signal, 0 is set to bit 7 (MCC) of PCC and oscillation of the main system clock is started. After the lapse of time required for stabilization of oscillation, the PCC is rewritten and the maximum-speed operation is resumed.

Caution: When subsystem clock is being operated while main system clock was stopped, if switching to the main system clock is made again, be sure to switch after securing oscillation stable time by software.

[Memo]

Chapter 7 16-Bit Timer/ Event Counter

7.1 16-bit Timer/Event Counter Function

16-bit timer/event counter (TM0) has the following functions:

- Interval timer
- PPG output
- Pulse width measurement
- External event counter
- Square wave output
- One-shot pulse output

(1) Interval timer

When 16-bit timer/event counter is used as an interval timer, it generates an interrupt request at predetermined time intervals.

(2) PPG output

16-bit timer/event counter can output a square wave whose frequency and output pulse width can be freely set.

(3) Pulse width measurement

16-bit timer/event counter can be used to measure the pulse width of a signal input from an external source.

(4) External event counter

16-bit timer/event counter can be used to measure the number of pulses of a signal input from an external source.

(5) Square wave output

16-bit timer/event counter can output a square wave any frequency.

(6) One-shot pulse output

16-bit timer/event counter can output a one-shot pulse with any output pulse width.

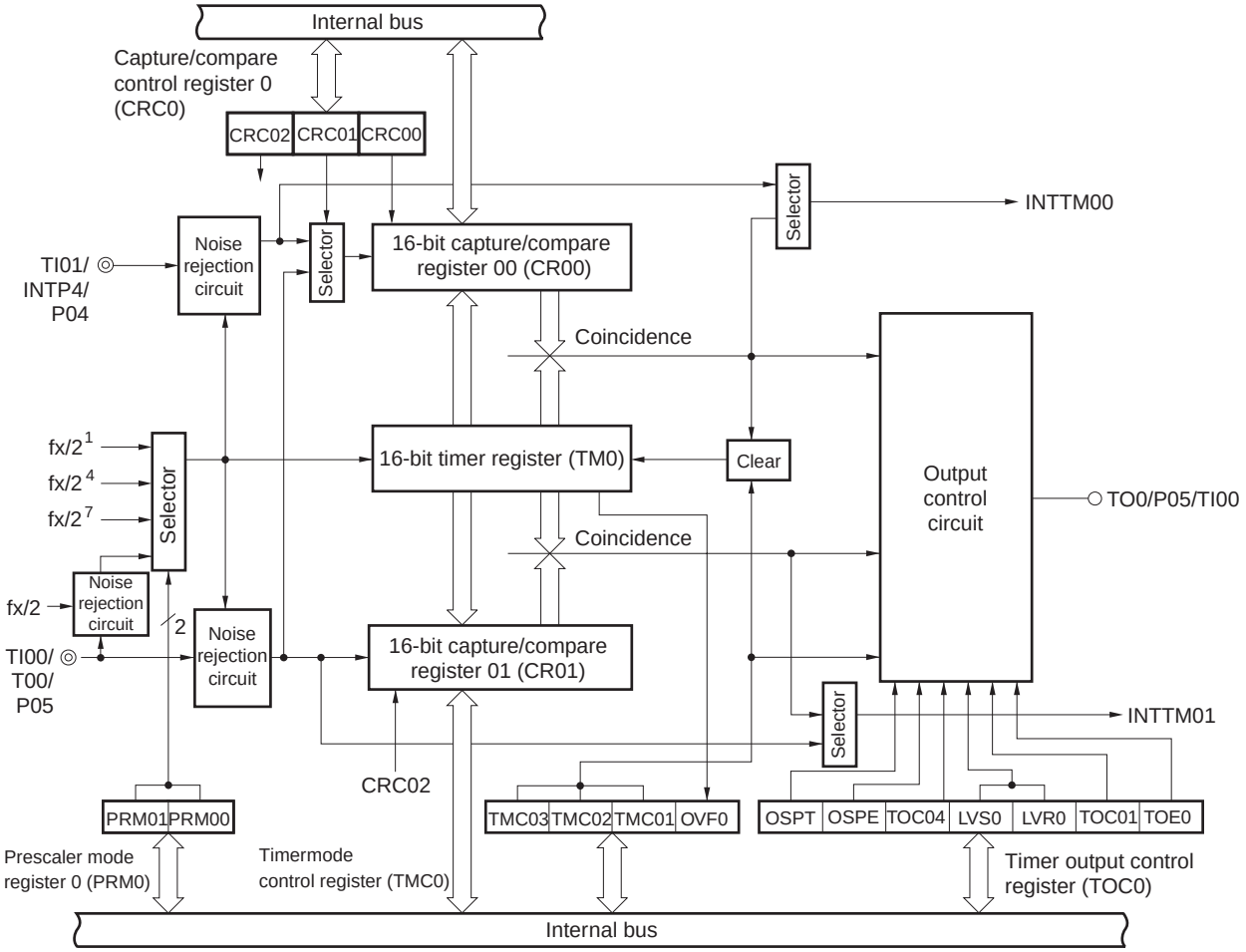
7.2 16-bit Timer/Event Counter Configuration

16-bit timer/event counter (TM0) consists of the following hardware:

Table 7-1: Configuration of 16-bit Timer/Event Counter (TM0)

Item	Configuration
Timer register	16 bits x 1 (TM0)
Register	Capture/compare register: 16 bits x 2 (CR00, CR01)
Timer output	1 (TO0)
Control register	16-bit timer mode control register (TMC0) Capture/compare register 0 (CRC0) 16-bit timer output control register (TOC0) Prescaler mode register 0 (PRM0) Port mode register 0 (PM0)

Figure 7-1: Block Diagram of 16-Bit Timer/Event Counter (TM0)



1) 16-bit timer register (TM0)

TM0 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. If the count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is reset to 0000H in the following cases:

<1> $\overline{\text{RESET}}$ is input.
<2> TMC03 and TMC02 are cleared.
<3> Valid edge of TI00 is input in the clear & start mode by inputting valid edge of TI00.
<4> TM0 and CR00 coincide with each other in the clear & start mode on coincidence between TM0 and CR00.
<5> Bit 6 of TOC0 (OSPT) is set or if the valid edge of TI00 is input in the one-shot pulse output mode.

2) Capture/compare register 00 (CR00)

CR00 is a 16-bit register that functions as a capture register and as a compare register. Whether this register functions as a capture or compare register is specified by using bit 0 (CRC00) of the capture/compare control register 0.

- **When using CR00 as compare register**
  The value set to CR00 is always compared with the count value of the 16-bit timer register (TM0). When the values of the two coincide, an interrupt request (INTTM00) is generated. When TM00 is used as an interval timer, CR00 can also be used as a register that includes the interval time.

- **When using CR00 as capture register**
  The valid edge of the TI00 or TI01 pin can be selected as a capture trigger. The valid edge of TI00 and TI01 is performed via the prescaler mode register 0 (PRM0).
  Tables 7-2 and 7-3 show the conditions that apply when the capture trigger is specified as the valid edge of the TI00 pin and the valid edge of the TI01 pin respectively.

Table 7-2: Valid Edge of TI00 Pin and Valid Edge of Capture Trigger of Capture/Compare Register

ES01	ES00	Valid Edge of TI00 Pin	Capture Trigger of CR00
0	0	Falling edge	Rising edge
0	1	Rising edge	Falling edge
1	0	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	No capture operation

Table 7-3: Valid Edge of TI01 Pin and Valid Edge of Capture Trigger of Capture/Compare Register

ES01	ES00	Valid Edge of TI00 Pin	Capture Trigger of CR00
0	0	Falling edge	Rising edge
0	1	Rising edge	Falling edge
1	0	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	Both rising and falling edges

CR00 is set by a 16-bit memory manipulation instruction.
After $\overline{\text{RESET}}$ input, the value of CR00 is undefined.

Caution: Set a value other than 0000H in CR00. This means, that an 1-pulse count operation cannot be performed when CR00 is used as an event counter.

(3) Capture/compare register 01 (CR01)

This is a 16-bit register that can be used as a capture register and a compare register. Whether it is used as a capture register or compare register is specified by bit 2 (CRC02) of the capture/compare control register 0.

- **When using CR01 as compare register**
  The value set to CR01 is always compared with the count value of the 16-bit timer register (TM0). When the values of the two coincide, an interrupt request (INTTM01) is generated.

- **When using CR01 as capture register**
  The valid edge of the TI00 pin can be selected as a capture trigger. The valid edge of TI00 is specified by using the prescaler mode register 0 (PRM0).
  CR01 is set by a 16-bit memory manipulation instruction.
  After $\overline{\text{RESET}}$ input, the value of CR00 is undefined.

Caution: Set a value other than 0000H in CR00. This means, that an 1-pulse count operation cannot be performed when CR00 is used as an event counter.

7.3 16-Bit Timer/Event Counter Control Register

The following four types of registers control 16-bit timer/event counter (TM0).

- 16-bit timer mode control register (TMC0)
- Capture/compare control register (CRC0)
- 16-bit timer output control register (TOC0)
- Prescaler mode register 0 (PRM0)
- Port mode register 0 (PM0)

(1) 16-bit timer mode control register (TMC0)

This register specifies the operation mode of the 16-bit timer and the clear mode, output timing, and overflow detection of the 16-bit timer register.

TMC0 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC0 to 00H.

Caution: The 16-bit timer register starts operating when a value other than 0, 0 (operation stop mode) is set to TMC02 and TMC03. To stop the operation, set 0, 0 to TMC02 and TMC03.

Figure 7-2: Format of 16-Bit Timer Mode Control Register (TMC0)

Address: FF60H After Reset: 00H R/W

Symbol	7	6	5	4	3	2	1	⓪
TMC0	0	0	0	0	TMC03	TMC02	TMC01	OVF0

TMC03	TMC02	TMC01	Operating Mode Clear mode and clear mode	Selection of TO0 output timing	Generation of interrupt
0	0	0	Operation stop (TM0 is cleared to 0).	Not affected	Does not generate.
0	0	1			
0	1	0	Free running mode	Coincidence between TM0 and CR00 or coincidence between TM0 and CR01	Generates on coincidence between TM0 and CR00 and coincidence between TM0 and CR01.
0	1	1		Coincidence between TM0 and CR00, coincidence between TM0 and CR01, or valid edge of TI00	
1	0	0	Clears and starts at valid edge of TI00.	Coincidence between TM0 and CR00 or coincidence between TM0 and CR01	
1	0	1		Coincidence between TM0 and CR00, coincidence between TM0 and CR01, or valid edge of TI00	
1	1	0	Clears and starts on coincidence between TM0 and CR00.	Coincidence between TM0 and cR00 or coincidence between TM0 and CR01	
1	1	1		Coincidence between TM0 and CR00, coincidence between TM0 and cR01, or valid edge of TI00	

0VF0	Detection of overflow of 16-bit timer register
0	Overflows.
1	Does not overflow.

Cautions
1. **Before changing the clear mode and TO0 output timing, be sure to stop the timer operation (reset TMC02 and TMC03 to 0, 0).**
2. **The valid edge of the TI00 pin is selected by using the prescaler mode register 0 (PRM0).**
3. **When a mode in which the timer is cleared and started on coincidence between TM0 and CR00, the OVF0 flag is set to 1 when the count value of TM0 changes from FFFFH to 0000H with CR00 set to FFFFH.**

Remark:
T00 : output pin of 16-bit timer/counter (TM0)
TI00 : input pin of 16-bit timer/counter (TM0)
TM0 : 16-bit timer register
CR00: compare register 00
CR01: compare register 01

(2) Capture/compare control register 0 (CRC0)

This register controls the operation of the capture/compare registers (CR00 and CR01).

CRC0 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CRC0 to 00H.

Figure 7-3: Format of Capture/Compare Control Register 0 (CRC0)

Address: FF62H After Reset: 04H R/W

Symbol	7	6	5	4	3	2	1	0
CRC0	0	0	0	0	0	CRC02	CRC01	CRC00

CRC02	Selection of operation mode of CR01
0	Operates as compare register
1	Operates as capture register

CRC01	Selection of capture trigger of CR00
0	Captured at valid edge of TI01
1	Captured in reverse phase of valid edge of TI00

CRC00	Selection of operation mode of CR00
0	Operates as compare register
1	Operates as capture register

Cautions: 1. Before setting CRC0, be sure to stop the timer operation.

2. When the mode in which the timer is cleared and started on coincidence between TM0 and CR00 is selected by the 16-bit timer mode control register (TMC0), do not specify CR00 as a capture register.

3. If valid edge of TI00 is both falling and rising, the capture operation is not available when CRC01 = 1.

(3) 16-bit timer output control register (TOC0)

This register controls the operation of the 16-bit timer/event counter (TM0) output control circuit by setting or resetting the R-S flip-flop (LV0), enabling or disabling reverse output, enabling or disabling output of 16-bit timer/counter (TM0), enabling or disabling one-shot pulse output operation, and selecting an output trigger for a one-shot pulse by software.

TOC0 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TOC0 to 00H.

Figure 7-4 shows the format of TOC0.

Figure 7-4: Format of 16-Bit Timer Output Control Register (TOC0)

Address: FF63H After Reset: 00H R/W

Symbol	7	⑥	⑤	4	③	②	1	⓪
TOC0	0	OSPT	OSPE	TOC04	LVS0	LVR0	TOC01	TOE0

OSPT	Output trigger control of one-shot pulse by software
0	No one-shot pulse trigger
1	Uses one-shot pulse trigger

OSPE	Controls of one-shot pulse output operation
0	Continuous pulse output
1	One-shot pulse output

TOC04	Timer output F/F control on coincidence between CR01 and TM0
0	Disables inversion timer output
1	Enables inversion timer output

LVS0	LVR0	Set status of timer output F/F of 16-bit timer/counter (TM0)
0	0	Not affected
0	1	Resets timer output F/F (0)
1	0	Sets timer output F/F (1)
1	1	Setting prohibited

TOC01	Timer output F/F control on coincidence between CR00 and TM0
0	Disables inversion timer output F/F
1	Enables inversion timer output F/F

TOE0	Output control of 16-bit timer/counter (TM0)
0	Disables output (port mode)
1	Enables output

Cautions: 1. Before setting TOC0, be sure to stop the timer operation.

2. LVS0 and LVR0 are 0 when read after data have been set to them.

3. OSPT is 0 when read because it is automatically cleared after data has been set.

(4) Prescaler mode register 0 (PRM0)

This register selects a count clock of the 16-bit timer/event counter (TM0) and the valid edge of TI00, TI01 input. PRM0 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PRM0 to 00H.

Figure 7-5: Format of Prescaler Mode Register 0 (PRM0)

Address: 0FF1CH After Reset : 00H R/W

Symbol	7	6	5	4	3	2	1	0
PRM0	ES11	ES10	ES01	ES00	0	0	PRM01	PRM00

ES11	ES10	Selection of valid edge of TI01
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

ES01	ES00	Selection of valid edge of TI00
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

PRM01	PRM00	Selection of count clock
0	0	$f_X/2^1$ (4 MHz)
0	1	$f_X/2^4$ (500 kHz)
1	0	$f_X/2^7$ (62.5 kHz)
1	1	Valid edge of TI00

Caution: When selecting the valid edge of TI00 as the count clock, do not specify the valid edge of TI00 to clear and start the timer and as a capture trigger.

Remark: Figures in parentheses apply to operation with f_X = 8.00 MHz.

(5) Port mode register 0 (PM0)

This register sets port 0 input/output in 1-bit units.

When using the P05/TO0/TI00 pin for timer output, set PM05 and the output latch of P05 to 0.

PM0 is et with an 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM0 value to FFH.

Figure 7-6: Port Mode Register 0 (PM0) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W

PM7n	P7n pin input/output mode selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

7.4 16-Bit Timer/Event Counter Operations

7.4.1 Operation as interval timer (16 bits)

The 16-bit timer/event counter operates as an interval timer when the 16-bit timer mode control register (TMC0) and capture/compare control register 0 (CRC0) are set as shown in Figure 7-7.

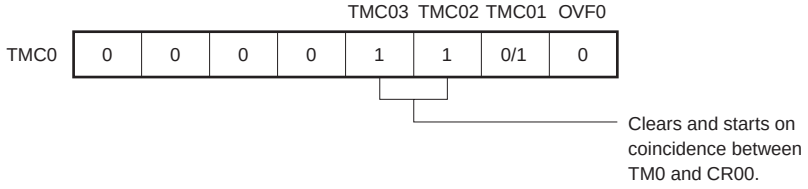
In this case, 16-bit timer/event counter repeatedly generates an interrupt at the time interval specified by the count value set in advance to the 16-bit capture/compare register 00 (CR00).

When the count value of the 16-bit timer register (TM0) coincides with the set value of CR00, the value of TM0 is cleared to 0, and the timer continues counting. At the same time, an interrupt request signal (INTTM00) is generated.

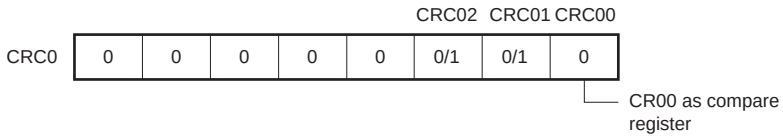
The count clock of the 16-bit timer/event counter can be selected by bits 0 and 1 (PRM00 and PRM01) of the prescaler mode register 0 (PRM0).

Figure 7-7: Control Register Settings When Timer 0 Operates as Interval Timer

(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the interval timer function. For details, refer to Figures 7-2 and 7-3.

Figure 7-8: Configuration of Interval Timer

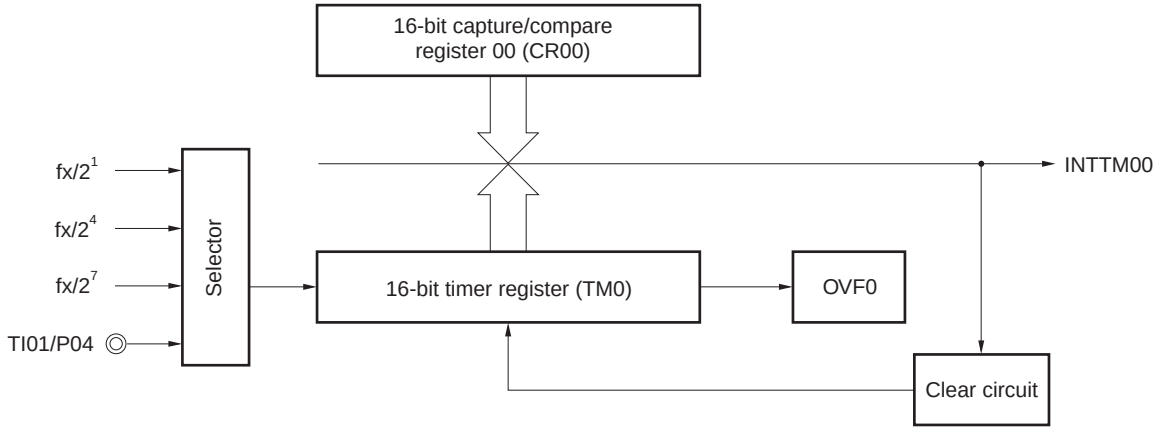
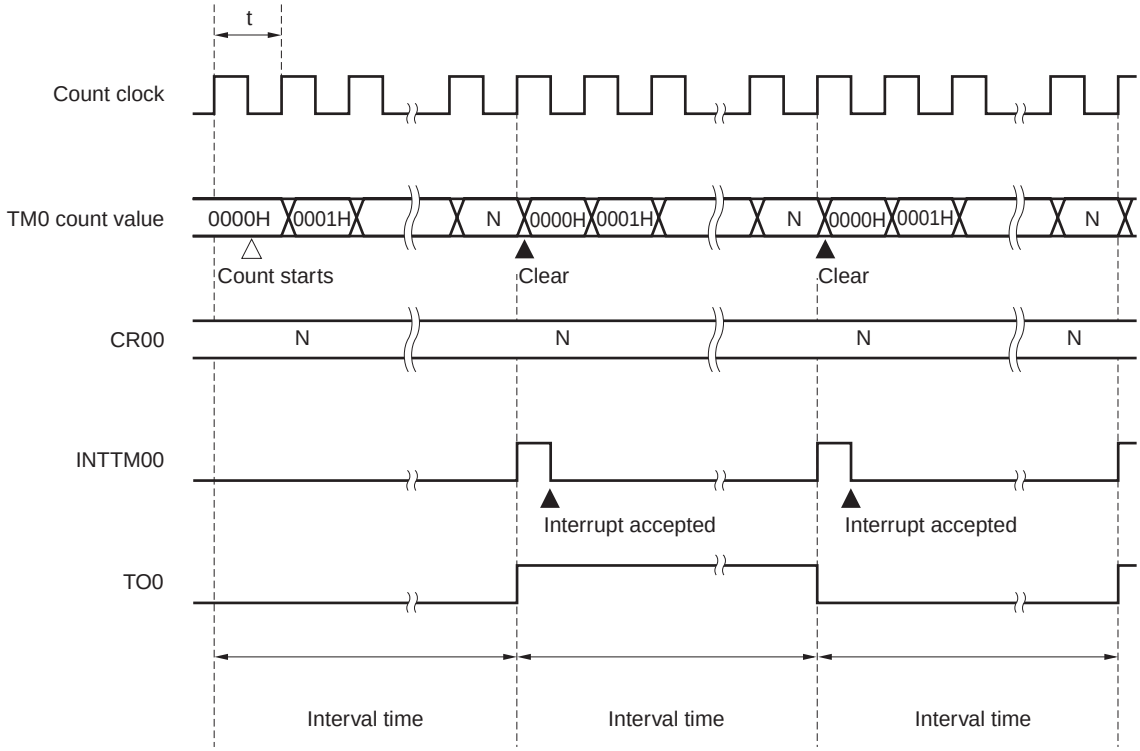


Figure 7-9: Timing of Interval Timer Operation



Remark: Interval time = (N+1) x t: N = 0000H to FFFFH

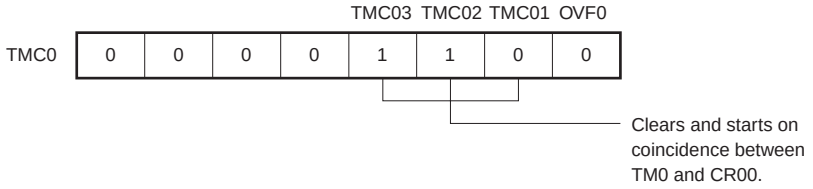
7.4.2 PPG output operation

The 16-bit timer/counter can be used for PPG (Programmable Pulse Generator) output by setting the 16-bit timer mode control register (TMC0) and capture/compare control register 0 (CRC0) as shown in Figure 7-10.

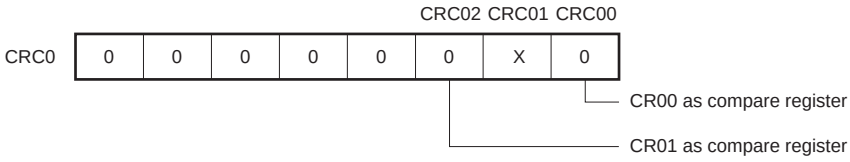
The PPG output function outputs a rectangular wave with a cycle specified by the count value set in advance to the 16-bit capture/compare register 00 (CR00) and a pulse width specified by the count value set in advance to the 16-bit capture/compare register 01 (CR01).

Figure 7-10: Control Register Settings in PPG Output Operation

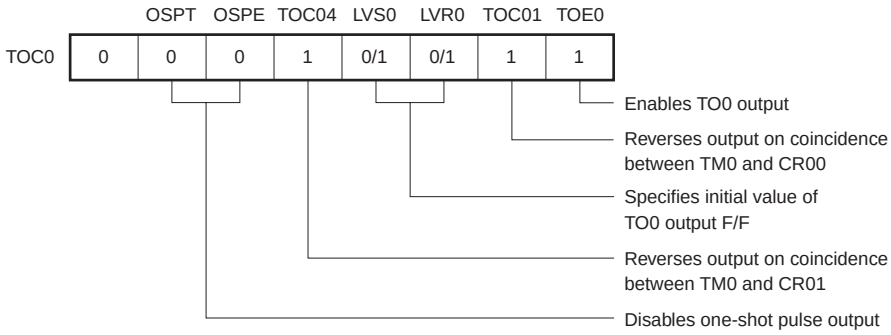
(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



(c) 16-bit timer output control register (TOC0)



Remark: x : don't care

Caution: Make sure that 0000H ≤ CR01 < CR00 ≤ FFFFH is set to CR00 and CR01.

7.4.3 Pulse width measurement

The 16-bit timer register (TM0) can be used to measure the pulse widths of the signals input to the TI00 and TI01 pins.

Measurement can be carried out with TM0 used as a free running counter or by restarting the timer in synchronization with the edge of the signal input to the TI00 pin.

(1) Pulse width measurement with free running counter and one capture register

If the edge specified by the prescaler mode register 0 (PRM0) is input to the TI00 pin when the 16-bit timer register (TM0) is used as a free running counter (refer to Figure 7-11), the value of TM0 is loaded to the 16-bit capture/compare register 01 (CR01), and an external interrupt request signal (INTTM01) is set.

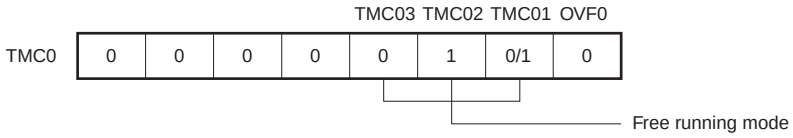
The edge is specified by using bits 6 and 7 (ES10 and ES11) of the prescaler mode register 0 (PRM0).

The rising edge, falling edge, or both the rising and falling edges can be selected.

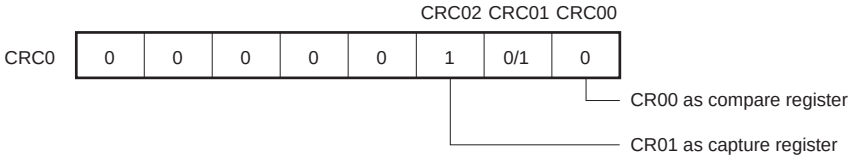
The valid edge is detected through sampling at a count clock cycle selected by the prescaler mode register 0n (PRM0), and the capture operation is not performed until the valid level is detected two times. Therefore, noise with a short pulse width can be rejected.

Figure 7-11: Control Register Settings for Pulse Width Measurement with Free Running Counter and One Capture Register

(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the pulse width measurement function. For details, refer to Figures 7-2 and 7-3.

Figure 7-12: Configuration for Pulse Width Measurement with Free Running Counter

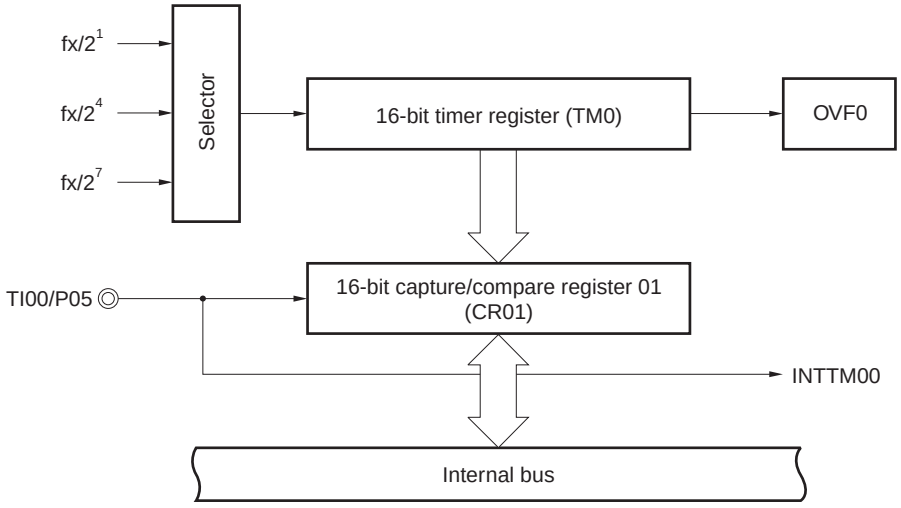
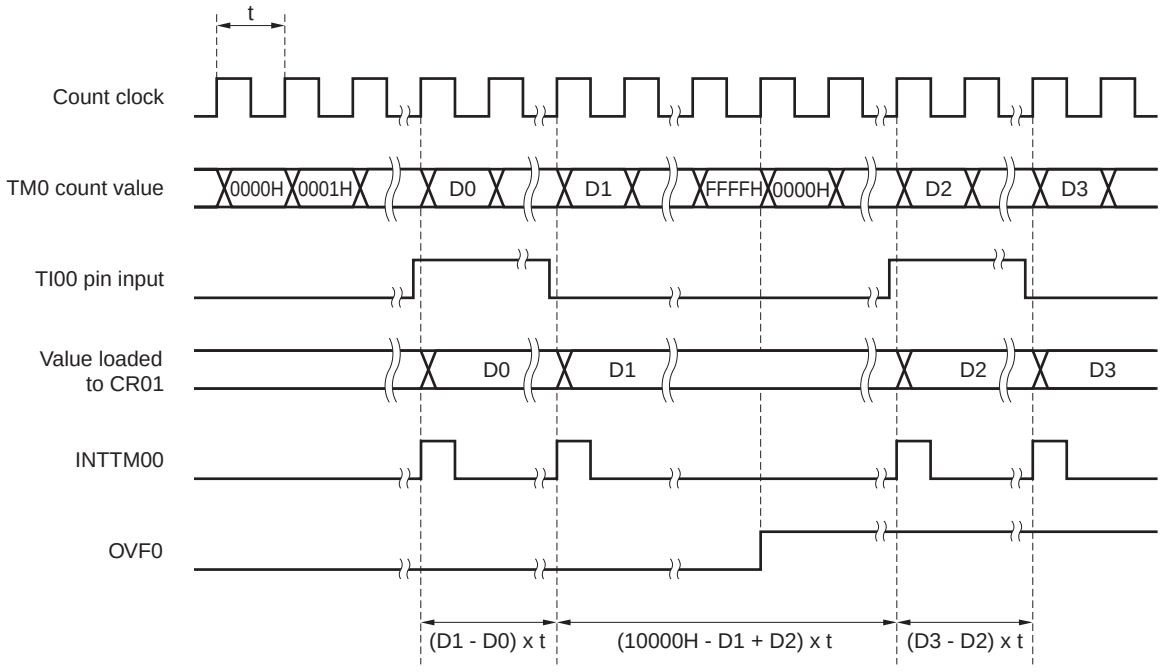


Figure 7-13: Timing of Pulse Width Measurement with Free Running Counter and One Capture Register (with both edges specified)



(2) Measurement of two pulse widths with free running counter

The pulse widths of the two signals respectively input to the TI00 and TI01 pins can be measured when the 16-bit timer register (TM0) is used as a free running counter (refer to Figure 7-14).

When the edge specified by bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0) is input to the TI00 pin, the value of the TM0 is loaded to the 16-bit capture/compare register 01 (CR01) and an external interrupt request signal (INTTM01) is set.

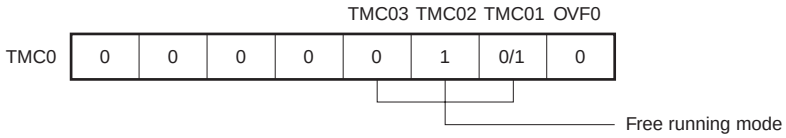
When the edge specified by bits 6 and 7 (ES10 and ES11) of the prescaler mode register 0 (PRM0) is input to the TI01 pin, the value of TM0 is loaded to the 16-bit capture/compare register 00 (CR00), and an external interrupt request signal (INTTM00) is set.

The edges of the TI00 and TI01 pins are specified by bits 4 and 5 (ES00 and ES01) and bits 6 and 7 (ES10 and ES11) of PRM0, respectively. The rising, falling, or both rising and falling edges can be specified.

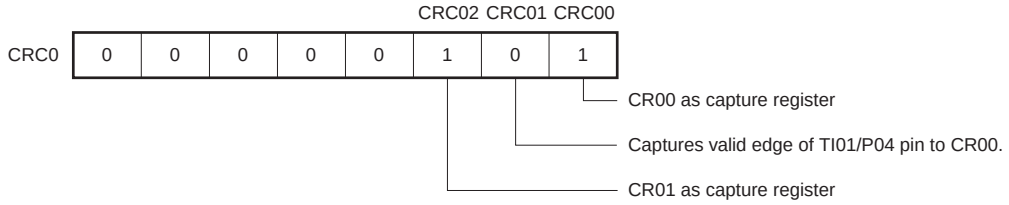
The valid edge of TI00/P05 pin and TI01/P04 pin is detected through sampling at a count clock cycle selected by the prescaler mode register 0 (PRM0), and the capture operation is not performed until the valid level is detected two times. Therefore, noise with a short pulse width can be rejected.

Figure 7-14: Control Register Settings for Measurement of Two Pulse Widths with Free Running Counter

(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the pulse width measurement function. For details, refer to Figures 7-2 and 7-3.

- **Capture operation (free running mode)**

The following figure illustrates the operation of the capture register when the capture trigger is input.

Figure 7-15: CR01 Capture Operation with Rising Edge Specified

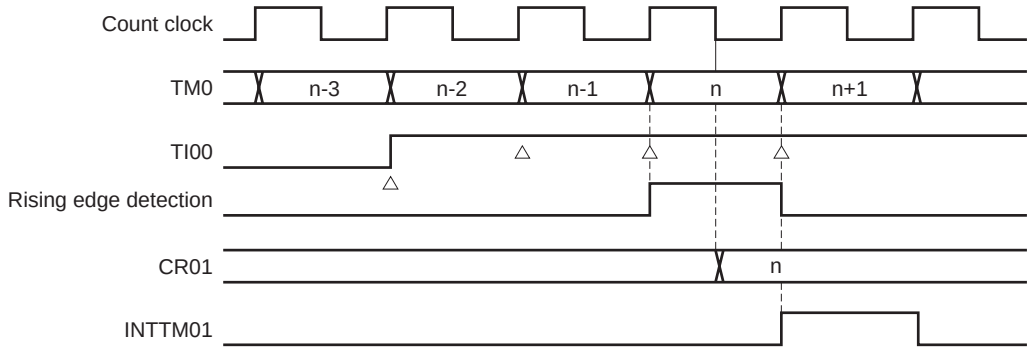
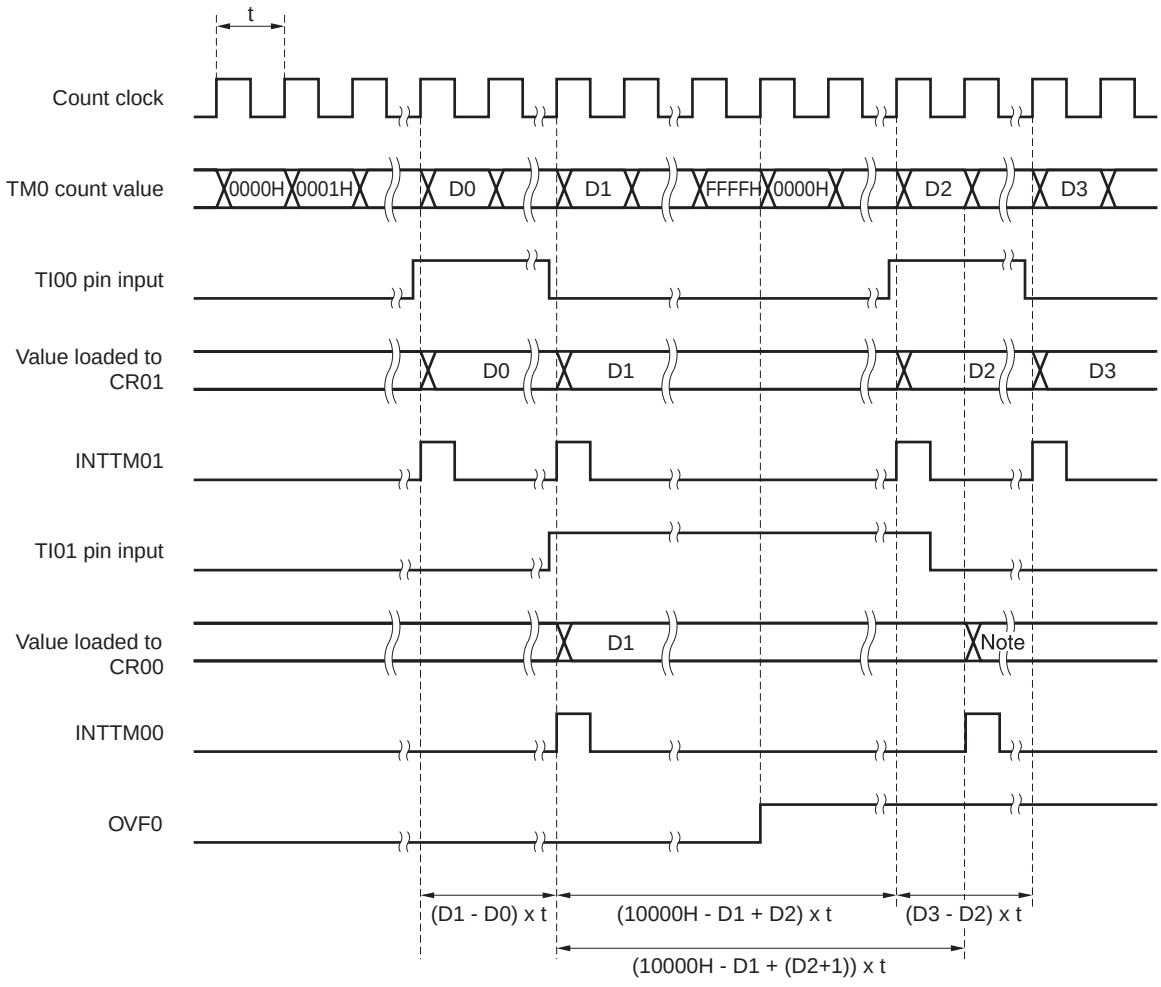


Figure 7-16: Timing of Pulse Width Measurement with Free Running Counter (with both edges specified)



Note: D2 + 1

(3) Pulse width measurement with free running counter and two capture registers

When the 16-bit timer register (TM0) is used as a free running counter (refer to Figure 7-17), the pulse width of the signal input to the TI00 pin can be measured.

When the edge specified by bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0) is input to the TI00 pin, the value of TM0 is loaded to the 16-bit capture/compare register 01 (CR01), and an external interrupt request signal (INTTM01) is set.

The value of TM0 is also loaded to the 16-bit capture/compare register 00 (CR00) when an edge reverse to the one that triggers capturing to CR01 is input.

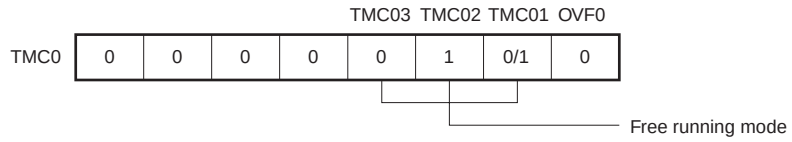
The edge of the TI00 pin is specified by bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0). The rising or falling edge can be specified.

The valid edge of TI00/P05 pin and TI01/P04 pin is detected through sampling at a count clock cycle selected by the prescaler mode register 0 (PRM0), and the capture operation is not performed until the valid level is detected two times. Therefore, noise with a short pulse width can be rejected.

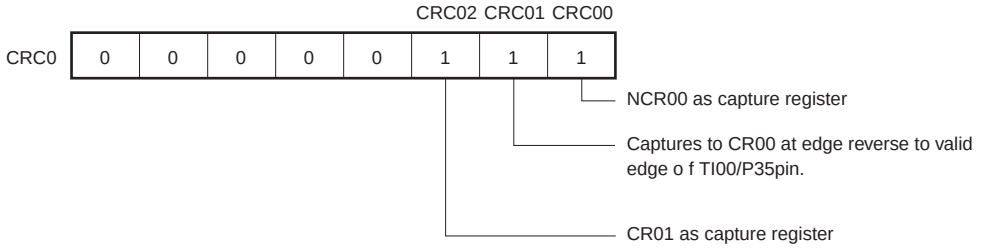
Caution: If the valid edge of the TI00 pin is specified to be both the rising and falling edges, the capture/compare register 00 (CR00) cannot perform its capture operation.

Figure 7-17: Control Register Settings for Pulse Width Measurement with Free Running Counter and Two Capture Registers

(a) 16-bit timer mode control register (TMC0)

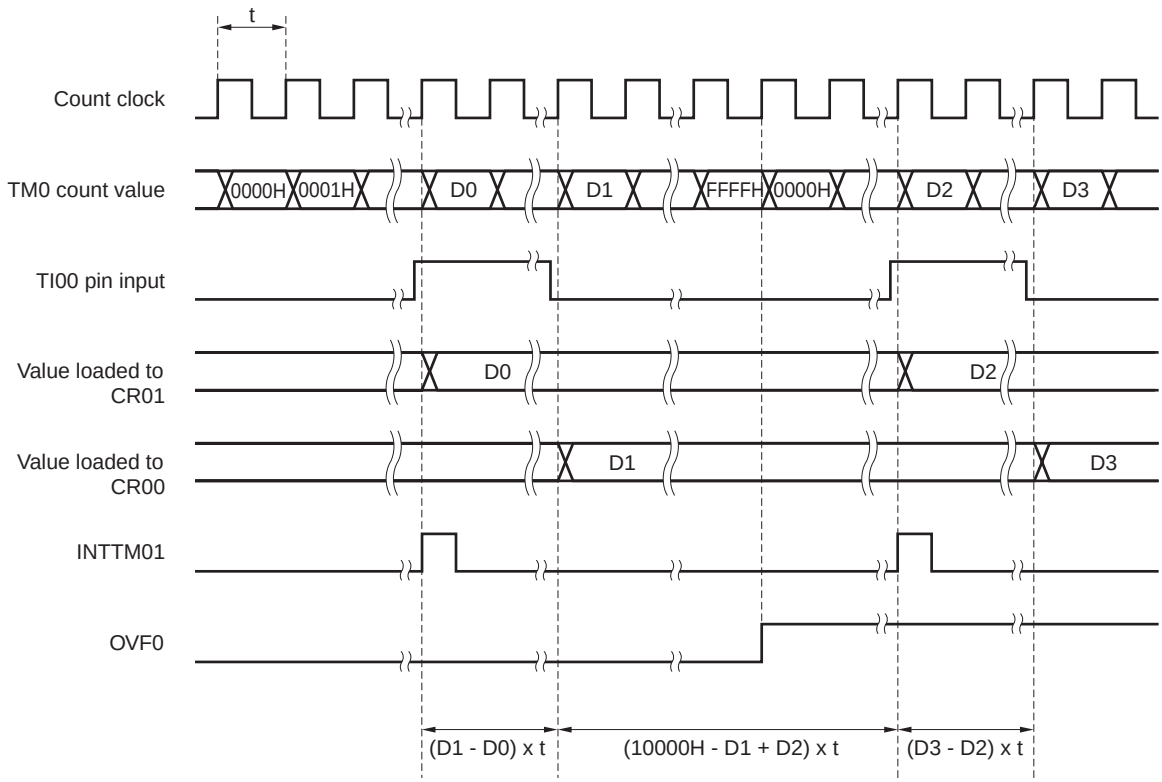


(b) Capture/compare control register 0 (CRC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the pulse width measurement function. For details, refer to Figures 7-2 and 7-3.

Figure 7-18: Timing of Pulse Width Measurement with Free Running Counter and Two Capture Registers (with rising edge specified)



(4) Pulse width measurement by restarting

When the valid edge of the TI00 pin is detected, the pulse width of the signal input to the TI00n pin can be measured by clearing the 16-bit timer register (TM0) once and then resuming counting after loading the count value of TM0 to the 16-bit capture/compare register 01 (CR01).

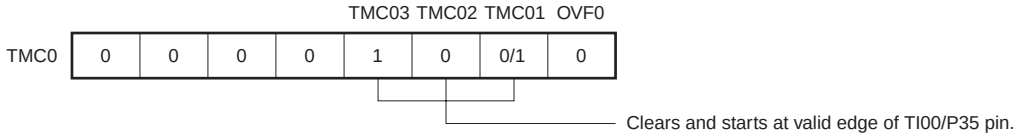
The edge of the TI00 pin is specified by bits 4 and 5 (ES00 and ES01) of PRM0. The rising or falling edge can be specified.

The valid edge is detected through sampling at a count clock cycle selected by the prescaler mode register 0 (PRM0), and the capture operation is not performed until the valid level is detected two times. Therefore, noise with a short pulse width can be rejected.

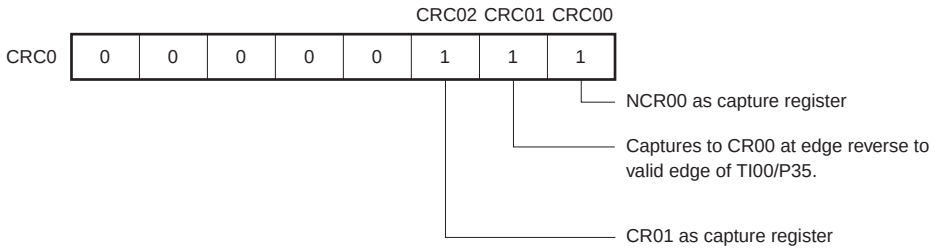
Caution: If the valid edge of the TI00 pin is specified to be both the rising and falling edges, the capture/compare register 00 (CR00) cannot perform its capture operation.

Figure 7-19: Control Register Settings for Pulse Width Measurement by Restarting

(a) 16-bit timer mode control register (TMC0)

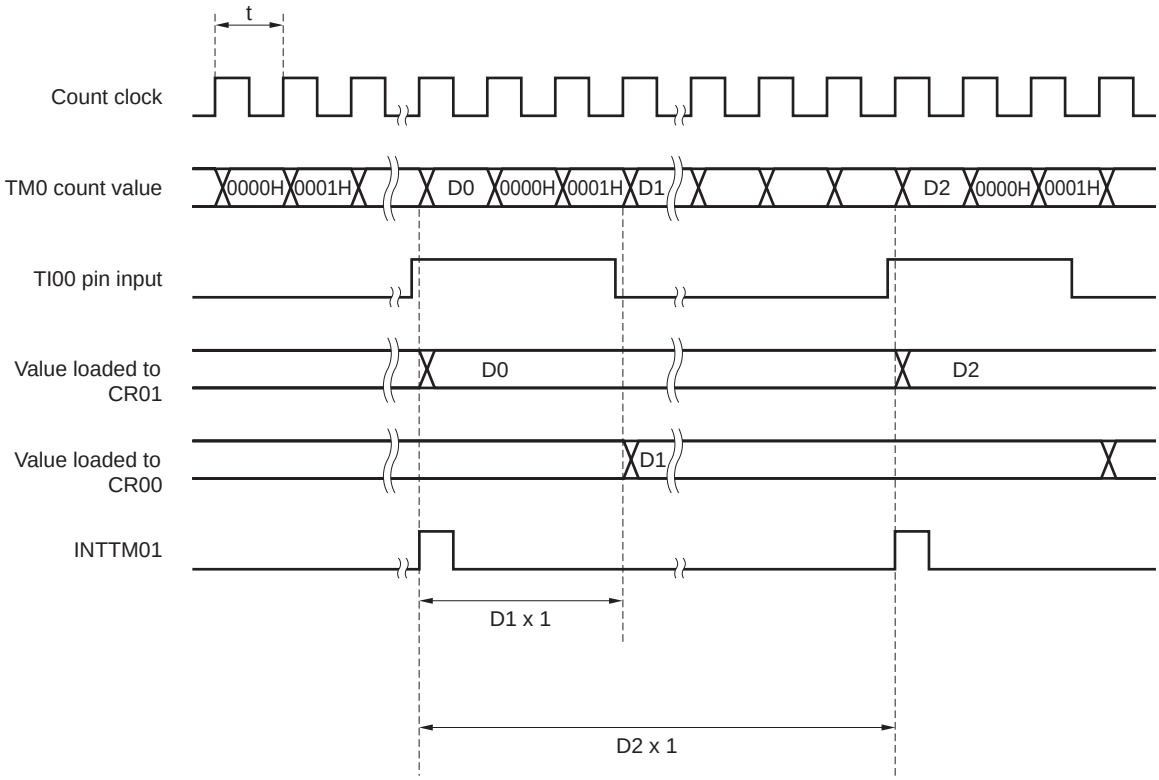


(b) Capture/compare control register 0 (CRC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the pulse width measurement function. For details, refer to Figures 7-2 and 7-3.

Figure 7-20: Timing of Pulse Width Measurement by Restarting (with rising edge specified)



7.4.4 Operation as external event counter

16-bit timer/event counter can be used as an external event counter which counts the number of clock pulses input to the TI00 pin from an external source by using the 16-bit timer register (TM0).

Each time the valid edge specified by the prescaler mode register 0 (PRM0) has been input to the TI00 pin, TM0 is incremented.

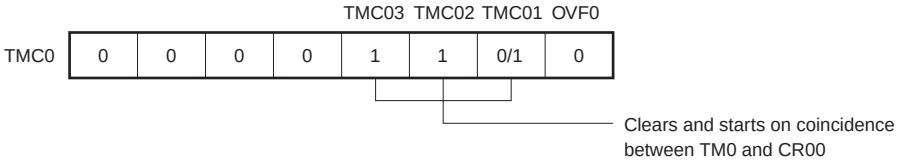
When the count value of TM0 coincides with the value of the 16-bit capture/compare register 00 (CR00), TM0 is cleared to 0, and an interrupt request signal (INTTM00) is generated.

The edge of the TI00 pin is specified by bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0). The rising, falling, or both the rising and falling edges can be specified.

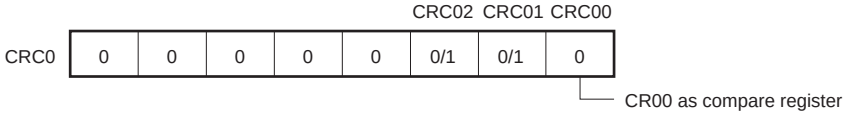
The valid edge is detected through sampling at a count clock cycle, selected by the prescaler mode register 0 (PRM0) and performed until the valid level is detected two times. Therefore, noise with a short pulse width can be rejected.

Figure 7-21: Control Register Settings in External Event Counter Mode

(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the external event counter function. For details, refer to Figures 7-2 and 7-3.

Figure 7-22: Configuration of External Event Counter

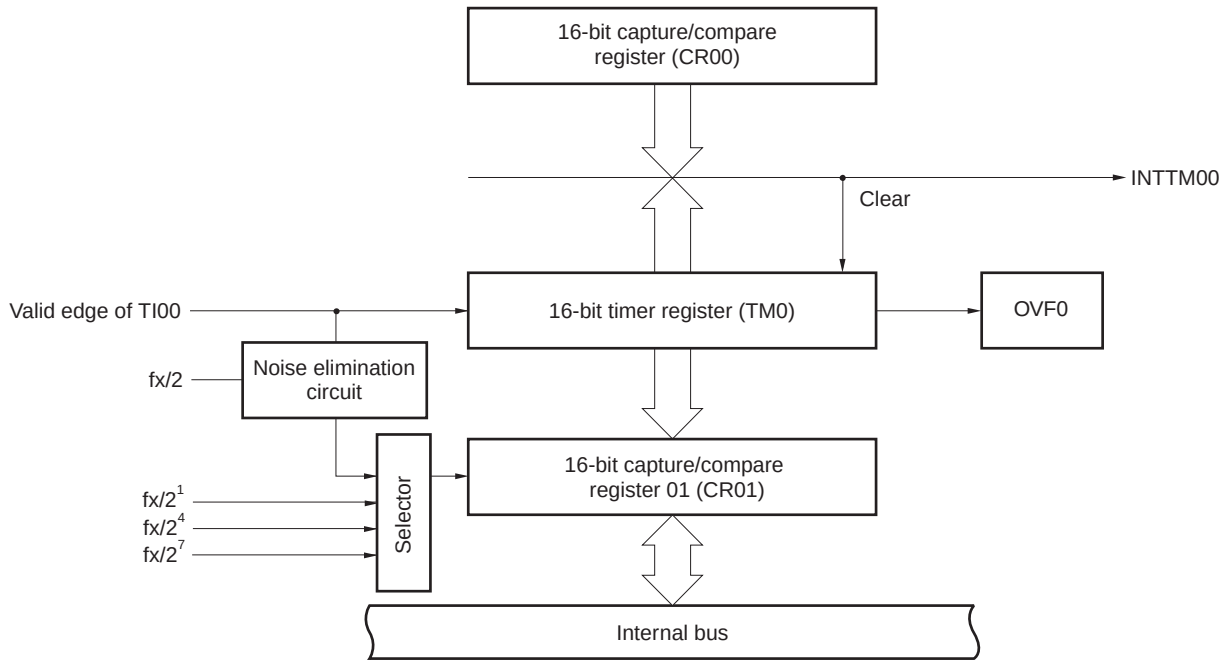
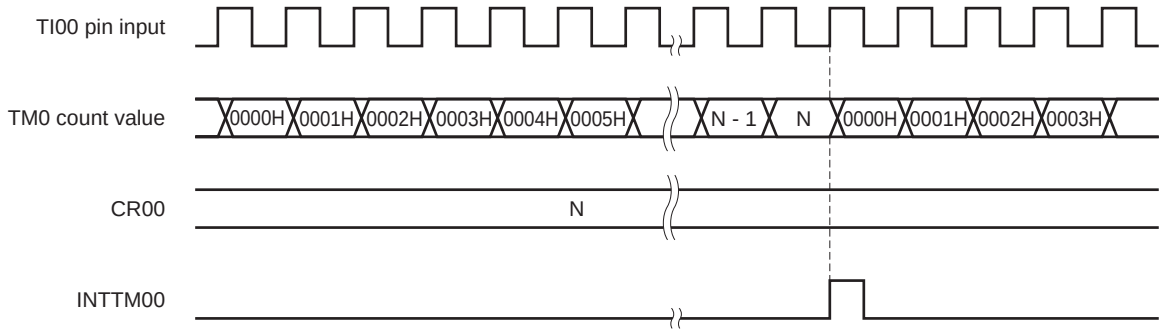


Figure 7-23: Timing of External Event Counter Operation (with rising edge specified)



Caution: Read TM0 when reading the count value of the external event counter.

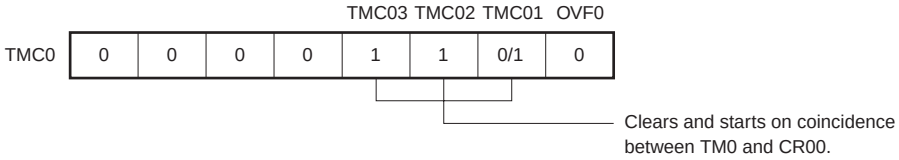
7.4.5 Operation to output square wave

The 16-bit timer/event counter can be used to output a square wave with any frequency at an interval specified by the count value set in advance to the 16-bit capture/compare register 00 (CR00).

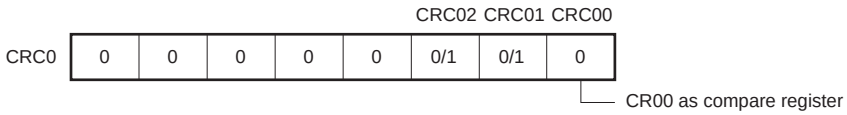
By setting bits 0 (TOE0) and 1 (TOC01) of the 16-bit timer output control register to 1, the output status of the TO0/P05 pin is reversed at an interval specified by the count value set in advance to CR00. In this way, a square wave of any frequency can be output.

Figure 7-24: Set Contents of Control Registers in Square Wave Output Mode

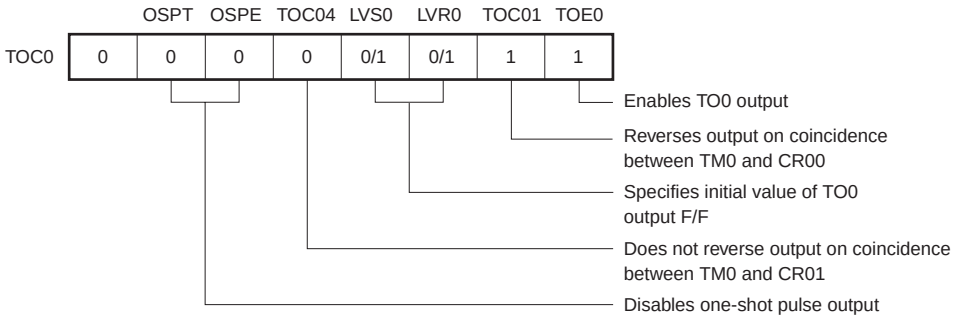
(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)

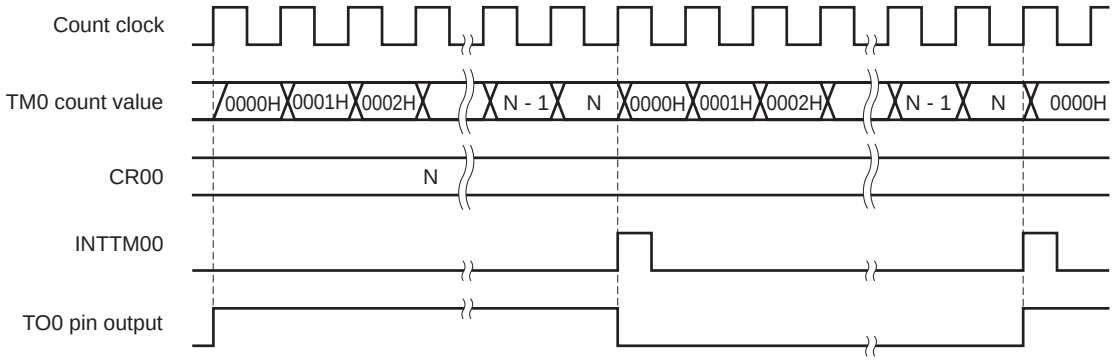


(c) 16-bit timer output control register (TOC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the square wave output function. For details, refer to Figures 7-2, 7-3, and 7-4.

Figure 7-25: Timing of Square Wave Output Operation



7.4.6 Operation to output one-shot pulse

16-bit timer/event counter can output a one-shot pulse in synchronization with a software trigger and an external trigger (TI00/TO0/P05 pin input).

(1) One-shot pulse output with software trigger

A one-shot pulse can be output from the TO0/P05 pin by setting the 16-bit timer mode control register (TMC0), capture/com7-26, and by setting bit 6 (OSPT) of TOC0 by software.

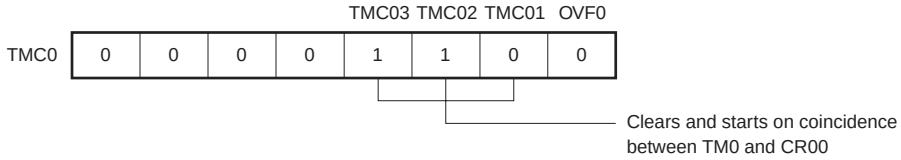
By setting OSPT to 1, the 16-bit timer/event counter is cleared and started, and its output is asserted active at the count value set in advance to the 16-bit capture/compare register 01 (CR01). After that, the output is deasserted inactive at the count value set in advance to the 16-bit capture/compare register 00 (CR00).

Even after the one-shot pulse has been output, TM0 continues its operation. To stop TM0, TMC0 must be reset to 00H.

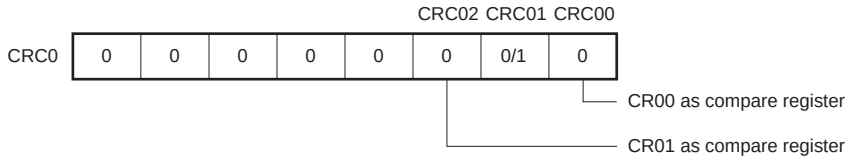
Caution: Do not set OSPT to 1 while the one-shot pulse is being output. To output the one-shot pulse again, wait until INTTM00, which occurs on coincidence between TM0 and CR00, occurs.

Figure 7-26: Control Register Settings for One-Shot Pulse Output with Software Trigger

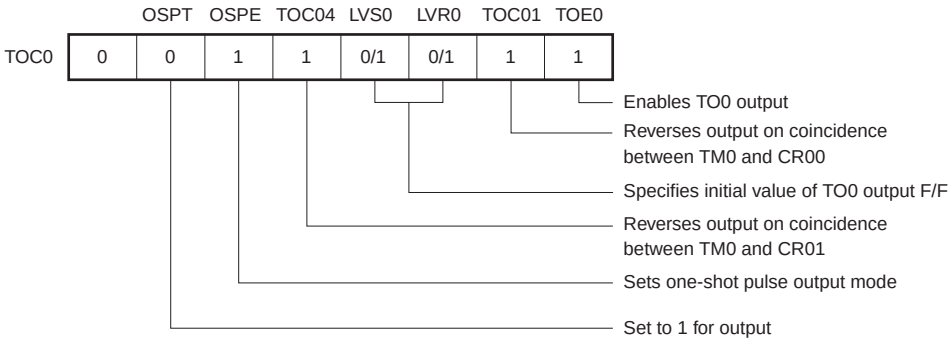
(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



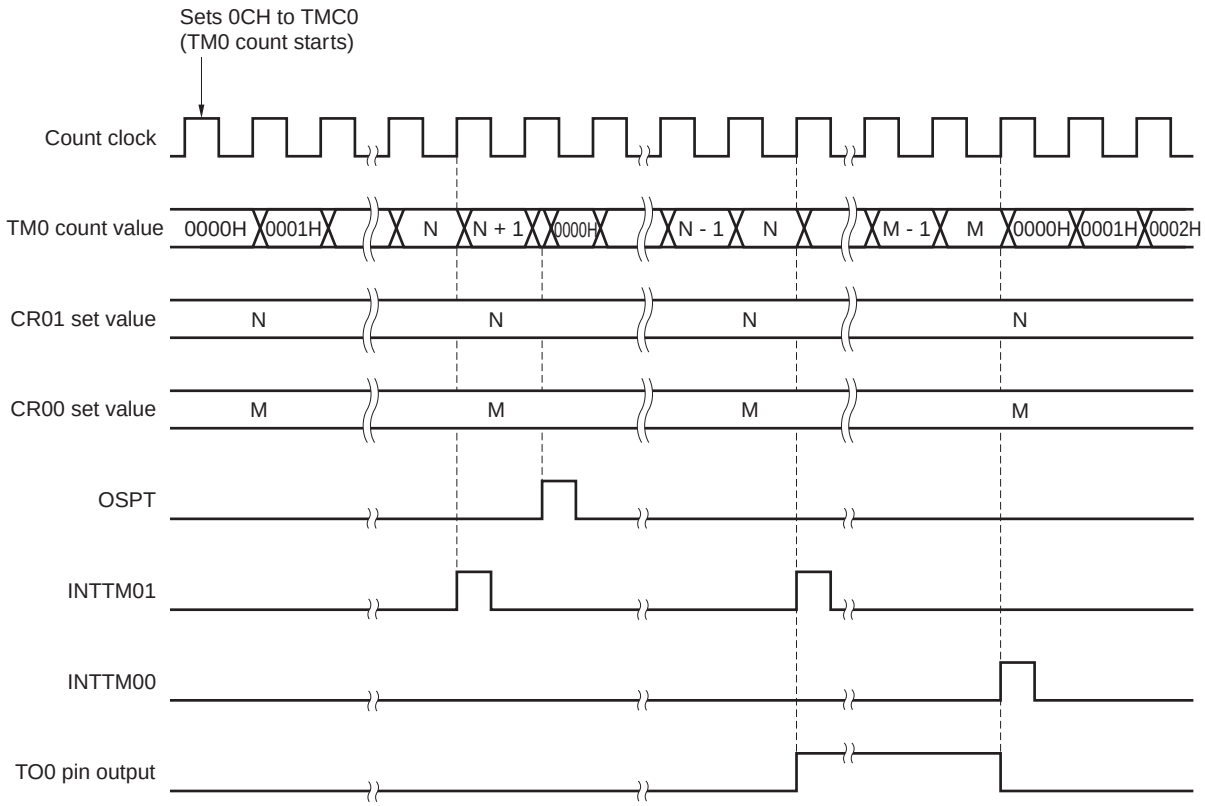
(c) 16-bit timer output control register (TOC0)



Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the one-shot pulse output function. For details, refer to Figures 7-2, 7-3, and 7-4.

Caution: Set a value in the following range to CR00 and CR01.
0000H - CR01 < CR00 - FFFFH

Figure 7-27: Timing of One-Shot Pulse Output Operation with Software Trigger



Caution: The 16-bit timer register starts operating as soon as a value other than 0, 0 (operation stop mode) has been set to TMC02 and TMC03.

(2) One-shot pulse output with external trigger

A one-shot pulse can be output from the TO0/TI00/P05 pin by setting the 16-bit timer mode control register (TMC0), capture/compare control register 0 (CRC0), and 16-bit timer output control register (TOC0) as shown in Figure 7-28, and by using the valid edge of the TO0/TI00/P05 pin as an external trigger.

The valid edge of the TI00/P05 pin is specified by bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0). The rising, falling, or both the rising and falling edges can be specified.

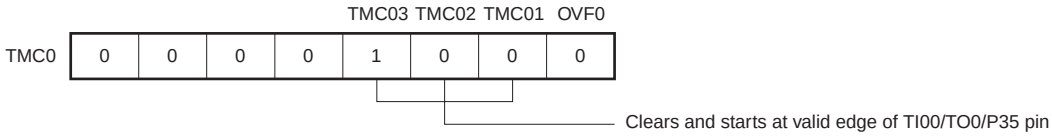
When the valid edge of the TI00 pin is detected, the 16-bit timer/event counter is cleared and started, and the output is asserted active at the count value set in advance to the 16-bit capture/compare register 01 (CR01).

After that, the output is deasserted inactive at the count value set in advance to the 16-bit capture/compare register 00 (CR00).

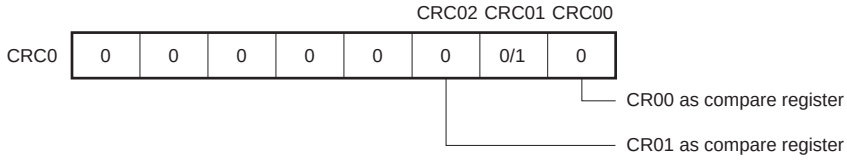
Caution: Even if the external trigger is generated again while the one-shot pulse is output, it is ignored.

Figure 7-28: Control Register Settings for One-Shot Pulse Output with External Trigger

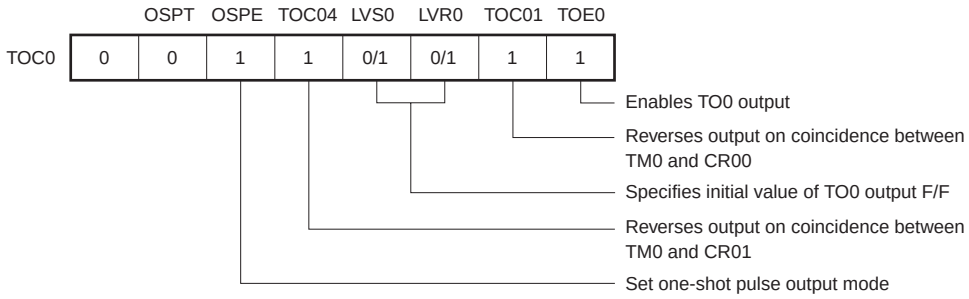
(a) 16-bit timer mode control register (TMC0)



(b) Capture/compare control register 0 (CRC0)



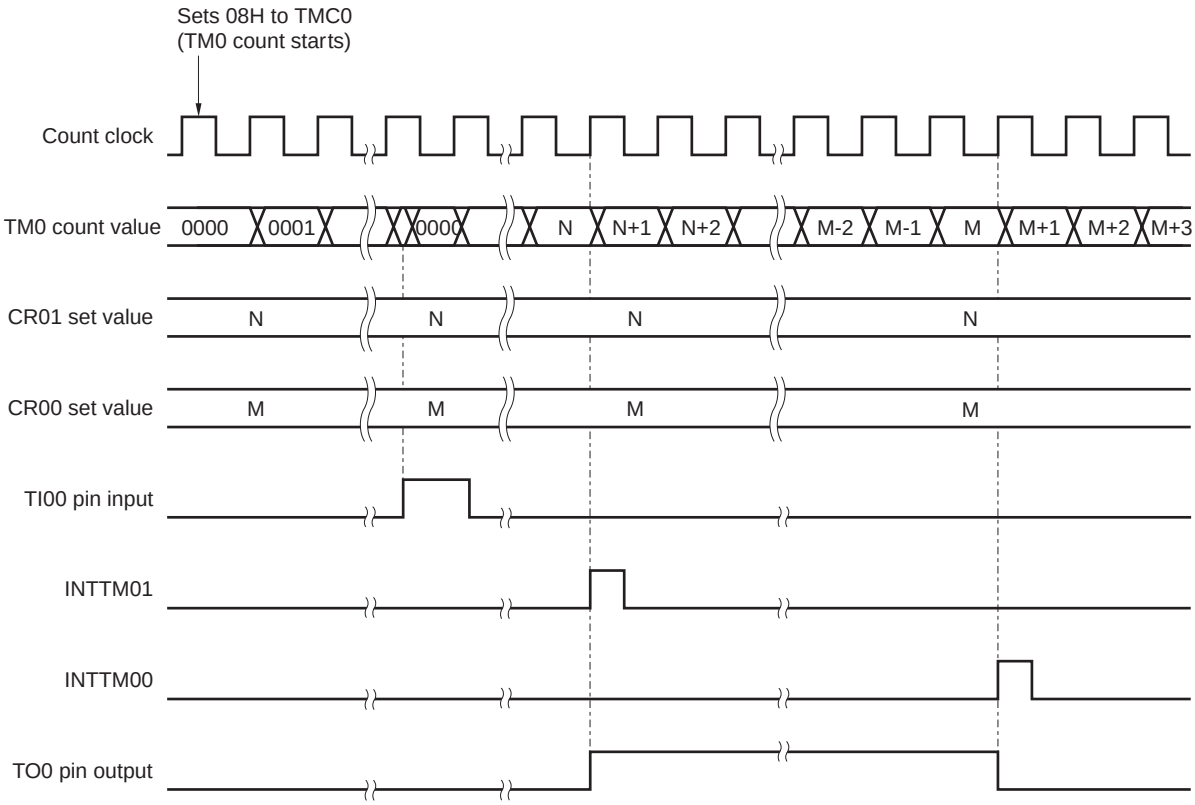
(c) 16-bit timer output control register (TOC0)



Caution: Set a value in the following range to CR00 and CR01.
0000H ≤ CR01 < CR00 ≤ FFFFH

Remark: 0/1: When these bits are reset to 0 or set to 1, the other functions can be used along with the one-shot pulse output function. For details, refer to Figures 7-2, 7-3, and 7-4.

Figure 7-29: Timing of One-Shot Pulse Output Operation with External Trigger (with rising edge specified)



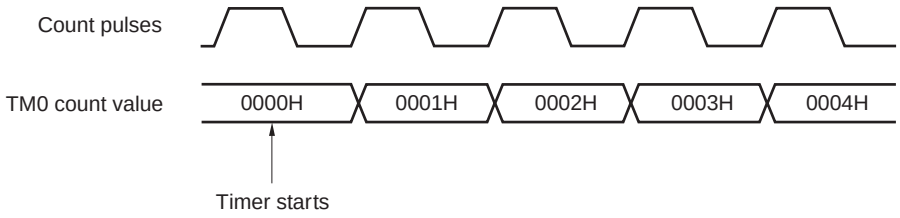
Caution: The 16-bit timer register starts operating as soon as a value other than 0, 0 (operation stop mode) has been set to TMC02 and TMC03.

7.5 16-Bit Timer/Event Counter Operating Precautions

(1) Error on starting timer

An error of up to 1 clock occurs before the coincidence signal is generated after the timer has been started. This is because the 16-bit timer register (TM0) is started asynchronously in respect to the count pulse.

Figure 7-30: Start Timing of 16-Bit Timer Register



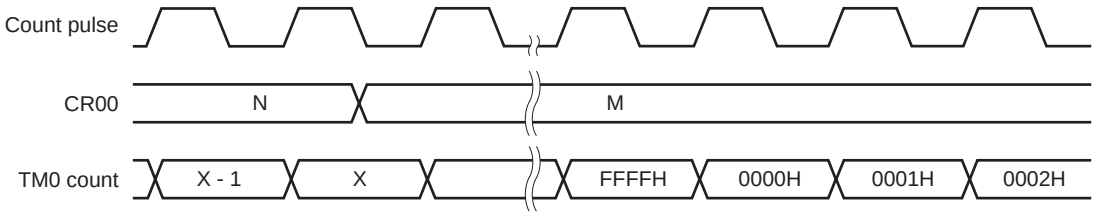
(2) 16-bit compare register setting

Set another value than 0000H to the 16-bit captured compare register CR00, CR01. This means, that a 1-pulse count operation cannot be performed, when it is used as event counter.

(3) Setting compare register during timer count operation

If the value to which the current value of the 16-bit capture/compare register 00 (CR00) has been changed is less than the value of the 16-bit timer register (TM0), TM0 continues counting, overflows, and starts counting again from 0. If the new value of CR00 (M) is less than the old value (N), the timer must be restarted after the value of CR00 has been changed.

Figure 7-31: Timing after Changing Compare Register during Timer Count Operation

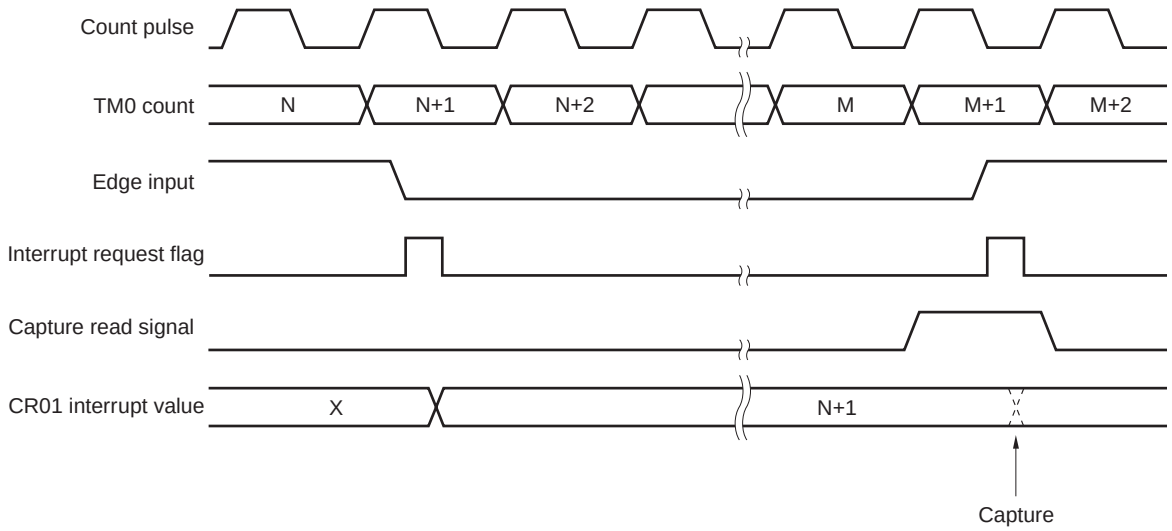


Remark: N > X > M

(4) Data hold timing of capture register

If the valid edge is input to the TI00 pin while the 16-bit capture/compare register 01 (CR01) is read, CR01 performs the capture operation, but this capture value is not guaranteed. However, the interrupt request flag (INTTM01) is set as a result of detection of the valid edge.

Figure 7-32: Data Hold Timing of Capture Register



(5) Setting valid edge

Before setting the valid edge of the TI00/TO0/P05 pin, stop the timer operation by resetting bits 2 and 3 (TMC02 and TMC03) of the 16-bit timer mode control register to 0, 0. Set the valid edge by using bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0).

(6) Re-triggering one-shot pulse

(a) One-shot pulse output by software

When a one-shot pulse is output, do not set OSPT to 1. Do not output the one-shot pulse again until INTTM00, which occurs on coincidence between TM0 and CR00, occurs.

(b) One-shot pulse output with external trigger

If the external trigger occurs while a one-shot pulse is output, it is ignored.

7) Operation of OVF0 flag

The OVF0 flag is set to 1 in the following case:

Select mode in which 16-bit timer/counter is cleared and started on coincidence between TM0 and CR00.

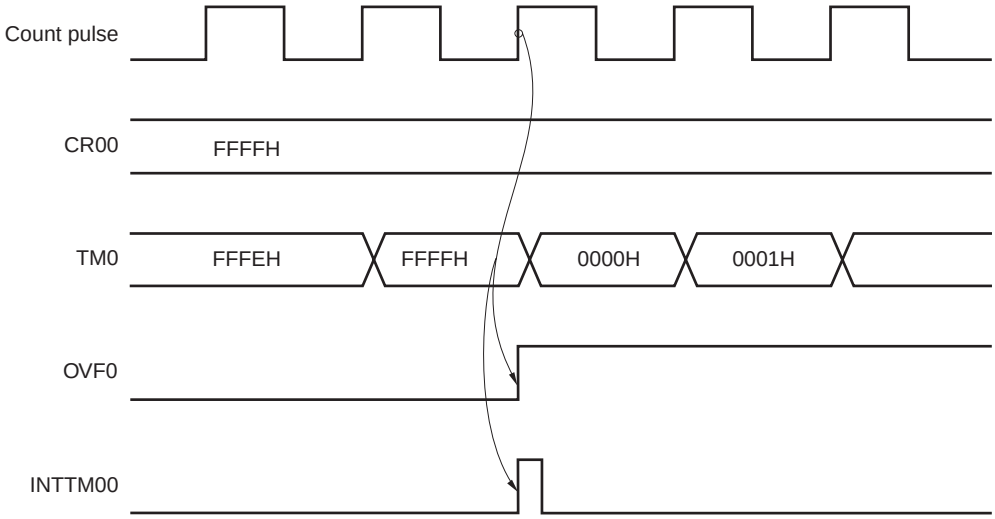
↓

Set CR00 to FFFFH

↓

When TM0 counts up from FFFFH to 0000H

Figure 7-33: Operation Timing of OVF0 Flag



(8) Contending operations

(a) The contending operation between the read time of 16-bit capture/compare register (CR00/CR01) and capture trigger input (CR00/CR01 used as capture register)

Capture/trigger input is prior to the other. The data read from CR00/CR01 is not defined.

(b) The coincidence timing of contending operation between the write period of 16-bit capture/compare register (CR00/CR01) and 16-bit timer register (TM0) (CR00/CR01 used as a compare register)

The coincidence discriminant is not performed normally. Do not write any data to CR00/CR01 near the coincidence timing.

[Memo]

Chapter 8 16-Bit Timer 2 TM2

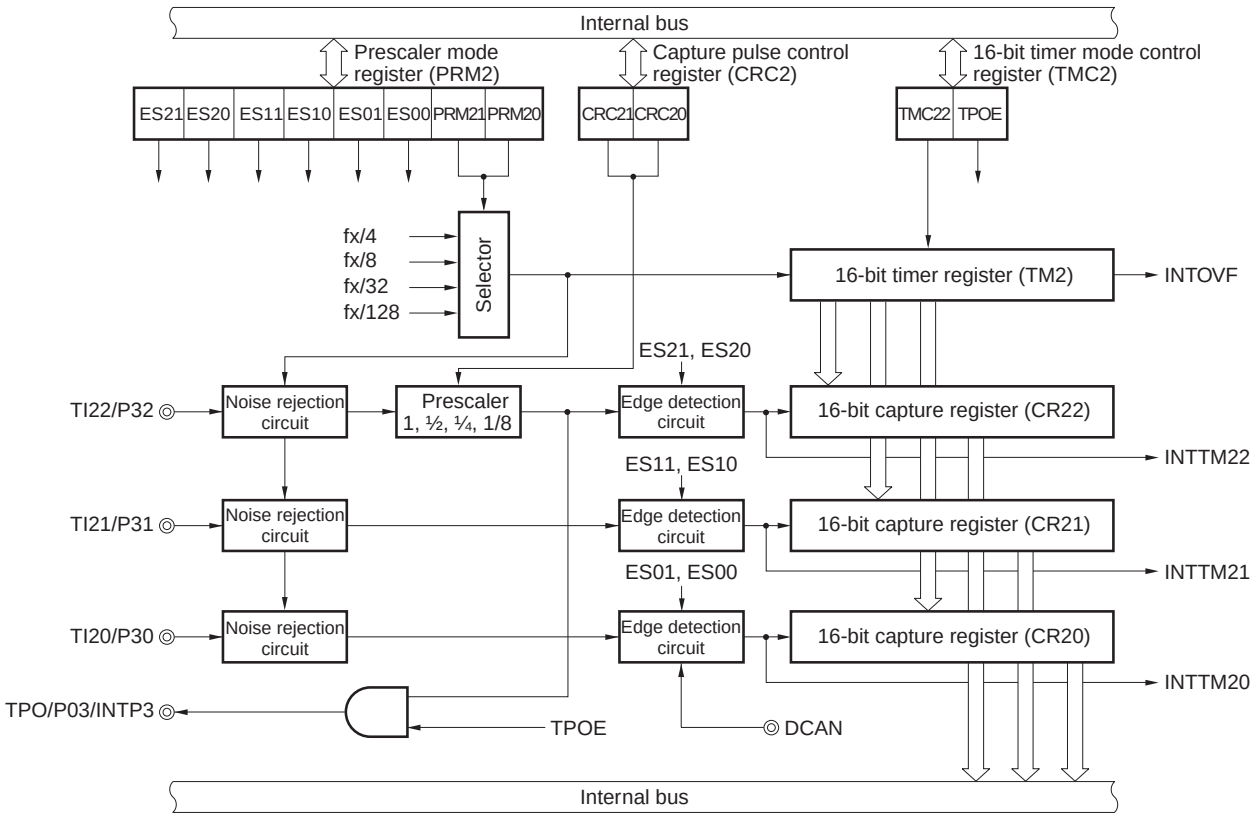
8.1 16-Bit Timer 2 Functions

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 8-1 shows 16-Bit Timer 2 Block Diagram.

Figure 8-1: Timer 0 (TM0) Block Diagram



(1) Pulse width measurement

TM2 can measure the pulse width of an externally 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).

8.2 16-Bit Timer 2 Configuration

Timer 2 consists of the following hardware.

Table 8-1: Timer 2 Configuration

Item	Configuration
Timer register	16 bits x 1 (TM2)
Register	Capture register: 16 bits x 3 (CR20 to CR22)
Control register	16 bit timer mode control register (TMC2)
	Capture pulse control register (CRC2)
	Prescaler mode register (PRM2)

(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. If the count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is reset to 0000H in the following cases:

<1> At $\overline{\text{RESET}}$ input

<2> If TMC22 is cleared

(2) Capture register 20 (CR20)

The valid edge of the TI20/P30 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/P31 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/P32 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.

8.3 16-Bit Timer 2 Control Registers

The following three 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)

(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 a 1-bit or 8-bit memory manipulation instruction.

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

Figure 8-2: 16-Bit Timer Mode Control Register (TMC2) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TMC2	0	0	0	0	0	TMC02	0	TPOE	FF65H	00H	R/W

TMC2	Timer 2 Operating Mode Selection
0	Operation stop (TM2 cleared to 0)
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 a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CRC2 value to 04H.

Figure 8-3: Capture Pulse Control Register (CRC2) Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
CRC2	0	0	0	0	0	0	CRC01	CRC00	FF67H	00H	R/W

CRC21	CRC20	Capture Pulse Selection
0	0	Does not divide capture pulse
0	1	Divides capture pulse by 2
1	0	Divides capture pulse by 4
1	1	Divides capture pulse by 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 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 8-4: Prescaler Mode Register (PRM2) Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
PRM2	ES21	ES20	ES11	ES10	ES01	ES00	PRM01	PRM00	FF66H	00H	R/W

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

PRM01	PRM00	Count Clock Selection
0	0	$fx/2^2$
0	1	$fx/2^3$
1	0	$fx/2^5$
1	1	$fx/2^7$

Caution: Timer operation must be stopped before setting PRM2.

8.4 16-Bit Timer 2 Operations

8.4.1 Pulse width measurement operations

It is possible to measure the pulse width of the signals input to the TI20/P30 to TI22/P32 pins by using the 16-bit timer register (TM2). TM2 is used in free-running mode.

(1) Pulse width measurement with free-running counter and one capture register (TI20)

When the edge specified by the prescaler mode register (PRM2) is input to the TI20/P30 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 8-5: Configuration Diagram for Pulse Width Measurement by Using the Free Running Counter

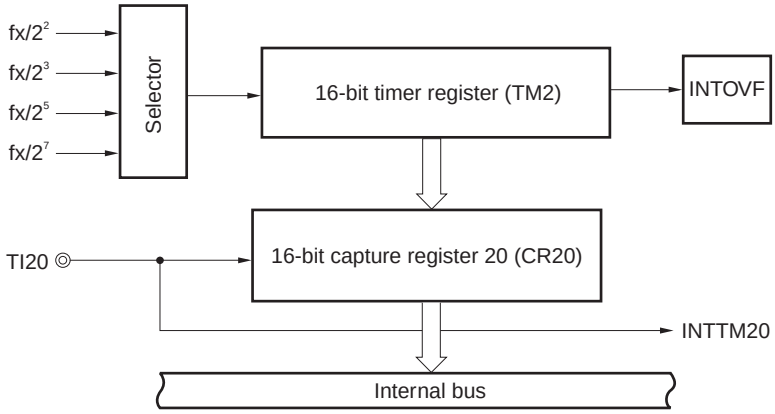
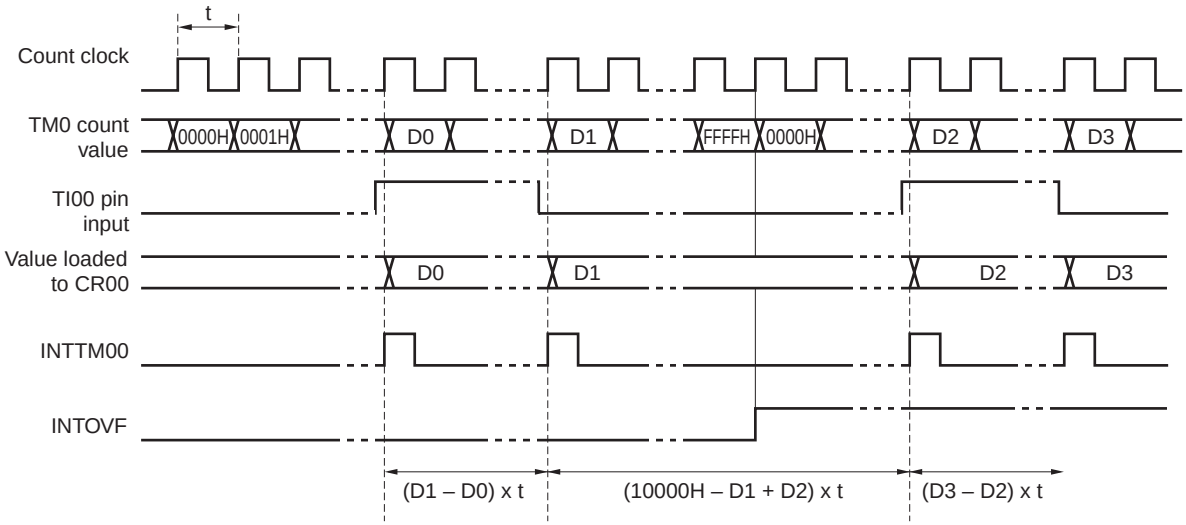


Figure 8-6: Timing of Pulse Width Measurement Operation by Using the Free Running Counter and One Capture Register (with Both Edges Specified)



(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/P30 to TI22/P32 pins.

When the edge specified by bits 2 and 3 (ES00 and ES01) of prescaler mode register (PRM2) is input to the TI20/P30 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/P31 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/P32 pin, the value of TM2 is taken into 16-bit capture register 22 (CR22) and external interrupt request signal (INTTM22) is set.

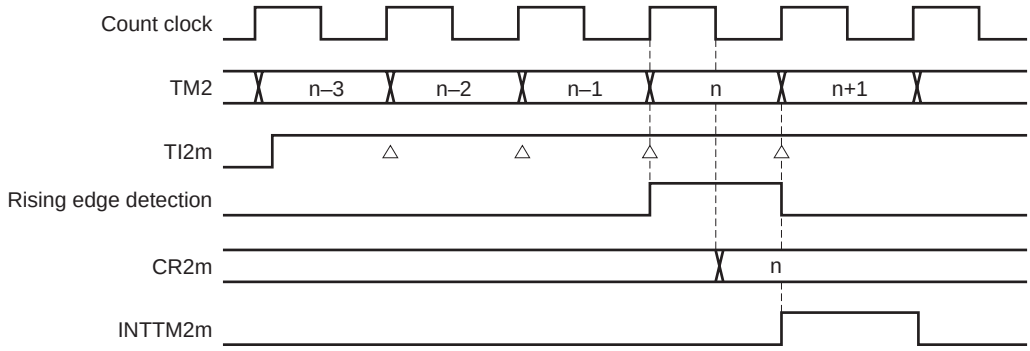
Any of three edge specifications can be selected - rising, falling, or both edges - as the valid edges for the TI20/P30 to TI22/P32 pins by means of bits 2 and 3 (ES00 and ES01), bits 4 and 5 (ES10 and ES11), and bits 6 and 7 (ES06 and ES07) of PRM2, respectively.

For TI20/P30 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**

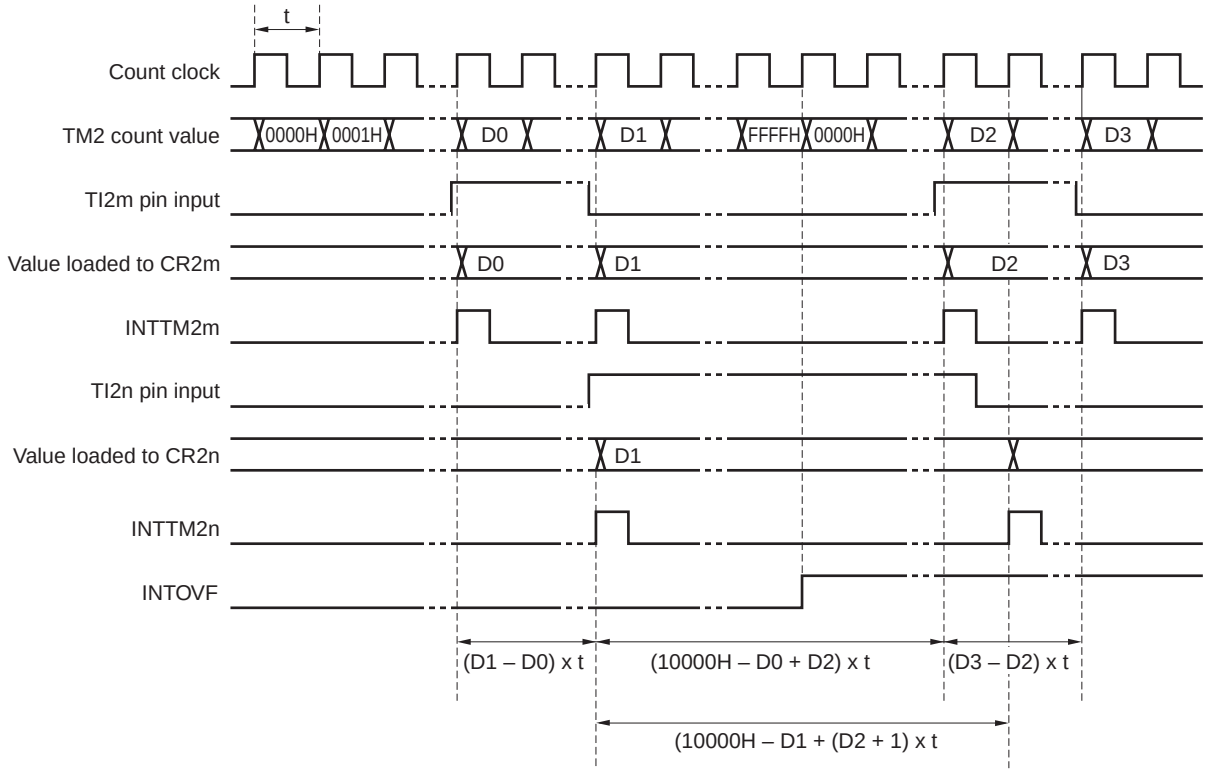
  Capture register operation in capture trigger input is shown.

Figure 8-7: CR2m Capture Operation with Rising Edge Specified



Remark: m = 0 to 2

Figure 8-8: Timing of Pulse Width Measurement Operation by Free Running Counter (with Both Edges Specified)



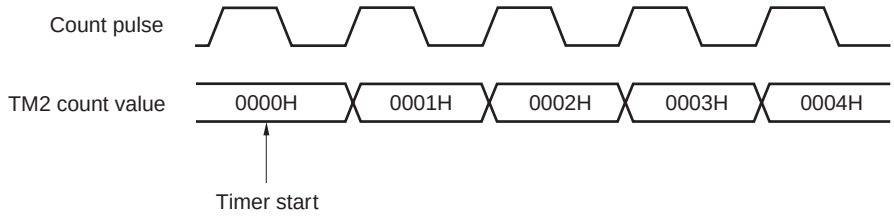
Remark: m = 0 to 2, n = 1, 2

8.5 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) is started asynchronously with the count pulse.

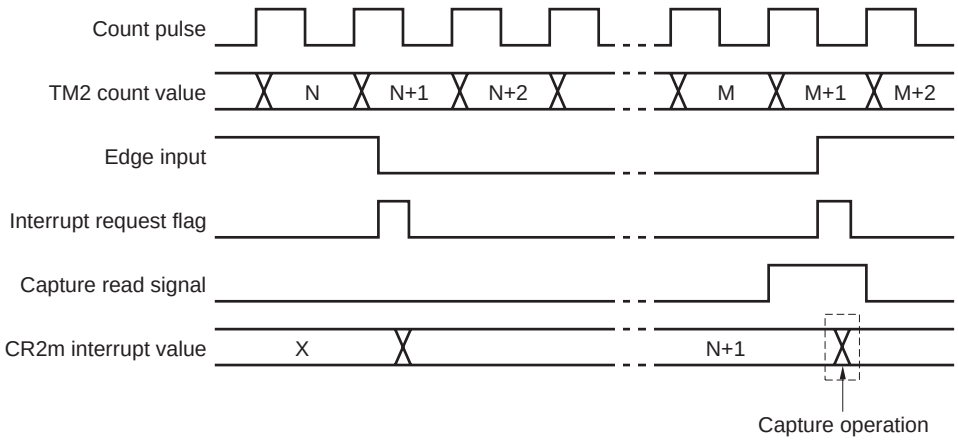
Figure 8-9: 16-Bit Timer Register Start Timing



(2) Capture register data retention timings

If the valid edge of the TI2m/P3m pin is input during the 16-bit capture register 0m (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 8-10: Capture Register Data Retention Timing



Remark: m = 0 to 2

(3) Valid edge setting

Set the valid edge of the TI2m/P3m pin after setting bit 2 (TMC02) 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.

[Memo]

Chapter 9 8-Bit Timer/Event Counters 50 and 51

9.1 8-Bit Timer/Event Counters 5 and 6 Functions

The 8-bit timer event counters 5 and 6 (TM5, TM6) have the following functions.

- Interval timer
- External event counter
- Square-wave output
- PWM output

(1) 8-bit interval timer

Interrupts are generated at the preset time intervals.

Table 9-1: 8-Bit Timer/Event Counter 50 Interval Times

Minimum Interval Width	Maximum Interval Width	Resolution
1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
2^9 x 1/fx (64 μs)	2^{17} x 1/fx (16 ms)	2^9 x 1/fx (64 μs)

Table 9-2: 8-Bit Timer/Event Counter 51 Interval Times

Minimum Interval Width	Maximum Interval Width	Resolution
1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
2^{12} x 1/fx (512 μs)	2^{20} x 1/fx (131 ms)	2^{12} x 1/fx (512 μs)

Remarks: **1.** fx: Main system clock oscillation frequency

2. Values in parentheses when operated at fx = 8.0 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 9-3: 8-Bit Timer/Event Counter 50 Square-Wave Output Ranges

Minimum Pulse Width	Maximum Pulse Width	Resolution
1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
2^9 x 1/fx (64 μs)	2^{17} x 1/fx (16 ms)	2^9 x 1/fx (64 μs)

Table 9-4: 8-Bit Timer/Event Counter 50 Square-Wave Output Ranges

Minimum Pulse Width	Maximum Pulse Width	Resolution
1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
2^{12} x 1/fx (512 μs)	2^{20} x 1/fx (131 ms)	2^{12} x 1/fx (512 μs)

Remarks: **1.** fx: Main system clock oscillation frequency

2. Values in parentheses when operated at fx = 8.0 MHz.

(4) PWM output

TM50 and TM51 can generate an 8-bit resolution PWM output.

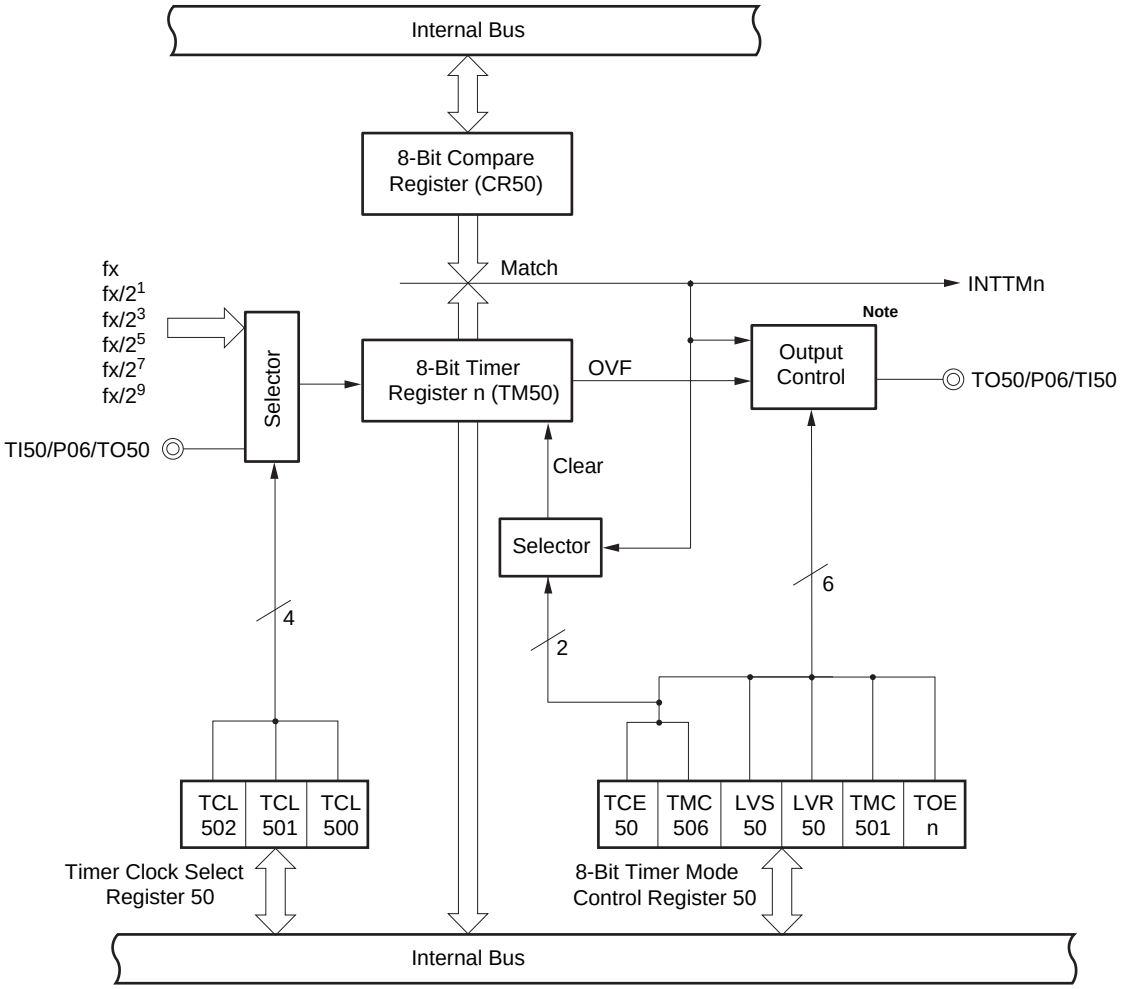
9.2 8-Bit Timer/Event Counters 50 and 51 Configurations

The 8-bit timer/event counters 50 and 51 consist of the following hardware.

Table 9-5: 8-Bit Timer/Event Counters 50 and 51 Configurations

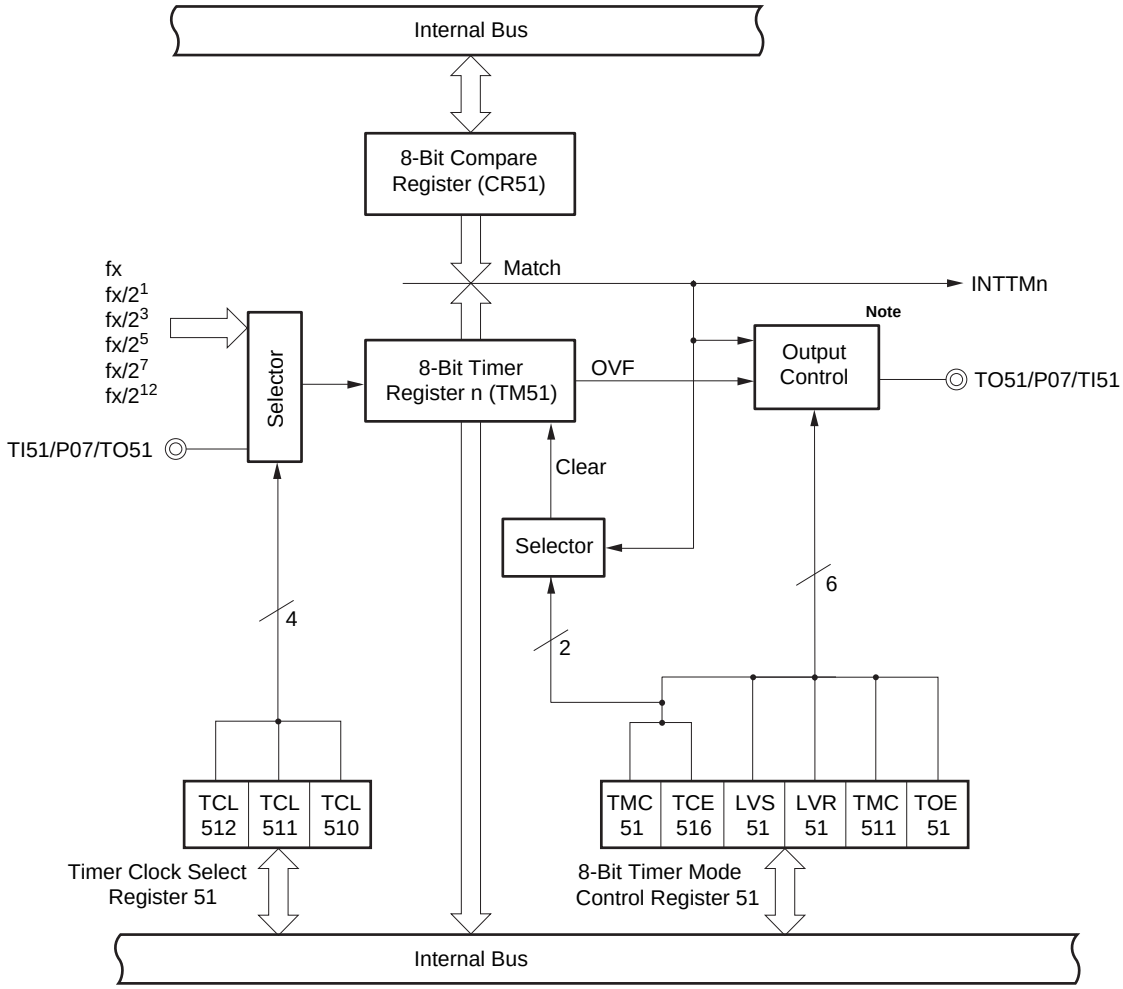
Item	Configuration
Timer register	8 bits x 2 (TM50, TM51)
Register	Compare register 8 bits x 2 (CR50, CR51)
Timer output	2 (TO50, TO51)
Control register	Timer clock select register 50 and 51 (TCL50, TCL51) 8-bit timer mode control registers 5 and 6 (TMC50, TMC51) Port mode registers 0 (PM0)

Figure 9-1: 8-Bit Timer/Event Counter 50 Block Diagram



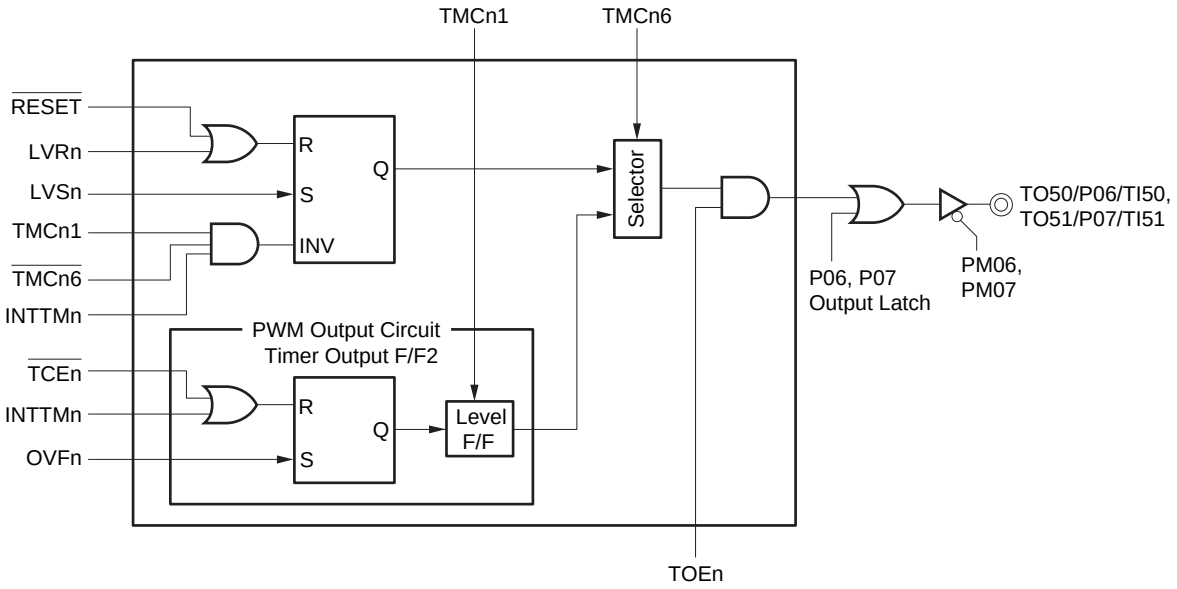
Note: Refer to Figure 9-2 for details of configurations of 8-bit timer/event counters 50 and 51 output control circuits.

Figure 9-2: 8-Bit Timer/Event Counter 51 Block Diagram



Note: Refer to Figure 9-3 for details of configurations of 8-bit timer/event counters 50 and 51 output control circuits.

Figure 9-3: 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, 51)

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 count pulses.

TM50 and TM51 are read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TM50 and TM51 to 00H.

9.3 8-Bit Timer/Event Counters 50 and 51 Control Registers

The following three 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 register 0 (PM0)

(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 9-4: Timer Clock Select Register 50 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	FF71H	00H	R/W

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_X (8.0 MHz)
0	1	1	$f_X/2^1$ (4.0 MHz)
1	0	0	$f_X/2^3$ (1.0 MHz)
1	0	1	$f_X/2^5$ (250 kHz)
1	1	0	$f_X/2^7$ (62.5 kHz)
1	1	1	$f_X/2^9$ (15.6 kHz)
Other than above			Setting prohibited

Note: When clock is input from the external, timer output (PWM output) cannot be used.

Caution: When rewriting TCL50 to other data, stop the timer operation beforehand.

Remarks: **1.** f_X: Main system clock oscillation frequency

2. TI50: 8-bit timer register 50 input pin

3. Values in parentheses apply to operation with f_X = 8.0 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 9-5: Timer Clock Select Register 51 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	FF75H	00H	R/W

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_X (8.0 MHz)
0	1	1	$f_X/2^1$ (4.0 MHz)
1	0	0	$f_X/2^3$ (1.0 MHz)
1	0	1	$f_X/2^5$ (250 kHz)
1	1	0	$f_X/2^7$ (62.5 kHz)
1	1	1	$f_X/2^{12}$ (1.9 kHz)
Other than above			Setting prohibited

Note: When clock is input from the external, timer output (PWM output) cannot be used.

Caution: When rewriting TCL51 to other data, stop the timer operation beforehand.

Remarks: **1.** f_X: Main system clock oscillation frequency

2. TI51: 8-bit timer register 51 input pin

3. Values in parentheses apply to operation with f_X = 8.0 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 a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC50 to 04H.

Figure 9-6: 8-Bit Timer Output Control Register Format

Symbol	<7>	6	5	4	<3>	<2>	1	<0>	Address	After Reset	R/W
TMC50	TCE50	TMC506	0	0	LVS50	LVR50	TMC501	TOE50	FF70H	04H	R/W

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 a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC51 to 04H.

Figure 9-7: 8-Bit Timer Output Control Register 51 Format

Symbol	<7>	6	5	4	<3>	<2>	1	<0>	Address	After Reset	R/W
TMC51	TCE51	TMC516	0	0	LVS51	LVR6	TMC511	TOE51	FF74H	04H	R/W

TOE51	8-Bit Timer/Event Counter 51 Output Control
0	Output disabled (Port mode)
6	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 4 and bit 5 to 0.

(5) Port mode register 0 (PM0)

This register sets port 0 input/output in 1-bit units.

When using the P06/TI50/TO50 and P07/TI51/TO51 pins for timer output, set PM06, PM07 and output latches of P06 and P07 to 0.

PM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM0 to FFH.

Figure 9-8: Port Mode Register 0 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W

PM0n	P0n Pin Input/Output Mode Selection (n=0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

9.4 8-Bit Timer/Event Counters 50 and 51 Operations

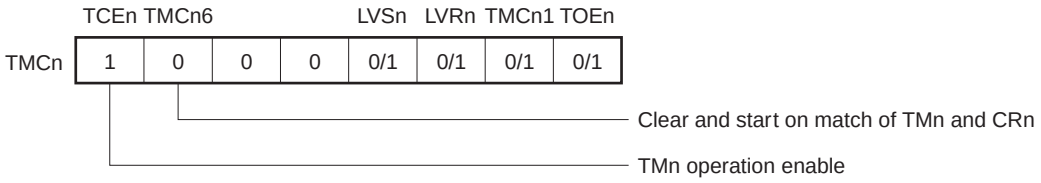
9.4.1 Interval timer operations

Setting the 8-bit timer mode control registers (TMC50 and TMC51) as shown in Figure 8-9 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 9-9: 8-Bit Timer Mode Control Register Settings for Interval Timer Operation

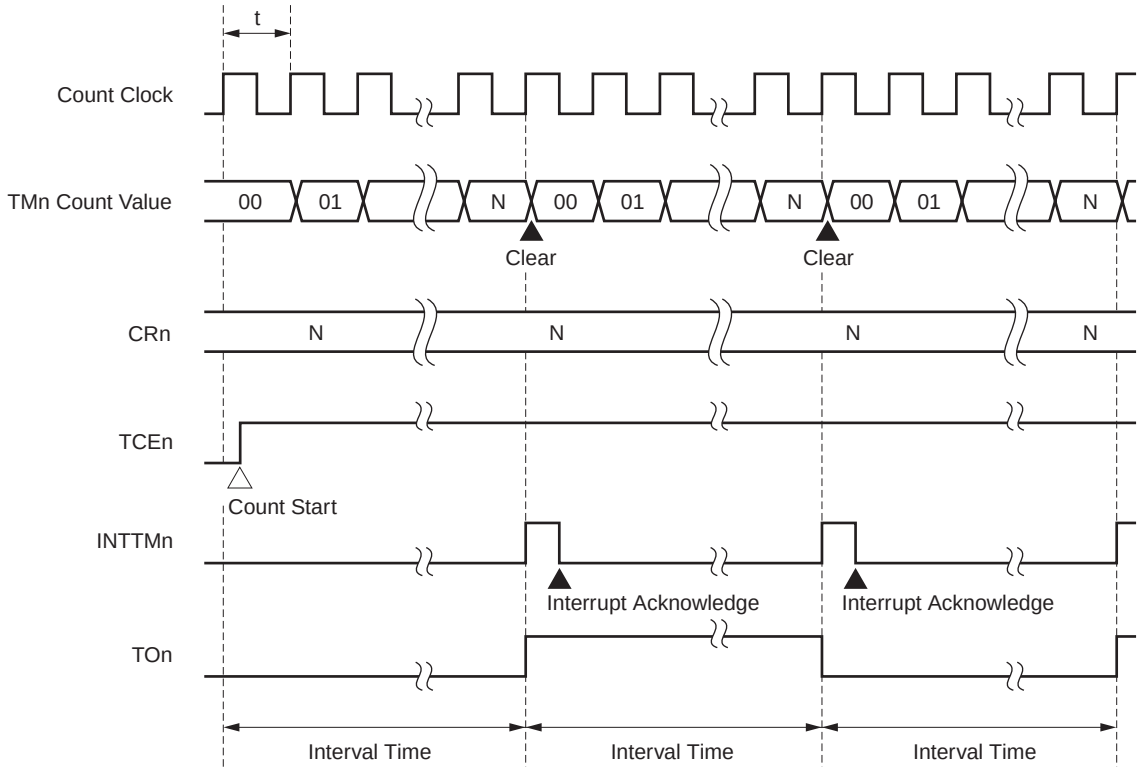


Remarks: **1.** 0/1: Setting 0 or 1 allows another function to be used simultaneously with the interval timer. See **9.3 (3), (4)** for details.

2. n = 50, 51

Figure 9-10: Interval Timer Operation Timings (1/3)

(a) When N = 00H to FFH

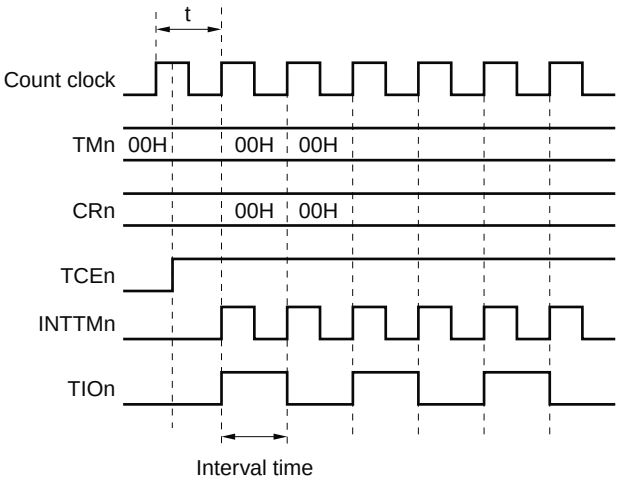


Remarks: **1.** Interval time = (N + 1) x t: N = 00H to FFH

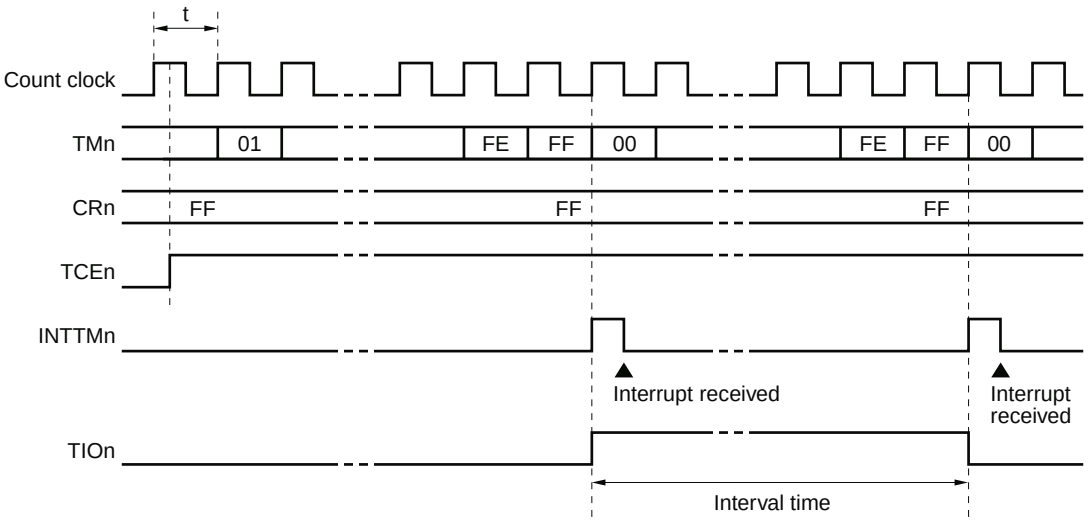
2. n = 50, 51

Figure 9-10: Interval Timer Operation Timings (2/3)

(b) When CRn = 00H



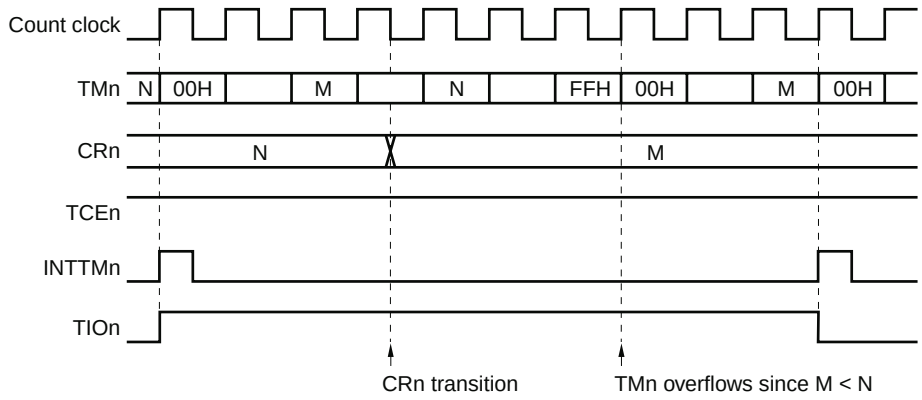
(c) When CRn = FFH



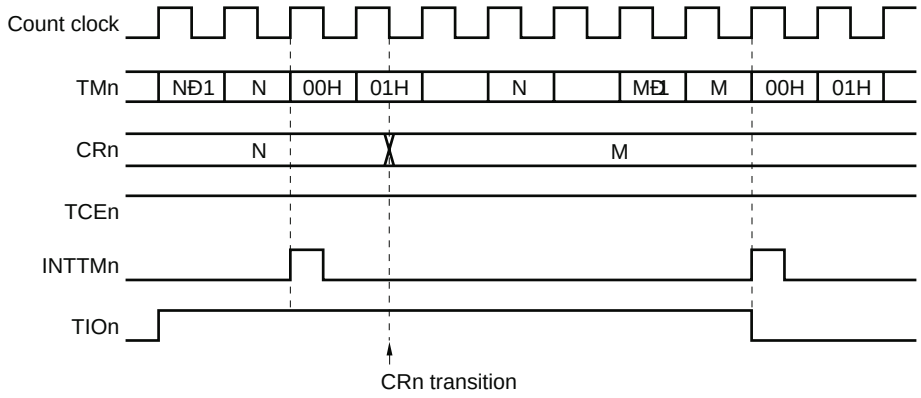
Remark: n = 50, 51

Figure 9-10: Interval Timer Operation Timings (3/3)

(d) Operated by CR5n transition (M < N)



(e) Operated by CR5n transition (M > N)



Remark: n = 50, 51

Table 9-6: 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 x Tin input cycle	Tin input edge input cycle
0	0	1	Tin input cycle	2^8 x Tin input cycle	Tin input edge input cycle
0	1	0	1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
0	1	1	2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
1	0	0	2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
1	0	1	2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
1	1	0	2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
1	1	1	2^9 x 1/fx (64 μs)	2^{17} x 1/fx (16 ms)	2^9 x 1/fx (64 μs)
Other than above			Setting prohibited		

Table 9-7: 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 x Tin input cycle	Tin input edge input cycle
0	0	1	Tin input cycle	2^8 x Tin input cycle	Tin input edge input cycle
0	1	0	1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
0	1	1	2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
1	0	0	2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
1	0	1	2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
1	1	0	2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
1	1	1	2^{12} x 1/fx (512 μs)	2^{20} x 1/fx (131 ms)	2^{12} x 1/fx (512 μs)
Other than above			Setting prohibited		

Remarks: **1.** fx: Main system clock oscillation frequency

2. Values in parentheses apply to operation with fx = 8.0 MHz.

3. n = 50, 51

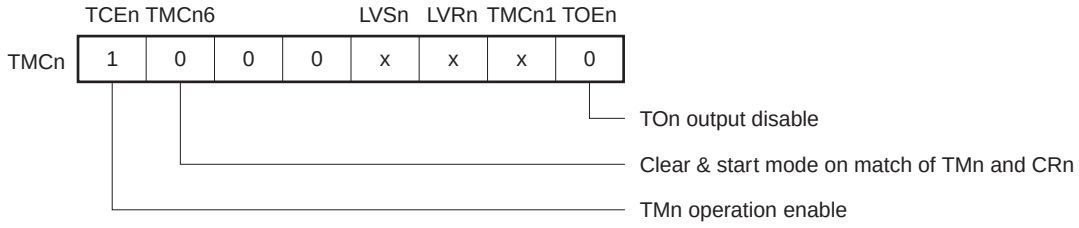
9.4.2 External event counter operation

The external event counter counts the number of external clock pulses to be input to the TI50/P06/TO50 and TI51/P07/TO51 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.

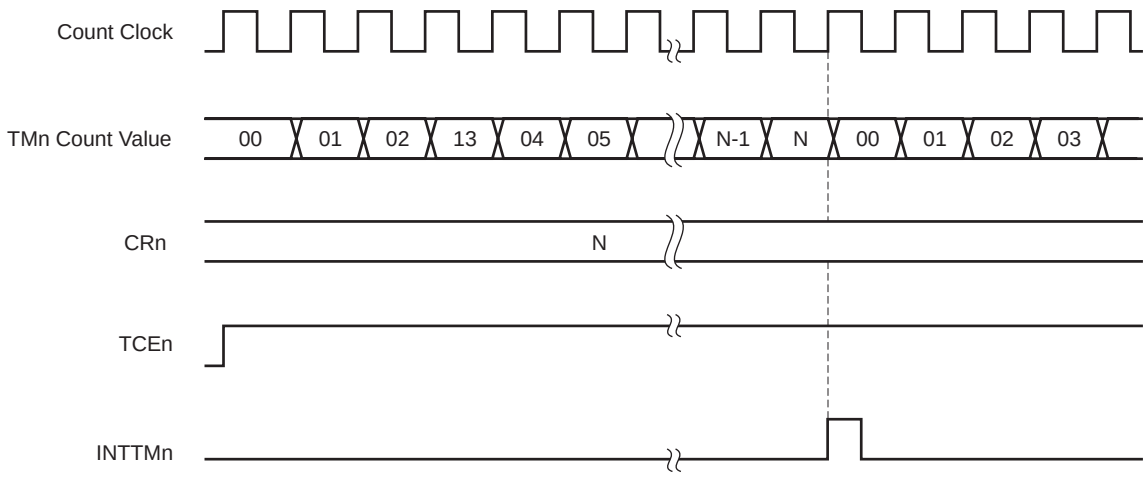
When the TM50 and TM51 counted values match the values of 8-bit compare registers (CR50 and CR51), TM50 and TM51 are cleared to 0 and the interrupt request signals (INTTM50 and INTTM51) are generated.

Figure 9-11: 8-Bit Timer Mode Control Register Setting for External Event Counter Operation



Remarks: **1.** n = 50, 51
2. x: don't care

Figure 9-12: External Event Counter Operation Timings (with Rising Edge Specified)



Remarks: **1.** N = 00H to FFH
2. n = 50, 51

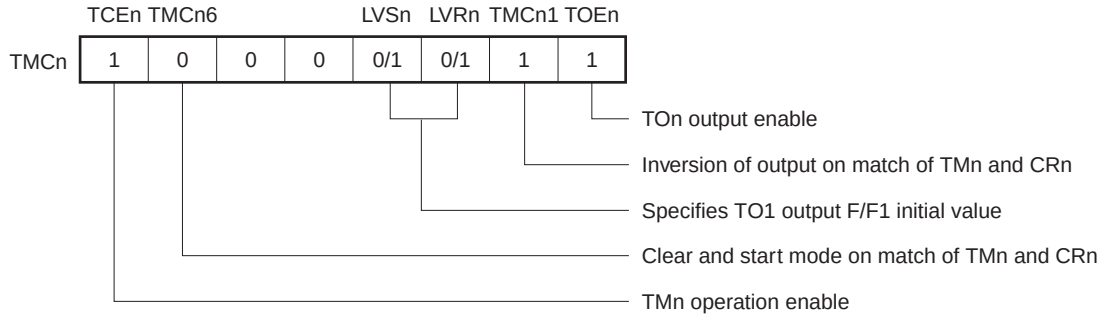
9.4.3 Square-wave output

A square wave with any selected frequency is output at intervals of the value preset to 8-bit compare registers (CR50 and CR51).

The TO50/P06/TI50 or TO51/P07/TI51 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 any selected frequency to be output.

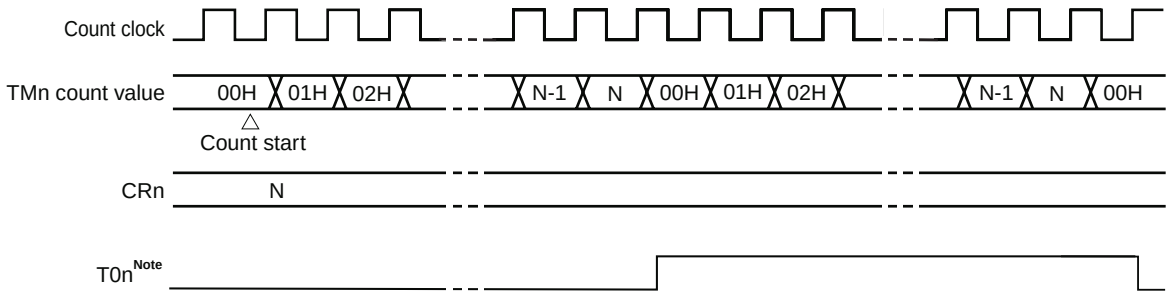
Figure 9-13: 8-Bit Timer Mode Control Register Settings for Square-Wave Output Operation



Caution: When TI50/P06/TO50 or TI51/P07/TO51 pin is used as the timer output, set port mode register (PM00 or PM07) and output latch to 0.

Remark: n = 50, 51

Figure 9-14: 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 9-8: 8-Bit Timer/Event Counters 50 Square-Wave Output Ranges

Minimum Pulse Time	Maximum Pulse Time	Resolution
1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
2^9 x 1/fx (64 μs)	2^{17} x 1/fx (16 ms)	2^9 x 1/fx (64 μs)

Table 9-9: 8-Bit Timer/Event Counters 51 Square-Wave Output Ranges

Minimum Pulse Time	Maximum Pulse Time	Resolution
1/fx (125 ns)	2^8 x 1/fx (32 μs)	1/fx (125 ns)
2^1 x 1/fx (250 ns)	2^9 x 1/fx (64 μs)	2^1 x 1/fx (250 ns)
2^3 x 1/fx (1 μs)	2^{11} x 1/fx (256 μs)	2^3 x 1/fx (1 μs)
2^5 x 1/fx (4 μs)	2^{13} x 1/fx (1 ms)	2^5 x 1/fx (4 μs)
2^7 x 1/fx (16 μs)	2^{15} x 1/fx (4 ms)	2^7 x 1/fx (16 μs)
2^{12} x 1/fx (512 μs)	2^{20} x 1/fx (131 ms)	2^{12} x 1/fx (512 μs)

Remarks: **1.** f: Main system clock oscillation frequency
2. Values in parentheses when operated at fx = 8.0 MHz.
3. n = 50, 51

9.4.4 PWM output operations

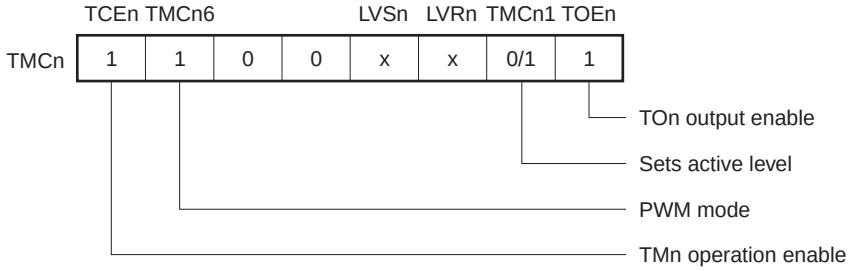
Setting the 8-bit timer mode control registers (TMC50 and TMC51) as shown in Figure 8-14 allows operation as PWM output. Pulses with the duty rate determined by the values preset in 8-bit compare registers (CR50 and CR51) output from the TO50/P06/TI50 or TO51/P07/TI51 pin.

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.

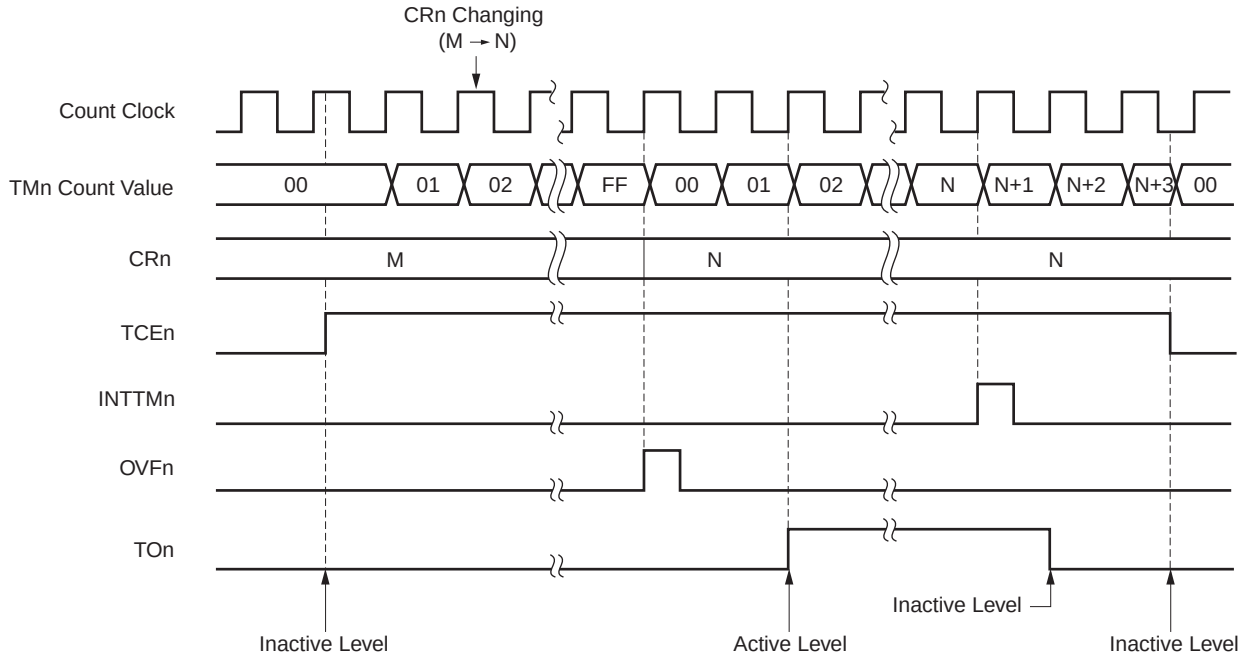
Figure 9-15: 8-Bit Timer Control Register Settings for PWM Output Operation



Remarks: **1.** n = 50, 51

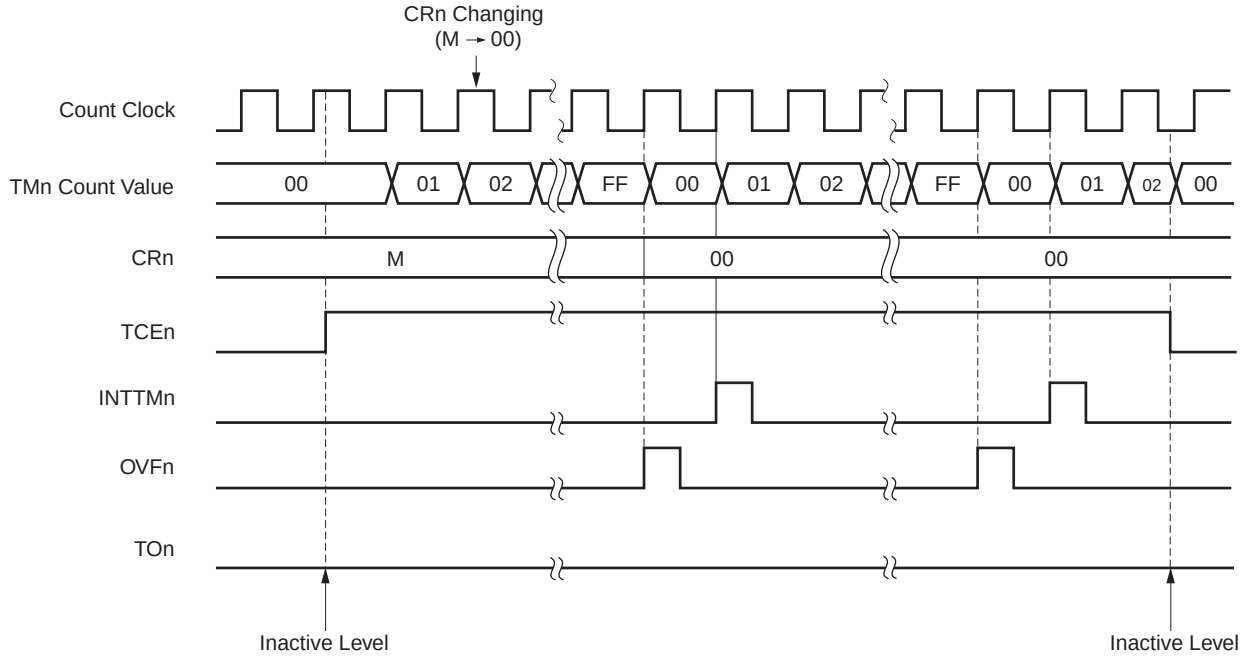
2. x: don't care

Figure 9-16: PWM Output Operation Timing (Active high setting)



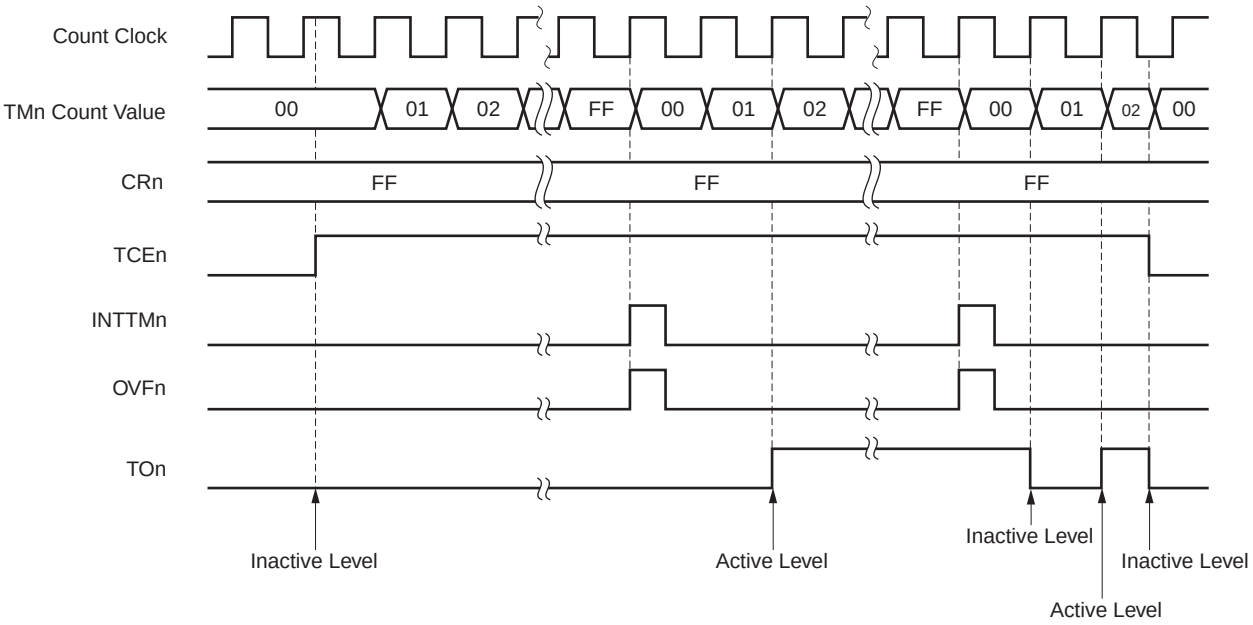
Remark: n = 50, 51

Figure 9-17: PWM Output Operation Timings (CRn0 = 00H, active high setting)



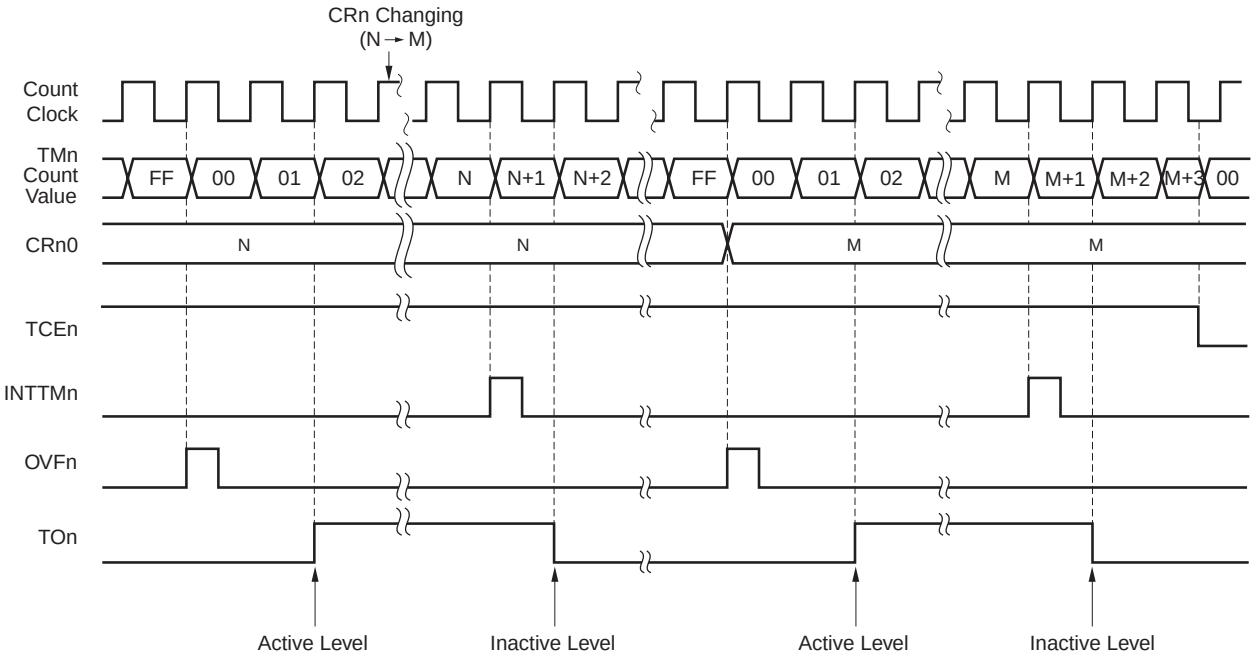
Remark: n = 50, 51

Figure 9-18: PWM Output Operation Timings (CRn = FFH, active high setting)



Remark: n = 50, 51

Figure 9-19: PWM Output Operation Timings (CRn changing, active high setting)



Remark: n = 50, 51

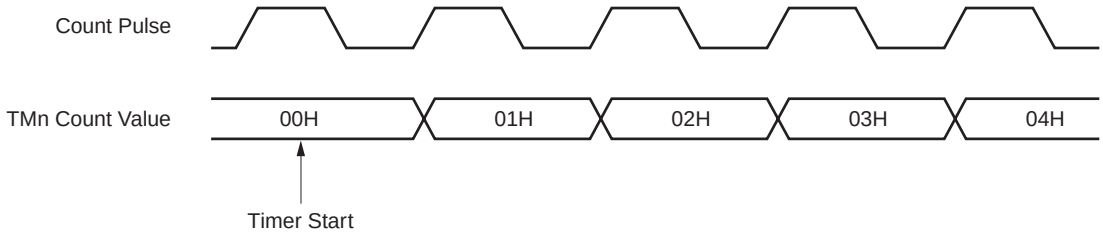
Caution: If CRn is changed during TMn operation, the value changed is not reflected until TMn overflows.

9.5 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 9-20: 8-bit Timer Registers 50 and 51 Start Timings



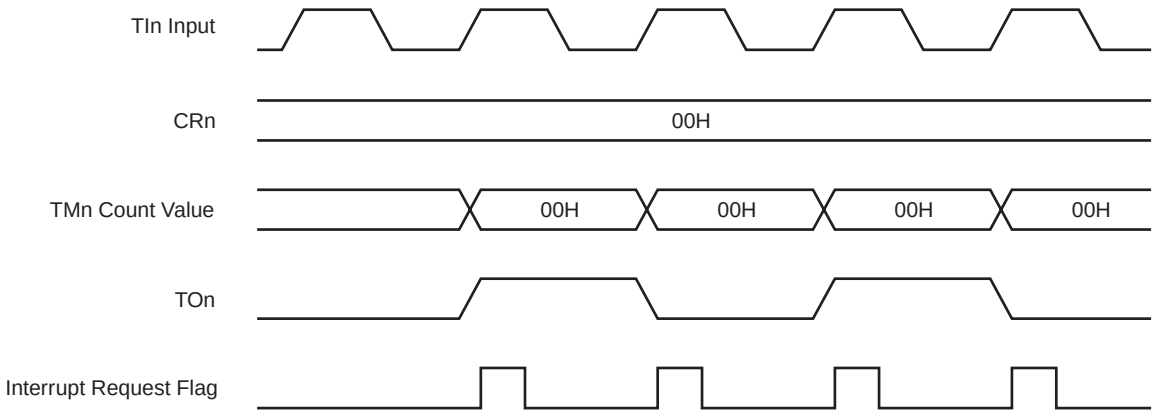
Remark: n = 50, 51

(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 9-21: External Event Counter Operation Timings

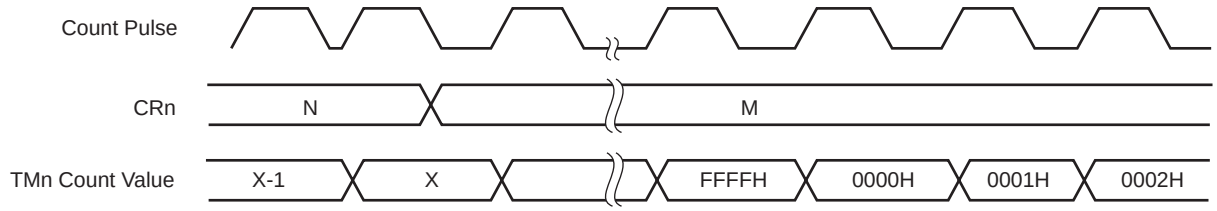


Remark: n = 50, 51

(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 9-22: Timings after Compare Register Change during Timer Count Operation



Remark: n = 50, 51

[Memo]

Chapter 10 Watch Timer

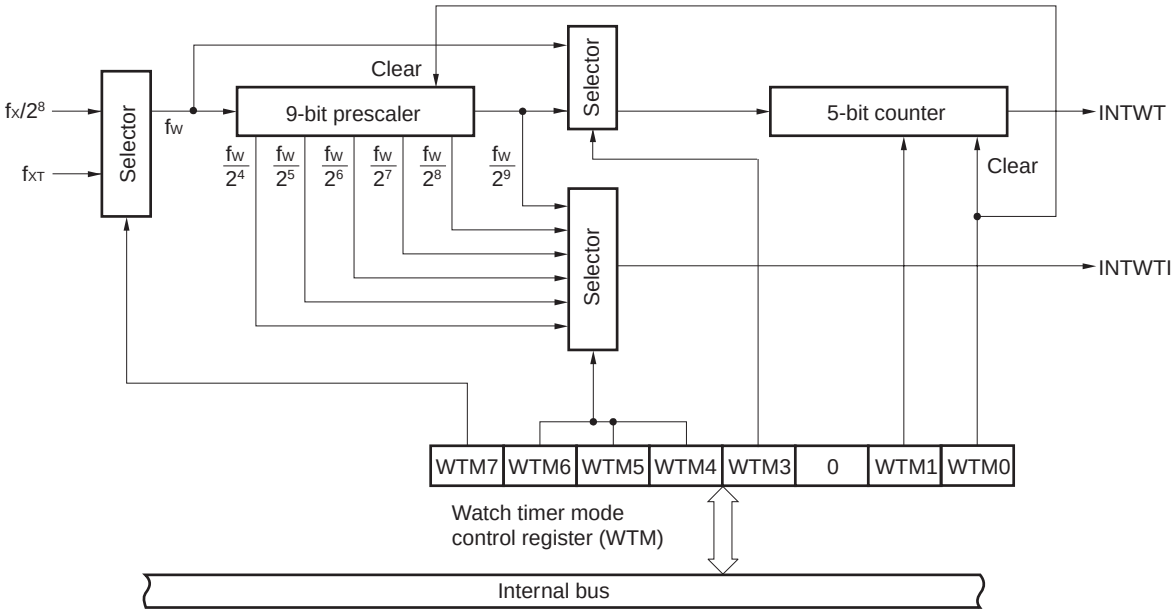
10.1 Watch Timer Functions

The watch timer has the following functions:

- Watch timer
- Interval timer

The watch timer and the interval timer can be used simultaneously.
The figure 10-1 shows Watch Timer Block Diagram.

Figure 10-1: Block Diagram of Watch Timer



(1) Watch timer

When the main system clock or subsystem clock is used, interrupt requests (INTWT) are generated at 0.5 second intervals.

(2) Interval timer

Interrupt requests (INTWTI) are generated at the preset time interval.

Table 10-1: Interval Timer Interval Time

Interval Time	When operated at f_X = 8.00 MHz	When operated at f_X = 5.00 MHz	When operated at f_{XT} = 32.768 kHz
$2^4/f_W$	512 μs	819 μs	488 μs
$2^5/f_W$	1 ms	1.6 ms	977 μs
$2^6/f_W$	2 ms	3.2 ms	1.95 ms
$2^7/f_W$	4 ms	6.55 ms	3.91 ms
$2^8/f_W$	8.19 ms	13.1 ms	7.81 ms
$2^9/f_W$	16.38 ms	26.2 ms	15.6 ms

Remark: f_X: Main system clock oscillation frequency
f_{XT}: Subsystem clock oscillation frequency

10.2 Watch Timer Configuration

The watch timer consists of the following hardware.

Table 10-2: Watch Timer Configuration

Item	Configuration
Counter	5 bits x 1
Prescaler	9 bits x 1
Control register	Watch timer mode control register (WTM)

10.3 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 a 1-bit or 8-bit memory manipulation instruction.

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

Figure 10-2: Watch Timer Mode Control Register (WTM) Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
WTM	WTM7	WTM6	WTM5	WTM4	WTM3	0	WTM1	WTM0	FF41H	00H	R/W

WTM7	Watch Timer Count Clock Selection
0	Input clock set to $fx/2^8$
1	Input clock set to f_{XT}

WTM6	WTM5	WTM4	Prescaler Interval Time Selection		
			fx = 8.08 MHz Operation	fx = 5.00 MHz Operation	fx = 32.768 kHz Operation
0	0	0	2^4/fw (512 μs)	2^4/fw (819 μs)	2^4/fw (488 μs)
0	0	1	2^5/fw (1 ms)	2^5/fw (1.6 ms)	2^5/fw (977 μs)
0	1	0	2^6/fw (2 ms)	2^6/fw (3.2 ms)	2^6/fw (1.95 ms)
0	1	1	2^7/fw (4 ms)	2^7/fw (6.55 ms)	2^7/fw (3.91 ms)
1	0	0	2^8/fw (8.19 ms)	2^8/fw (13.1 ms)	2^8/fw (7.81 ms)
1	0	1	2^9/fw (16.38 ms)	2^9/fw (26.2 ms)	2^9/fw (15.6 ms)
Other than above			Setting prohibited		

WTM3	Watch Operating Mode Selections
0	Normal operating mode (interrupt generation at 2^{14}/fw)
1	Fast feed operating mode (interrupt generation at 2^5/fw)

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. fw: Watch timer clock frequency ($fx/2^8$ or fxt)
2. fx: Main system clock oscillation frequency
3. fxt: Subsystem clock oscillation frequency

10.4 Watch Timer Operations

10.4.1 Watch timer operation

When the 32.768-kHz subsystem clock is used, the timer operates as a watch timer with a 0.5-second interval.

The watch timer is generated interrupt request at the constant time interval.

When bit 0 (WTM0) and bit 1 (WTM1) of the watch timer mode control register is set to 1, the 5-bit counter is cleared and the count operation stops.

For simultaneous operation of the interval timer, zero-second start can be achieved by setting WTM1 to 0.

10.4.2 Interval timer operation

The watch timer operates as interval timer which generates interrupt request repeatedly at an interval of the preset count value.

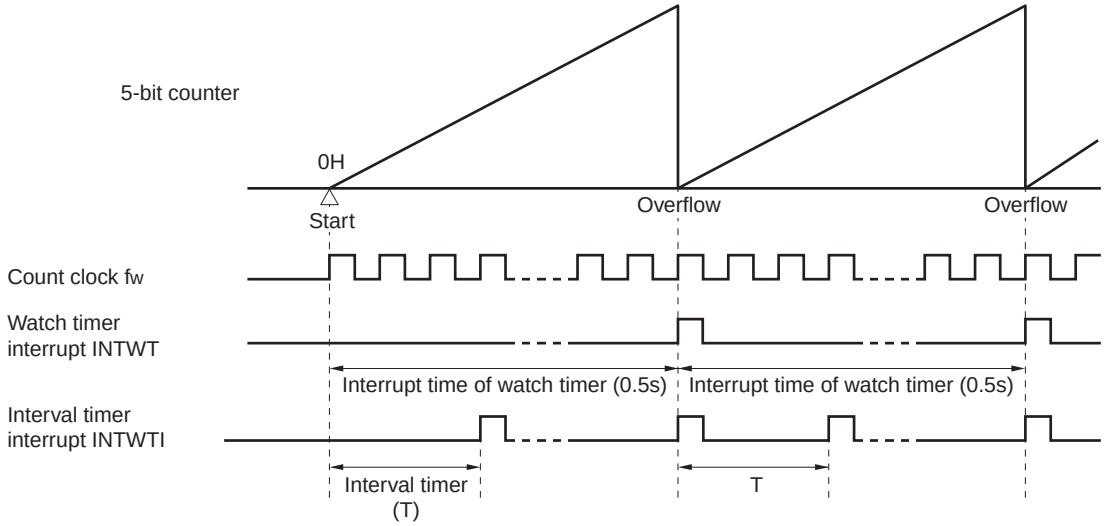
The interval time can be selected with bits 4 to 6 (WTM4 to WTM6) of the watch timer mode control register (WTM).

Table 10-3: Interval Timer Operation

WTM6	WTM5	WTM4	Interval Time	fx = 8.08 MHz Operation	fx = 5.00 MHz Operation	fx = 32.768 MHz Operation
0	0	0	2^4 x 1/fw	512 μs	819 μs	488 μs
0	0	1	2^5 x 1/fw	1 ms	1.6 ms	977 μs
0	1	0	2^6 x 1/fw	2 ms	3.2 ms	1.95 ms
0	1	1	2^7 x 1/fw	4 ms	6.55 ms	3.91 ms
1	0	0	2^8 x 1/fw	8.19 ms	13.1 ms	7.81 ms
1	0	1	2^9 x 1/fw	16.38 ms	26.2 ms	15.6 ms
Other than above			Setting prohibited			

Remark: f_X: Main system clock oscillation frequency
f_{XT}: Subsystem clock oscillation frequency
f_W: Watch timer clock frequency

Figure 10-3: Operation Timing of Watch Timer/Interval Timer



Remark: fw: Watch timer clock frequency

[Memo]

Chapter 11 Watchdog Timer

11.1 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

An inadvertent program loop is detected. Upon detection of the inadvertent program loop, a non-maskable interrupt request or $\overline{\text{RESET}}$ can be generated.

Table 11-1: Watchdog Timer Inadvertent Program Overrun Detection Times

Runaway Detection Time	
2^{12} x 1/fx	2^{12} x 1/fx (512 μs)
2^{13} x 1/fx	2^{13} x 1/fx (1 ms)
2^{14} x 1/fx	2^{14} x 1/fx (2 ms)
2^{15} x 1/fx	2^{15} x 1/fx (4 ms)
2^{16} x 1/fx	2^{16} x 1/fx (8.19 ms)
2^{17} x 1/fx	2^{17} x 1/fx (16.38 ms)
2^{18} x 1/fx	2^{18} x 1/fx (32.76 ms)
2^{20} x 1/fx	2^{20} x 1/fx (131 ms)

Remark: Figures in parentheses apply to operation with fx = 8.0 MHz.

(2) Interval timer mode

Interrupts are generated at the preset time intervals.

Table 11-2: Interval Times

Interval Time	
2^{12} x 1/fx	2^{12} x 1/fx (512 μs)
2^{13} x 1/fx	2^{13} x 1/fx (1 ms)
2^{14} x 1/fx	2^{14} x 1/fx (2 ms)
2^{15} x 1/fx	2^{15} x 1/fx (4 ms)
2^{16} x 1/fx	2^{16} x 1/fx (8.19 ms)
2^{17} x 1/fx	2^{17} x 1/fx (16.38 ms)
2^{18} x 1/fx	2^{18} x 1/fx (32.76 ms)
2^{20} x 1/fx	2^{20} x 1/fx (131 ms)

Remark: Figures in parentheses apply to operation with fx = 8.0 MHz.

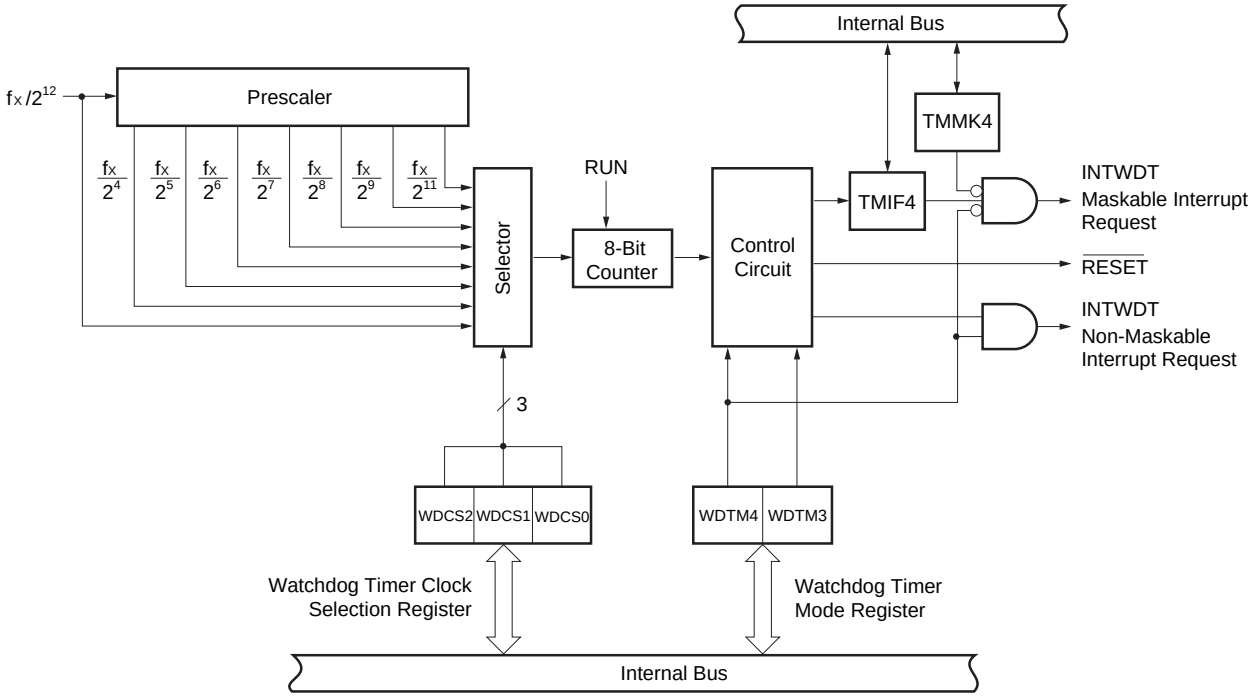
11.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 11-3: Watchdog Timer Configuration

Item	Configuration
Control register	Timer clock select register (WDCS) Watchdog timer mode register (WDTM)

Figure 11-1: Watchdog Timer Block Diagram



11.3 Watchdog Timer Control Registers

The following two types of registers are used to control the watchdog timer.

- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)

(1) Watchdog timer clock select register (WDCS)

This register sets the watchdog timer count clock.

WDCS is set with 8-bit memory manipulation instruction.

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

Figure 11-2: Watchdog Timer Clock Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Overflow time of watchdog 1 interval timer
0	0	0	$fx/2^{12}$ (512 μs)
0	0	1	$fx/2^{13}$ (1 ms)
0	1	0	$fx/2^{14}$ (2 ms)
0	1	1	$fx/2^{15}$ (4 ms)
1	0	0	$fx/2^{16}$ (8.19 ms)
1	0	1	$fx/2^{17}$ (16.38 ms)
1	1	0	$fx/2^{18}$ (32.76 ms)
1	1	1	$fx/2^{20}$ (131 ms)

Caution: When rewriting WDCS to other data, stop the timer operation beforehand.

Remarks: 1. fx: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 8.0 MHz.

(2) Watchdog timer mode register (WDTM)

This register sets the watchdog timer operating mode and enables/disables counting.

WDTM is set with a 1-bit or 8-bit memory manipulation instruction.

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

Figure 11-3: Watchdog Timer Mode Register Format

Symbol	⑦	6	5	4	3	2	1	0	Address	AfterReset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

WDTM4	WDTM3	Watchdog Timer Operation Mode Selection Note 1
0	X	Interval timer mode (Maskable interrupt occurs upon generation of an overflow)
1	0	Watchdog timer mode 1 (Non-maskable interrupt occurs upon generation of an overflow)
1	1	Watchdog timer mode 2 (Reset operation is activated upon generation of an overflow)

RUN	Watchdog Timer Operation Mode Selection Note 2
0	Count stop
1	Counter is cleared and counting starts

Notes: **1.** Once set to 1, WDTM3 and WDTM4 cannot be cleared to 0 by software.

2. Once set to 1, RUN cannot be cleared to 0 by software.
Thus, once counting starts, it can only be stopped by $\overline{\text{RESET}}$ input.

Caution: **When 1 is set in RUN so that the watchdog timer is cleared, the actual overflow time is up to 0.5 % shorter than the time set by watchdog timer clock select register.**

Remark: x = don't care.

11.4 Watchdog Timer Operations

11.4.1 Watchdog timer operation

When bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1, the watchdog timer is operated to detect any inadvertent program loop.

The watchdog timer count clock (inadvertent program loop detection time interval) can be selected with bits 0 to 2 (WDCS0 to WDCS2) of the timer clock select register (WDCS).

Watchdog timer starts by setting bit 7 (RUN) of WDTM to 1. After the watchdog timer is started, set RUN to 1 within the set overrun detection time interval. The watchdog timer can be cleared and counting is started by setting RUN to 1. If RUN is not set to 1 and the inadvertent program loop detection time is past, system reset or a non-maskable interrupt request is generated according to the WDTM bit 3 (WDTM3) value.

The watchdog timer can be cleared when RUN is set to 1.

The watchdog timer continues operating in the HALT mode but it stops in the STOP mode. Thus, set RUN to 1 before the STOP mode is set, clear the watchdog timer and then execute the STOP instruction.

Cautions 1. The actual overrun detection time may be shorter than the set time by a maximum of 0.5 %.

2. When the subsystem clock is selected for CPU clock, watchdog timer count operation is stopped.

Table 11-4: Watchdog Timer Overrun Detection Time

WDCS2	WDCS1	WDCS0	Runaway Detection Time
0	0	0	$fx/2^{12}$ (512 μs)
0	0	1	$fx/2^{13}$ (1 ms)
0	1	0	$fx/2^{14}$ (2 ms)
0	1	1	$fx/2^{15}$ (4 ms)
1	0	0	$fx/2^{16}$ (8.19 ms)
1	0	1	$fx/2^{17}$ (16.38 ms)
1	1	0	$fx/2^{18}$ (32.76 ms)
1	1	1	$fx/2^{20}$ (131 ms)

Remarks: 1. fx: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with fx = 8.0 MHz.

11.4.2 Interval timer operation

The watchdog timer operates as an interval timer which generates interrupts repeatedly at an interval of the preset count value when bit 3 (WDTM3) of the watchdog timer mode register (WDTM) is set to 0, respectively.

When the watchdog timer operates as interval timer, the interrupt mask flag (TMMK4) and priority specify flag (TMPR4) are validated and the maskable interrupt request (INTWDT) can be generated. Among maskable interrupts, the INTWDT default has the highest priority.

The interval timer continues operating in the HALT mode but it stops in STOP mode. Thus, set bit 7 (RUN) of WDTM to 1 before the STOP mode is set, clear the interval timer and then execute the STOP instruction.

Cautions:
1. **Once bit 4 (WDTM4) of WDTM is set to 1 (with the watchdog timer mode selected), the interval timer mode is not set unless $\overline{\text{RESET}}$ input is applied.**
2. **The interval time just after setting with WDTM may be shorter than the set time by a maximum of 0.5 %.**
3. **When the subsystem clock is selected for CPU clock, watchdog timer count operation is stopped.**

Table 11-5: Interval Timer Interval Time

WDCS2	WDCS1	WDCS0	Interval Time
0	0	0	$fx/2^{12}$ (512 μs)
0	0	1	$fx/2^{13}$ (1 ms)
0	1	0	$fx/2^{14}$ (2 ms)
0	1	1	$fx/2^{15}$ (4 ms)
1	0	0	$fx/2^{16}$ (8.19 ms)
1	0	1	$fx/2^{17}$ (16.38 ms)
1	1	0	$fx/2^{18}$ (32.76 ms)
1	1	1	$fx/2^{20}$ (131 ms)

Remarks:
1. fx: Main system clock oscillation frequency
2. Figures in parentheses apply to operation with fx = 8.0 MHz.

[Memo]

Chapter 12 Clock Output Control Circuit

12.1 Clock Output Control Circuit Functions

The clock output control circuit is intended for carrier output during remote controlled transmission and clock output for supply to peripheral LSI. Clocks selected with the clock output selection register (CKS) are output from the PCL/P33/SGOA pin.

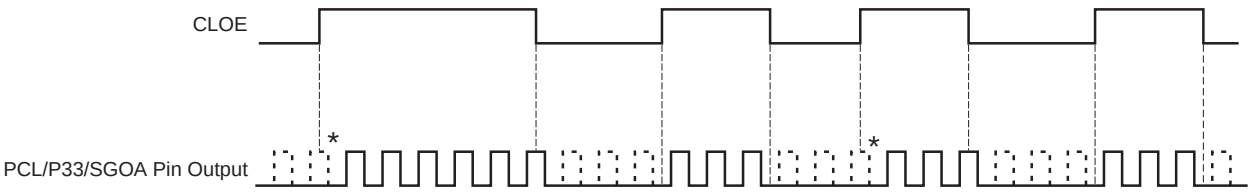
Follow the procedure below to output clock pulses.

(1) Select the clock pulse output frequency (with clock pulse output disabled) with bits 0 to 3 (CCS0 to CCS2) of CKS.
(2) Set the P33 output latch to 0.
(3) Set bit 3 (PM33) of port mode register 3 to 0 (set to output mode).
(4) Set bit 4 (CLOE) of clock output selection register to 1.

Caution: Clock output cannot be used when setting P33 output latch to 1.

Remark: When clock output enable/disable is switched, the clock output control circuit does not output pulses with small widths (See the portions marked with ***** in Figure 12-1).

Figure 12-1: Remote Controlled Output Application Example



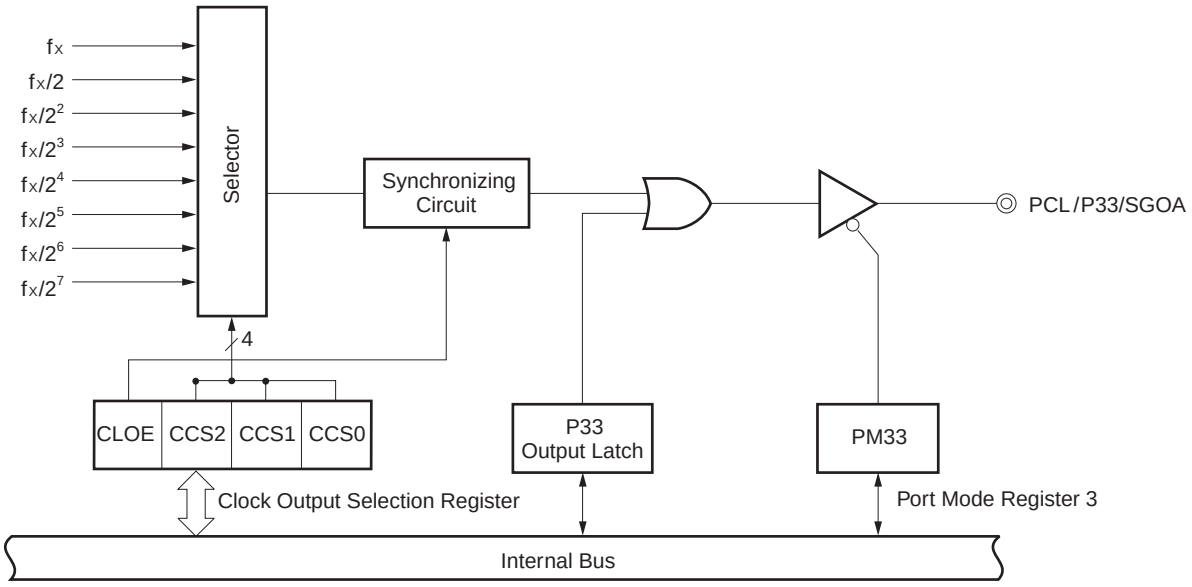
12.2 Clock Output Control Circuit Configuration

The clock output control circuit consists of the following hardware.

Table 12-1: Clock Output Control Circuit Configuration

Item	Configuration
Control register	Clock output selection register (CKS) Port mode register 3 (PM3)

Figure 12-2: Clock Output Control Circuit Block Diagram



12.3 Clock Output Function Control Registers

The following two types of registers are used to control the clock output function.

- Clock output selection register (CKS)
- Port mode register 3 (PM3)

(1) Clock Output Selection Register (CKS)

This register sets PCL output clock.

CKS is set with a 1-bit or 8-bit memory manipulation instruction.

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

Caution: When enabling PCL output, set CCS50 to CCS52, then set 1 in CLOE with a 1-bit memory manipulation instruction.

Figure 12-3: Clock Output Selection Register Format

Symbol	⑦	6	5	4	3	2	1	0	Address	AfterReset	R/W
CKS	0	0	0	CLOE	0	CCS2	CCS1	CCS0	FF40H	00H	R/W

CCS2	CCS1	CCS0	PCL Output Clock Selection
0	0	0	fx (8 MHz)
0	0	1	$f_X/2^1$ (4 MHz)
0	1	0	$f_X/2^2$ (2 MHz)
0	1	1	$f_X/2^3$ (1 MHz)
1	0	0	$f_X/2^4$ (500 kHz)
1	0	1	$f_X/2^5$ (250 kHz)
1	1	0	$f_X/2^6$ (125 kHz)
1	1	1	$f_X/2^7$ (62.5 kHz)
Other than above			Setting prohibited

CLOE	PCL Output Control
0	Output disable
1	Output enable

Remarks: **1.** fx: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with f_X = 8.0 MHz.

(2) Port mode register 3 (PM3)

This register set port 3 input/output in 1-bit units.

When using the P33/PCL/SGOA pin for clock output function, set PM33 and output latch of P33 to 0.

PM3 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM3 to FFH.

Figure 12-4: Port Mode Register 3 Format

Symbol	7	6	5	4	3	2	1	0	Address	AfterReset	R/W
PM3	0	0	0	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W

PM3n	P3n pin input/output mode selection (n = 0 to 4)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

[Memo]

Chapter 13 A/D Converter

13.1 A/D Converter Functions

The A/D converter is an 8-bit resolution converter that converts analog inputs into digital values. It can control up to 8 analog input channels (ANI0 to ANI7).

This A/D converter has the following functions:

(1) A/D conversion with 8-bit resolution

One channel of analog input is selected from ANI0 to ANI7, and A/D conversion is repeatedly executed with a resolution of 8 bits. Each time the conversion has been completed, an interrupt request (INTAD) is generated.

(2) Power-fail detection function

This function is to detect a voltage drop in the battery of an automobile. The result of A/D conversion (value of the ADCR1 register) and the value of PFT register (PFT: power-fail compare threshold value register) are compared. If the condition for comparison is satisfied, the INTAD is generated.

Figure 13-1: A/D Converter Block Diagram

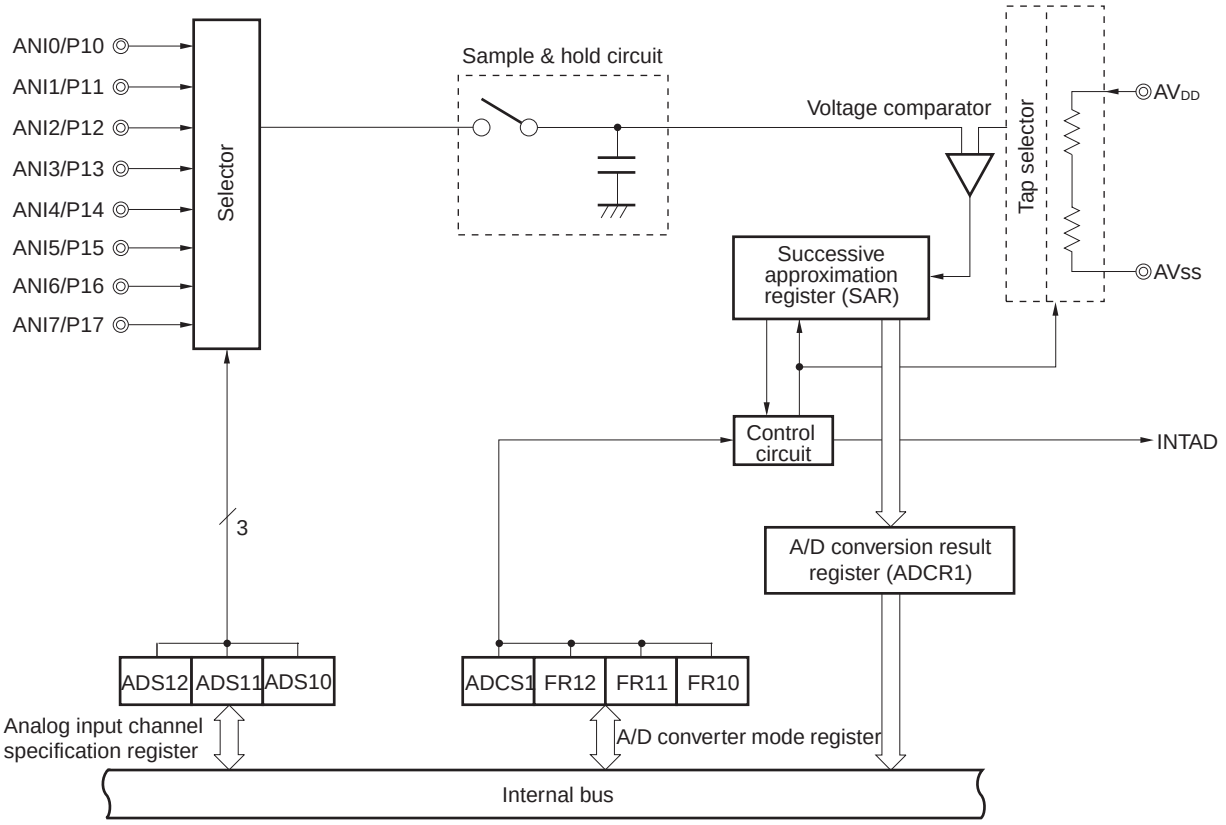
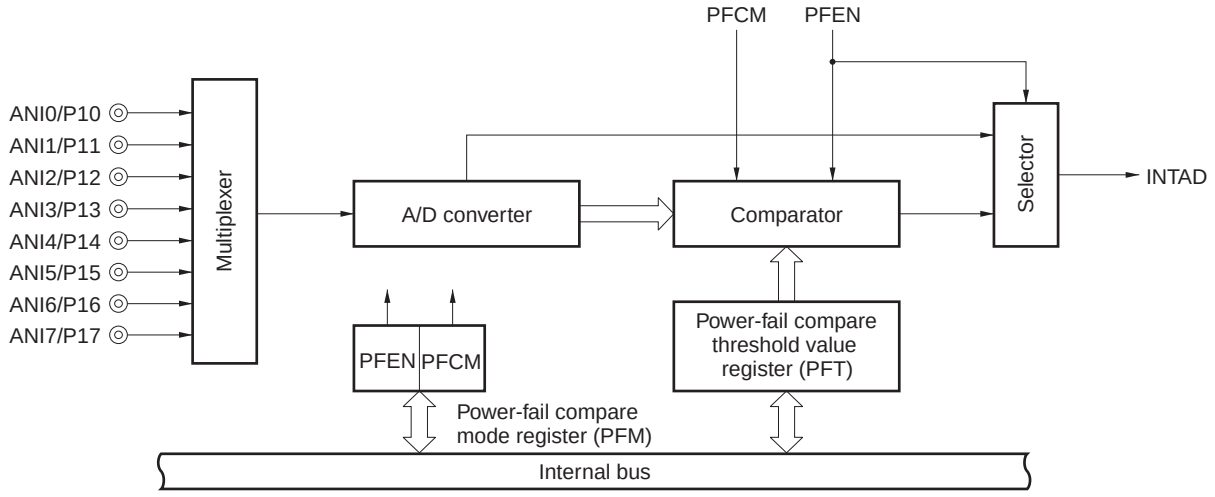


Figure 13-2: Power-Fail Detection Function Block Diagram



13.2 A/D Converter Configuration

A/D converter consists of the following hardware.

Table 13-1: A/D Converter Configuration

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Register	Successive approximation register (SAR) A/D conversion result register (ADCR1)
Control register	A/D converter mode register (ADM1) Analog input channel specification register (ADS1) Power-fail compare mode register (PFM) Power-fail compare threshold value register (PFT)

(1) Successive approximation register (SAR)

This register compares the analog input voltage value to the voltage tap (compare voltage) value applied from the series resistor string, and holds the result from the most significant bit (MSB). When up to the least significant bit (LSB) is set (end of A/D conversion), the SAR contents are transferred to the A/D conversion result register.

(2) A/D conversion result register (ADCR1)

This register holds the A/D conversion result. Each time when the A/D conversion ends, the conversion result is loaded from the successive approximation register.

ADCR1 is read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ADCR1 to 00H.

Caution: If a write operation is executed to the A/D converter mode register (ADM1) and the analog input channel specification register (ADS1) the contents of ADCR1 are undefined. Read the conversion result before a write operation is executed to ADM1 and ADS1. If a timing other than the above is used, the correct conversion result may not be read.

(3) Sample & hold circuit

The sample & hold circuit samples each analog input sequentially applied from the input circuit, and sends it to the voltage comparator. This circuit holds the sampled analog input voltage value during A/D conversion.

(4) Voltage comparator

The voltage comparator compares the analog input to the series resistor string output voltage.

(5) Series resistor string

The series resistor string is in AV_{DD} to AV_{SS}, and generates a voltage to be compared to the analog input.

(6) ANI0 to ANI7 pins

These are eight analog input pins to input analog signals to the A/D converter. ANI0 to ANI7 are alternate-function pins that can also be used for digital input.

Caution: Use ANI0 to ANI4 input voltages within the specification 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.

(7) AV_{DD} pin (Shared with AV_{REF} pin)

This pin inputs the A/D converter reference voltage and is used as the AD-converter power supply pin. The supply power has to be connected when the A/D converter is used.

It converts signals input to ANI0 to ANI7 into digital signals according to the voltage applied between AV_{DD} and AV_{SS}.

The current flowing in the series resistor string can be reduced by setting the voltage to be input to the AV_{DD} pin to AV_{SS} level in the standby mode.

(8) AV_{SS} pin

This is the GND potential pin of the A/D converter. Always keep it at the same potential as the V_{SS} pin even when not using the A/D converter.

13.3 A/D Converter Control Registers

The following 4 types of registers are used to control A/D converter.

- A/D converter mode register (ADM1)
- Analog input channel specification register (ADS1)
- Power-fail compare mode register (PFM)
- Power-fail compare threshold value register (PFT)

(1) A/D converter mode register (ADM1)

This register sets the conversion time for analog input to be A/D converted, conversion start/stop and external trigger. ADM1 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ADM1 to 00H.

Figure 13-3: A/D Converter Mode Register (ADM1) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
ADM1	ADCS1	0	FR12	FR11	FR10	0	0	0	FF98H	00H	R/W

ADCS1	A/D Conversion Operation Control
0	Stop conversion operation
1	Enable conversion operation

FR12	FR11	FR10	Conversion Time Selection [Note]
0	0	0	144/fx
0	0	1	120/fx
0	1	0	96/fx
1	0	0	72/fx
1	0	1	60/fx
1	1	0	48/fx
Other than above			Setting prohibited

Note: Set so that the A/D conversion time is 18 μs or more.

Caution: Bits 0 to 2 and bit 6 must be set to 0.

Remark: fx: Main system clock oscillation frequency

(2) Analog input channel specification register (ADS1)

This register specifies the analog voltage input port for A/D conversion.

ADS1 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ADS1 to 00H.

Figure 13-4: Analog Input Channel Specification Register (ADS1) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
ADS1	0	0	0	0	0	ADS12	ADS11	ADS10	FF99H	00H	R/W

ADS12	ADS11	ADS10	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

Caution: Bits 3 to 7 must be set to 0.

(3) Power-fail compare mode register (PFM)

The power-fail compare mode register (PFM) controls a comparison operation.

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

Figure 13-5: Power-Fail Compare Mode Register (PFM) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PFM	PFEN	PFCM	0	0	0	0	0	0	FF9AH	00H	R/W

PFEN	Enables Power-Fail Comparison
0	Disables power-fail comparison (used as normal A/D converter)
1	Enables power-fail comparison (used to detect power failure)

PFCM		Power-Fail Compare Mode Selection
0	ADCR1 ≥ PFT	Generates interrupt request signal INTAD
	ADCR1 < PFT	Does not generate interrupt request signal INTAD
1	ADCR1 ≥ PFT	Does not generate interrupt request signal INTAD
	ADCR1 < PFT	Generates interrupt request signal INTAD

Caution: Bits 0 to 5 must be set to 0.

(4) Power-fail compare threshold value register (PFT)

The power-fail compare threshold value register (PFT) sets a threshold value against which the result of A/D conversion is to be compared.

PFT is set with an 8-bit memory manipulation instruction.

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

Figure 13-6: Power-fail compare threshold value register (PFT)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PFT	PFT7	PFT6	PFT5	PFT4	PFT3	PFT2	PFT1	PFT0	FF9BH	00H	R/W

13.4 A/D Converter Operations

13.4.1 Basic operations of A/D converter

<1> Select one channel for A/D conversion with the analog input channel specification register (ADS1).

<2> The voltage input to the selected analog input channel is sampled by the sample & hold circuit.

<3> 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 is ended.

<4> Set bit 7 of the successive approximation register (SAR) so that the tap selector sets the series resistor string voltage tap to (1/2) AV_{DD}.

<5> The voltage difference between the series resistor string voltage tap and analog input is compared with the voltage comparator. If the analog input is greater than (1/2) AV_{DD}, the MSB of SAR remains set. If the analog input is smaller than (1/2) AV_{DD}, the MSB is reset.

<6> Next, bit 6 of SAR is automatically set, and the operation proceeds to the next comparison. The series resistor string voltage tap is selected according to the preset value of bit 7, as described below.

- Bit 7 = 1: (3/4) AV_{DD}
- Bit 7 = 0: (1/4) AV_{DD}

The voltage tap and analog input voltage are compared and bit 6 of SAR is manipulated as follows.

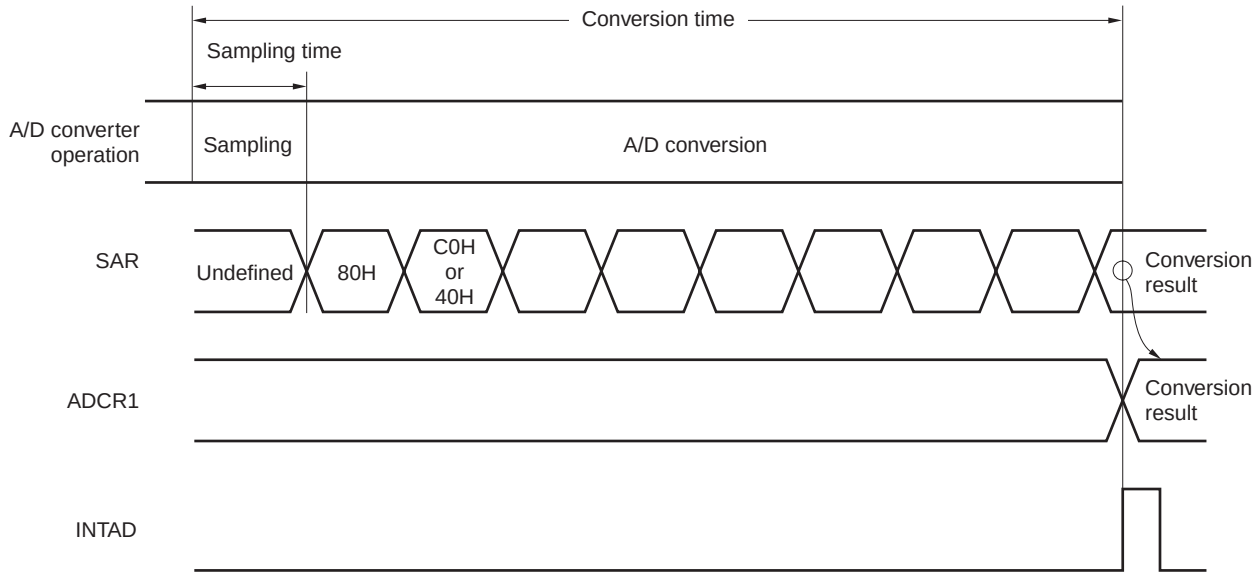
- Analog input voltage ≥ Voltage tap: Bit 6 = 1
- Analog input voltage < Voltage tap: Bit 6 = 0

<7> Comparison is continued in this way up to bit 0 of SAR.

<8> Upon completion of the comparison of 8 bits, an effective digital result value remains in SAR, and the result value is transferred to and latched in the A/D conversion result register (ADCR1). At the same time, the A/D conversion end interrupt request (INTAD) can also be generated.

Caution: The first A/D conversion value just after A/D conversion is undefined.

Figure 13-7: Basic Operation of 8-Bit A/D Converter



A/D conversion operations are performed continuously until bit 7 (ADCS1) of the A/D converter mode register (ADM1) is reset (to 0) by software.

If a write operation to the ADM1 and analog input channel specification register (ADS1) is performed during an A/D conversion operation, the conversion operation is initialized, and if the ADCS1 bit is set (to 1), conversion starts again from the beginning.

$\overline{\text{RESET}}$ input sets the A/D conversion result register (ADCR1) to 00H.

13.4.2 Input voltage and conversion results

The relation between the analog input voltage input to the analog input pins (ANI0 to ANI7) and the A/D conversion result (stored in the A/D conversion result register (ADCR1)) is shown by the following expression.

$$ADCR1 = INT\left(\frac{VIN}{AV_{DD}} \times 256 + 0.5\right)$$

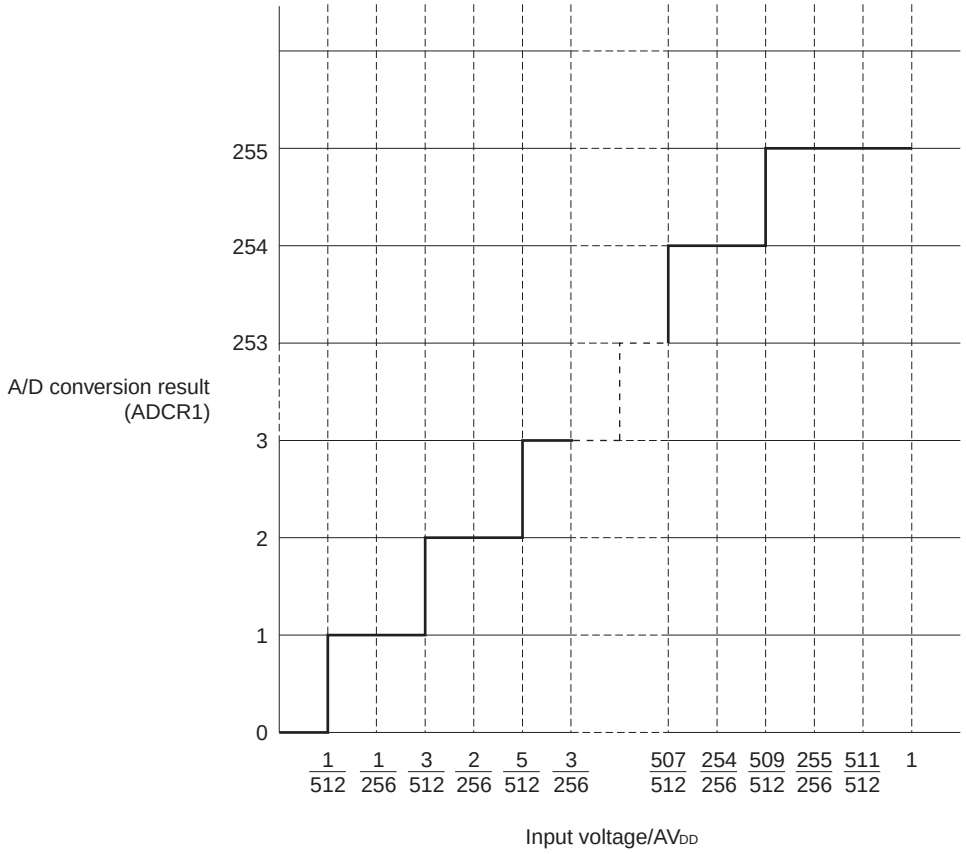
or

$$(ADCR1 - 0.5) \times \frac{AV_{DD}}{256} - V_{IN} < (ADCR1 + 0.5) \times \frac{AV_{DD}}{256}$$

where, INT( ) : Function which returns integer part of value in parentheses
V_{IN} : Analog input voltage
AV_{DD} : AV_{DD} pin voltage
ADCR1 : A/D conversion result register (ADCR1) value

Figure 13-8 shows the relation between the analog input voltage and the A/D conversion result.

Figure 13-8: Relation between Analog Input Voltage and A/D Conversion Result



13.4.3 A/D converter operation mode

The operation mode of the A/D converter is the select mode. One analog input channel is selected from among ANI0 to ANI7 with the analog input channel specification register (ADS1) and A/D conversion is performed.

The following two types of functions can be selected by setting the PFEN flag of the PFM register.

(1) Normal 8-bit A/D converter (PFEN = 0)
(2) Power-fail detection function (PFEN = 1)

(1) A/D conversion (when PFEN = 0)

When bit 7 (ADCS1) of the A/D converter mode register (ADM1) is set to 1 and bit 7 of the power-fail compare mode register (PFM) is set to 0, A/D conversion of the voltage applied to the analog input pin specified with the analog input channel specification register (ADS1) starts.

Upon the end of the A/D conversion, the conversion result is stored in the A/D conversion result register (ADCR1), and the interrupt request signal (INTAD) is generated. After one A/D conversion operation is started and ended, the next conversion operation is immediately started. A/D conversion operations are repeated until new data is written to ADS1.

If ADS1 is rewritten during A/D conversion operation, the A/D conversion operation under execution is stopped, and A/D conversion of a newly selected analog input channel is started.

If data with ADCS1 set to 0 is written to ADM1 during A/D conversion operation, the A/D conversion operation stops immediately.

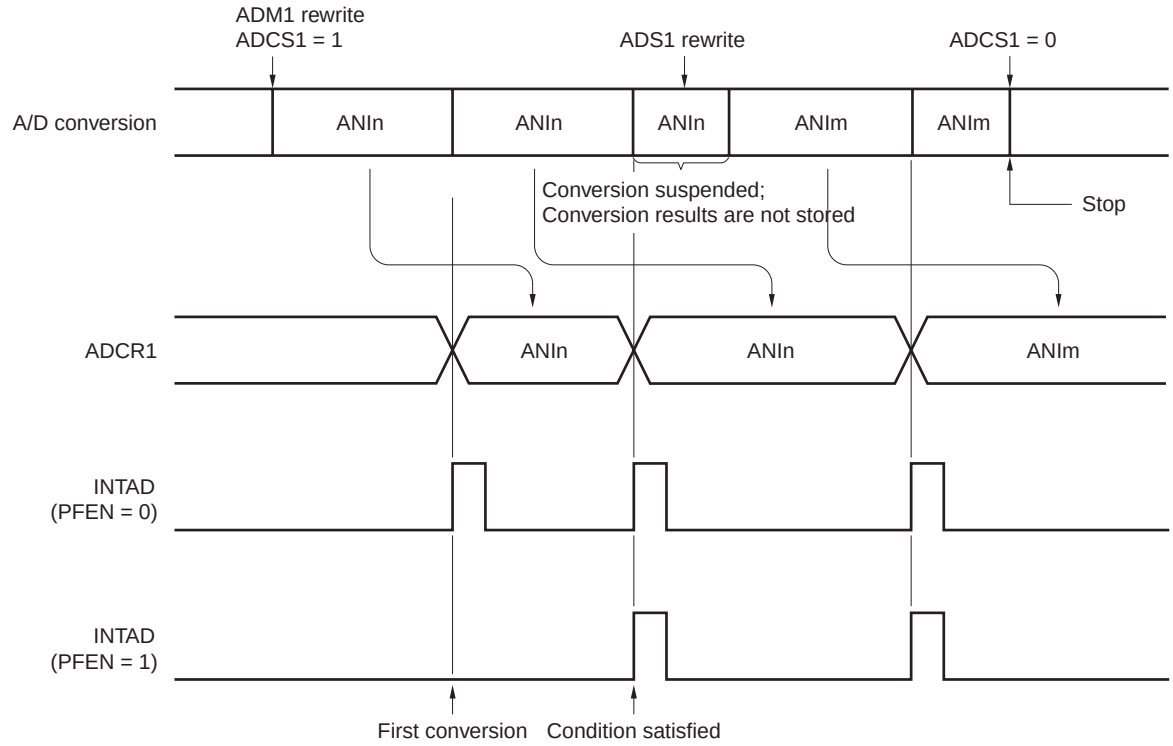
(2) Power-fail detection function (when PFEN = 1)

When bit 7 (ADCS1) of the A/D converter mode register (ADM1) and bit 7 (PFEN) of the power-fail compare mode register (PFM) are set to 1, A/D conversion of the voltage applied to the analog input pin specified with the analog input channel specification register (ADS1) starts.

Upon the end of the A/D conversion, the conversion result is stored in the A/D conversion result register (ADCR1), compared with the value of the power-fail compare threshold value register (PFT), and INTAD is generated under the condition specified by the PFCM flag of the PFM register.

Caution: When executing power-fail comparison, the interrupt request signal (INTAD) is not generated on completion of the first conversion after ADCS1 has been set to 1. INTAD is valid from completion of the second conversion.

Figure 13-9: A/D Conversion



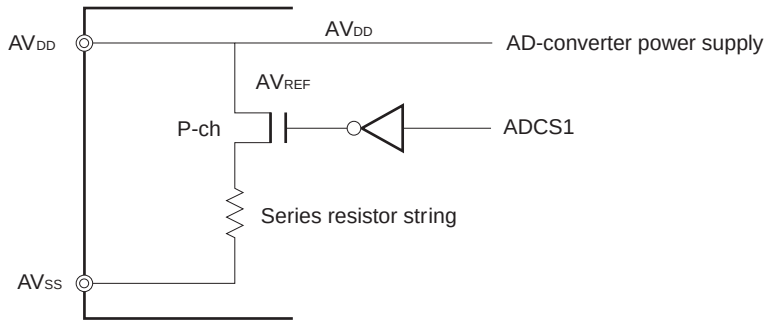
Remarks: **1.** n = 0, 1, ..., 7
2. m = 0, 1, ..., 7

13.5 A/D Converter Precautions

(1) Current consumption in standby mode

A/D converter stops operating in the standby mode. At this time, current consumption can be reduced by setting bit 7 (ADCS1) of the A/D converter mode register (ADM1) to 0 to stop conversion. Figure 13-10 shows how to reduce the current consumption in the standby mode.

Figure 13-10: Example Method of Reducing Current Consumption in Standby Mode



(2) Input range of ANI0 to ANI7

The input voltages of ANI0 to ANI7 should be within the specification range. In particular, if a voltage higher than AV_{DD} or lower than AV_{SS} is input (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.

(3) Contending operations

<1> Contention between A/D conversion result register (ADCR1) write and ADCR1 read by instruction upon the end of conversion

ADCR1 read is given priority. After the read operation, the new conversion result is written to ADCR1.

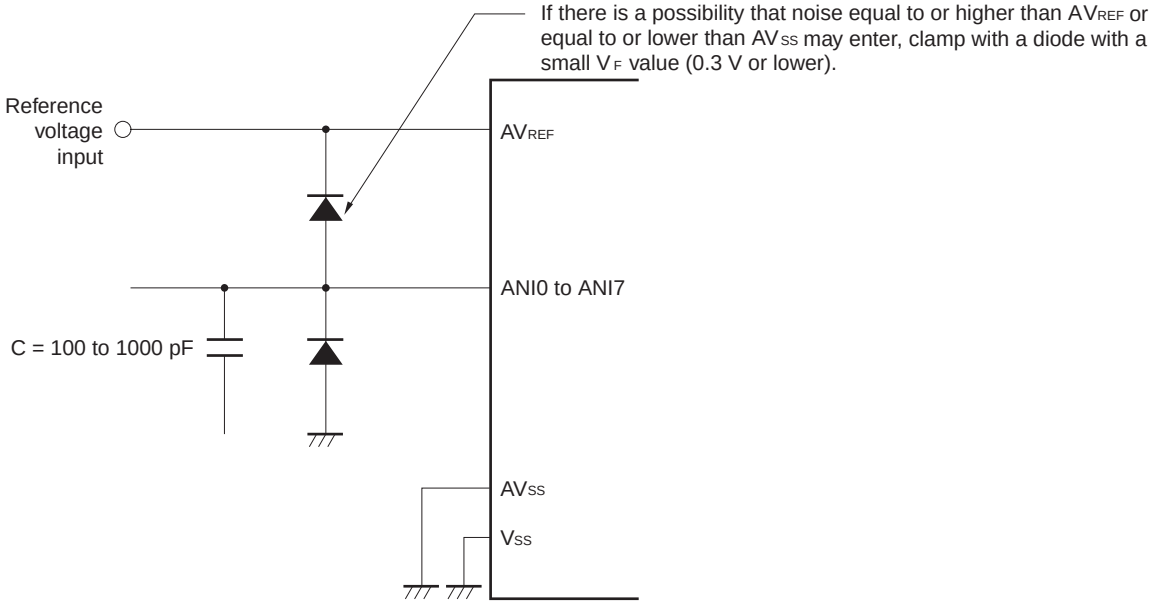
<2> Contention between ADCR1 write and A/D converter mode register (ADM1) write or analog input channel specification register (ADS1) write upon the end of conversion

ADM1 or ADS1 write is given priority. ADCR1 write is not performed, nor is the conversion end interrupt request signal (INTAD) generated.

(4) Noise countermeasures

To maintain 8-bit resolution, attention must be paid to noise input to pin AV_{DD} 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 13-11 to reduce noise.

Figure 13-11: Analog Input Pin Handling



(5) ANI0 to ANI7

The analog input pins (ANI0 to ANI7) also function as input port pins (P10 to P17).

When A/D conversion is performed with any of pins ANI0 to ANI7 selected, do not execute a port input instruction while conversion is in progress, as this may reduce the conversion resolution.

Also, if digital pulses are applied to a pin adjacent to the pin in the process of A/D conversion, the expected A/D conversion value may not be obtainable due to coupling noise. Therefore, avoid applying pulses to pins adjacent to the pin undergoing A/D conversion.

(6) AV_{REF} pin input impedance

A series resistor string of approximately 21 kΩ is connected between the AV_{DD} pin and the AV_{SS} pin. Therefore, if the output impedance of the reference voltage is high, this will result in parallel connection to the series resistor string between the AV_{DD} pin and the AV_{SS} pin, and there will be a large reference voltage error.

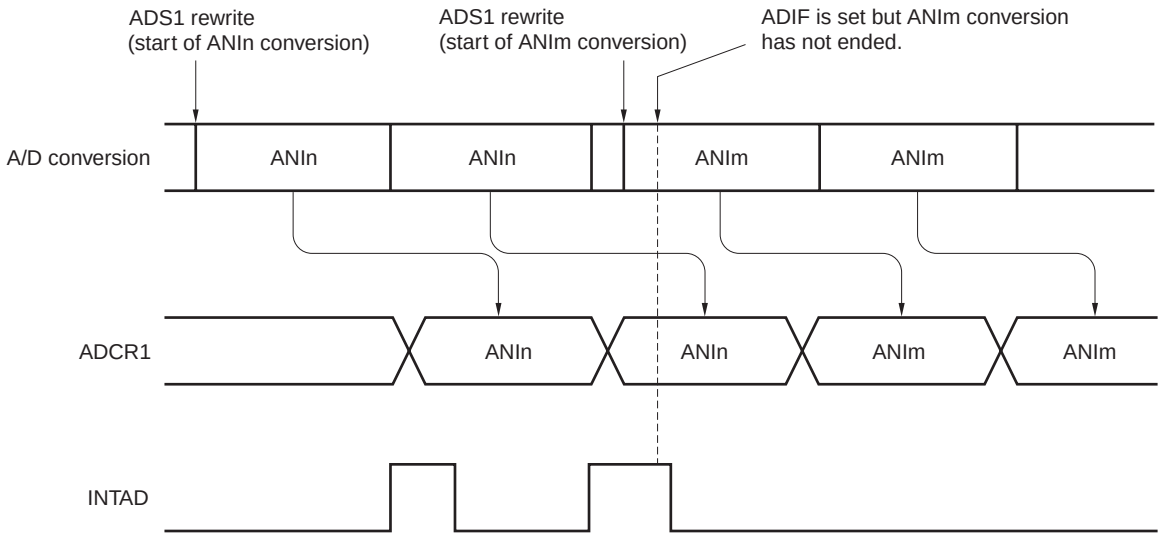
(7) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the analog input channel specification register (ADS1) is changed.

Caution is therefore required if a change of analog input pin is performed during A/D conversion. The A/D conversion result and conversion end interrupt request flag for the pre-change analog input may be set just before the ADS1 rewrite, if the ADIF is read immediately after the ADS1 rewrite, the ADIF may be set despite to the fact that the A/D conversion for the post-change analog input has not ended.

When the A/D conversion is stopped and then resumed, clear ADIF before the A/D conversion operation is resumed.

Figure 13-12: A/D Conversion End Interrupt Request Generation Timing



Remarks: **1.** n = 0, 1, ..., 7

2. m = 0, 1, ..., 7

(8) Read of A/D conversion result register (ADCR1)

When a write operation is executed to A/D converter mode register (ADM1) and analog input channel specification register (ADS1), the contents of ADCR1 are undefined. Read the conversion result before write operation is executed to ADM1, ADS1. If a timing other than the above is used, the correct conversion result may not be read.

13.6 Cautions on Emulation

To perform debugging with an in-circuit emulator (IE-78001-R-A), the D/A converter mode register (DAM0) must be set. DAM0 is a register used to set the I/O board (IE-780948-SL-EM1).

13.6.1 D/A converter mode register (DAM0)

DAM0 is necessary if the power-fail detection function is used. Unless DAM0 is set, the power-fail detection function cannot be used. DAM0 is a write-only register.

Because the IE-780948-SL-EM1 uses an external analog comparator and a D/A converter to implement part of the power-fail detection function, the reference voltage must be controlled. Therefore, set bit 0 (DACE) of DAM0 to 1 when using the power-fail detection function.

Figure 13-13: D/A Converter Mode Register (DAM0) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DAM0	0	0	0	0	0	0	0	DACE	FF9CH	00H	W

DACE	Reference Voltage Control
0	Disabled
1	Enabled (when power-fail detection function is used)

Cautions: 1. DAM0 is a special register that must be set when debugging is performed with an in-circuit emulator. Even if this register is used, the operation of the μPD780948 Subseries is not affected. However, delete the instruction that manipulates this register from the program at the final stage of debugging.

2. Bits 7 to 1 must be set to 0.

[Memo]

Chapter 14 Serial Interface Outline

14.1 Serial Interface Outline

The μPD780948 subseries incorporates three channels of serial interfaces.

Table 14-1: Differences between the Serial Interface Channels

Serial Transfer Mode	μPD780948	μPD78F0948	μPD780949	μPD78F0949
SIO 30 (3-wire serial I/O)	O	O	O	O
SIO 31 (2-wire serial I/O)	O	O	O	O
UART0	O	O	O	O

Remark: **O** : Provided
— : Not provided

[Memo]

Chapter 15 Serial Interface Channel 30

15.1 Serial Interface Channel 30 Functions

The SIO30 has the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used if serial transfer is not performed. For details, see **15.5.1 Operation Stop Mode**.

(2) 3-wire serial I/O mode (fixed as MSB first)

This is an 8-bit data transfer mode using three lines: a serial clock line ($\overline{SCK0}$), serial output line (SO0), and serial input line (SI0).

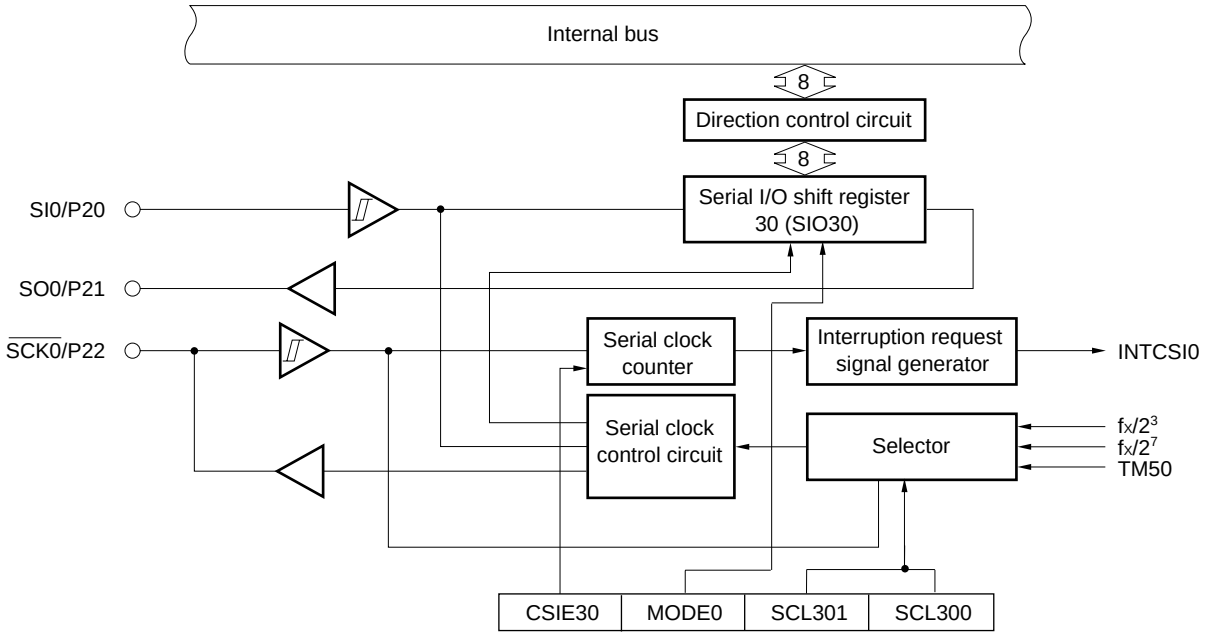
Since simultaneous transmit and receive operations are enabled in 3-wire serial I/O mode, the processing time for data transfers is reduced.

The first bit in the 8-bit data in serial transfers is fixed as the MSB.

3-wire serial I/O mode is useful for connection to a peripheral I/O device that includes a clock-synchronous serial interface, like a display controller, etc. For details see **15.5.2 Three-Wire Serial I/O Mode**.

Figure 15-1 shows a block diagram of the SIO30.

Figure 15-1: Block Diagram of SIO30



15.2 Serial Interface Channel 30 Configuration

The SIO30 includes the following hardware.

Table 15-1: Composition of SIO30

Item	Configuration
Registers	Serial I/O shift register 30 (SIO30)
Control registers	Serial operation mode register 30 (CSIM30)

(1) Serial I/O shift register 30 (SIO30)

This is an 8-bit register that performs parallel-serial conversion and serial transmit/receive (shift operations) synchronized with the serial clock.

SIO30 is set by an 8-bit memory manipulation instruction.

When “1” is set to bit 7 (CSIE30) of the serial operation mode register 30 (CSIM30), a serial operation can be started by writing data to or reading data from SIO30.

When transmitting, data written to SIO30 is output via the serial output (SO0).

When receiving, data is read from the serial input (SI0) and written to SIO30.

The $\overline{\text{RESET}}$ signal resets the register value to 00H.

Caution: Do not access SIO30 during a transmit operation unless the access is triggered by a transfer start. (Read is disabled when MODE = 0 and write is disabled when MODE = 1.)

15.3 List of SFRs (Special Function Registers)

Table 15-2: List of SFRs (Special Function Registers)

SFR name	Symbol	R/W	Units available for bit manipulation			Value when reset
			1 bit	8 bits	16 bits	
Serial operation mode register 30	CSIM30	R/W	○	○	—	00H
Serial I/O shift register 30	SIO30		—	○	—	

15.4 Serial Interface Control Registers

The SIO3 uses the following type of register for control functions.

- Serial operation mode register 30 (CSIM30)

(1) Serial operation mode register 30 (CSIM30)

This register is used to enable or disable SIO30's serial clock, operation modes, and specific operations.
CSIM30 can be set via a 1-bit or 8-bit memory manipulation instruction.
The $\overline{\text{RESET}}$ input sets the value to 00H.

Figure 15-2: Format of Serial Operation Mode Register 30 (CSIM30)

Address: FFA8H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIM30	CSIE30	0	0	0	0	MODE0	SCL301	SCL300

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port Note 1
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial operation + port function

MODE0	Transfer operation modes and flags		
	Operation mode	Transfer start trigger	P21/SO0
0	Transmit/receive mode	Write to SIO30	SO0 output
1	Receive-only mode Note 2	Read from SIO30	Port function

SCL301	SCL300	Clock selection (f_X = 8.00 MHz)
0	0	External clock input
0	1	8-bit timer 0 (TM50) output
1	0	2^3
1	1	2^7

Notes:
1. When CSIE30 = 0 (SIO30 operation stop status), the pins connected to SI0 and SO0 can be used for port functions.
2. When MODE0 = 1 (Receive mode), pin P21 can be used for port function.

15.5 Serial Interface Operations

This section explains on two modes of SIO3.

15.5.1 Operation stop mode

This mode is used if the serial transfers are not performed to reduce power consumption.
During the operation stop mode, the pins can be used as normal I/O ports as well.

(1) Register settings

The operation stop mode can be set via the serial operation mode register 30 (CSIM30).
CSIM30 can be set via 1-bit or 8-bit memory manipulation instructions.
The $\overline{\text{RESET}}$ input sets the value to 00H.

Figure 15-3: Format of Serial Operation Mode Register 30 (CSIM30)

Address: FFA8H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIM30	CSIE30	0	0	0	0	MODE0	SCL301	SCL300

CSIE30	SIO30 operation enable/disable specification		
	Shift register operation	Serial counter	Port Note
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial operation + port function

Note: When CSIE30 = 0 (SIO30 operation stop status), the pins connected to SI0 and SO0 can be used for port functions.

15.5.2 Three-wire serial I/O mode

The three-wire serial I/O mode is useful when connecting a peripheral I/O device that includes a clock-synchronous serial interface, a display controller, etc.

This mode executes the data transfer via three lines: a serial clock line ($\overline{SCK0}$), serial output line (SO0), and serial input line (SI0).

(1) Register settings

The 3-wire serial I/O mode is set via serial operation mode register 30 (CSIM30).

CSIM30 can be set via 1-bit or 8-bit memory manipulation instructions.

The $\overline{RESET}$ input set the value to 00H .

Figure 15-4: Format of Serial Operation Mode Register 30 (CSIM30)

Address: FFA8H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CSIM30	CSIE30	0	0	0	0	MODE0	SCL301	SCL300

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port Note 1
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial operation + port function

MODE0	Transfer operation modes and flags		
	Operation mode	Transfer start trigger	P21/SO0
0	Transmit/receive mode	Write to SIO30	SO0 output
1	Receive-only mode Note 2	Read from SIO30	Port function

SCL301	SCL300	Clock selection (f_X = 8.00 MHz)
0	0	External clock input
0	1	8-bit timer 0 (TM50) output
1	0	2^3
1	1	2^7

Note: **1.** When CSIE30 = 0 (SIO30 operation stop status), the pins connected to SI0 and SO0 can be used for port functions.

2. When M0DE0 = 1 (Receive mode), pin P21 can be used for port function.

(2) Communication Operations

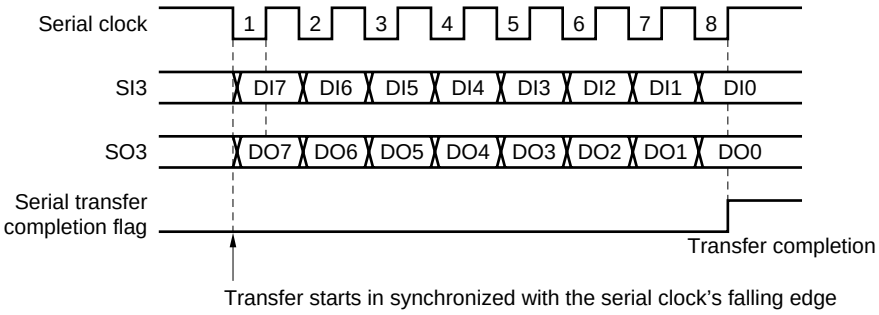
In the three-wire serial I/O mode, data is transmitted and received in 8-bit units. Each bit of data is sent or received synchronized with the serial clock.

The serial I/O shift register 30 (SIO30) is shifted synchronized with the falling edge of the serial clock.

The transmission data is held in the SO0 latch and is output from the SO0 pin. The data is received via the SI30 pin synchronized with the rising edge of the serial clock is latched to SIO30.

The completion of an 8-bit transfer automatically stops operation of SIO30 and sets a serial transfer completion flag.

Figure 15-5: Timing of Three-wire Serial I/O Mode



(3) Transfer start

A serial transfer starts when the following two conditions have been satisfied and transfer data has been set to serial I/O shift register 30 (SIO30).

- The SIO30 operation control bit (CSIE30) = 1
- After an 8-bit serial transfer, the internal serial clock is either stopped or is set to high level.
- Transmit/receive mode
  When CSIE30 = 1 and MODE0 = 0, transfer starts when writing to SIO30.
- Receive-only mode
  When CSIE30 = 1 and MODE0 = 1, transfer starts when reading from SIO30.

Caution: After the data has been written to SIO30, the transfer will not start even if the CSIE30 bit value is set to "1".

The completion of an 8-bit transfer automatically stops the serial transfer operation and sets a serial transfer completion flag.

[Memo]

Chapter 16 Serial Interface Channel 31

16.1 Serial Interface Channel 31 Functions

The SIO3 has the following two modes.

- Operation stop mode
- 2-wire serial I/O mode

(1) Operation stop mode

This mode is used if the serial transfers are not performed. For details, see **16.5.1 Operation Stop Mode**.

(2) 2-wire serial I/O mode (fixed as MSB first)

This is an 8-bit data transfer mode using two lines: a serial clock line ($\overline{SCK1}$), and serial input/output line (SIO1).

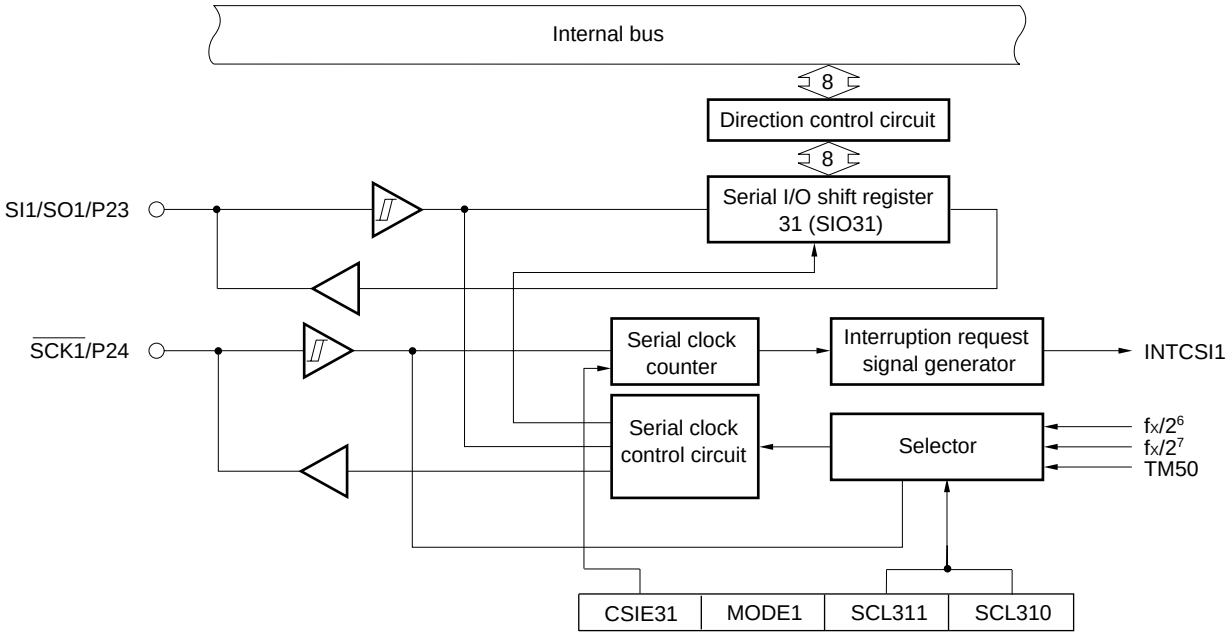
Since simultaneous transmit and receive operations are enabled in the 2-wire serial I/O mode, the processing time for data transfers is reduced.

The first bit in the 8-bit data in serial transfers is fixed as the MSB.

2-wire serial I/O mode is useful for connection to a peripheral I/O device that includes a clock-synchronous serial interface, like a display controller, etc.

Figure 16-1 shows a block diagram of the SIO30 macro.

Figure 16-1: Block Diagram of SIO3 Macro



16.2 Serial Interface Channel 31 Configuration

The SIO31 includes the following hardware.

Table 16-1: Composition of SIO30

Item	Configuration
Registers	Serial I/O shift register 30 (SIO30)
Control registers	Serial operation mode register 30 (CSIM30)

(1) Serial I/O shift register 31 (SIO31)

This is an 8-bit register that performs parallel-serial conversion and serial transmit/receive (shift operations) synchronized with the serial clock.

SIO31 is set by an 8-bit memory manipulation instruction.

When “1” is set to bit 7 (CSIE31) of the serial operation mode register 30 (CSIM31), a serial operation can be started by writing data to or reading data from SIO31.

When transmitting the data is written to SIO31 and is output via the serial output (SO31).

When receiving, data is read from the serial input (SI30) and written to SIO31.

The $\overline{\text{RESET}}$ signal resets the register value to 00H.

Caution: Do not access SIO31 during a transmit operation unless the access is triggered by a transfer start.

16.3 List of SFRs (Special Function Registers)

Table 16-2: List of SFRs (Special Function Registers)

SFR name	Symbol	R/W	Units available for bit manipulation			Value when reset
			1 bit	8 bits	16 bits	
Serial operation mode register 31	CSIM31	R/W	○	○	—	00H
Serial I/O shift register 31	SIO31		—	○	—	

16.4 Serial Interface Control Registers

The SIO31 uses the following type of register for control functions.

- Serial operation mode register 31 (CSIM31)

(1) Serial operation mode register 31 (CSIM31)

This register is used to enable or disable SIO31's serial clock, operation modes, and specific operations.

CSIM31 can be set via a 1-bit or 8-bit memory manipulation instruction.

The RESET input sets the value to 00H.

Figure 16-2: Format of Serial Operation Mode Register 31 (CSIM31)

Symbol	7	6	5	4	3	2	1	0	Address	When Reset	R/W
CSIM31	CSIE31	0	0	0	0	MODE1	SCL311	SCL310	FFAAH	00H	R/W

CSIE31	Enable/disable specification for SIO31		
	Shift register operation	Serial counter	Port [Note]
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial function

MODE1	Transfer operation modes and flags		
	Operation mode	Transfer start trigger	P23/SO1/SI1
0	Transmit/receive mode	Write to SIO31	SO1/SI1
1	Receive-only mode	Read from SIO31	SI1

SCL311	SCL310	Clock selection (fx = 8.00 MHz)
0	0	External clock input
0	1	8-bit timer 0 (TM50) output
1	0	2^6
1	1	2^7

Note: When CSIE31 = 0 (SIO31 operation stop status), the pins connected to SI1/SO1 and SCK1 can be used for port functions.

16.5 Serial Interface Channel 31 Operations

This section explains on two modes of SIO31.

16.5.1 Operation Stop Mode

This mode is used if the serial transfers is performed to reduce power consumption.
When in operation stop mode, the pins can be used as normal I/O ports as well.

(1) Register settings

Operation stop mode are set via serial operation mode register 31 (CSIM31).
CSIM31 can be set via 1-bit or 8-bit memory manipulation instructions.
The RESET input sets the value to 00H.

Figure 16-3: Format of Serial Operation Mode Register 31 (CSIM31)

Symbol	7	6	5	4	3	2	1	0	Address	When Reset	R/W
CSIM31	CSIE31	0	0	0	0	MODE1	SCL311	SCL310	FFAAH	00H	R/W

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port [Note]
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial function + port function

Note: When CSIE31 = 0 (SIO31 operation stop status), the pin connected to SI1/SO1 can be used for port function.

16.5.2 Two-wire serial I/O mode

The two-wire serial I/O mode is useful when connecting a peripheral I/O device that includes a clock-synchronous serial interface, like display controller, etc.

This mode executes data transfers via two lines: a serial clock line ($\overline{SCK1}$) and serial input/output line (SI1/SO1).

(1) Register settings

2-wire serial I/O mode is set via serial operation mode register 31 (CSIM31).

CSIM31 can be set via 1-bit or 8-bit memory manipulation instructions.

The $\overline{RESET}$ input set the value to 00H .

Figure 16-4: Format of Serial Operation Mode Register 31 (CSIM31)

Symbol	7	6	5	4	3	2	1	0	Address	When Reset	R/W
CSIM31	CSIE31	0	0	0	0	MODE1	SCL311	SCL310	FFAAH	00H	R/W

CSIE31	Enable/disable specification for SIO31		
	Shift register operation	Serial counter	Port Note
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial function

MODE1	Transfer operation modes and flags		
	Operation mode	Transfer start trigger	P23/SO1/SI1
0	Transmit/receive mode	Write to SIO31	SO1/SI1
1	Receive-only mode	Read from SIO31	SI1

SCL311	SCL310	Clock selection (fx = 8.00 MHz)
0	0	External clock input
0	1	8-bit timer 0 (TM50) output
1	0	2^6
1	1	2^7

Note: When CSIE31 = 0 (SIO31 operation stop status), the pin connected to SI1/SO1 can be used for port function.

(2) Communication Operations

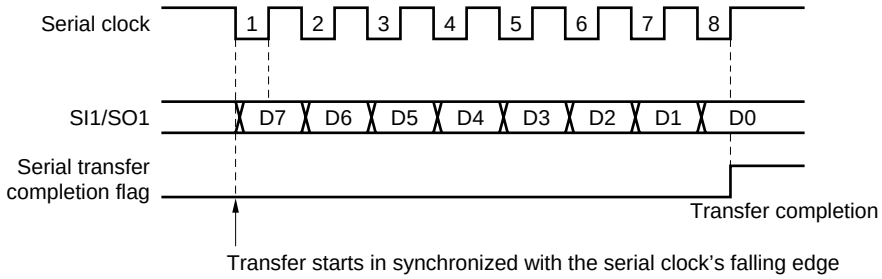
In the two-wire serial I/O mode, the data is transmitted and received in 8-bit units. Each bit of data is sent or received synchronized with the serial clock.

The serial I/O shift register 31 (SIO31) is shifted synchronized with the falling edge of the serial clock.

Transmission data is held in the SO31 latch and is output from the SO31 pin. The data that is received via the SI31 pin synchronized with the rising edge of the serial clock is latched to SIO31.

The completion of an 8-bit transfer automatically stops operation of SIO31 and sets a serial transfer completion flag.

Figure 16-5: Timing of Three-wire Serial I/O Mode



(3) Operation start

A serial operation starts when the following two conditions have been satisfied and transfer data has been set to serial I/O shift register 31 (SIO31).

- The SIO31 operation control bit (CSIE31) = 1
- After an 8-bit serial transfer, the internal serial clock is either stopped or is set to high level.
- Transmit/receive mode
  When CSIE31 = 1 and MODE1 = 0, transfer starts when writing to SIO31.
- Receive-only mode
  When CSIE31 = 1 and MODE1 = 0, transfer starts when reading from SIO31.

Caution: After data has been written to SIO31, transfer will not start even if the CSIE31 bit value is set to "1".

Completion of an 8-bit transfer automatically stops the serial transfer operation and sets a serial transfer completion flag.

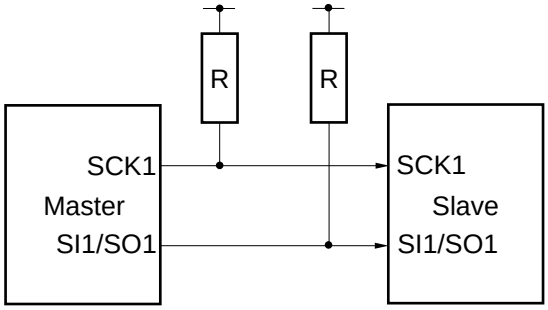
(4) 2-wire serial communication

The SCK1 and SI1/SO0 pins can be used with N-ch open drain output buffer. Therefore, the external pull-up resistors have to be used as in figure 16-6. In order to set these pins to N-ch open drain type, write 1 to PF24 and PF23 registers.

When this product is used as a master, PM23, PM24, the output latch P23 and P24 should be 0. When used as a slave, PM23 and PM24 should be 1. A static output by software is always possible by manipulating the output latches.

If it is necessary to turn off the N-ch transistor for data reception, FFH must be written to SIO1 register in advance.

Figure 16-6: 2-Wire Mode Connection



[Memo]

Chapter 17 Serial Interface UART

17.1 Serial Interface UART Functions

The serial interface UART has the following two modes.

(1) Operation stop mode

This mode is used if the serial transfer is performed to reduce power consumption.

For details, see **17.5.1 Operation Stop Mode**.

(2) Asynchronous serial interface (UART) mode

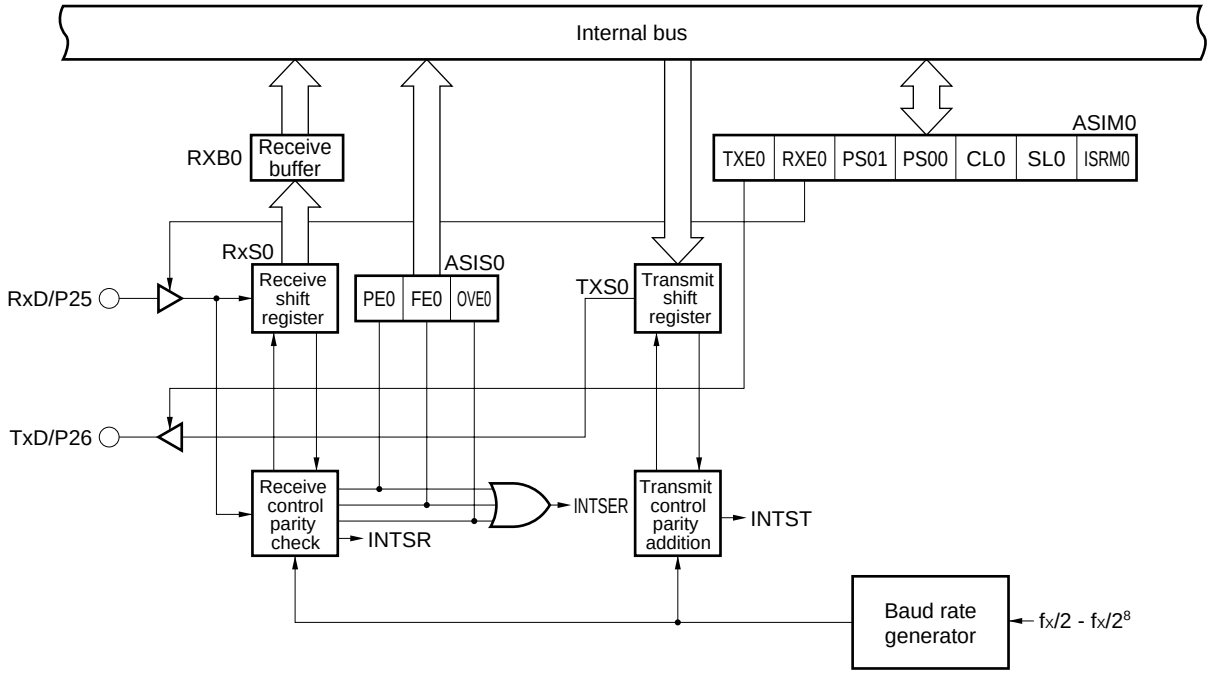
This mode enables the full-duplex operation where one byte of data is transmitted and received after the start bit.

The on-chip dedicated UART baud rate generator enables communications using a wide range of selectable baud rates.

For details, see **17.5.2 Asynchronous Serial Interface (UART) Mode**.

Figure 17-1 shows a block diagram of the UART macro.

Figure 17-1: Block Diagram of UART



17.2 Serial Interface UART Configuration

The UART includes the following hardware.

Table 17-1: Configuration of UART

Item	Configuration
Registers	Transmit shift register 1 (TXS0) Receive shift register 1 (RXS0) Receive buffer register (RXB0)
Control registers	Asynchronous serial interface mode register (ASIM0) Asynchronous serial interface status register (ASIS0) Baud rate generator control register (BRGC0)

(1) Transmit shift register 1 (TXS0)

This register is for setting the transmit data. The data is written to TXS0 for transmission as serial data.

When the data length is set as 7 bits, bits 0 to 6 of the data written to TXS0 are transmitted as serial data. Writing data to TXS0 starts the transmit operation.

TXS0 can be written via 8-bit memory manipulation instructions. It cannot be read.

When $\overline{\text{RESET}}$ is input, its value is FFH.

Caution: Do not write to TXS0 during a transmit operation.
The same address is assigned to TXS0 and the receive buffer register (RXB0). A read operation reads values from RXB0.

(2) Receive shift register 1 (RXS0)

This register converts serial data input via the RxD pin to parallel data. When one byte of the data is received at this register, the receive data is transferred to the receive buffer register (RXB0).

RXS0 cannot be manipulated directly by a program.

(3) Receive buffer register (RXB0)

This register is used to hold receive data. When one byte of data is received, one byte of new receive data is transferred from the receive shift register (RXS0).

When the data length is set as 7 bits, receive data is sent to bits 0 to 6 of RXB0. The MSB must be set to “0” in RXB0.

RXB0 can be read to via 8-bit memory manipulation instructions. It cannot be written to.

When $\overline{\text{RESET}}$ is input, its value is FFH.

Caution: The same address is assigned to RXB0 and the transmit shift register (TXS0). During a write operation, values are written to TXS0.

(4) Transmission control circuit

The transmission control circuit controls transmit operations, such as adding a start bit, parity bit, and stop bit to data that is written to the transmit shift register (TXS0), based on the values set to the asynchronous serial interface mode register (ASIM0).

(5) Reception control circuit

The reception control circuit controls the receive operations based on the values set to the asynchronous serial interface mode register (ASIM0). During a receive operation, it performs error checking, such as parity errors, and sets various values to the asynchronous serial interface status register (ASIS0) according to the type of error that is detected.

17.3 List of SFRS (Special Function Registers)

Table 17-2: List of SFRs (Special Function Registers)

SFR name	Symbol	R/W	Units available for bit manipulation			Value when reset
			1 bit	8 bits	16 bits	
Transmit shift register	TXS0	W	—	○	—	FFH
Receive buffer register	RXB0	R				
Asynchronous serial interface mode register	ASIM0	R/W	○	○	—	00H
Asynchronous serial interface status register	ASIS0	W	—	○	—	
Baud rate generator control register	BRGC0	R/W	—	○	—	

17.4 Serial Interface Control Registers

The UART uses the following three types of registers for control functions.

- Asynchronous serial interface mode register (ASIM0)
- Asynchronous serial interface status register (ASIS0)
- Baud rate generator control register (BRGC0)

(1) Asynchronous serial interface mode register (ASIM0)

This is an 8-bit register that controls the UART serial transfer operation.

ASIM0 can be set by 1-bit or 8-bit memory manipulation instructions.

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

Figure 17-2 shows the format of ASIM0.

Figure 17-2: Format of Asynchronous Serial Interface Mode Register (ASIM0)

Address: FFA0H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0

TXE0	RXE0	Operation mode	RxD0/P25 pin function	TxD0/P26 pin function
0	0	Operation stop	Port function	Port function
0	1	UART0 mode (receive only)	Serial operation	Port function
1	0	UART0 mode (transmit only)	Port function	Serial operation
1	1	UART0 mode (transmit and receive)	Serial operation	Serial operation

PS01	PS00	Parity bit specification
0	0	No parity
0	1	Zero parity always added during transmittion No parity detection during reception (parity errors do not occur)
1	0	Odd aprity
1	1	Even parity

CL0	Character length specification
0	7 bits
1	8 bits

SL0	Stop bit length specification for transmit data
0	1 bit
1	2 bits

ISRM0	Receive completion interrupt control when error occurs
0	Receive completion interrupt is issued when an error occurs
1	Receive completion interrupt is not issued when an error occurs

Caution: Do not switch the operation mode until after the current serial transmit/receive operation has stopped.

(2) Asynchronous serial interface status register (ASIS0)

When a receive error occurs during UART mode, this register indicates the type of error.

ASIS0 can be read using an 8-bit memory manipulation instruction.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 17-3: Format of Asynchronous Serial Interface Status Register (ASIS0)

Address: FFA1H When reset: 00H R

Symbol	7	6	5	4	3	2	1	0
ASIS0	0	0	0	0	0	PE0	FE0	OVE0

PE0	Parity error flag
0	No parity error
0	Parity error (Incorrect parity bit detected)

FE0	Framing error flag
0	No framing error
1	Framing error**Note 1** (Stop bit not detected)

OVE0	Overrun error flag
0	No overrun error
1	Overrun error**Note 2** (Next receive operation was completed before data was read from receive buffer register)

Notes:

1. Even if a stop bit length of two bits has been set to bit 2 (SL0) in the asynchronous serial interface mode register (ASIM0), the stop bit detection during a receive operation only applies to a stop bit length of 1 bit.

2. Be sure to read the contents of the receive buffer register (RXB0) when an overrun error has occurred.

Until the contents of RXB0 are read, further overrun errors will occur when receiving data.

(3) Baud rate generator control register (BRGC)

This register sets the serial clock for UART.

BRGC can be set via an 8-bit memory manipulation instruction.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 17-4 shows the format of BRGC.

Figure 17-4: Format of Baud Rate Generator Control Register (BRGC0)

Address: FFA2H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00

(f_X = 8.00 MHz)

TPS02	TPS01	TPS00	Source clock selection for 5-bit counter	n
0	0	0	$f_X/2^1$	1
0	0	1	$f_X/2^2$	2
0	1	0	$f_X/2^3$	3
0	1	1	$f_X/2^4$	4
1	0	0	$f_X/2^5$	5
1	0	1	$f_X/2^6$	6
1	1	0	$f_X/2^7$	7
1	1	1	$f_X/2^8$	8

MDL03	MDL02	MDL01	MDL00	Input clock selection for baud rate generator	k
0	0	0	0	f_{SCK}/16	0
0	0	0	1	f_{SCK}/17	1
0	0	1	0	f_{SCK}/18	2
0	0	1	1	f_{SCK}/19	3
0	1	0	0	f_{SCK}/20	4
0	1	0	1	f_{SCK}/21	5
0	1	1	0	f_{SCK}/22	6
0	1	1	1	f_{SCK}/23	7
1	0	0	0	f_{SCK}/24	8
1	0	0	1	f_{SCK}/25	9
1	0	1	0	f_{SCK}/26	10
1	0	1	1	f_{SCK}/27	11
1	1	0	0	f_{SCK}/28	12
1	1	0	1	f_{SCK}/29	13
1	1	1	0	f_{SCK}/30	14
1	1	1	1	Setting prohibit	—

Caution: Writing to BRGC0 during a communication operation may cause abnormal output from the baud rate generator and disable further communication operations. Therefore, do not write to BRGC0 during a communication operation.

Remarks: 1. f_{SCK}: Source clock for 5-bit counter

2. n: Value set via TPS00 to TPS02 ($1 \leq n \leq 8$)

3. k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

17.5 Serial Interface Operations

This section explains the three modes of the UART.

17.5.1 Operation stop mode

This mode is used when serial transfers are not performed to reduce power consumption.

In the operation stop mode, pins can be used as ordinary ports.

(1) Register settings

Operation stop mode settings are made via the asynchronous serial interface mode register (ASIM).

ASIM0 can be set via 1-bit or 8-bit memory manipulation instructions.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 17-5: Register Settings

Address: FFA0H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0

TXE0	RXE0	Operation mode	RxD0/PXX pin function	TxD0/PXX pin function
0	0	Operation stop	Port function	Port function
0	1	UART0 mode (receive only)	Serial operation	Port function
1	0	UART0 mode (transmit only)	Port function	Serial operation
1	1	UART0 mode (transmit and receive)	Serial operation	Serial operation

Caution: Do not switch the operation mode until after the current serial transmit/receive operation has stopped.

17.5.2 Asynchronous serial interface (UART) mode

This mode enables full-duplex operation where one byte of the data is transmitted or received after the start bit.

The on-chip dedicated UART baud rate generator enables communications by using a wide range of selectable baud rates.

(1) Register settings

The UART mode settings are made via the asynchronous serial interface mode register (ASIM0), asynchronous serial interface status register (ASIS0), and the baud rate generator control register (BRGC0).

(a) Asynchronous serial interface mode register (ASIM0)

ASIM0 can be set by 1-bit or 8-bit memory manipulation instructions.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 17-6: Asynchronous serial interface mode register (ASIM0)

Address: FFA0H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0

TXE0	PEX0	Operation mode	RxD0/PXX pin function	TxD0/PXX pin function
0	0	Operation stop	Port function	Port function
0	1	UART0 mode (receive only)	Serial operation	Port function
1	0	UART0 mode (transmit only)	Port function	Serial operation
1	1	UART0 mode (transmit and receive)	Serial operation	Serial operation

PS01	PS00	Parity bit specification
0	0	No parity
0	1	Zero parity always added during transmittion No parity detection during reception (parity errors do not occur)
1	0	Odd aprity
1	1	Even parity

CL0	Character length specification
0	7 bits
0	8 bits

SL0	Stop bit length specification for transmit data
0	1 bit
1	2 bits

ISRM0	Receive completion interrupt control when error occurs
0	Receive completion interrupt is issued when an error occurs
1	Receive completion interrupt is not issued when an error occurs

Caution: Do not switch the operation mode until after the current serial transmit/receive operation has stopped.

(b) Asynchronous serial interface status register (ASIS0)

ASIS0 can be read using an 8-bit memory manipulation instruction.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 17-7: Asynchronous serial interface status register (ASIS0)

Address: FFA1H When reset: 00H R

Symbol	7	6	5	4	3	2	1	0
ASIS0	0	0	0	0	0	PE0	FE0	OVE0

PE0	Parity error flag
0	No parity error
1	Parity error (Incorrect parity bit detected)

FE0	Framing error flag
0	No framing error
1	Framing error **Note 1** (Stop bit not detected)

OVE0	Overrun error flag
0	No overrun error
1	Overrun error **Note 2** (Next receive operation was completed before data was read from receive buffer register)

Notes:

1. Even if a stop bit length of two bits has been set to bit 2 (SL0) in the asynchronous serial interface mode register (ASIM0), stop bit detection during a receive operation only applies to a stop bit length of 1 bit.

2. Be sure to read the contents of the receive buffer register (RXB0) when an overrun error has occurred.

Until the contents of RXB0 are read, further overrun errors will occur when receiving data.

(c) Baud rate generator control register (BRGC0)

BRGC0 can be set by an 8-bit memory manipulation instruction.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 17-8: Baud rate generator control register (BRGC0)

Address: FFA2H When reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00

(f_X = 8.00 MHz)

TPS02	TPS01	TPS00	Source clock selection for 5-bit counter	n
0	0	0	$f_X/2^1$	1
0	0	1	$f_X/2^2$	2
0	1	0	$f_X/2^3$	3
0	1	1	$f_X/2^4$	4
1	0	0	$f_X/2^5$	5
1	0	1	$f_X/2^6$	6
1	1	0	$f_X/2^7$	7
1	1	1	$f_X/2^8$	8

MDL03	MDL02	MDL01	MDL00	Input clock selection for baud rate generator	k
0	0	0	0	$f_{SCK}/16$	0
0	0	0	1	$f_{SCK}/17$	1
0	0	1	0	$f_{SCK}/18$	2
0	0	1	1	$f_{SCK}/19$	3
0	1	0	0	$f_{SCK}/20$	4
0	1	0	1	$f_{SCK}/21$	5
0	1	1	0	$f_{SCK}/22$	6
0	1	1	1	$f_{SCK}/23$	7
1	0	0	0	$f_{SCK}/24$	8
1	0	0	1	$f_{SCK}/25$	9
1	0	1	0	$f_{SCK}/26$	10
1	0	1	1	$f_{SCK}/27$	11
1	1	0	0	$f_{SCK}/28$	12
1	1	0	1	$f_{SCK}/29$	13
1	1	1	0	$f_{SCK}/30$	14
1	1	1	1	Setting prohibit	—

Caution: Writing to BRGC0 during a communication operation may cause abnormal output from the baud rate generator and disable further communication operations. Therefore, do not write to BRGC0 during a communication operation.

Remarks: 1. f_{SCK}: Source clock for 5-bit counter

2. n: Value set via TPS00 to TPS02 ($1 \leq n \leq 8$)

3. k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

The transmit/receive clock that is used to generate the baud rate is obtained by dividing the main system clock.

- Use of main system clock to generate a transmit/receive clock for baud rate
  The main system clock is divided to generate the transmit/receive clock. The baud rate generated by the main system clock is determined according to the following formula.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1}(k + 16)} \quad [\text{Hz}]$$

f_X: Oscillation frequency of main system clock
n : Value set via TPS00 to TPS02 ($1 \leq n \leq 8$)
For details, see Table 17-3.
k : Value set via MDL00 to MDL02 ($0 \leq k \leq 14$)

Table 17-3 shows the relation between the 5-bit counter's source clock assigned to bits 4 to 6 (TPS00 to TPS02) of BRGC0 and the "n" value in the above formula.

Table 17-3: Relation between 5-bit Counter's Source Clock and "n" Value

TPS02	TPS01	TPS00	5-bit counter's source clock selected	n
0	0	0	$f_X/2^1$	1
0	0	1	$f_X/2^2$	2
0	1	0	$f_X/2^3$	3
0	1	1	$f_X/2^4$	4
1	0	0	$f_X/2^5$	5
1	0	1	$f_X/2^6$	6
1	1	0	$f_X/2^7$	7
1	1	1	$f_X/2^8$	8

Remark: f_X: Oscillation frequency of main system clock.

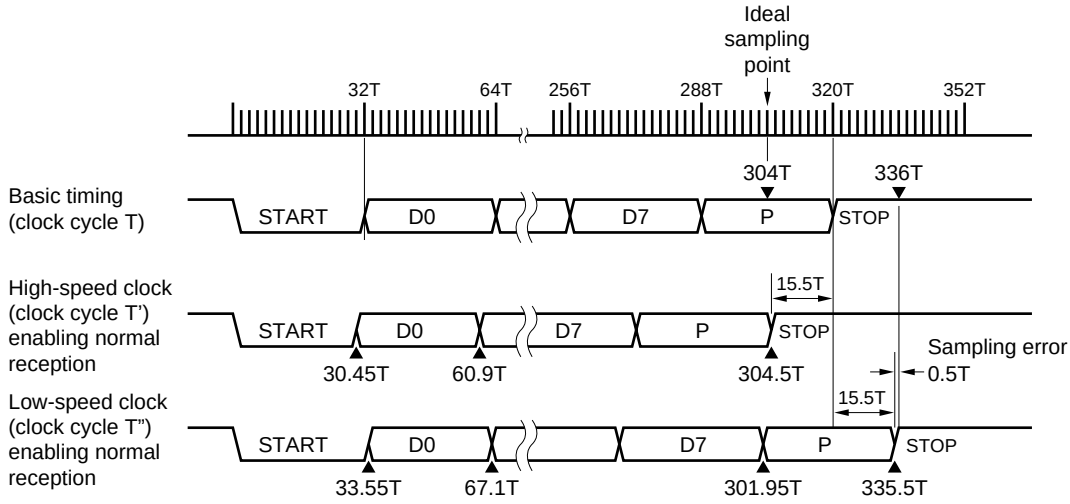
- **Error tolerance range for baud rates**
  The tolerance range for baud rates depends on the number of bits per frame and the counter's division rate [1/(16 + k)].
  Table 17-4 describes the relation between the main system clock and the baud rate and Figure 17-9 shows an example of a baud rate error tolerance range.

Table 17-4: Relation between Main System Clock and Baud Rate

Baud rate (bps)	f_X = 8.386 MHz		f_X = 8.000 MHz		f_X = 7.3728 MHz		f_X = 5.000 MHz		f_X = 4.1943 MHz	
	BRGC0	ERR (%)	BRGC0	ERR (%)	BRGC0	ERR (%)	BRGC0	ERR (%)	BRGC0	ERR (%)
600	–	–	–	–	–	–	–	–	7BH	1.14
1200	7BH	1.10	7AH	0.16	78H	0	70H	1.73	6BH	1.14
2400	6BH	1.10	6AH	0.16	68H	0	60H	1.73	5BH	1.14
4800	5BH	1.10	5AH	0.16	58H	0	50H	1.73	4BH	1.14
9600	4BH	1.10	4AH	0.16	48H	0	40H	1.73	3BH	1.14
19200	3BH	1.10	3AH	0.16	38H	0	30H	1.73	2BH	1.14
31250	31H	–1.3	30H	0	2DH	1.70	24H	0	21H	–1.3
38400	2BH	1.10	2AH	0.16	28H	0	20H	1.73	1BH	1.14
76800	1BH	1.10	1AH	0.16	18H	0	10H	1.73	–	–
115200	12H	1.10	11H	2.12	10H	0	–	–	–	–

Remarks: **1.** f_X: Oscillation frequency of main system clock
2. n: Value set via TPS00 to TPS02 ($1 \leq n \leq 8$)
3. k: Value set via MDL00 to MDL03 ($0 \leq k \leq 14$)

Figure 17-9: Error Tolerance (when k = 0), including Sampling Errors



Remark: T: 5-bit counter's source clock cycle

$$\text{Baud rate error tolerance (when k = 0)} = \frac{\pm 15.5}{320} \times 100 = 4.8438\ (\%)$$

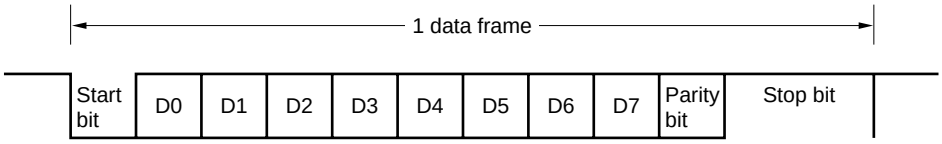
(2) Communication operations

(a) Data format

As shown in Figure 17-10, the format of the transmit/receive data consists of a start bit, character bits, a parity bit, and one or more stop bits.

The asynchronous serial interface mode register (ASIM0) is used to set the character bit length, parity selection, and stop bit length within each data frame.

Figure 17-10: Format of Transmit/Receive Data in Asynchronous Serial Interface



- Start bit ..... 1 bit
- Character bits ... 7 bits or 8 bits
- Parity bit ..... Even parity, odd parity, zero parity, or no parity
- Stop bit(s) ..... 1 bit or 2 bits

When “7 bits” is selected as the number of character bits, only the low-order 7 bits (bits 0 to 6) are valid, so that during a transmission the highest bit (bit 7) is ignored and during reception the highest bit (bit 7) must be set to “0”.

The asynchronous serial interface mode register (ASIM0) and the baud rate generator control register (BRGC0) are used to set the serial transfer rate.

If a receive error occurs, information about the receive error can be recognized by reading the asynchronous serial interface status register (ASIS0).

(b) Parity types and operations

The parity bit is used to detect bit errors in transfer data. Usually, the same type of parity bit is used by the transmitting and receiving sides. When odd parity or even parity is set, errors in the parity bit (the odd-number bit) can be detected. When zero parity or no parity is set, errors are not detected.

(i) Even parity

- During transmission
  The number of bits in transmit data that includes a parity bit is controlled so that there are an even number of "1" bits. The value of the parity bit is as follows.

  If the transmit data contains an odd number of "1" bits : the parity bit value is "1"
  If the transmit data contains an even number of "1" bits: the parity bit value is "0"

- During reception
  The number of "1" bits is counted among the transfer data that include a parity bit, and a parity error occurs when the result is an odd number.

(ii) Odd parity

- During transmission
  The number of bits in transmit data that includes a parity bit is controlled so that there is an odd number of "1" bits. The value of the parity bit is as follows.

  If the transmit data contains an odd number of "1" bits : the parity bit value is "0"
  If the transmit data contains an even number of "1" bits: the parity bit value is "1"

- During reception
  The number of "1" bits is counted among the transfer data that include a parity bit, and a parity error occurs when the result is an even number.

(iii) Zero parity

During transmission, the parity bit is set to "0" regardless of the transmit data.
During reception, the parity bit is not checked. Therefore, no parity errors will occur regardless of whether the parity bit is a "0" or a "1".

(iv) No parity

No parity bit is added to the transmit data.
During reception, receive data is regarded as having no parity bit. Since there is no parity bit, no parity errors will occur.

(c) Transmission

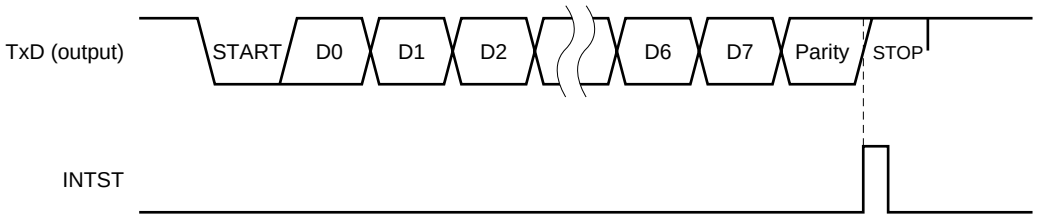
The transmit operation is started when transmit data is written to the transmit shift register (TXS0). A start bit, parity bit, and stop bit(s) are automatically added to the data.

Starting the transmit operation shifts out the data in TXS0, thereby emptying TXS0, after which a transmit completion interrupt (INTST) is issued.

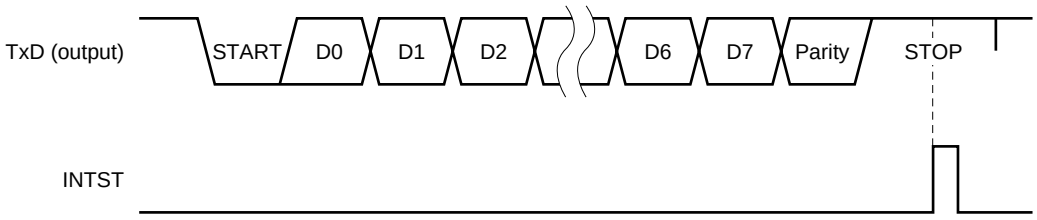
The timing of the transmit completion interrupt is shown in Figure 17-11.

Figure 17-11: Timing of Asynchronous Serial Interface Transmit Completion Interrupt

(i) Stop bit length: 1 bit



(ii) Stop bit length: 2 bits



Caution: Do not write to the asynchronous serial interface mode register (ASIM0) during a transmit operation. Writing to ASIM0 during a transmit operation may disable further transmit operations (in such cases, enter a $\overline{\text{RESET}}$ to restore normal operation). Whether or not a transmit operation is in progress can be determined via software using the transmit completion interrupt (INTST) or the interrupt request flag (STIF) that is set by INTST.

(d) Reception

The receive operation is enabled when "1" is set to bit 6 (RXE0) of the asynchronous serial interface mode register (ASIM0), and input data via RxD pin is sampled.

The serial clock specified by ASIM0 is used when sampling the RxD pin.

When the RxD pin goes low, the 5-bit counter begins counting and the start timing signal for data sampling is output if half of the specified baud rate time has elapsed. If the sampling of the RxD0 pin input of this start timing signal yields a low-level result, a start bit is recognized, after which the 5-bit counter is initialized and starts counting and data sampling begins. After the start bit is recognized, the character data, parity bit, and one-bit stop bit are detected, at which point reception of one data frame is completed.

Once the reception of one data frame is completed, the receive data in the shift register is transferred to the receive buffer register (RXB0) and a receive completion interrupt (INTSR) occurs.

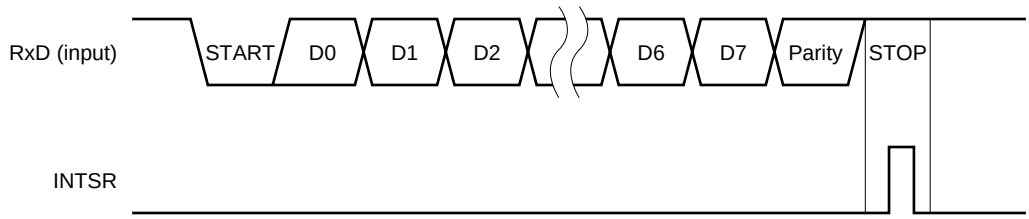
Even if an error has occurred, the receive data in which the error occurred is still transferred to RXB0 and INTSR occurs (see Figure 17-9).

If the RXE0 bit is reset (to "0") during a receive operation, the receive operation is stopped immediately.

At this time, neither the contents of RXB0 and ASIS0 do not change, nor does INTSR or INTSER occur.

Figure 17-12 shows the timing of the asynchronous serial interface receive completion interrupt.

Figure 17-12: Timing of Asynchronous Serial Interface Receive Completion Interrupt



Caution: Be sure to read the contents of the receive buffer register (RXB0) even when a receive error has occurred. Overrun errors will occur during the next data receive operations and the receive error status will remain until the contents of RXB0 are read.

(e) Receive errors

Three types of errors can occur during a receive operation: parity error, framing error, or overrun error.

If, as the result of the data reception, an error flag is set to the asynchronous serial interface status register (ASIS0), a receive error interrupt (INTSER) will occur. Receive error interrupts are generated before receive interrupts (INTSR). Table 17-5 lists the causes behind receive errors.

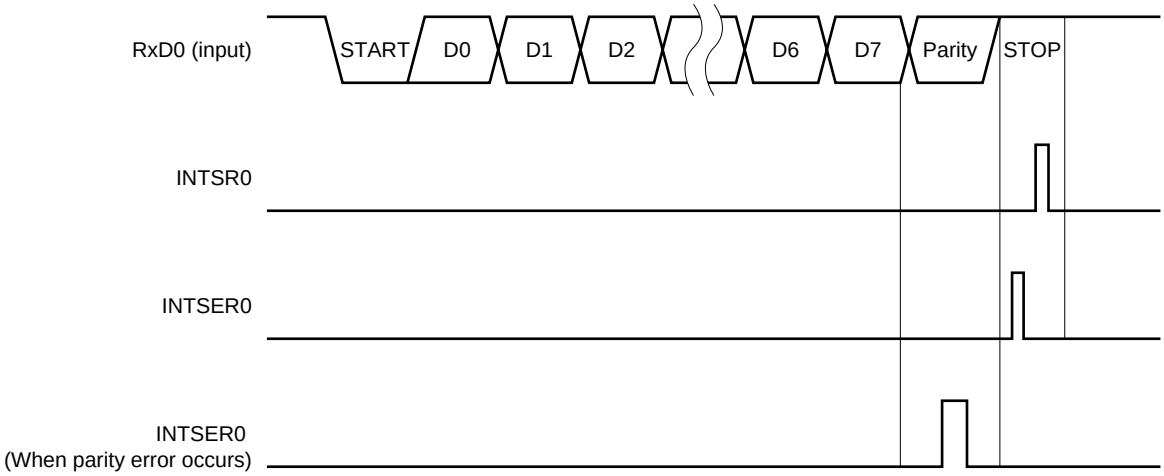
As part of receive error interrupt (INTSER) servicing, the contents of ASIS0 can be read to determine which type of error occurred during the receive operation (see Table 17-5 and Figure 17-13).

The content of ASIS0 is reset (to "0") if the receive buffer register (RXB0) is read or when the next data is received (if the next data contains an error, another error flag will be set).

Table 17-5: Causes of Receive Errors

Receive error	Cause	ASIS0 value
Parity error	Parity specified during transmission does not match parity of receive data	04H
Framing error	Stop bit was not detected	02H
Overrun error	Reception of the next data was completed before data was read from the receive buffer register	01H

Figure 17-13: Receive Error Timing



Cautions:

1. **The contents of ASIS0 are reset (to "0") when the receive buffer register (RXB0) is read or when the next data is received. To obtain information about the error, be sure to read the contents of ASIS0 before reading RXB0.**
2. **Be sure to read the contents of the receive buffer register (RXB0) even when a receive error has occurred. Overrun errors will occur during the next data receive operations and the receive error status will remain until the contents of RXB0 are read.**

17.6 Standby Function

Serial transfer operations can be performed during HALT mode.

During STOP mode, serial transfer operations are stopped and the values in the asynchronous serial interface mode register (ASIM0), transmit shift register (TXS0), receive shift register (RxS0), and receive buffer register (RXB0) remain as they were just before the clock was stopped.

Output from the TxD pin retains the immediately previous data if the clock is stopped (if the system enters STOP mode) during a transmit operation. If the clock is stopped during a receive operation, the data received before the clock was stopped is retained and all subsequent operations are stopped. The receive operation can be restarted once the clock is restarted.

[Memo]

Chapter 18 CAN Controller

Table 18-1: Outline of the Function

Feature	Details
Protocol	CAN2.0 with active extended frame capability (Bosch specification 2.0 part B)
Baudrate	Max. 500Kb at 8 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	Up to 16 received messages including 2 masks Two transmit channels
Message sorting	Unique identifier on all 16 received messages Up to 2 messages 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
Time functions	Support of time stamp and global time system
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

18.1 Protocol

CAN is an abreviation of "Controller Area Network", and is a class C high speed multiplexed communication protocol for real time communication in vehicle. CAN is being standardized in ISO (International Organization for Standardization) and SAE (Society of Automotive Engineers). For more detailed information please refer to Bosch, CAN specification 2.0 from September 1991.

18.1.1 Protocol mode function

(1) Standard format mode

- This mode supports an 11-bit message identifier thus making it possible to differentiate between 2032 types of messages.

(2) Extension format mode

- In the standard format mode, the identifier has 11 bits. However, in the extension format mode, the identifier is extended to 29 bits (11 + 18).
- When the IDE bits of the arbitration field is "recessive", it becomes the extension format mode.
- When the message of the extension format mode and the remote frame of the standard format mode are simultaneously transmitted, the node transmitting the message with the standard mode wins 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".
- For example, in case of a wired-AND implementation of the bus, the "dominant" levelwould be represented by a logical "0" and the "recessive" level by a logical 1.
- Physical states (e.g. electrical voltage, light) that represent the logical levels are not given in this specification.

18.1.2 Message format

The CAN protocol message supports different types of frames. The output conditions of each frame are as follows:

- Data frame: Carries the data from a transmitter to the receiver.
- Remote frame: Transmission demand frame from the requesting node.
- Error frame: Frame output on error detection.
- Overload frame: Frame output when a frame would be overwritten by the next one before the receiving mode could process it. The reception side does not finish its preparation.

18.1.3 Data frame/remote frame

Figure 18-1: Data Frame

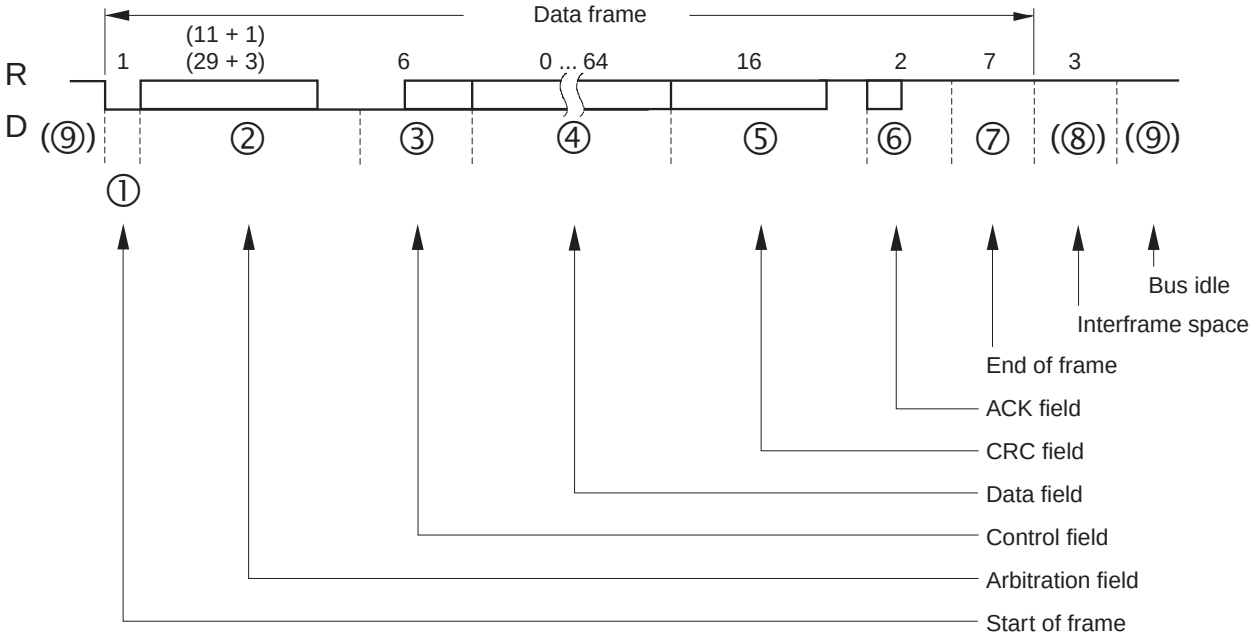
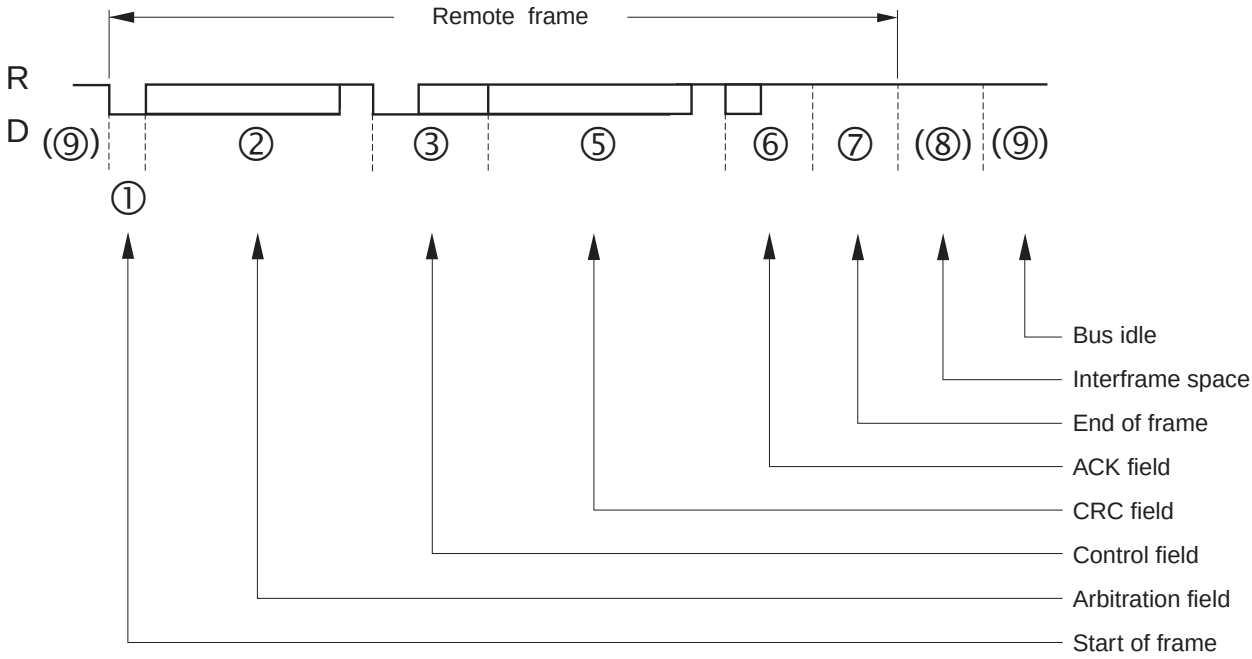


Figure 18-2: Remote Frame

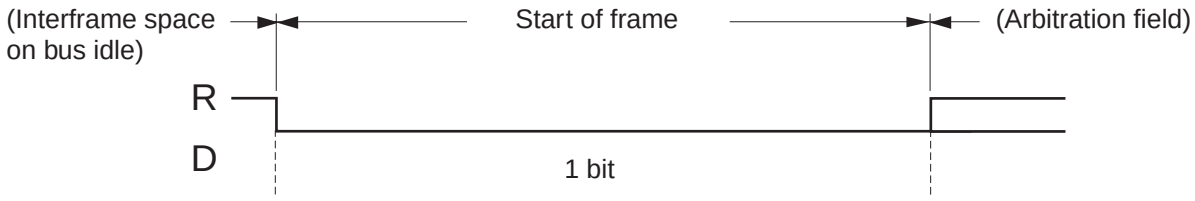


Note: This frame is transmitted when the reception node requests transmission. Data field is not transmitted even if the data length code ≠ '0' in the control field.

Description of each field

1. Start of frame: The start of data frame and remote frame are indicated.

Figure 18-3: Data Frame



- The start of frame 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, data frame/remote frame and protocol mode.

Figure 18-4: Arbitration Field/Standard Format Mode

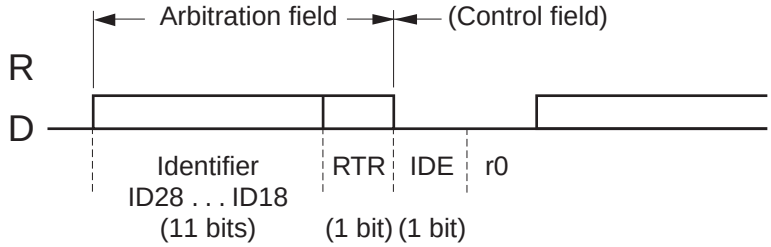
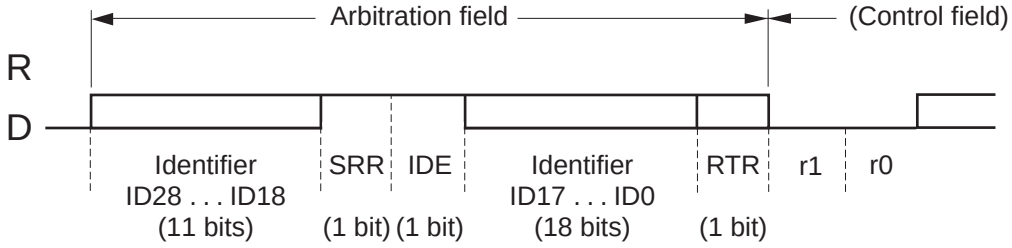


Figure 18-5: Arbitration Field/Expanded Format Mode



- ID28 - ID0 is the identifier.
- The identifier is transmitted at MSB first.

Table 18-2: Bit Number of the Identifier

Protocol Mode	Identifier Number
Standard format mode	11 bits
Expanded format mode	29 bits

Table 18-3: RTR Setting

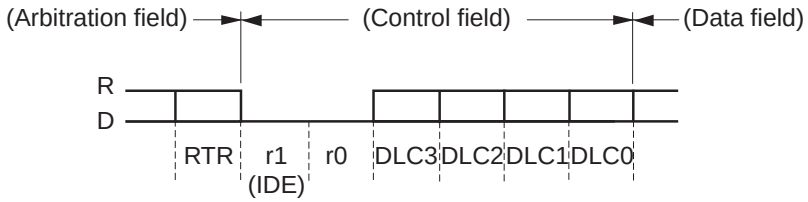
Frame Type	RTR Bit
Data frame	0
Remote frame	1

Table 18-4: Mode Setting

Protocol Mode	IDE Bit
Standard format mode	0
Expanded format mode	1

3. Control field: Data byte number N in the data field is set (N: 0 to 8).

Figure 18-6: Control Field



- IDE bit and r1 bit in the arbitration field are in the same at the standard format mode.

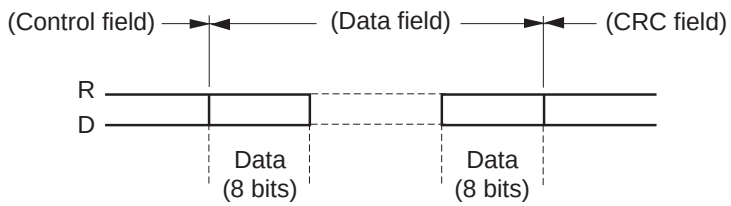
Table 18-5: Data Length Code Setting

Data Length Code				Number of Data Bytes
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
. . .	. . .	. . .	. . .	
0	1	1	1	7
1	0	0	0	8

Note: In case of the remote frame, data field is not generated even if data length code ≠ '0'.

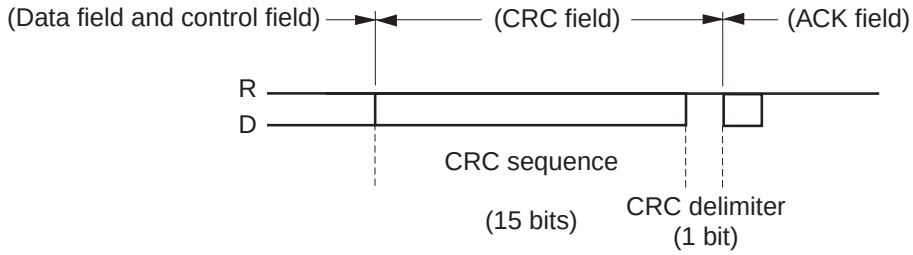
4. Data field: Data group of the number is set in the control field. Up to 8 bits can be set.

Figure 18-7: Data Field



5. CRC field: 15 bits CRC sequence to check the transmission error.

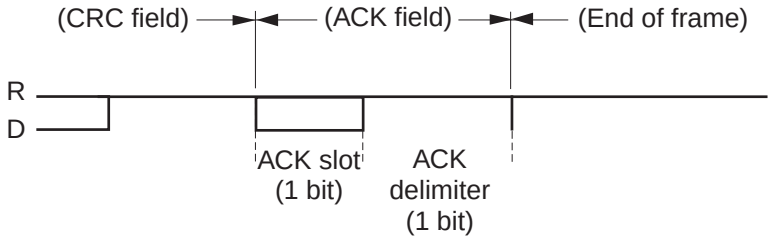
Figure 18-8: CRC Field



- 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 all basic data bit that is not bit stuffed from, the start of frame, arbitration field, control field and data field.
- Reception node: Compares CRC sequence calculated from the data bit except the reception data stuff bit and the CRC sequence in CRC field. In case when these are disagreement, the node shifts to the error frame.

6. ACK field: For check of normal reception.

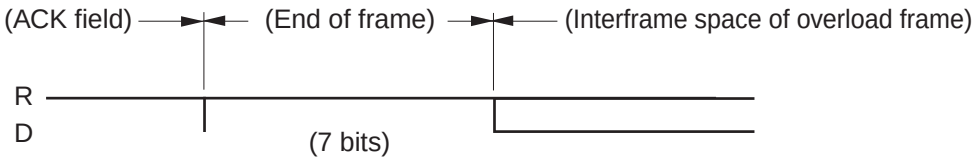
Figure 18-9: ACK Field



- Receive node sets the ACK slot to dominant level if no error was detected.
- Transmission node outputs 'Recessive level' of 2 bits, and checks the reception state of the reception node.

7. End of frame: Indicates the end of the transmission/reception.

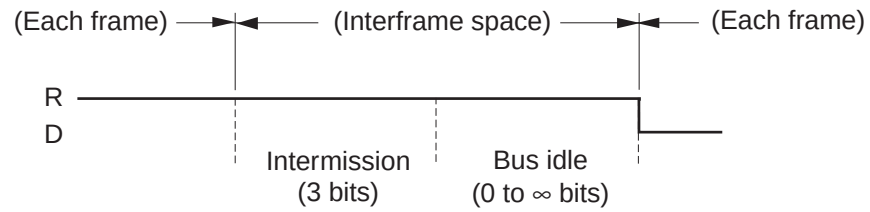
Figure 18-10: End of Frame



8. Interframe space: This frame is inserted between the data frame, remote frame, error frame, overload frame and the next frame to indicate partitions between each frame.

(A) Error active: Consists of 3 bits intermission and bus idle.

Figure 18-11: Interframe Space/Error Active



B) Error passive: Consists intermission, suspend transmission and bus idle.

Figure 18-12: Interframe Space/Error Passive

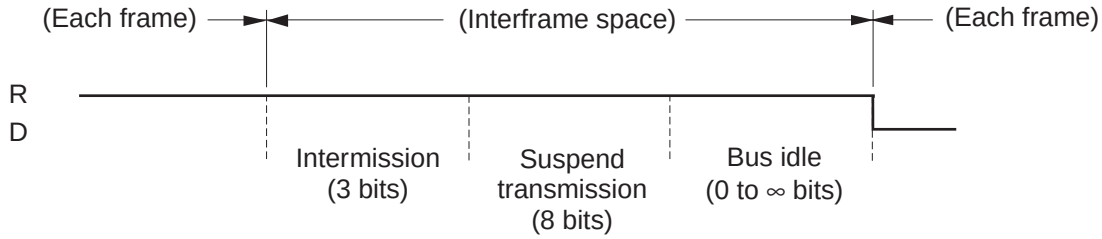


Table 18-6: Bit Length of the Intermission

Protocol Mode	Bit Length
Standard format mode	3 bits

Table 18-7: Operation in the Error State

Error State	Operation
Error active	State of each code can be transmitted at bus idle. The node required to transmit starts the transmission.
Error passive	State of each code can be transmitted after 8 bits bus idle has continued. When the other nodes begin the transmission, it becomes the reception state.

18.1.4 Error frame

- This frame is output from the node if an error is detected.
- When other nodes output 'Dominant level' to flag the passive error, the dominant level continues for 6 consecutive bits. The passive error flag consists of 6 consecutive Recessive bits unless a dominant bit from other nodes overwrite it.

Figure 18-13: Error Frame

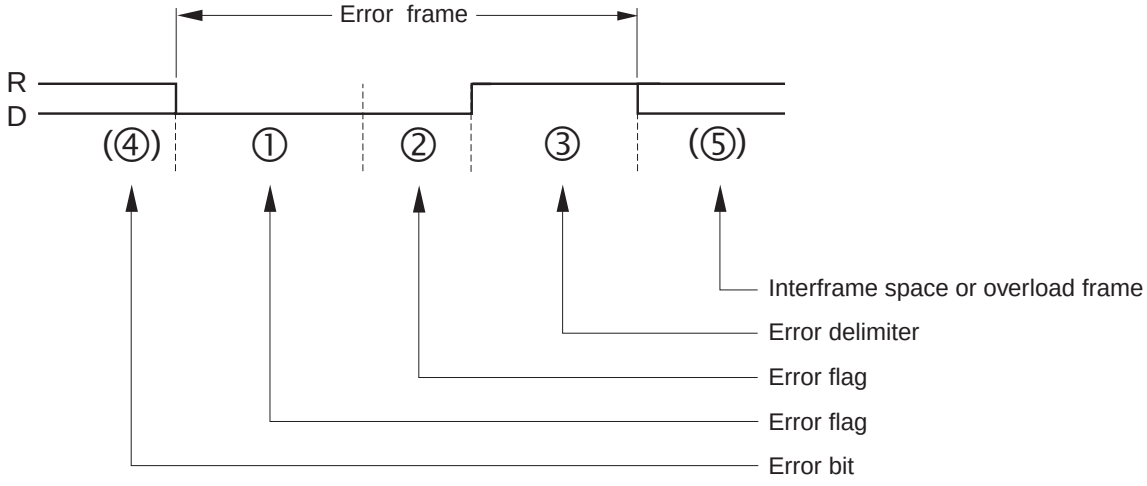


Table 18-8: Definition of each Field

No.	Name	Bit Number	Definition
①	Error flag	6	Error active node: output 6 bits ‘dominant level’ continuously. Error passive node: output 6 bits ‘recessive level’ continuously.
②	Error flag	0 to 6	Node received ‘error flag’ detects the bit stuff error and outputs ‘error flag’ again.
③	Error delimiter	8	Output 8 bits ‘recessive level’ continuously. In case of monitoring ‘dominant level’ at 8th bit, the overload frame is transmitted after the next bit.
④	Error bit	—	Output continuously after the bit where error has occured (in case of the CRC error, output continues after the Ack delimiter).
⑤	Interframe space/ overload frame	3/14 20 MAX	‘Interframe space’ or ‘overload frame’ continues.

18.1.5 Overload frame

- This frame is output from the first bit of the intermission when the reception node has not completed the receiving operation.
- When the bit error is detected in the intermission, this frame is output following the next bit after the bit error detection.

Figure 18-14: Overload Frame

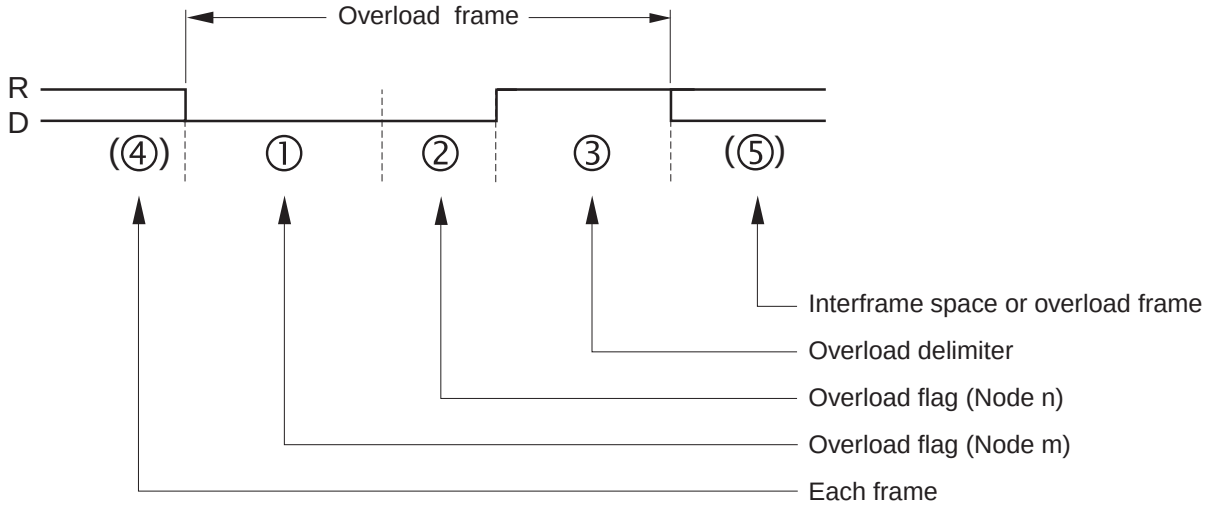


Table 18-9: Definition of each Frame

No.	Name	Bit Number	Definition
①	Overload flag	6	Output 6 bits 'dominant level' continuously.
②	Overload flag from node n	0 to 6	Node n that receives 'overload flag' in 'interframe space' outputs 'overload flag'.
③	Overload delimiter	8	Output 8 bits 'recessive level' continuously. In case of monitoring 'dominant level' at 8th bit, the overload frame is transmitted from the next bit.
④	Each 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.

Note: Node n/node m means any mode

18.2 Function

18.2.1 Bus priority decision

(1) When 1 node starts transmission
- During bus idle, the node having the output data can transmit.

(2) When more than 2 nodes start transmission
- The node with the lower identifier wins the arbitration.
- The transmission node compares its output arbitration field and the data level on the bus.
- It looses arbitration, when it outputs recessive level and sees dominant on bus.

Table 18-10: Bus Priority Decision

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 the data frame and remote frame are on the bus, the data frame has priority in which RTR is 'Dominant level'. The data frame wins the arbitration.

18.2.2 Bit stuffing

When the same level continues for more than 5 bits, bit stuffing (insert 1 bit with inverse level) takes place to prevent an error.

Table 18-11: 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 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 ACK field, the reception is continued by deleting following 1 bit.

18.2.3 Multi master

As the bus priority is determined by the identifier, any node can be the bus master.

18.2.4 Multi cast

If there are different nodes, is only one, an identifier has to be transmitted only one node. The same reception can be done in more than 2 nodes simultaneously.

18.2.5 Sleep mode/Stop function

This is a function to put the CAN controller in waiting mode to achieve low power consumption. The SLEEP mode is defined like in the CAN specification.

Additional to this SLEEP mode, which can be woken up by bus activities, the STOP mode is fully controlled by the CPU device.

18.2.6 Error control function

(1) Error types

Table 18-12: 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	Continuous 6 bits of the same level data	Transmission/ reception node	Start of frame to CRC sequence
CRC error	Comparison of the CRC generated from the reception data and 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 18-13: Output Timing of the Error Frame

Type	Output timing
Bit error, stuff error, form error, ACK error	Error frame is output from the next bit timing which detects error
CRC error	Error frame is output from the next bit timing of the ACK delimiter

(3) Measures when error occurs

- Transmission node re-transmits the data frame or the remote frame after the error frame.

(4) Error state

1. Types of error state

- Three types of error state. These are error active, error passive and bus off.
- The transmission error counter and the reception error counter control the error state.
- The error counter is incremented at each error.
- Output error flag is different whether the operation of error state is transmission or reception.
- If the value of error counter exceeds 96, warning level for an error passive is reached.
- When only one node is active at start-up, ACK is not returned even if data is transmitted so that the re-transmission of the error frame and data are repeated. In this case, the state does not become bus off. In adddition, the bus off state is also not entered even if the error state is repeated at a node transmitting wake up message.
- Receiving operation can be executed even if the transmitting operation is in bus off state.

Table 18-14: Types of Error State

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	256 or more	Communication cannot be made
			When the following states are generated 128 times, the state can be returned to the error active by error counter = 0. 1. 'Recessive level' continues 11 bits
	Reception	—	Does not exist

2. Error counter

- Error counter counts up when error has occured, and counts down when transmission and reception are operated normally. Timing of count up and count down is first bit of the error deliminiter.

Table 18-15: Error Counter

State	Transmission Error Counter	Reception Error Counter
When reception node detects error (except bit error in the active error flag and overload flag).	No change	+1
When reception node detects 'dominant level' next to the error flag of the error frame.	No change	+8
When transmission node transmits error flag [Error counter = ±0] ① When ACK error is detected in the error passive state and 'dominant level' is not detected in outputting passive error flag. ② Stuff error generation in arbitration field.	+8	No change
Bit error detection during active error flag and overload flag are output (transmission node of error active).	+8	No change
Bit error detection during active error flag and overload flag are output (reception node of error active).	No change	+8
When each node detects fourteen continuous 'dominant level' from the beginning of active error flag and overload flag, and every time eight continuous 'dominant level' after that are detected. Every time when each node detects eight continuous 'dominant level' after the passive error flag.	+8	+8
When the transmission node has completed to receive 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) Sets 127 (REC > 127)

Note: REC: Reception error counter

3. Bit error generation during intermission

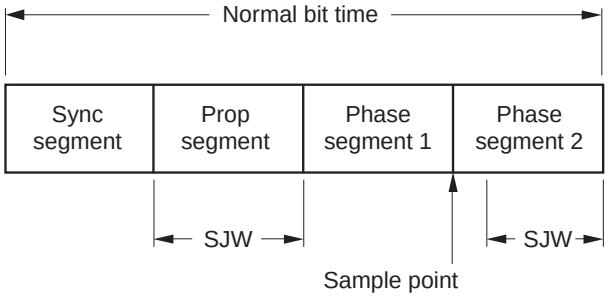
- Overload frame is generated.

18.2.7 Baud rate control function

1. Nominal bit time (8 to 25 time quantum)

- Definition of 1 data bit time is as follows.

Figure 18-15: Nominal Bit Time (8 to 25 Time Quantum)



[1 Minimum time quantum = 1/fx]

- Sync segment: This segment becomes beginning when bit snchronization is taken.
- Prop segment: This segment is to absorb delays of the output buffer, CAN bus and input buffer. It is to make ACK return until the phase segment 1 starting.
  Prop segment time ≥ (output buffer delay) + (CAN bus delay) + (input buffer delay).
- Phase segment 1/2: This segment is to compensate the data bit time error. The larger the size is, the larger is the tolerable error.
- SJW: This means resynchronization Jump Width. This is a bit to set a bit synchronization range.

Table 18-16: Segment Name and Segment Length

Segment Name	Segment Length
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 value (IPT[Note] = 0 to 2) of phase segment 1, + IPT (IPT = 0 to 2)
SJW	Programmable 1 to 4

Note: IPT = Information Processing Time

2. Adjusting synchronization of the data bit

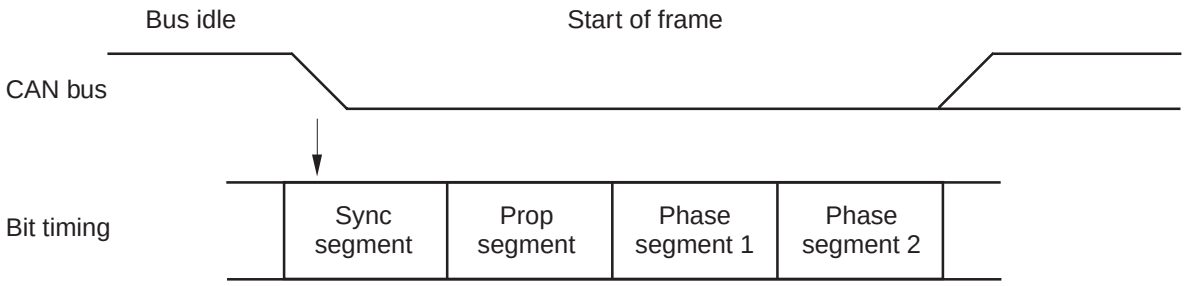
- Transmission node transmits data synchronizing with the transmission node bit timing.
- Reception node adjusts synchronization at the level changing on the bus caused on hardware or software synchronization.

(A) Hardware synchronization

Bit synchronization adjustment when reception node detects the start of frame in the bus idle state.

- Gets sampling a level on the bus to detect falling edge, that bit and the next bit become sync segment and prop segment, respectively. In this case, synchronization is adjusted to have no relation with SJW.
- After resetting, it is necessary to take bit synchronization after the wake up, only the first level change on the bus is taken for hardware synchronization (afterwards, following bit synchronization is performed).

Figure 18-16: Adjusting Synchronization of the Data Bit

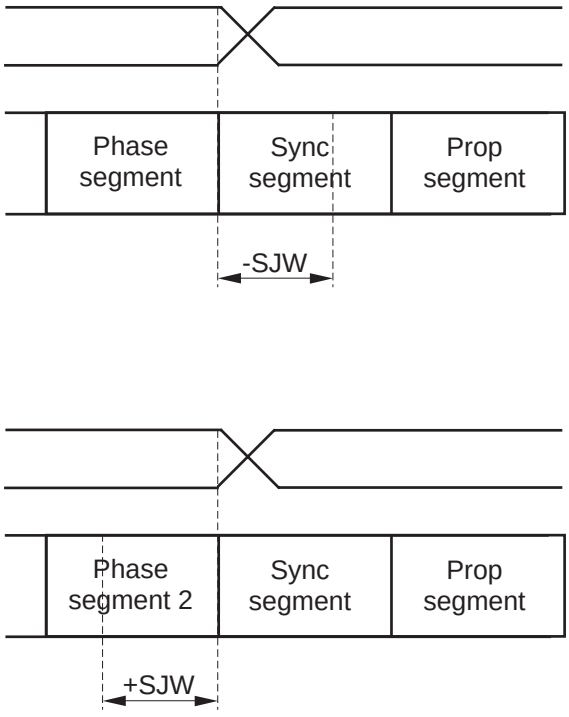


(B) Bit synchronization

When the level change on the bus is detected during reception, bit synchronization is performed.

- 2 types of synchronization can be performed.
  Normal operation: Level falling edge
  Low speed operation: Level falling edge and rising edge
- Synchronization is performed only when an edge is detected while the bit timing specified as SJW.
- By a baud rate 'difference' between transmission node and reception node, data sample point at the reception node shifts relatively.
- Allowable range of 'difference' is defined as 'SJW'. Range of SJW is set in front and behind (+/- of baud rate) of the sync segment. When an edge is generated at the SJW range, synchronization is achieved. When the edge is generated out of SJW range, synchronization is not achieved.
- The bit detected in the edge becomes sync segment forcibly, and the next one becomes prop segment, bit rate starts again.

Figure 18-17: Bit Synchronization



18.2.8 State shift chart

Figure 18-18: Transmission State Shift Chart

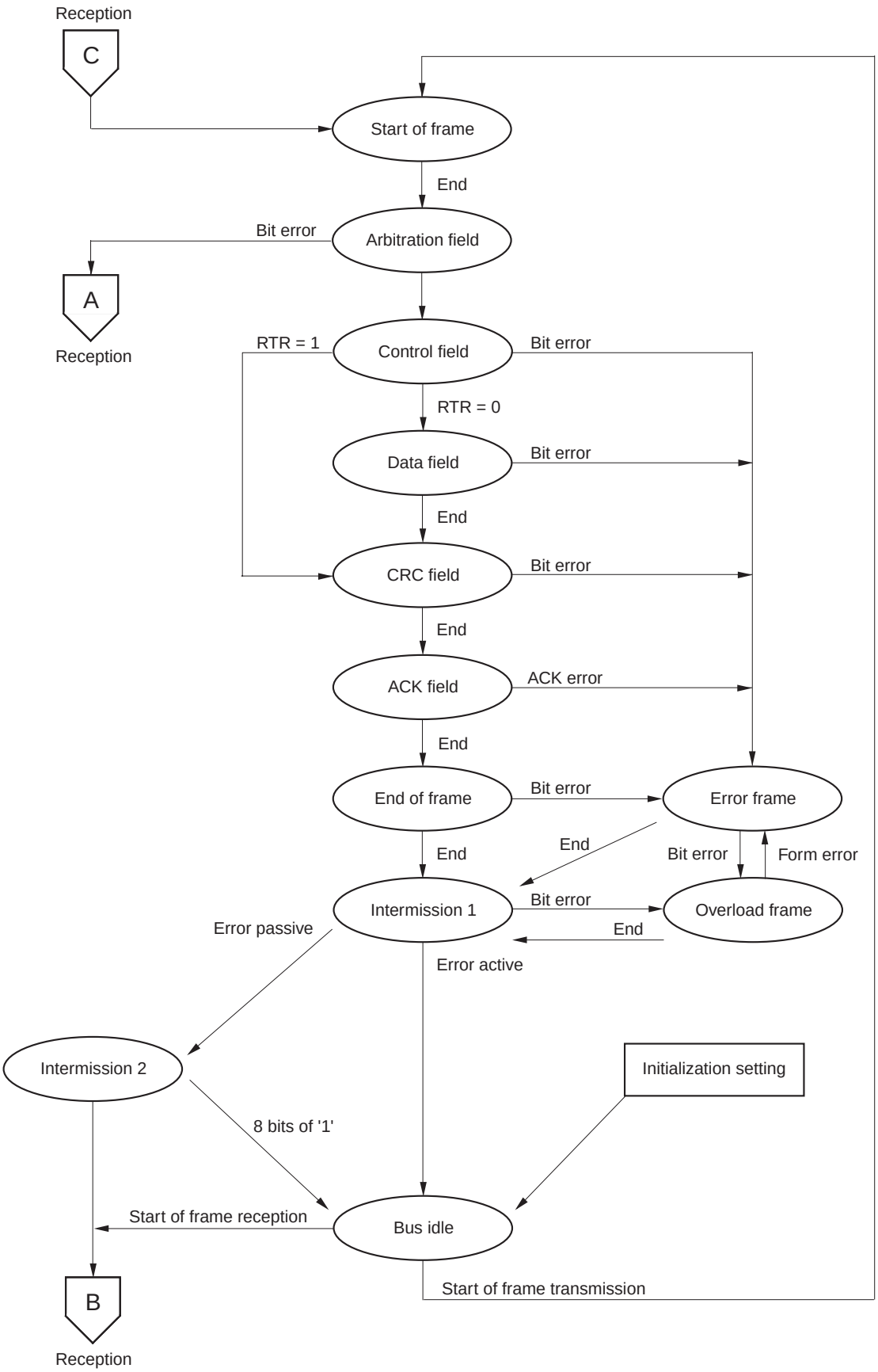


Figure 18-19: Reception State Shift Chart

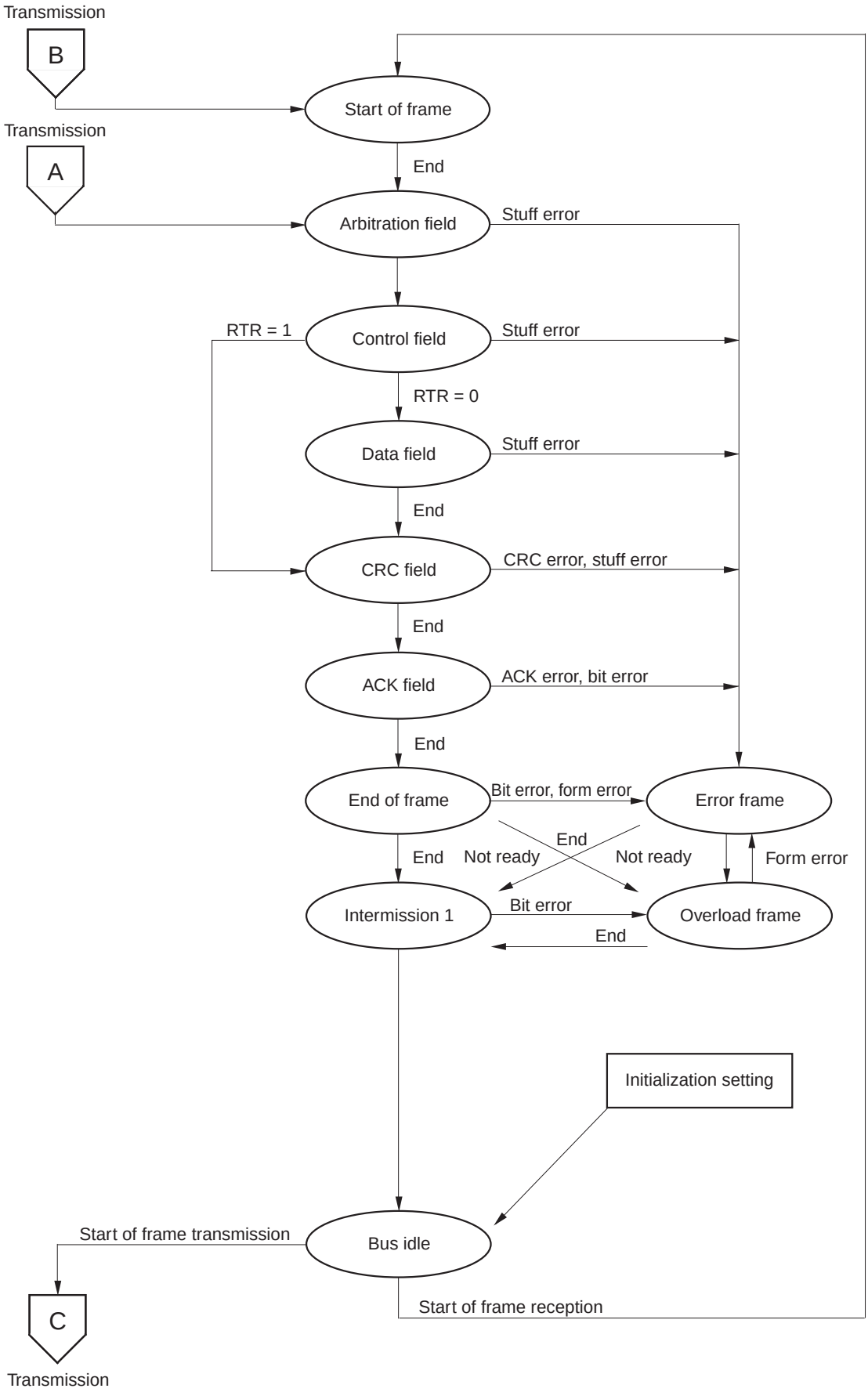
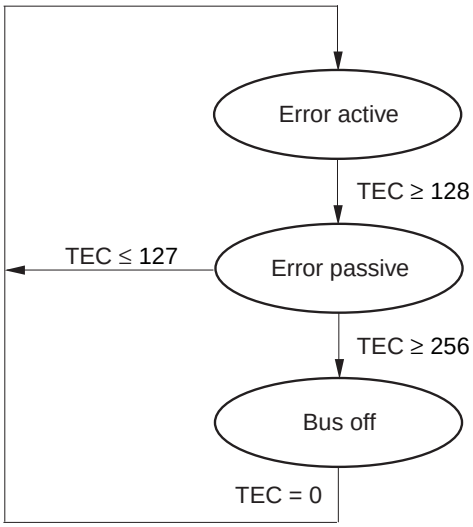


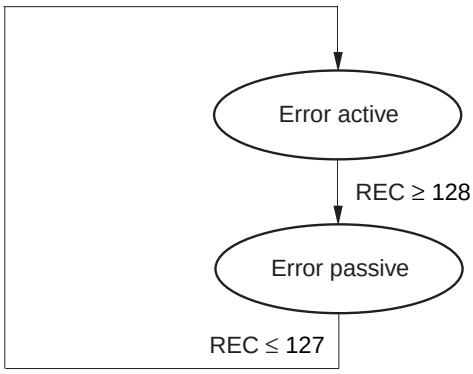
Figure 18-20: Error State Shift Chart

(A) Transmisssion



TEC = Transmission error counter

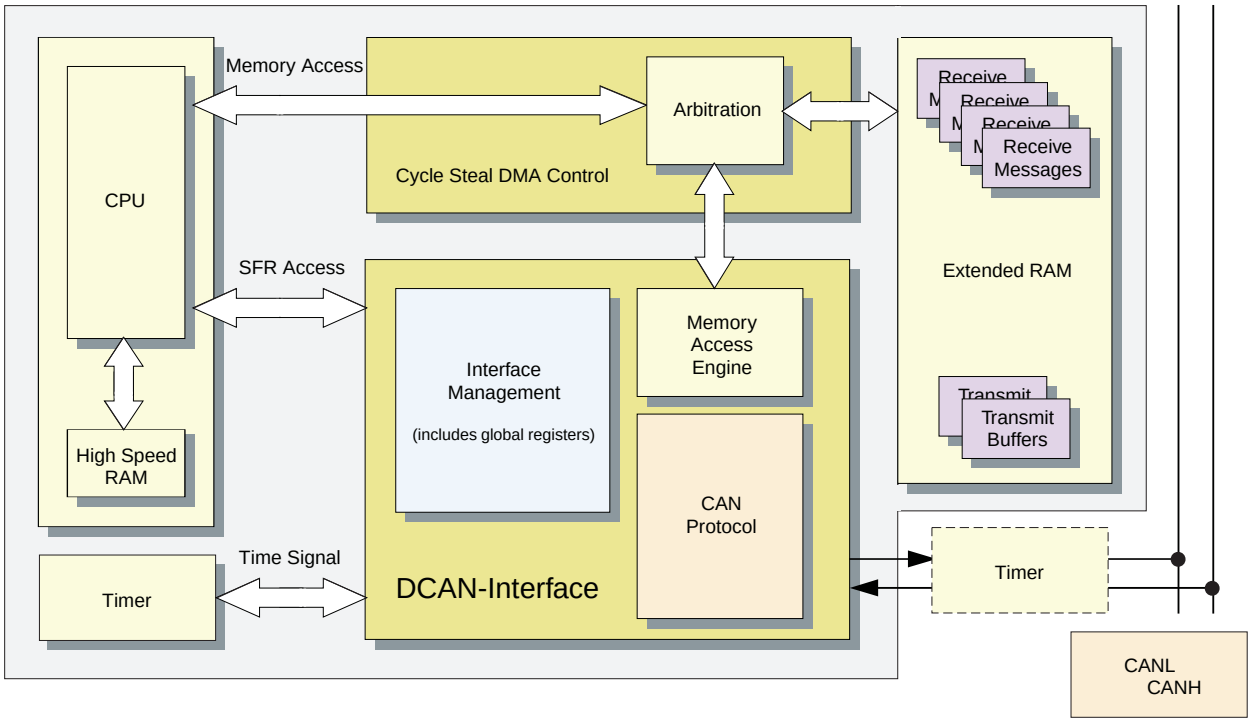
(B) Reception



REC = Reception error counter

18.3 Outline Description

Figure 18-21: Structural Block Diagram



This interface part handles all protocol activities by hardware in the CAN protocol part. The memory access engine fetches information for 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.

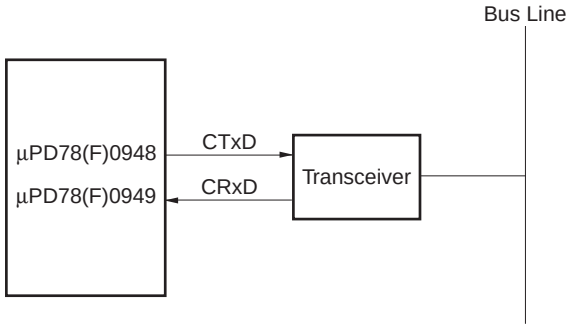
The DCAN interfaces directly to the RAM area that is accessable by the DCAN and by the CPU without any influence to the CPU.

The CAN 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.

18.4 Connection with Target System

The μPD78(F)0948/μPD78(F)0949 has to be connected to the CAN bus with an external transceiver.

Figure 18-22: Connection to the CAN Bus



18.5 CAN Module Configuration

The CAN-module consists of the following hardware.

Table 18-17: CAN Configuration

Item	Configuration
Message definition	In RAM areas
CAN input/output	1 (CTxD) 1 (CRxD)
Control register	CAN control register (CANC) Transmit control register (TCR) Received 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)

18.6 Operation

18.6.1 Special function register for CAN-module

Table 18-18: SFR Definitions

Address	Register Name	Symbol	R/W	Bit Manipulation Units			After
				1 Bit	8 Bit	16 Bit	Reset
FFB0H	CAN control register	CANC	R/W	O	O	—	01H
FFB1H	Transmit control register	TCR	R/W	—	O	—	00H
FFB2H	Received message register	RMES	R	—	O	—	00H
FFB3H	Redefinition control register	REDEF	R/W	O	O	—	00H
FFB4H	CAN error status register	CANES	R/W	—	O	—	00H
FFB5H	Transmit error counter	TEC	R	—	O	—	00H
FFB6H	Receive error counter	REC	R	—	O	—	00H
FFB7H	Message count register	MCNT	R	—	O	—	C0H
FFB8H	Bit rate prescaler	BRPRS	R/W	—	O	—	00H
FFB9H	Synchronous control register 0	SYMC0	R/W	—	O	—	18H
FFBAH	Synchronous control register 1	SYNC1	R/W	—	O	—	0EH
FFBBH	Mask control register	MASKC	R/W	—	O	—	00H

The following SFR-bits can be defined as 1-bit instruction. The other SFR-registers have to be defined by 8-bit instruction.

Table 18-19: 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

18.7 Message and Buffer Configuation

Table 18-20: Message and Buffer Structure

Address[Note2]	Register Name	Symbol	R/W	After Reset
00xH	Transmit buffer 0		R/W	Note1
01xH	Transmit buffer 1		R/W	Note1
02xH	Rec. message 0 / Mask 0		R/W	Note1
03xH	Rec. message 1		R/W	Note1
04xH	Rec. message 2 / Mask 1		R/W	Note1
05xH	Rec. message 3		R/W	Note1
06xH	Rec. message 4		R/W	Note1
07xH	Rec. message 5		R/W	Note1
08xH	Rec. message 6		R/W	Note1
09xH	Rec. message 7		R/W	Note1
0AxH	Rec. message 8		R/W	Note1
0BxH	Rec. message 9		R/W	Note1
0CxH	Rec. message 10		R/W	Note1
0DxH	Rec. message 11		R/W	Note1
0ExH	Rec. message 12		R/W	Note1
0FxH	Rec. message 13		R/W	Note1
10xH	Rec. message 14		R/W	Note1
11xH	Rec. message 15		R/W	Note1

Notes: **1.** Contents is undefined, because data resides in normal RAM area.

2. This addess is an offset to the RAM area starting address, defined with CADD in the Message Count Register (MCNT).

18.8 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 a possibility to send 8 message data bytes. The structure of the transmit buffer is similar to the structure of the receive buffers. The CPU can use by itself unused addresses, unused message data addresses, and unused transmit buffer. The control bits, the identification and the message data has 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 flag.

In the case that both transmit are used, the transmit priorities can be set. For this purpose the DCAN has the TXP flag in the TCR register. The application software has to set this priority befor the transmission is started.

The two transmit buffers have two independent vectorized interrupts.

18.9 Transmit Message

Table 18-21: Transmit Message Structure

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 addess is an relative offset to the starting address of the transfer buffer.

Transmit Message Definition

This register controls the message definition bits of the control field of the CAN protocol.

TCON is set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets TCON to an undefined value.

Figure 18-23: 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 part describes the type of Frame that is generated and its length. The reserved bits of the CAN protocol are always transferred in dominant state (0).

Note: The data length code selects number of bytes which has to be transmitted. Valid entries for the Data Length Code are 0 to 8. If a value greater than 8 is selected, 8 bytes are transmitted in the data frame with the Data Length Code specified in DLC.

Transmit Identifier Definition

These registers set the message identifier in the arbitration field of the CAN protocol.
IDTX0 to IDTX4 can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IDTX0 to IDTX4 to an undefined value.

Figure 18-24: Transmit Identifier

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

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDTX1	ID20	ID19	ID18	0	0	0	0	0	xxx3H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDTX2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDTX3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDTX4	ID1	ID0	0	0	0	0	0	0	xxx6H	undefined	R/W

Transmit Data Definition

These registers set the transmit message data of the data field in the CAN frame.
DATA0 to DATA7 can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to an undefined value.

Figure 18-25: 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

18.10 Transmit 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 receive data is stored direktly in this RAM area.

The 16 buffers have a 16 byte data structure for standard and extended frames with a possibility to send 8 message data bytes. 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. In case of 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/disabled for each used buffer separately.

18.11 Receive Message

Table 18-22: Receive Message Structure

Name	Address [Note]	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	0
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							

Note: This addess is an relative offset to the starting address of the receive buffer.

Receive Control Bits Definition

This register sets the receive control bits of the control field of the CAN protocol.
IDCON can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IDCON to an undefined value.

Figure 18-26: 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 should be received in the associated buffer.

Note: The user has to define whether he wants to create a receive interrupt request, when new data is received on this buffer.

Receive Status Bits Definition

This register sets the receive status bits of the arbitration field of the CAN protocol.
DSTAT can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DSTAT to an undefined value.

Figure 18-27: Receive Status Bits

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 this message. It signals whether new data is stored, or if the DCAN currently tranfers data into this buffer.
In addition the data length of the last transfered data and the reserved bits of the protocol are shown.

DN	Data New
0	No change in data
1	Data changed

The CAN-module sets DN, when it starts transferring a message into the buffer.

The CPU can clear this bit, to signal that it has read the data.

The CPU must clear this bits during initialization of the message area. Otherwise the CPU has no information on the status of the messages after start of the CAN activities.

MUC	Memory Update
0	CAN does not access data part
1	CAN is transfering new data to message buffer

The CAN-module sets MUC, when it starts transferring a message into the buffer and clears the MUC bit, when the transfer data 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”

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

This register is written by the DCAN two times during message storage.
As the first access to this buffer area. DN = 1, MUC = 1 , reserved bits, DLC are written.
As the last access to this buffer area. DN = 1, MUC = 0 , reserved bits, 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 received in the data frame with the data length code specified in DLC.

Receive Identifier Definition

These registers set the receive identifier definition of the control field of the CAN protocol. IDREC0 to IDREC4 can be set with a 1-bit or an 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets IDREC0 to IDREC4 to an undefined value.

Figure 18-28: Receive Identifier

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

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDREC1	ID20	ID19	ID18	0	0	0	0	0	xxx3H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDREC2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx44H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDREC3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	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 messages without mask function this data is only read by the DCAN.

The identifier of the receive messages should not changed without being in the initialization phase or the the receive buffer is set 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, what may cause unusable receptions in this buffer.

Note: The unused parts of the identifier may be written by the DCAN to “0”. They are not open to other use by the CPU.

Receive Message Data Part

These registers set the receive message data part of the CAN protocol.
DATA0 to DATA7 can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to an undefined value.

Figure 18-29: 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 fits with the identifier are stored in the receive buffer memory area.

If the DLC is less than eight, the DCAN will not write the additional bytes up to eight. The DCAN stores a maximum of 8 bytes (conforming to the CAN protocol rules) even when the received DLC is greater than eight.

18.12 Mask Function

Table 18-23: Mask Function

Name	Address Note1	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							

Message Buffer 0 and Buffer 2 may be switched for masked operation with the Mask Control Register. In this case the message does not hold message identifier and data, but only holds identifier and RTR mask information for masked compares on the next higher message number. In the case of a global mask select, it keeps mask information for all higher messages.

A mask does not store any information about identifier length. The same mask can therefor be used for both types of frames (standard and extended) during global mask operation.

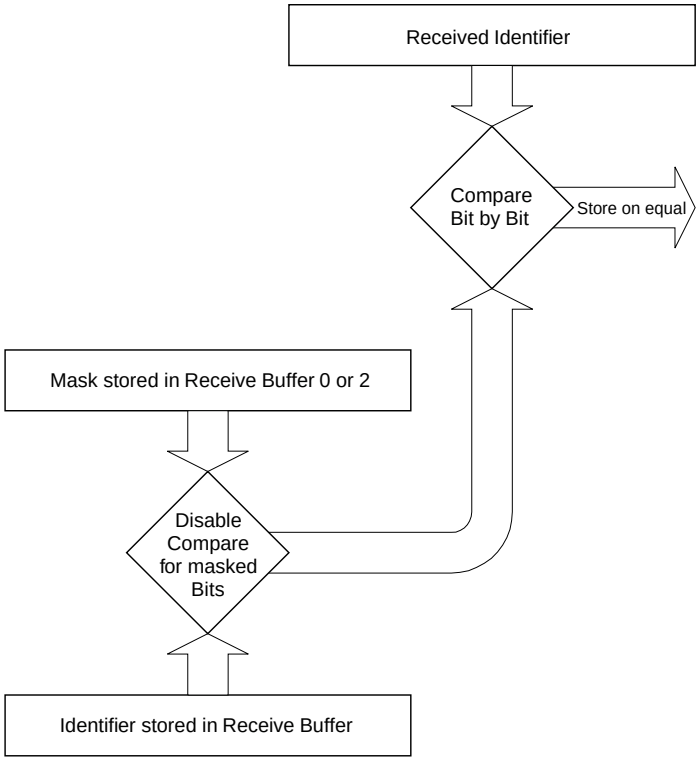
Identifier Compare with Mask

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. A global mask can serve for standard receive buffers at the same time as for extended receive buffer.

Figure 18-30: Identifier Compare with Mask



This function implements the so-called basic-CAN behavior.

The type of identifier is in this case 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 RAM:

- Identifier (11/29 bit as defined by IDE bit)
- Remote bit. RTR
- Reserved bits
- Data length code. DLC
- Data bytes as defined by DLC.

Caution: All writes into DCAN are byte accesses. Unused bits in the same byte will be written zero.

Mask Identifier Control Bit Definition

This register sets the mask identifier control bit of the CAN protocol.
MCON can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MCON to an undefined value.

Figure 18-31: 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
1	Receive message independent from RTR bit

Mask Identifier Definition

These register sets the mask identifier definition of the DCAN.
MREC0 to MREC4 can be set with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MREC0 to MREC4 to an undefined value.

Figure 18-32: 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

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MREC1	MID20	MID19	MID18	0	0	0	0	0	xxx3H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MREC2	MID17	MID16	MID15	MID14	MID13	MID12	MID11	MID10	xxx4H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MREC3	MID9	MID8	MID7	MID6	MID5	MID4	MID3	MID2	xxx5H	undefined	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	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 of received message
1	Receive message independent from IDn bit

18.13 CAN

18.13.1 Status register

CAN Control Register

These register sets the CAN control definition of the CAN module.

CANC can be set with a 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CANC to 01H.

Figure 18-33: CAN Control Register

Symbol	7	6	5	④	3	②	1	⓪	Address	After Reset	R/W
CANC	RxF	TxF	0	SOFE	SOFSEL	SLEEP	STOP	INIT	FFB0H	01h	R/W

INIT	Initialize
0	Normal operation
1	Initialization mode

RxF and TxF are read only bits.

INIT is used configure savely all bus and compare parameters, without creating unsolicited behaviour.

INIT initiates the DCAN to set all internal activities to inactive states. Due to actual bus protocol activities this may need some time. The actual status is shown in the INITSTATE bit in the CANES register.

The CTxD output stays recessive (logical high) during this initialization.

Changing of the following registers is only permitted when INIT is active:

MCNT, SYNC0, SYNC1, MASKC.

Any write to these registers when INIT = 0 is ignored.

SLEEP	Sleep/Stop Request for CAN protocol
0	Normal operation
1	CAN protocol goes to sleep or stop mode depending on STOP bit

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

The clock supply to the DCAN is switched off during sleep and stop condition of the DCAN.

The sleep condition is cleared under following conditions:

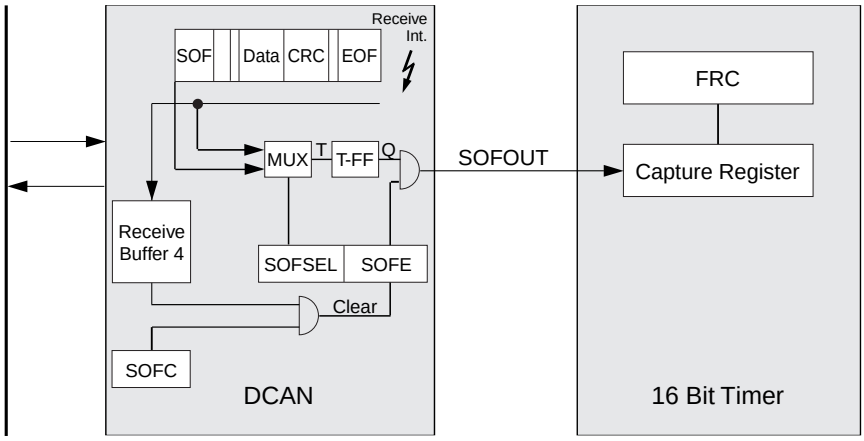
a) CPU clears the SLEEP bit.
b) Transition on CAN Bus. Only when STOP = 0.
c) CPU sets SLEEP, but CAN protocol is active due to bus activity.

The WAKE bit in CANES is set under condition b) and c). An error interrupt is activated at the same time.

SOFE	Start of Frame Enable
0	SOFOUT does not change
1	SOFOUT toggles depending on the selected mode

SOFSEL	Start of Frame Output Function Select
0	SOFOUT works as time stamp function
1	SOFOUT signals SOF on the bus / Global time function

Figure 18-34: DCAN Support



The generation of an SOFOUT signal can be used for time measurements and for global time base synchronisation of different CAN nodes.

Table 18-24: Possible Setup of the SOFOUT Function

SOFSEL	SOFC	SOFE	SOFOUT Function
x	x	0	SOFOUT does not change
0	x	1	Time stamp mode Toggle with each receive interrupt
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 Clear SOFE bit when DCAN starts to store a message in receive buffer 4

SOFC is located in the synchronisation register SYNC1.
$\overline{\text{RESET}}$ and setting of the INIT bit clear the SOFOUT to 0.

Table 18-25: 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

Figure 18-35: Time Stamp Function

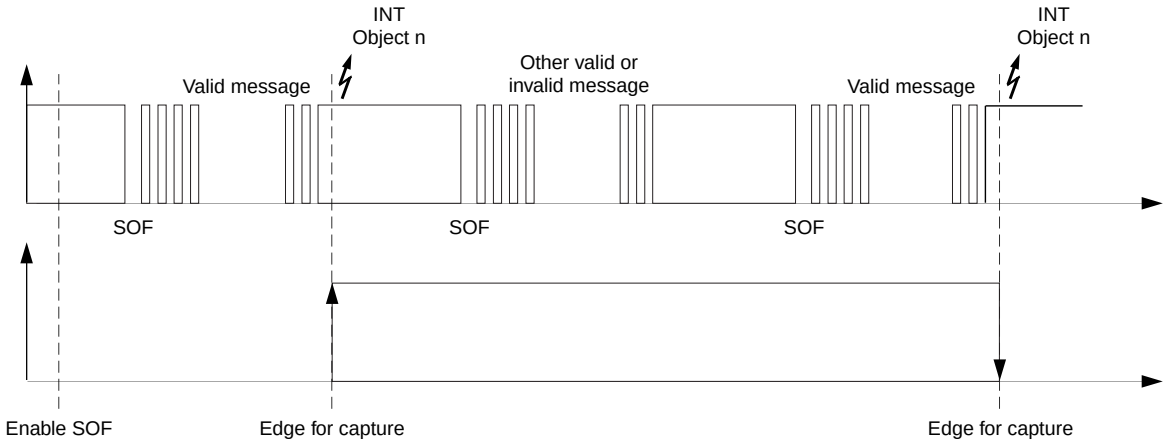


Figure 18-36: SOFOUT Toggle Function

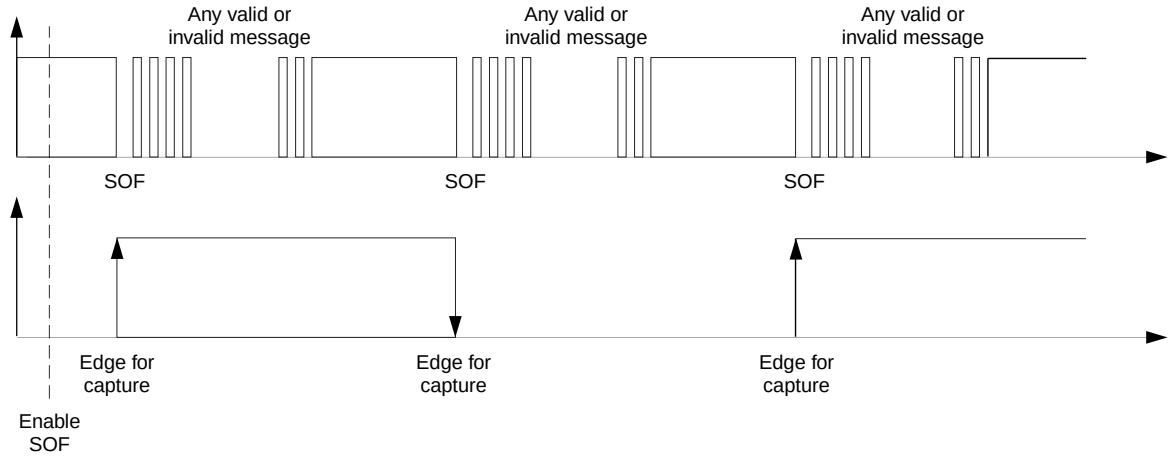
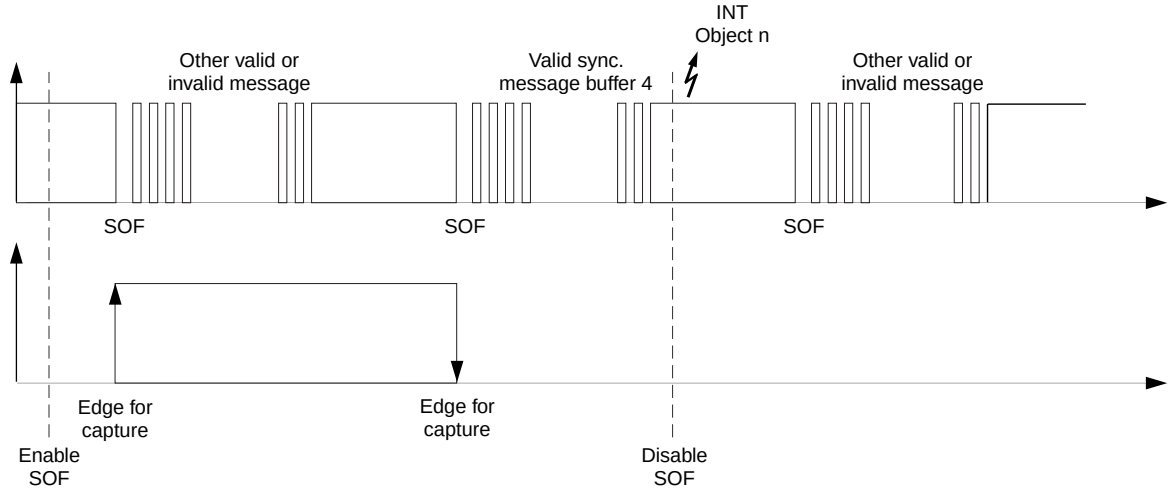


Figure 18-37: Global Time System Function



18.13.2 CAN error status register

These register sets the CAN error status of the transmission and reception.
CANES has to be set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CANC to 00H.

Figure 18-38: CAN Error Status Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
CANES	BOFF	RECS	TECS	0	INITSTATE	VALID	WAKE	OVER	FFB4H	00h	R/W

BOFF, RECS, TECS and INITSTATE are read only bits.

BOFF	Bus Off Flag
0	Transmission error counter ≤ 255
1	Transmission error counter = 255

It changes when the contents of the transmission error counter has changed.

RECS	Reception error counter status
0	Reception error counter < 96
1	Reception error counter ≥ 96 / Warning level for error passive reached

It changes when the contents of the reception error counter changes.

TECS	Transmission error counter status
0	Transmission error counter < 96
1	Transmission error counter ≥ 96 / Warning level for error passive reached

It changes when the contents of the reception error counter changes.

INITSTATE	INIT accepted
0	CAN is in normal operation
1	CAN is stopped and ready to accept new configuration data

It changes with a delay to the INIT bit in CANC. The delay is dependent on running bus activity and the time to set all internal activities to inactive state. Time can be several bit times long.

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.

Whenever a frame is successsfully received by the protocol, it is irrespective of the identifier and mask settings of the DCAN. It is set at the end of the frame part.

Cautions: 1. The VALID bit is cleared if CPU writes “1” to it, or when the INIT-bit in CANC is set.
2. Writing a 0 to the valid bit has no influence.

WAKE	Wakeup Condition
0	Normal operation
1	Sleep mode has been cancelled

This bit is set and an error interrupt is generated under the following circumstances:

a) A CAN bus activities occurs during SLEEP condition of CAN protocol.

b) Any try to set sleep mode in CAN control register during receive or transmit operation will immediately set the Wake up condition.

The CPU must clear this bit after recognition, 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 "1" to it, or when the INIT-bit in CANC is set.
2. Writing a 0 to the WAKE bit has no influence.

OVER	Overrun Condition
0	Normal operation
1	Overrun occured during access to RAM

The Overrun condition is set whenever the CAN can not make all RAM accesses that are necessary for sorting and storing receive data or fetching transmit data. An error interrupt is generated at the same time.

An overrun condition will **not** occur under the following circumstances:
The DCAN clock as defined with the PRM bits in the BRPRS register is at minimum 16 times of the CAN baudrate **and** the selected CPU clock (defined in the DCC register) is at minimum 8 times of the baudrate.

Possible reasons 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 18-26: 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 e. TXRQ1/0 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.

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 zero "0" to them do not change them.
- Writing a one "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.

CAN Transmit Error Counter

These register builds the transmit error counter of the data transmission.
TEC can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets TEC to 00H.

Figure 18-39: Transmit Error Counter

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TEC	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	FFB5H	00H	R

The transmit error counters reflects the status of the error counter for transmission errors as it is defined in the CAN Protocol.

CAN Receive Error Counter

These register builds the receive error counter of the data reception.
REC can be read with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets REC to 00H.

Figure 18-40: Receive Error Counter

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
REC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	FFB6H	00H	R

The receive error counters reflects the status of the error counter for reception errors as it is defined in the CAN Protocol.

Message Count Register

These register sets the number of receive message buffers and 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 18-41: Message Count Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MCNT	CADD1	CADD0	0	MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	FFB7H	C0H	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	No receive message handling
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
					Setting prohibited, will be automatically changed to 16

CADD1	CADD0	DCAN Address Definition
0	0	Setting prohibited
0	1	Setting prohibited
1	0	DCAN uses address range starting at 0F400h
1	1	DCAN uses address range starting at 0F600h, reset value

The DCAN can use different address areas in the IXRAM. This register defines the lower starting address for the DCAN area. The upper limit is defined by the number of messages defined with MCNT.

The DCAN address definition gives the possibility to initialize the 78K0 with different memory configuration with the IXS register. In case of this product it is recommended to use a short address of the expansion RAM (F400h) and to set CADD1 and CADD0 to 1.0.

18.14 Baudrate Generation

Bitrate Prescaler

These registers set the bitrate prescaler for the DCAN.

BRPRS can be set with an 8-bit memory manipulation instruction.

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

Figure 18-42: Bit Rate Prescaler

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRPRS	PRM1	PRM0	BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	FFB8H	00H	R/W

The bit rate prescaler defines the clock sources for DCAN operation.

PRM1	PRM0	Clock Selector for DCAN
0	0	fx is input for DCAN
0	1	fx/2 is input for DCAN
1	0	fx/4 is input for DCAN
1	1	CCLK is input for DCAN

This PRM selector influences all DCAN activities.

BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	Bit Rate Prescaler
0	0	0	0	0	0	DCAN clock / 2
0	0	0	0	0	1	DCAN clock / 4
0	0	0	0	1	0	DCAN clock / 6
0	0	0	0	1	1	DCAN clock / 8
.	.	.	.	.	.	
.	.	.	.	.	.	
.	.	.	.	.	.	
1	1	1	0	1	0	DCAN clock / 118
1	1	1	0	1	1	DCAN clock / 120
1	1	1	1	0	0	DCAN clock / 122
1	1	1	1	0	1	DCAN clock / 124
1	1	1	1	1	0	DCAN clock / 126
1	1	1	1	1	1	DCAN clock / 128

$$\text{Baudrate} = \frac{\text{DCAN clock (PRMn)}}{2 \times \text{BRPRS} + 2}$$

The BRPRS selects the base clock system for the protocol part of the DCAN. The Baudrate is defined by this setting in conjunction with the SYNC1 and SYNC0 registers.

Writing to the BRPRS register is only possible when INIT bit of CANC register is set to initialization mode.

Synchronisation Control Registers 0 and 1

These registers define the bit timing of the DCAN. It selects the length of one data bit on the CAN bus and the position of the sample point during the bit timing.

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 18-43: Synchronization Control Register 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	FFB9H	18H	R/W

SYNC1

DBT4	DBT3	DBT2	DBT1	DBT0	Data Bit Time
Other than under					Setting prohibited
0	0	1	1	1	(Output cycle of BRPRS) x 8
0	1	0	0	0	(Output cycle of BRPRS) x 9
0	1	0	0	1	(Output cycle of BRPRS) x 10
0	1	0	1	0	(Output cycle of BRPRS) x 11
0	1	0	1	1	(Output cycle of BRPRS) x 12
0	1	1	0	0	(Output cycle of BRPRS) x 13
0	1	1	0	1	(Output cycle of BRPRS) x 14
0	1	1	1	0	(Output cycle of BRPRS) x 15
0	1	1	1	1	(Output cycle of BRPRS) x 16
1	0	0	0	0	(Output cycle of BRPRS) x 17
1	0	0	0	1	(Output cycle of BRPRS) x 18
1	0	0	1	0	(Output cycle of BRPRS) x 19
1	0	0	1	1	(Output cycle of BRPRS) x 20
1	0	1	0	0	(Output cycle of BRPRS) x 21
1	0	1	0	1	(Output cycle of BRPRS) x 22
1	0	1	1	0	(Output cycle of BRPRS) x 23
1	0	1	1	1	(Output cycle of BRPRS) x 24
1	1	0	0	0	(Output cycle of BRPRS) x 25
Other than above					Setting prohibited

The length of one data bit on the CAN bus is selected.

Figure 18-43: Synchronization Control Register 0 and 1 (2/2)

SPT4	SPT3	SPT2	SPT1	SPT0	Sample Point Position
Other than under					Setting prohibited
0	0	0	0	1	(Output cycle of BRPRS) x 2
0	0	0	1	0	(Output cycle of BRPRS) x 3
0	0	0	1	1	(Output cycle of BRPRS) x 4
0	0	1	0	0	(Output cycle of BRPRS) x 5
0	0	1	0	1	(Output cycle of BRPRS) x 6
0	0	1	1	0	(Output cycle of BRPRS) x 7
0	0	1	1	1	(Output cycle of BRPRS) x 8
0	1	0	0	0	(Output cycle of BRPRS) x 9
0	1	0	0	1	(Output cycle of BRPRS) x 10
0	1	0	1	0	(Output cycle of BRPRS) x 11
0	1	0	1	1	(Output cycle of BRPRS) x 12
0	1	1	0	0	(Output cycle of BRPRS) x 13
0	1	1	0	1	(Output cycle of BRPRS) x 14
0	1	1	1	0	(Output cycle of BRPRS) x 15
0	1	1	1	1	(Output cycle of BRPRS) x 16
1	0	0	0	0	(Output cycle of BRPRS) x 17
Other than above					Setting prohibited

The sample point during the bit timing is selected.

Synchronization Control Register 1

This register defines the bit timing of the DCAN. It defines the synchronization jump width. This gives the possible range of resynchronization to different transmission speeds.

SYNC1 can be read or written with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets SYNC1 to 0EH.

Figure 18-44: Synchronization Control Register 1

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SYNC1	0	SOFC	SAMP	RXONLY	SJW1	SJW0	SPT4	SPT3	FFBAH	0EH	R/W

SYNC0

SJW1	SJW0	Synchronization Jump Width
0	0	(Output cycle of BRPRS) x 1
0	1	(Output cycle of BRPRS) x 2
1	0	(Output cycle of BRPRS) x 3
1	1	(Output cycle of BRPRS) x 4

SJW0 and SJW1 define the synchronisation jump width as specified in the Bosch CAN specification 2.0. This determines the possible range of synchronisation to different transmission speeds.

Limits on defining bit timing:

$2 \leq SPT \leq 16$
$7 \leq DBT \leq 24$
$2 \leq (DBT - SPT) \leq 8$
$SJW \leq (DBT - SPT -1)$

The following rule must be kept to be conform with the CAN protocol specification:
$1 \leq (2 \times SPT - DBT) \leq 8$

These limits represent the CAN protocol limits. Violating these limits cause to violate the CAN protocol.

Example:

System clock:	fx	8 MHz
CAN parameter:	Baudrate	500kBaud
	Sample point	75%
	SJW	2 BTL

BRPRS =	00h	(Clock selector = fx:	PRM = 00b)
		(DCAN Prescaler = 1/2:	DBT = 000000b)
SYNC0 =	A7h	(CAN Bit in BTL = 8	
		[7 < (fx/DBT/Baudrate) < 26]	DBT = 00111b)
SYNC1 =	x4h	(sample point 75% = 6:	SPT = 00101b)
		(SJW = 2	SJW = 01b)

x depends on the setting of:
- Number of sampling points
- Receive only function
- Use of time stamp or global time system

RXONLY	Receive Only Operation
0	Normal operation
0	Only receive operation, CAN does not activate transmit line

This operation mode can be used for baudrate detection. It makes it possible to try different baudrate configurations without disturbing other CAN nodes on the bus.

Differences to CAN protocol:

- Never sends acknowledge, error frames or transmit messages.
- Error counters do not count.

The VALID bit in CANES shows whether the protocol receives any valid message.

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	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

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 the SFR register and the time function mode.

Caution: CPU can read SYNC0/1 register at any time. Writing to the SYNC0/1 registers is only possible when INIT bit of CANC register is set to initialization mode.

18.15 Function Control

18.15.1 Transmit control

Transmit Control Register

This register controls the transmission of the DCAN-module. The Transmit Control register 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.
$\overline{\text{RESET}}$ input sets TCR to 00H.

Figure 18-45: Transmit Control Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCR	TXP	0	TXC1	TXC0	TXA1	TXA0	TXRQ1	TXRQ0	FFB1H	00H	R/W

TXC1 and TXC0 are read only bits.

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 buffer is requested, TXP has no influence.

This bit is checked by the DCAN immediately before the frame is started. The order in which the TXRQ1/0 bits will be set does not influence as long as the first requested frame is not started on the bus.

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

Setting the TXAn bit by the CPU request the CAN to free this buffer.

The TXAn bits have 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 is cleared.

This abort does not affect any rules of the CAN protocol. An already started frame will continue to its end.

An abort operation can cause different results dependent on the time it is set..

a) Abort is received before the start of the arbitration for transmit.
The TXCn bit is reset showing that the buffer was not send to other nodes.
b) Abort is received during the arbitration, but arbitration is lost.
The TXCn bit is reset showing that the buffer was not send to other nodes.
c) Abort is received during frame transmission, but transmission ends with an error.
The TXCn bit is reset showing that the buffer was not send to other nodes.
d) Abort is received during the frame transmission and transmission ends without error.
The TXCn bit is set showing a successful transfer of the data before the abort gets active.

In all cases the TXRQn bit and the TXAn bit will be cleared at the end of the abort operation, when the receive buffer is available again.

Cautions: 1. The bits are cleared when the INIT-bit in CANC is set.
2. Writing a 0 to TXAn bit has no influence

The TXCn bit are updated at the End of every frame transmission or abort.

TXRQn	Transmission Request Flag
0	Write: no transmission requested for this buffer Read: transmit buffer is free
1	Write: request transmission for buffer n Read: transmit buffer is occupied by former transmit request

The TXRQn bits have dual function:

1. Request the transmission of a transmit buffer.

2. Inform the CPU whether a buffer is available or is still occupied by a former transmit request.

Setting the transmission request bit requests the CAN to sent the buffer contents onto the bus. The CAN clears the bit after completion of the transmit. Completion is either a normal transfer free of 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.
2. Writing a 0 to TXR0n bit has no influence
3. Do not use bit operations on this register.
4. Do not change data in transmit buffer n when TXRQn is set.

18.15.2 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 in 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.

Receive Message Register

These register controls the reception of the messages of the DCAN-module

RMES can be read with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets RMES to 00H.

Figure 18-46: Receive Message Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
RMES	DN7	DN6	DN5	DN4	DN3	DN2	DN1	DN0	FFB2H	00H	R

DNn	Data New Bit for Message n (n = 0...7)
0	No message received on message n, CPU has cleared DN bit in message n
1	New data received in message n

This register is read only and it is cleared when the INIT bit in CANC is set.

DN0 has no meaninig when Rec. Buffer 0 is configured for mask operation in the Mask Control Register.

DN2 has no meaninig when Rec. Buffer 2 is configured for mask operation in the Mask Control Register.

18.15.3 Mask control

The mask Control Register defines whether the DCAN compares the identifier of a received message in its whole length or some bits are not used for comparison.

This functionality is provided by the use of 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 in this register. This buffer is not used for normal message storage then.

Mask Control Register

These register controls the mask function of the receive messages of the DCAN-module.

MASKC can be written with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MASKC to 00H.

Figure 18-47: Mask Control Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MASKC	0	0	0	0	0	GLOBAL	MSK1	MSK0	FFBBH	00H	R/W

MSK0	Mask 0 Enable
0	Rec. buffer 0 and 1 in normal operation
1	Rec. buffer 0 is mask for buffer 1

MSK1	Mask 1 Enable
0	Rec. buffer 2 and 3 in normal operation
1	Rec. buffer 2 is mask for buffer 3

GLOBAL	Enable Global Mask
0	Normal mask operation
1	Highest defined mask is active for all following buffers

This register is readable at any time. Write is only permitted when the CAN is in initialization mode.

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 as mask information and 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 18-27: 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 mask
1	0	1	Mask0	Compare ID & mask0	Compare ID & mask0	Compare ID & mask0	Compare ID & mask0	Global mask
1	1	0	Compare ID	Compare ID	Mask1	Compare ID & mask1	Compare ID & mask1	Two normal, rest global mask
1	1	1	Mask0	Compare ID & mask0	Mask1	Compare ID & 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 an incoming message. In this case it is necessary to priorize the receive buffers.

The priority of the 16 receive buffers depend on the setup of the mask register.

The rules for priority are:

- Lower numbered receive buffers have higher priority
- A masked receive buffer has a lower priority than all non-masked receive buffers.

18.15.4 Special functions

Redefinition Control Register

These register controls redefinition of an identifier of a receive messages of the DCAN-module.
REDEF can be written with a 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets REDEF to 00H.

Figure 18-48: Redefinition Control Register

Symbol	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
REDEF	DEF	0	0	0	SEL3	SEL2	SEL1	SEL0	FFB3H	00H	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.

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. Do not change DEF and SEL at the same time. Change SEL only when DEF is cleared.
2. Write first SEL with DEF cleared. Write than SEL with DEF, or use bit manipulation instruction.
3. Clear DEF but keeping SEL set to same value.

Setting the redefinition bit removes the selected receive buffer from the list of possible ID hits during identifier comparisions.

There is one special case:

The message is complete and the DCAN started the storage into the RAM. In this case the complete message is stored and the redefinition request is ignored. Programmers should be prepared to receive this message immediately during redefinition. The user can identify this situation because the data new bit (DN) in the receive buffer will be set. The user must monitor this bit when he changes the identifier and the buffer is in mask operation, because in this case the DCAN also writes the identifier part of the message.

18.16 Interrupt Information

18.16.1 Interrupt vectors

The DCAN peripheral supports four interrupt sources as shown in the following table.

Table 18-28: 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

18.16.2 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 this bits mean that the buffer gets free for writing a new message into it. The clear can occur due to a successful transmission or due to an abort of a transmission and can only cleared by the DCAN.

18.16.3 Receive interrupt

The receive interrupt is generated when all following conditions are fulfilled:

- CAN protocol port marks received frame valid
- Memory access engine finds a message buffer with a identifier/mask combination that fits to the receive frame
- 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 compare and storage function can delay the interrupt up to the 7th bit of the next frame.

18.16.4 Error interrupt

The error interrupt is generated when all following conditions are fulfilled:

- Transmission error counter (BOFF) changes it state.
- Transmission error counter status (TECS) changes it state.
- Reception error status (RECS) changes it state.
- Overrun during RAM access (OVER) changes it state.
- The wake-up condition (WAKE) occurs.

The internal WAKE conditions sets the interrupt high. The interrupt is kept high until the DCAN has got enough clock cycles to safely restart all internal activities. This may need several bit periods of the CAN bus.

18.17 Standby Function

18.17.1 CPU halt mode

The Halt mode is possible in conjunction with CAN sleep mode.

18.17.2 CPU stop mode

The DCAN stops any activity when its clock supply stops. This may cause an errorneous behavior on the CAN bus when the clock is not sychronized to bus activities.

The DCAN will reach an overrun condition, when it receives clock supply again.

Stop mode is possible in conjunction with CAN sleep mode.

18.17.3 DCAN sleep mode

The DCAN has a sleep mode that is original defined in the CAN specification from Bosch.

The CPU requests the sleep mode from DCAN. The DCAN will signal, if it is not possible to enter the sleep mode due to ongoing bus activities. After a successful switch to the sleep mode, the CPU can safely go into HALT or STOP mode. The DCAN can futhermore scan the CAN bus and interrupt the CPU. This WAKEUP is independent from the clock. The release time for the STOP mode of the device is not critical, since the DCAN synchronizes again to the CAN bus after clock supply has started.

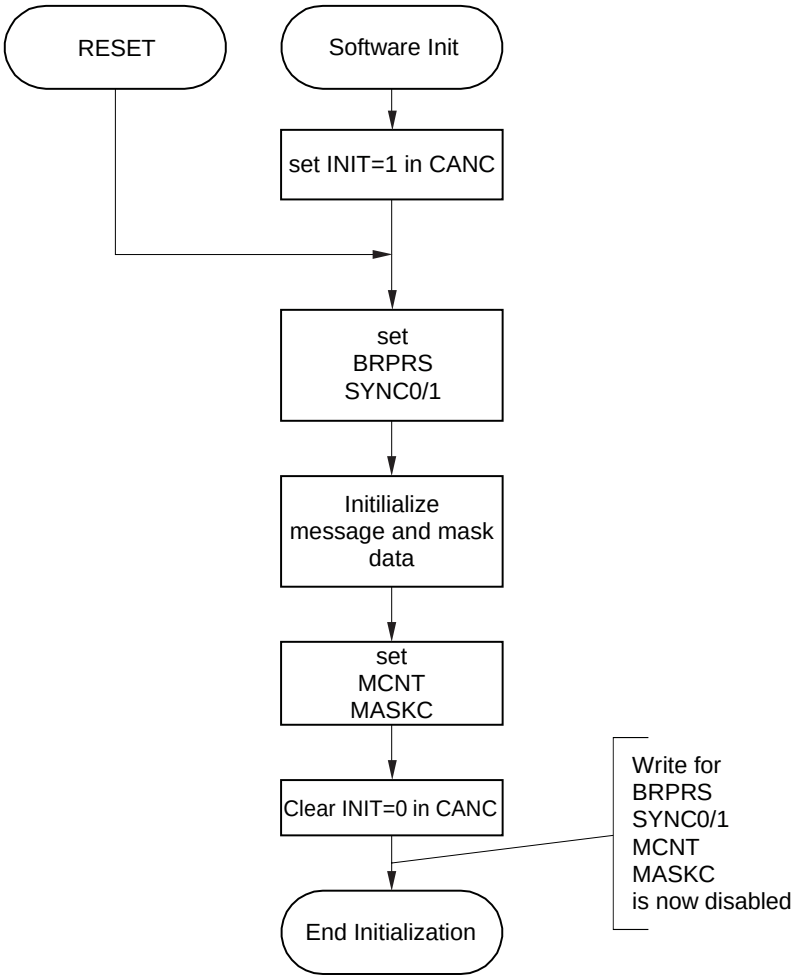
18.17.4 DCAN stop mode

The CPU requests the stop mode from DCAN. The DCAN will signal, if it is not possible to enter the stop mode due to ongoing bus activities. After a successful switch to the stop mode, the CPU can safely go into HALT or STOP mode. The DCAN can only wake up by the CPU in order to clear the SLEEP bit in the CANC register.

18.18 Functional Description by Flowcharts

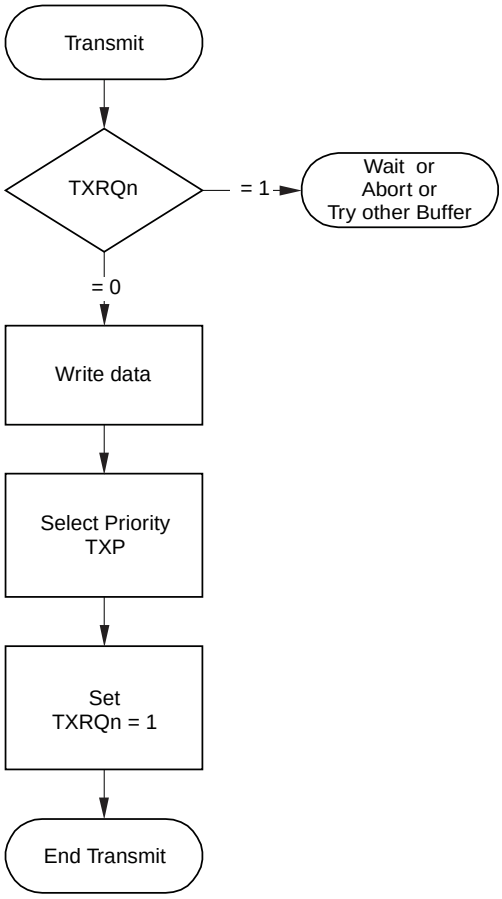
18.18.1 Initialization

Figure 18-49: Initialization Flow Chart



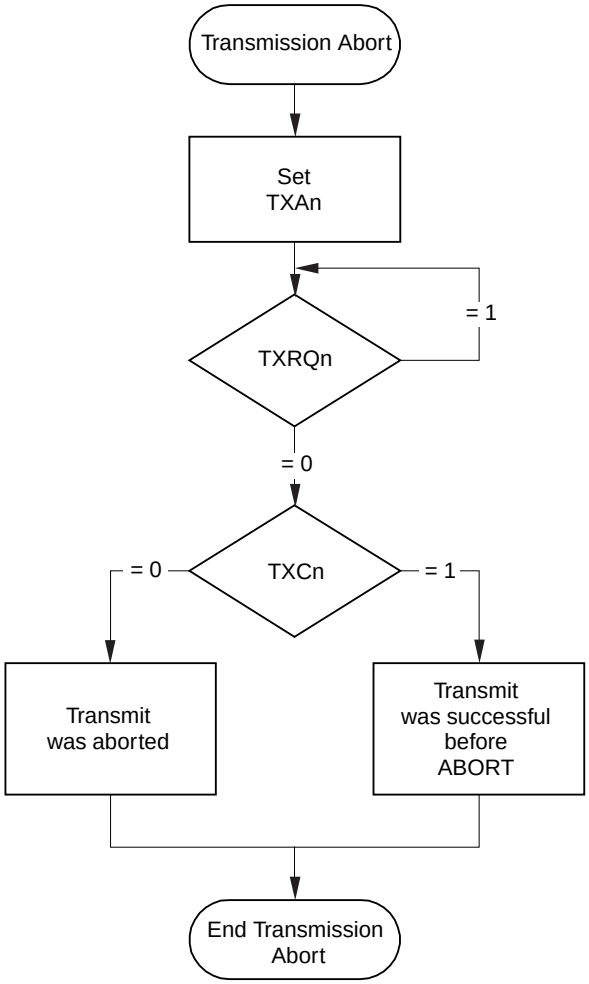
18.18.2 Prepare transmit

Figure 18-50: Transmit Preparation



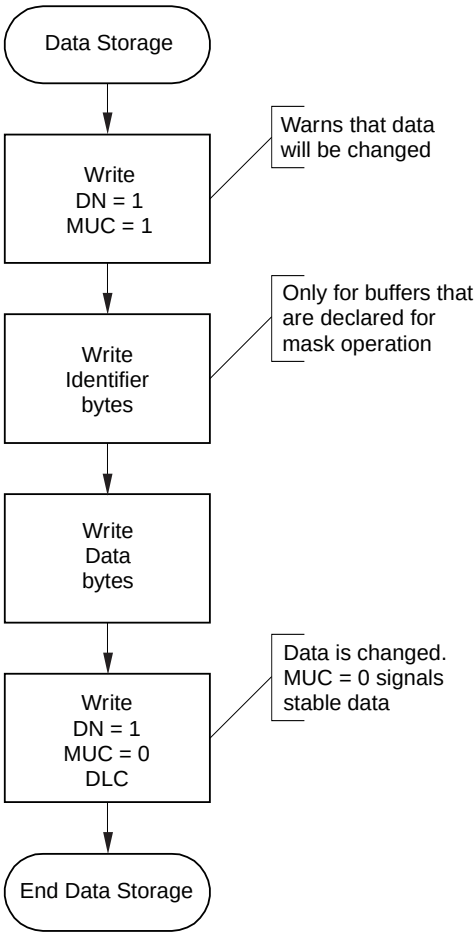
18.18.3 Abort transmit

Figure 18-51: Transmit Abort, Software Flow



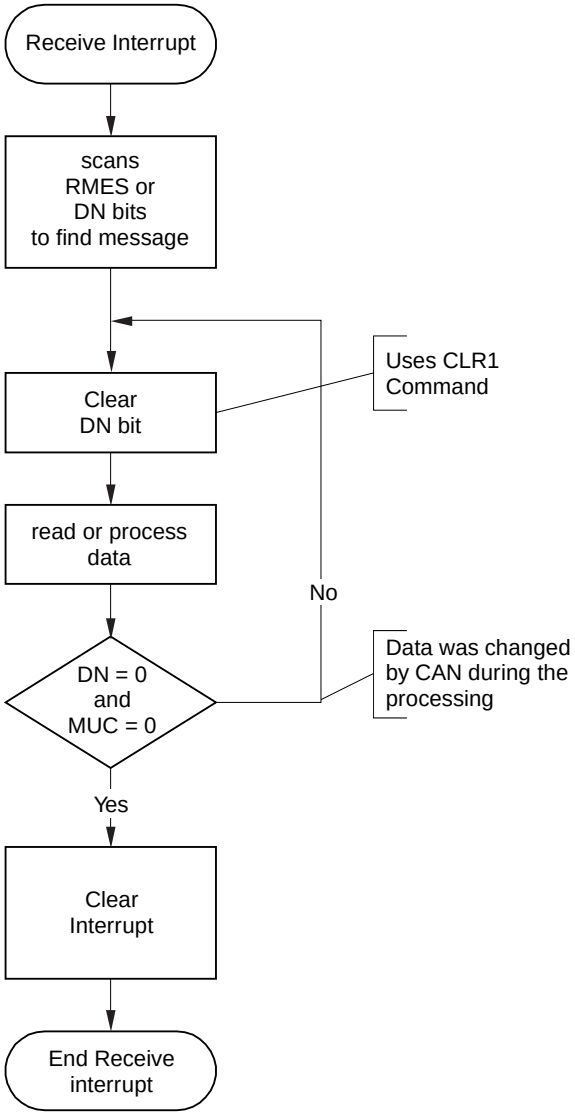
18.18.4 Handling by the DCAN

Figure 18-52: Handling of Semaphore Bits by DCAN-Module



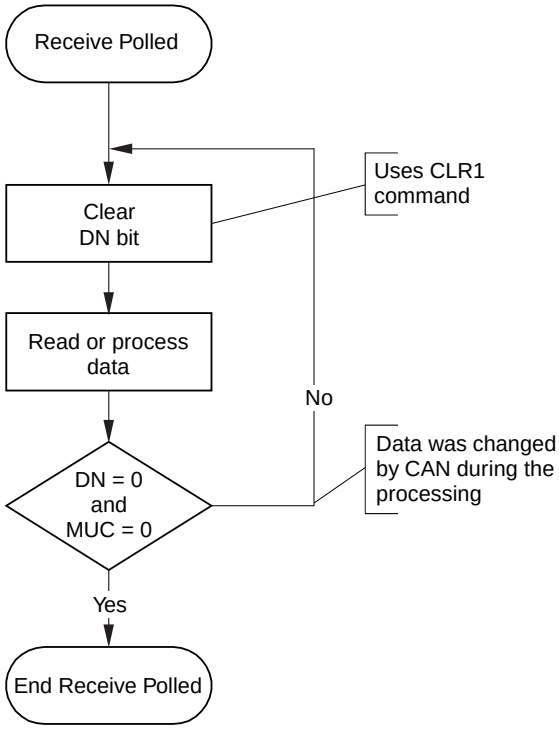
18.18.5 Receive event oriented

Figure 18-53: Receive with Interrupt, Software Flow



18.18.6 Receive task oriented

Figure 18-54: Receive, Software Polling



[Memo]

Chapter 19 LCD Controller/Driver

19.1 LCD Controller/Driver Functions

The functions of the LCD controller/driver incorporated in the μPD780949 Subseries are shown 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 40 segment signal outputs (S0 to S39); 4 common signal outputs (COM0 to COM3). All segment outputs can be switched to input/output ports.
P147/S0 to P140/S7, P137/S8 to P130/S15 and P127/S16 to P120/S23 are bytewise switchable.
P77/S24 to P70/S31 and P57/S32 to P50/S39 are bitwise switchable.
(5) The operation with the subsystem clock is not available.

The maximum number of displayable pixels is shown in Table 19-1.

Table 19-1: Maximum Number of Display Pixels

Bias Method	Time Division	Common Signals Used	Maximum Number of Display Pixels
1/3	4	COM0 to COM3	160 (40 segments x 4 commons)

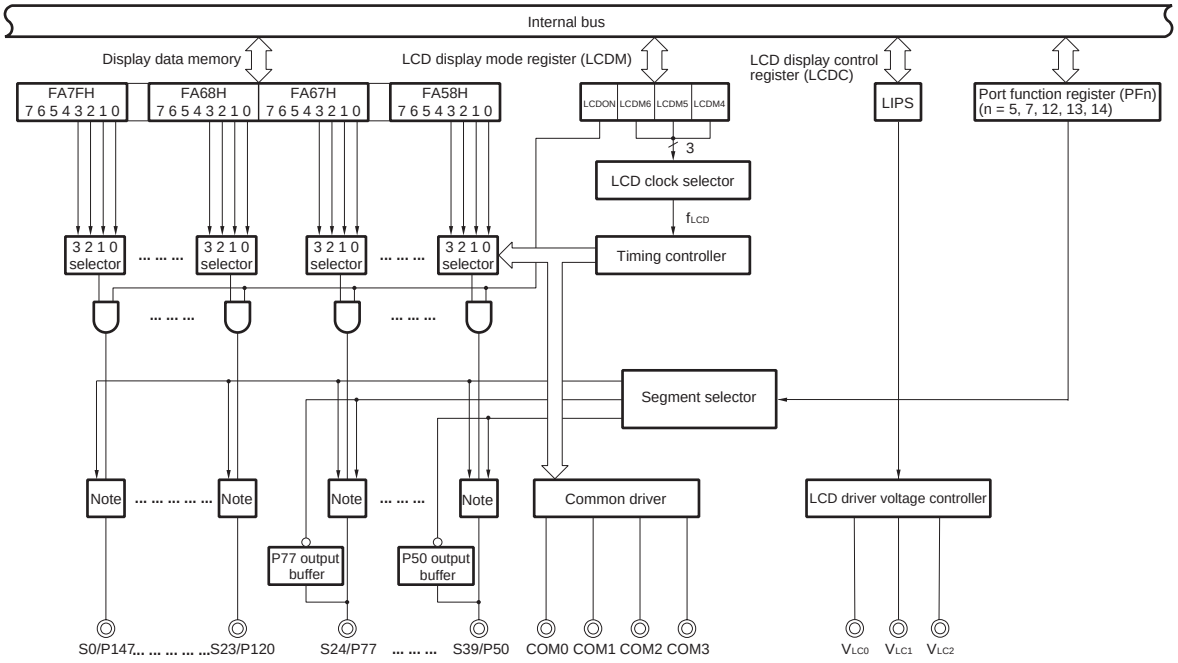
19.2 LCD Controller/Driver Configuration

The LCD controller/driver consists of the following hardware.

Table 19-2: LCD Controller/Driver Configuration

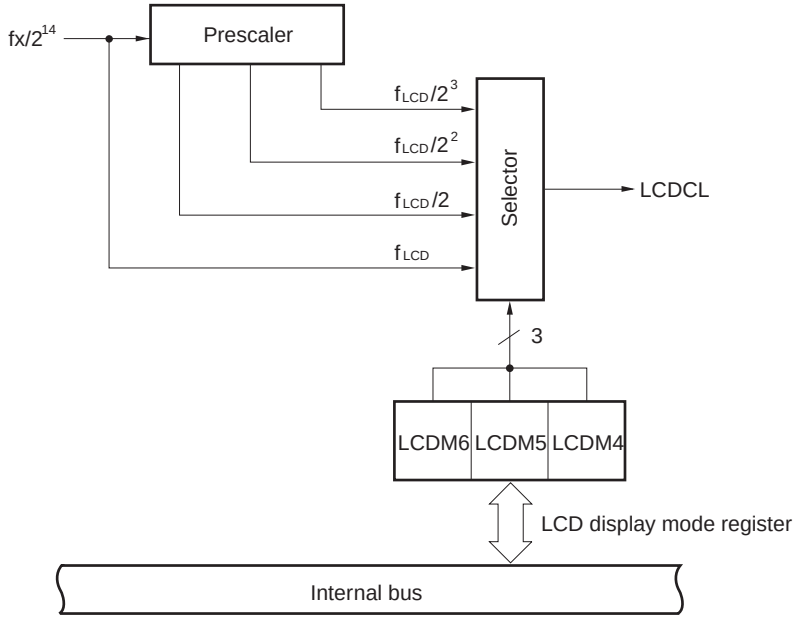
Item	Configuration
Display outputs	Segment signals: 40 Segment signal with alternate function: 40 Common signals: 4 (COM0 to COM3)
Control registers	LCD display mode register (LCDM) LCD display control register (LCDC)

Figure 19-1: LCD Controller/Driver Block Diagram



Note: Segment driver

Figure 19-2: LCD Clock Select Circuit Block Diagram



Remarks: **1.** LCDCL : LCD clock
2. f_{LCD} : LCD clock frequency

19.3 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 sets display operation enabling/disabling, the LCD clock, frame frequency.

LCDM is set with a 1-bit or 8-bit memory manipulation instruction.

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

Figure 19-3: 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 Display Enable/Disable
0	Display off (all segment outputs are non-select signal outputs)
1	Display on

LCDM6	LCDM5	LCDM4	LCD Clock Selection (fx = 8.00 MHz)
0	0	0	$fx/2^{17}$ (61 Hz)
0	0	1	$fx/2^{16}$ (122 Hz)
0	1	0	$fx/2^{15}$ (244 Hz)
0	1	1	$fx/2^{14}$ (488 Hz)
Other than above			Setting prohibited

Remark: fx = Main system 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 a 1-bit or 8-bit memory manipulation instruction.

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

Figure 19-4: 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	00H	R/W

LIPS	LCD Driving Power Supply Selection
0	Does not supply power to LCD
1	Supplies power to LCD from V_{DD} pin

Caution: Set bit7 to 1 and bit 1 to bit 6 to 0.

19.4 LCD Controller/Driver Settings

LCD controller/driver settings should be performed as shown below.

<1> Set the initial value in the display data memory (FA58H to FA7FH).
<2> Set the pins to be used as segment outputs in port function registers (PF5, PF7, PF12, PF13 and PF14).
<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.

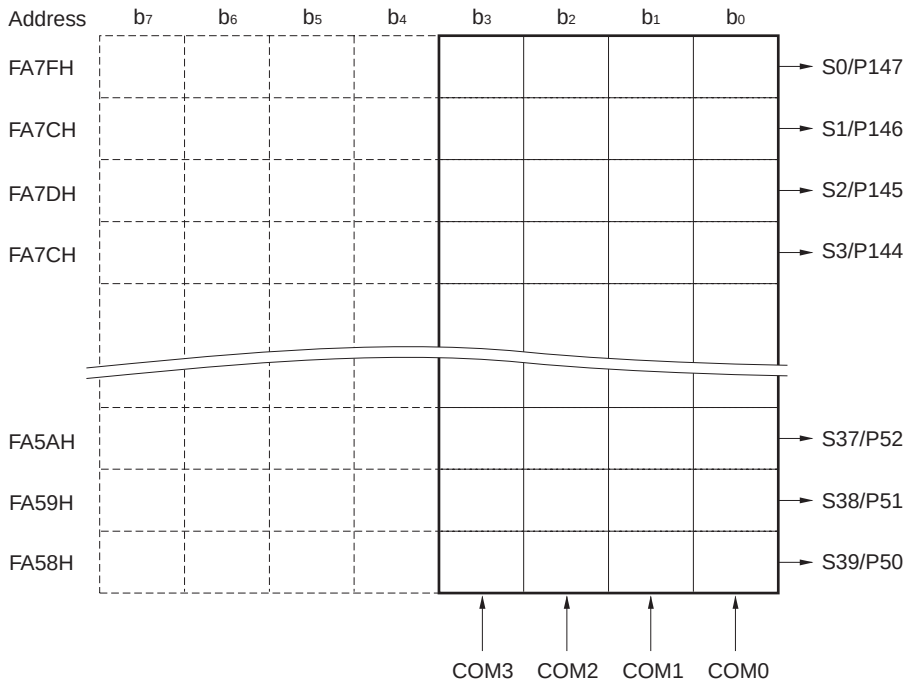
19.5 LCD Display Data Memory

The LCD display data memory is mapped onto addresses FA58H to FA7FH. The data stored in the LCD display data memory can be displayed on an LCD panel by the LCD controller/driver.

Figure 19-5 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.

Figure 19-5: Relationship between LCD Display Data Memory Contents and Segment/Common Outputs



Caution: The higher 4 bits of the LCD display data memory do not incorporate memory. Be sure to set them to 0.

Note: The data of S0 is stored at the highest address in the LCD display data memory.

19.6 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 19-3, and operations are repeated with these as the cycle.

Table 19-3: COM Signals

COM Signal / Time Division	COM0	COM1	COM2	COM3
4-Time Division	▲			►

(2) Segment signals

Segment signals correspond to a 40-byte LCD display data memory (FA58H 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 output to a segment pin (S0 to S39) (S39 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 form the display pattern, and then write 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 19-4 are output in the common signals and segment signals.

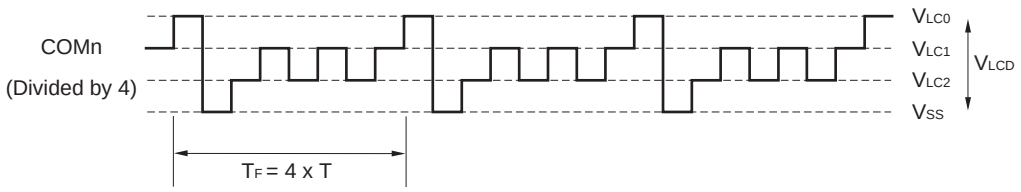
The $\pm V_{LCD}$ ON voltage is only produced when the common signal and segment signal are both at the selection voltage; other combinations produce the OFF voltage.

Table 19-4: 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}

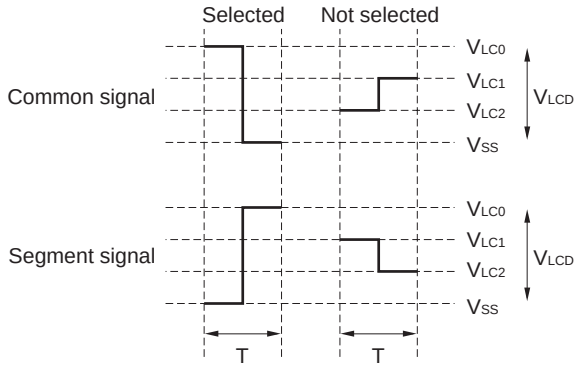
Figure 19-6 shows the common signal waveform, and Figure 19-7 shows the common signal and segment signal voltages and phases.

Figure 19-6: Common Signal Waveform



T: One LCDCL cycle
TF: Frame frequency

Figure 19-7: Common Signal and Segment Signal Voltages and Phases



T: One LCDCL cycle

19.7 Supplying LCD Drive Voltage V_{LC0}, V_{LC1}, and V_{LC2}

The μPD780949 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.

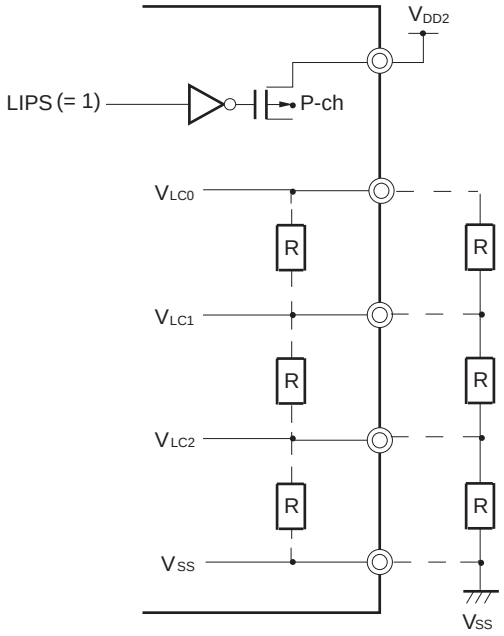
Table 19-5: LCD Drive Voltage Suupy

Bias Method / LCD Drive Voltage	1/3 Bias Method
V_{LC0}	V_{LC0}
V_{LC1}	2/3 V_{LC0}
V_{LC2}	1/3 V_{LC0}

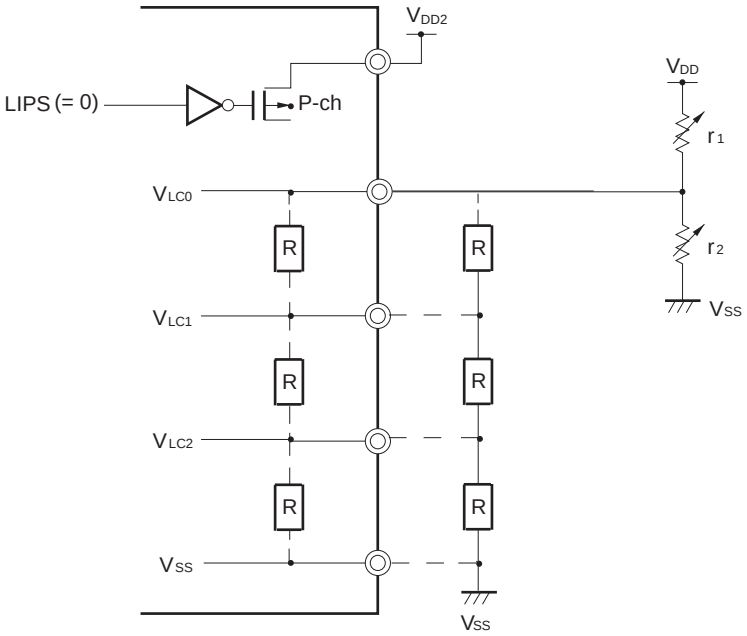
Figure 19-8 shows an example of supplying an LCD drive voltage from an internal source according to Table 19-5. By using variable resistors r1 and r2, a non-stepwise LCD drive voltage can be supplied.

Figure 19-8: Example of Connection of LCD Drive Power Supply

(a) To supply LCD drive voltage from V_{DD}



(b) To supply LCD drive voltage from external source



Remarks: **1.** The flash version μPD78F0948/μPD78F0949 has no interval split resistor.
2. The Mask version μPD780948/μPD780949 has the possibility to implement interval split resistors.

19.8 Display Mode

19.8.1 4-time-division display example

Figure 19-10 shows the connection of a 4-time-division type 10-digit LCD panel with the display pattern shown in Figure 19-9 with the μPD780949 Subseries segment signals (S0 to S19) and common signals (COM0 to COM3). The display example is "1234567890," and the display data memory contents (addresses FA59H to FA6CH) correspond to this.

An explanation is given here taking the example of the 5th digit "6" (). In accordance with the display pattern in Figure 19-9, selection and non-selection voltages must be output to pins S8 and S9 as shown in Table 19-6 at the COM0 to COM3 common signal timings.

Table 19-6: Selection and Non-Selection Voltages (COM0 to COM3)

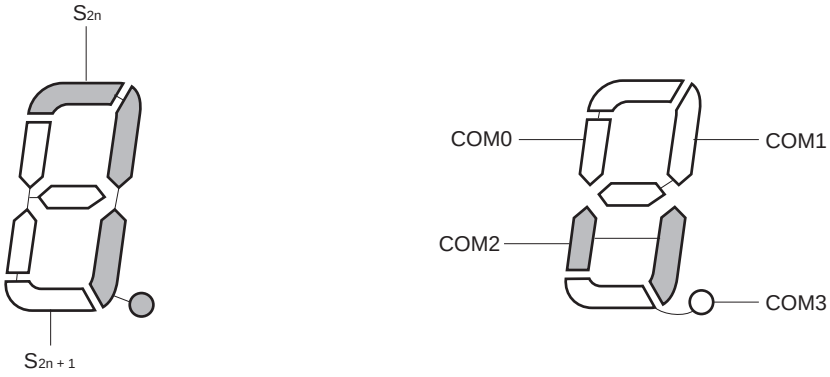
Segment / Common	S8	S9
COM0	S	S
COM1	NS	S
COM2	S	S
COM3	NS	S

S: Selection, NS: Non-selection

From this, it can be seen that 0101 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 19-11 (for the sake of simplicity, waveforms for COM2 and COM3 have been omitted). When S8 is at 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 19-9: 4-Time-Division LCD Display Pattern and Electrode Connections



n = 0 to 9

Figure 19-10: 4-Time-Division LCD Panel Connection Example

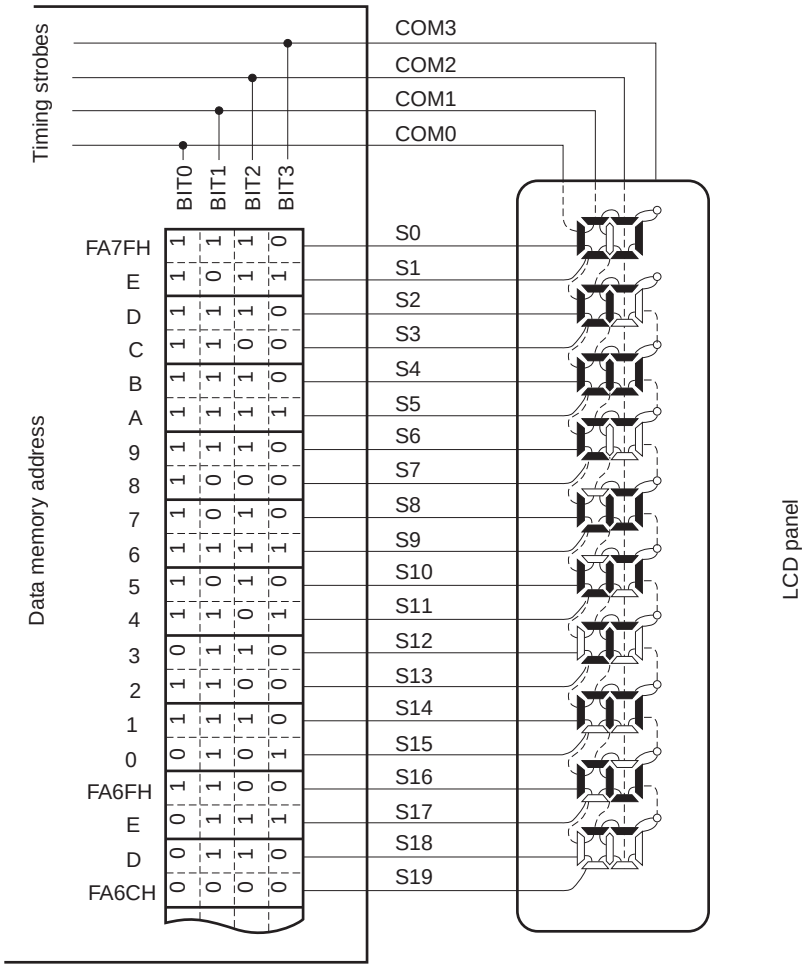
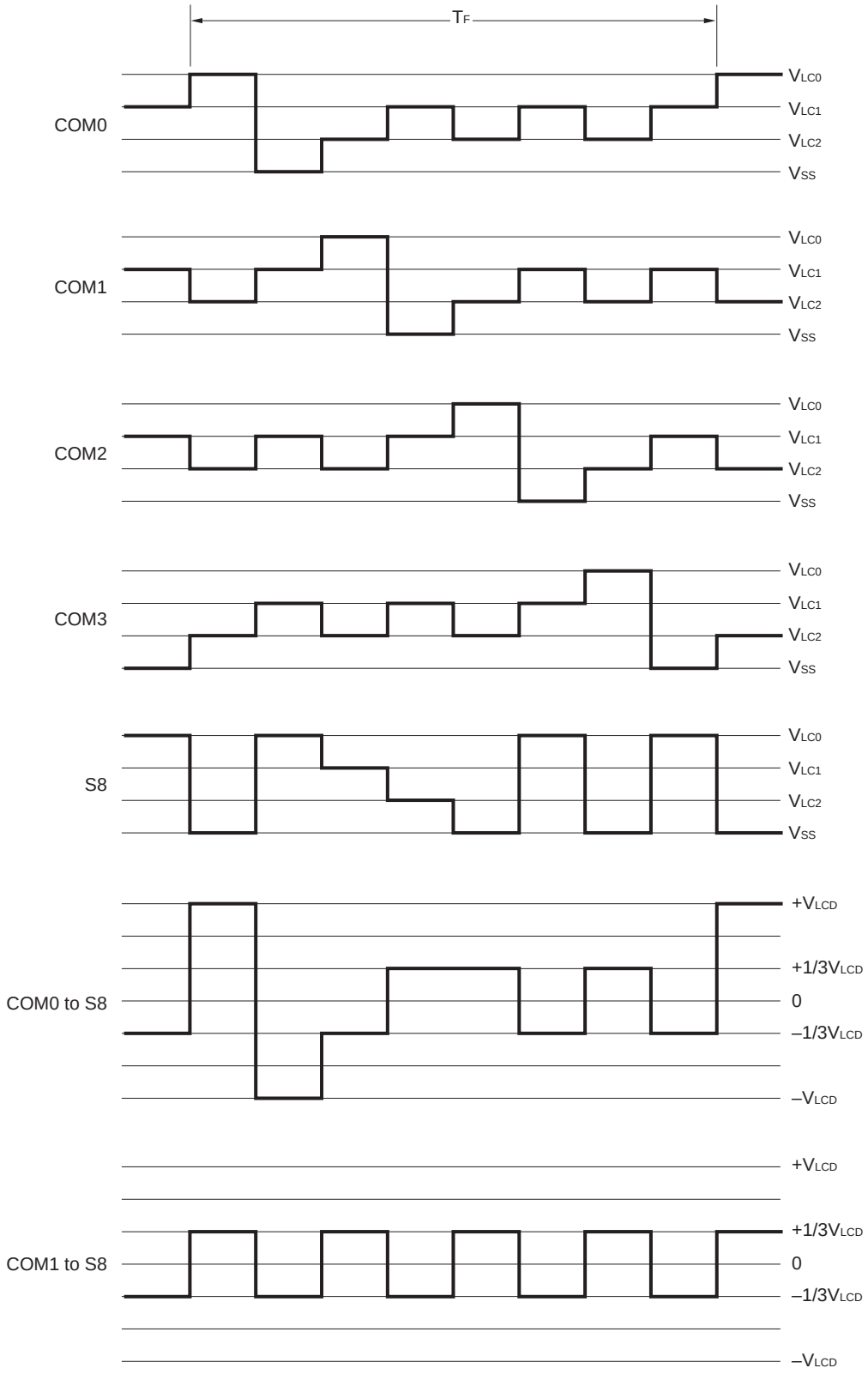


Figure 19-11: 4-Time-Division LCD Drive Waveform Examples (1/3 Bias Method)



19.9 Cautions on Emulation

To perform debugging with an in-circuit emulator (IE-78001-R-A), the LCD timer control register (LCDTM) must be set. LCDTM is a register used to set a I/O board (IE-780948-SL-EM1).

19.9.1 LCD timer control register (LCDTM)

LCDTM is a write-only register that controls supply of the LCD clock. Unless LCDTM is set, the LCD controller/ driver does not operate. Therefore, set bit 1 (TMC21) of LCDTM to 1 when using the LCD controller/driver.

Figure 19-12: LCD Timer Control Register (LCDTM) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
LCDTM	0	0	0	0	0	0	TMC21	0	FF4AH	00H	W

TMC21	LCD Clock Supply Control
0	LCD controller/driver stop mode (supply of LCD clock is stopped)
1	LCD controller/driver operating mode (supply of LCD clock is enabled)

Cautions: 1. LCDTM is a special register that must be set when debugging is performed with an in-circuit emulator. Even if this register is used, the operation of the μPD780948 Subseries is not affected. However, delete the instruction that manipulates this register from the program at the final stage of debugging.

2. Bits 7 to 2, and bit 0 must be set to 0.

[Memo]

Chapter 20 Sound Generator

20.1 Sound Generator Function

The sound generator has the function to sound the buzzer from an external speaker, and the following two signals are output.

(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 fx = 8.38 MHz) can be output. 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 enable control of the buzzer sound volume.

(2) Amplitude output signal
A PWM signal with a 7-bit resolution for variable amplitude can be independently output.

Figure 20-1 shows the sound generator block diagram and Figure 24-2 shows the concept of each signal.

Figure 20-1: Sound Generator Block Diagram

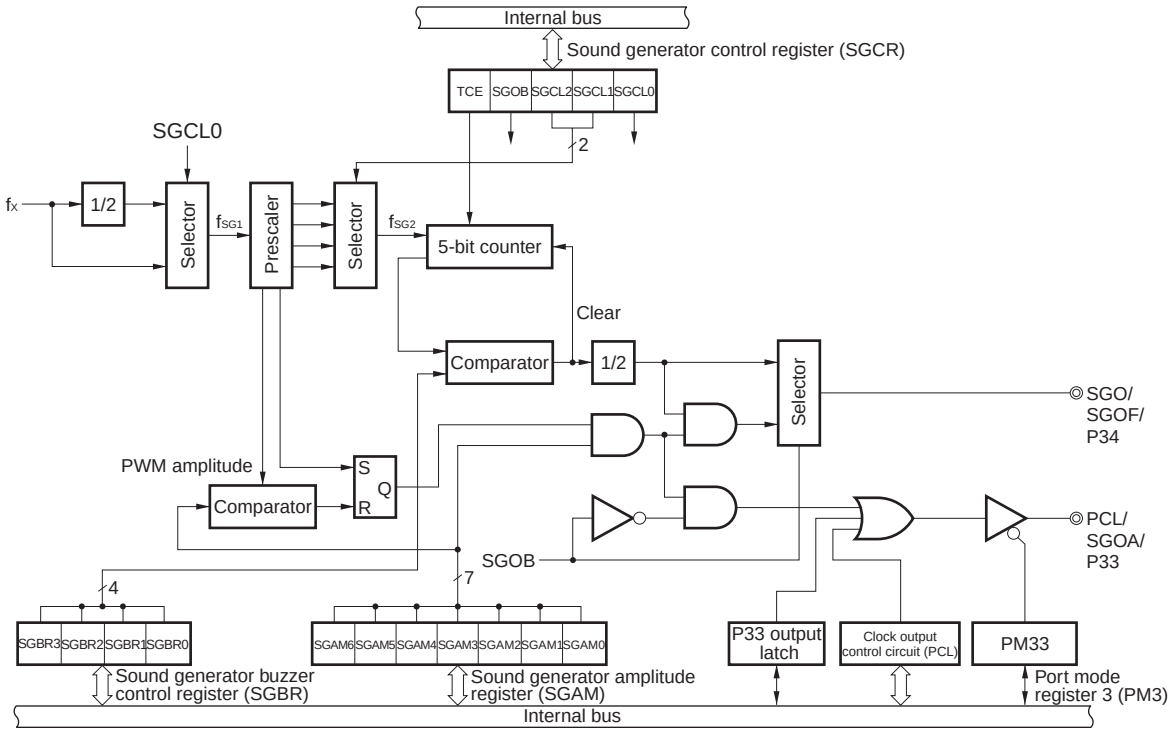
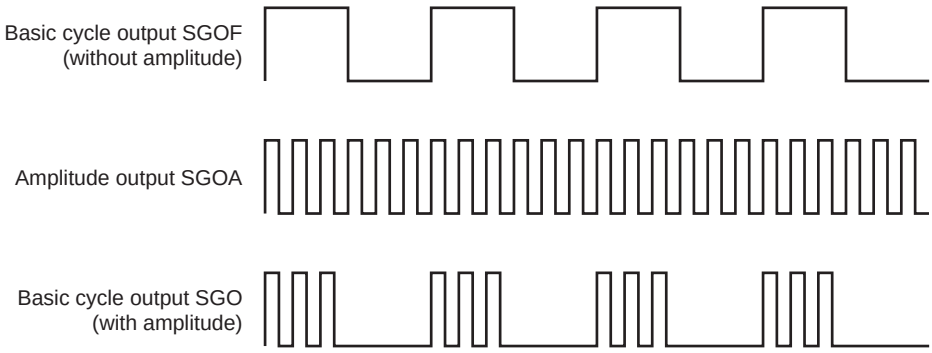


Figure 20-2: Concept of Each Signal



20.2 Sound Generator Configuration

The sound generator consists of the following hardware.

Table 20-1: Sound Generator Configuration

Item	Configuration
Counter	8 bits x 1, 5 bits x 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)

20.3 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 20-3 shows the SGCR format.

Figure 20-3: Sound Generator Control Register (SGCR) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SGCR	TCE	0	0	0	SGOB	SGCL2	SGCL1	SGCL0	FFC0H	04H	R/W

TCE	Sound Generator Output Selection
0	Timer operation stopped SGOF/SGO and SGOA for low-level output
1	Sound generator operation SGOF/SGO and SGOA for output

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	Selects SGOF and SGOA outputs
1	Selects SGO and PCL outputs

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	$f_{SG2} = f_{SG1}/2^8$

SGCL0	Sound Generator Input Frequency Selection
0	$f_{SG1} = f_X/2$
1	$f_{SG1} = f_X$

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.

Table 20-2: Maximum and Minimum Values of the Buzzer Output Frequency

SGCL2	SGCL1	SGCL0	Maximum and Minimum Values of Buzzer Output				
			f_{SG2}	fx = 8 MHz		fx = 8.38 MHz	
				Max. (kHz)	Min. (kHz)	Max. (kHz)	Min. (kHz)
0	0	0	$f_{SG1}/2^6$	3.677	1.953	3.851	2.046
0	0	1	$f_{SG1}/2^5$	7.354	3.906	7.702	4.092
0	1	0	$f_{SG1}/2^7$	1.838	0.976	1.926	1.024
0	1	1	$f_{SG1}/2^6$	3.677	1.953	0.481	2.046
1	0	0	$f_{SG1}/2^8$	0.919	0.488	0.963	0.512
1	0	1	$f_{SG1}/2^7$	1.838	0.976	1.926	1.024
1	1	0	$f_{SG1}/2^9$	0.460	0.244	0.481	0.256
1	1	1	$f_{SG1}/2^8$	0.919	0.488	0.963	0.512

The sound generator output frequency f_{SG} can be calculated by the following expression.

$$f_{SG} = 2^{(SGCL0 - SGCL1 - 2 \times SGCL2 - 7)} \times \{fx/(SGBR + 17)\}$$

Substitute set 0 or 1 to SGCL0 to SGCL2 in the above expression. Substitute a decimal value to SGBR. Where fx = 8 MHz, SGCL0 to SGCL2 is (1, 0, 0), and SGBR0 to SGBR3 is (1, 1, 1, 1), SGBR = 15. Therefore,

$$f_{SG} = 2^{(1 - 0 - 2 \times 0 - 7)} \times \{fx/(15 + 17)\}$$
$$= 3.906 \text{ kHz}$$

(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 a 1-bit or 8-bit memory manipulation instruction.

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

Figure 20-4 shows the SGBR format.

Figure 20-4: 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	Buzzer Output Frequency (kHz) Note	
				fx = 8 MHz)	fx = 8.38 MHz)
0	0	0	0	3.677	3.851
0	0	0	1	3.472	3.637
0	0	1	0	3.290	3.446
0	0	1	1	3.125	3.273
0	1	0	0	2.976	3.117
0	1	0	1	2.841	2.976
0	1	1	0	2.717	2.847
0	1	1	1	2.604	2.728
1	0	0	0	2.500	2.619
1	0	0	1	2.404	2.518
1	0	1	0	2.315	2.425
1	0	1	1	2.232	2.339
1	1	0	0	2.155	2.258
1	1	0	1	2.083	2.182
1	1	1	0	2.016	2.112
1	1	1	1	1.953	2.046

Note: Output frequency where SGCL0, SGCL1, and SGCL2 are 0, 0, and 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.

(3) Sound generator amplitude register (SGAM)

SGAM is a register that sets the amplitude of the sound generator output signal.

SGAM is set with a 1-bit or 8-bit memory manipulation instruction.

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

Figure 20-5 shows the SGAM format.

Figure 20-5: 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.

20.4 Sound Generator Operations

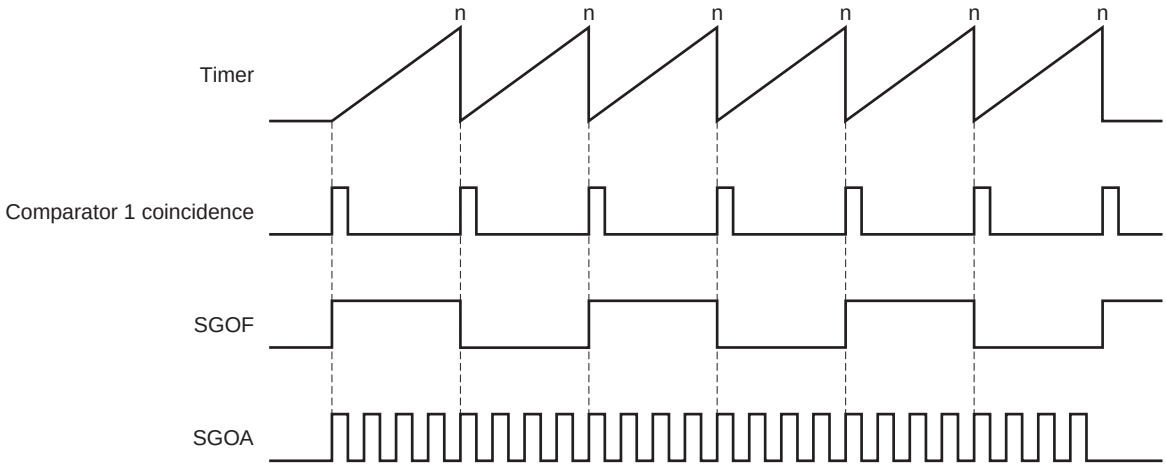
20.4.1 To output basic cycle signal SGOF (without amplitude)

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 20-6: Sound Generator Output Operation Timing



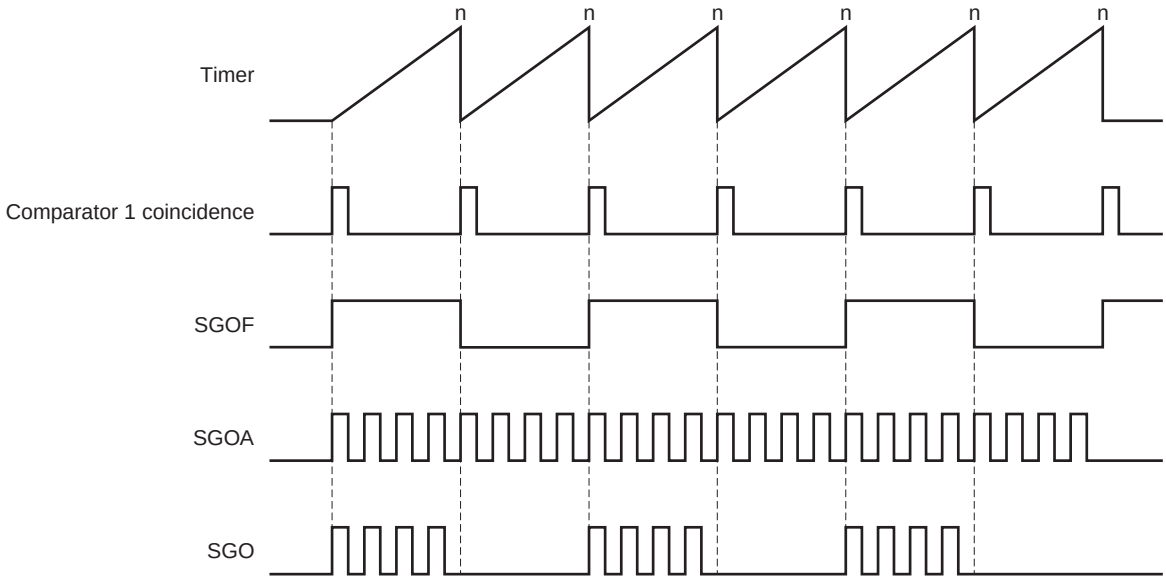
20.4.2 To output basic cycle signal SGO (with amplitude)

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 20-7: Sound Generator Output Operation Timing



[Memo]

Chapter 21 Interrupt Functions

21.1 Interrupt Function Types

The following three types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally even in a disabled state. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.

It generates a standby release signal.

The non-maskable interrupt has one source of interrupt request from the watchdog timer.

(2) 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, and PR1L).

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 21-1).

A standby release signal is generated.

The maskable interrupt has seven sources of external interrupt requests and fifteen sources of internal interrupt requests.

(3) 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.

21.2 Interrupt Sources and Configuration

There are total of 24 non-maskable, maskable, and software interrupts in the interrupt sources.

Table 21-1: Interrupt Source List

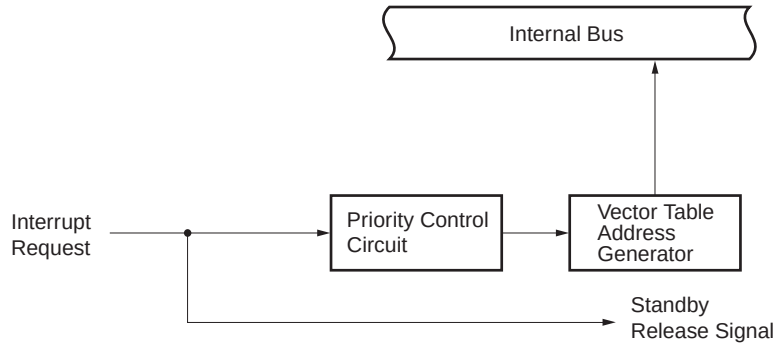
Maskability	Note 1 Interrupt Priority	Interrupt Source		Internal/ Vector	External Address	Note 2 Basic Structure Type
		Name	Trigger			
Non-maskable	–	INTWDT	Overflow of watchdog timer (When the watchdog timer NMI is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (When the interval timer mode is selected)			(B)
	1	INTAD	End of A/D converter conversion		0006H	
	2	INTOVF	Overflow of 16-bit timer 2		0008H	
	3	INTTM20	Generation of 16-bit timer 2 capture register (CR20) match signal		000AH	
	4	INTTM21	Generation of 16-bit timer 2 capture register (CR21) match signal		000CH	
	5	INTTM22	Generation of 16-bit timer 2 capture register (CR22) match signal		000EH	
	6	INTP0	Pin input edge detection	External	0010H	(C)
	7	INTP1			0012H	
	8	INTP2			0014H	
	9	INTP3			0016H	
	10	INTP4			0018H	
	11	INTCE	CAN Error	Internal	001AH	(B)
	12	INTCR	CAN Receive		001CH	
	13	INTCT0	CAN Transmitbuffer 0		001EH	
	14	INTCT1	CAN Transmitbuffer 1		0020H	
	15	INTCSI0	End of serial interface channel 0 transfer		0022H	
	16	INTCSI1	End of serial interface channel 1 transfer		0024H	
	17	INTSER	Serial interface channel 1 UART reception error generation		0026H	
	18	INTSR	End of serial interface channel 1 UART reception		0028H	
	19	INTST	End of serial interface channel 1 UART transfer		002AH	
	20	INTTM00	Generation of 16-bit timer 0 capture/compare register (CR00) match signal		002CH	
	21	INTTM01	Generation of 16-bit timer 0 capture/compare register (CR01) match signal		002EH	
	22	INTTM50	Generation of 8-bit timer/event counter 50 match signal		0030H	
	23	INTTM51	Generation of 8-bit timer/event counter 51 match signal		0032H	
	24	INTWE	EEPROM write completion interrupt		0034H	
	25	INTWTI	Reference time interval signal from watch timer		0036H	
	26	INTWT	Reference time interval signal from watch timer			
Software	–	BRK	BRK instruction execution	Internal	003EH	(D)

Notes: **1.** Default priorities are intended for two or more simultaneously generated maskable interrupt requests. 0 is the highest priority and 26 is the lowest priority.

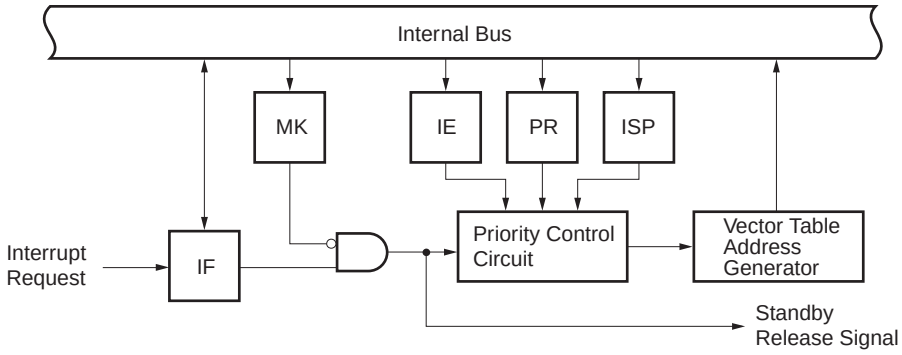
2. Basic configuration types (A) to (E) correspond to (A) to (E) of Figure 21-1.

Figure 21-1: Basic Configuration of Interrupt Function (1/2)

(A) Internal non-maskable interrupt



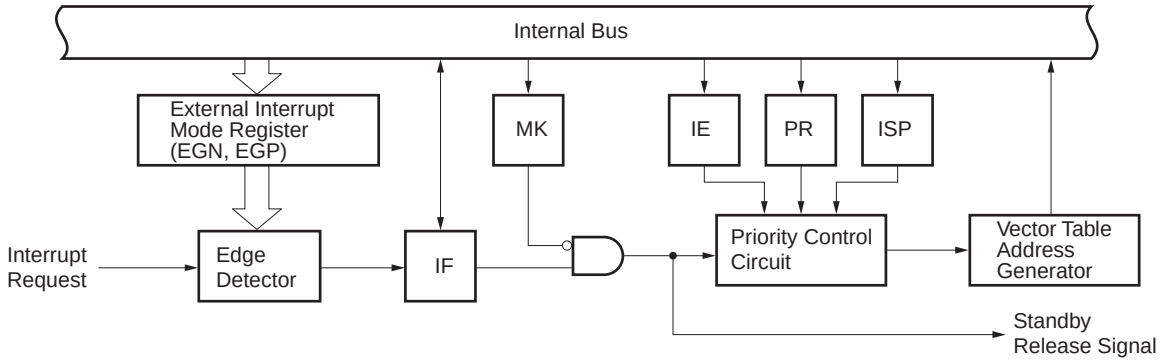
(B) Internal maskable interrupt



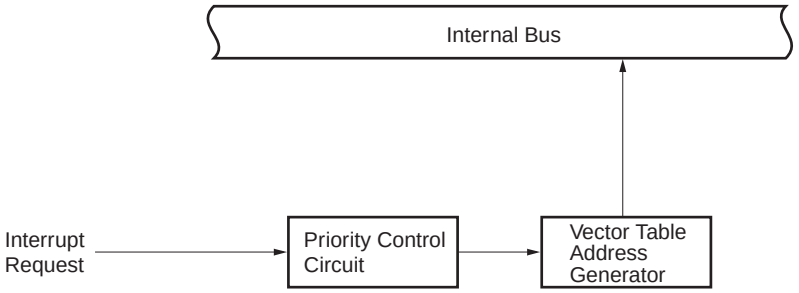
IF : Interrupt request flag
IE : Interrupt enable flag
ISP : Inservice priority flag
MK : Interrupt mask flag
PR : Priority specify flag

Figure 21-1: Basic Configuration of Interrupt Function (2/2)

(D) External maskable interrupt (except INTP0)



(E) Software interrupt



IF : Interrupt request flag
IE : Interrupt enable flag
ISP : Inservice priority flag
MK : Interrupt mask flag
PR : Priority specify flag

21.3 Interrupt Function Control Registers

The following six 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 21-2 gives a listing of interrupt request flags, interrupt mask flags, and priority specify flags corresponding to interrupt request sources.

Table 21-2: Various Flags Corresponding to Interrupt Request Sources

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag	Priority Specify Flag
INTP0	PIF0	PMK0	PPR0
INTP1	PIF1	PMK1	PPR1
INTP2	PIF2	PMK2	PPR2
INTP3	PIF3	PMK3	PPR3
INTP4	PIF4	PMK4	PPR4
INTTM00	TMIF00	TMMK00	TMPR00
INTTM01	TMIF01	TMMK01	TMPR01
INTOVF	OVFIF	OVFMK	OVFPR
INTTM20	TMIF20	TMMK20	TMPR20
INTTM21	TMIF21	TMMK21	TMPR21
INTTM22	TMIF22	TMMK22	TMPR22
INTM50	TMIF50	TMMK50	TMPR50
INTM51	TMIF51	TMMK51	TMPR51
INTWTI	WTIIF	WTIMK	WTIPR
INTWI	WTIF	WTMK	WTPR
INTWDT	TMIF4	TMMK4	TMPR4
INTAD	ADIF	ADMK	ADPR
INTCSI0	CSIIF0	CSIMK0	CSIPR0
INTCSI1	CSIIF1	CSIMK1	CSIPR1
INTSER	SERIF	SERMK	SERPR
INTSR	SRIF	SRMK	SRPR
INTST	STIF	STMK	STPR
INTCE	CEIF	CEMK	CEPR
INTCR	RRF	CRMK	CRPR
INTCT0	CTIF0	CTMK0	CTPR0
INTCT1	CTIF1	CTMK1	CTPR1
INTWE	WEIF	WEMK	WEPR

(1) Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H)

The interrupt request flag is set to 1 when the corresponding interrupt request is generated or an instruction is executed. 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 a 1-bit or 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register IF0, use a 16-bit memory manipulation instruction for the setting.

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

Figure 21-2: Interrupt Request Flag Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IF0L	PIF1	PIF0	TMIF22	TMIF21	TMIF20	OVFIF	ADIF	TMIF4	FFE0H	00H	R/W
IF0H	CSIIF0	CTIF1	CTIF0	CRIF	CEIF	PIF4	PIF3	PIF2	FFE1H	00H	R/W
IF1L	TMIF51	TMIF50	TMIF01	TMIF00	STIF	SRIF	SERIF	CSIIF1	FFE2H	00H	R/W
IF1H	0	0	0	0	0	WTIF	WTIIF	WEIF	FFE3H	00H	R/W

xxIFx	Interrupt Request Flag
0	No interrupt request signal
1	Interrupt request signal is generated; Interrupt request state

Cautions: 1. TMIF4 flag is R/W enabled only when a watchdog timer is used as an interval timer. If used in the watchdog timer mode 1, set TMIF4 flag to 0.

2. Set always 0 in IF1H bit 3 to bit 7.

(2) Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H)

The interrupt mask flag is used to enable/disable the corresponding maskable interrupt service and to set standby clear enable/disable.

MK0L, MK0H, MK1L and MK1H are set with a 1-bit or 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register MK0, use a 16-bit memory manipulation instruction for the setting.

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

Figure 21-3: Interrupt Mask Flag Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MK0L	PMK1	PMK0	TMMK22	TMMK21	TMMK20	OVFMK	ADMK	TMMK4	FFE4H	00H	R/W
MK0H	CSIMK0	CTMK1	CTMK0	CRMK	CEMK	PMK4	PMK3	PMK2	FFE5H	00H	R/W
MK1L	TMMK51	TMMK50	TMMK01	TMMK00	STMK	SRMK	SERMK	CSIMK1	FFE6H	00H	R/W
MK1H	1	1	1	1	1	WMKF	WTIMK	WEMK	FFE7H	00H	R/W

xxMKx	Interrupt Servicing Control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions: 1. If TMMK4 flag is read when a watchdog timer is used as a non-maskable interrupt, MK0 value becomes undefined.

2. Set always 1 in MK1H bit 3 to bit 7.

(3) Priority specify flag registers (PR0L, PR0H, PR1L, PR1H)

The priority specify flag is used to set the corresponding maskable interrupt priority orders.

PR0L, PR0H, PR1L and PR1H are set with a 1-bit or 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register PR0, use a 16-bit memory manipulation instruction for the setting.

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

Figure 21-4: Priority Specify Flag Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PR0L	PPR1	PPR0	TMPR22	TMPR21	TMPR20	OVFPR	ADPR	TMPR4	FFE8H	00H	R/W
PR0H	CSIPR0	CTPR1	CTPR0	CRPR	CEPR	PPR4	PPR3	PPR2	FFE9H	00H	R/W
PR1L	TMPR51	TMPR50	TMPR01	TMPR00	STPR	SRPR	SERPR	CSIPR1	FFEAH	00H	R/W
PR1H	1	1	1	1	1	WPRF	WTIPR	WEPR	FFEBH	00H	R/W

xxPRx	Priority Level Selection
0	High priority level
1	Low priority level

Cautions: 1. When a watchdog timer is used as a non-maskable interrupt, set 1 in TMPR4 flag.
2. Set always 1 in PR1H bit 3 to bit 7.

(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 P04.

EGP and EGN can be read or written to with a 1-bit or 8-bit memory manipulation instruction.

These registers are set to 00H when the $\overline{\text{RESET}}$ signal is output.

Figure 21-5: 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	On Reset	R/W
EGP	0	0	0	EGP4	EGP3	EGP2	EGP1	EGP0	FF48H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	On Reset	R/W
EGN	0	0	0	EGN4	EGN3	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	Valid edge of INTPn pin (n = 0 – 4)
0	0	Interrupt disable
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

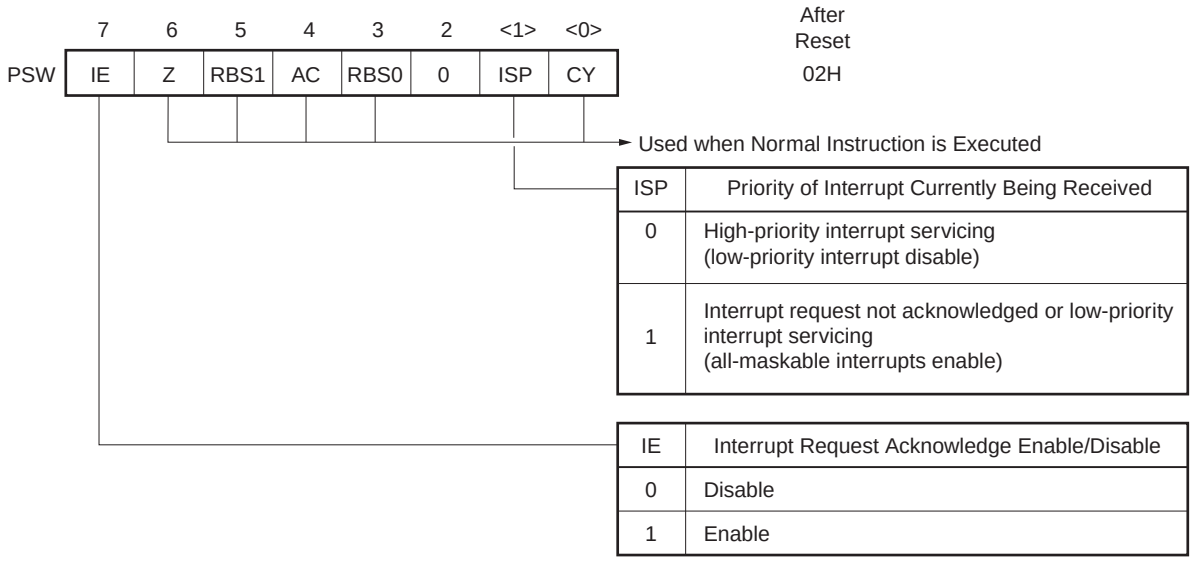
(5) Program status word (PSW)

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 interrupt 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 into a 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 into the stack with the PUSH PSW instruction. It is reset from the stack with the RETI, RETB, and POP PSW instructions.

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

Figure 21-6: Program Status Word Format



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

21.4 Interrupt Servicing Operations

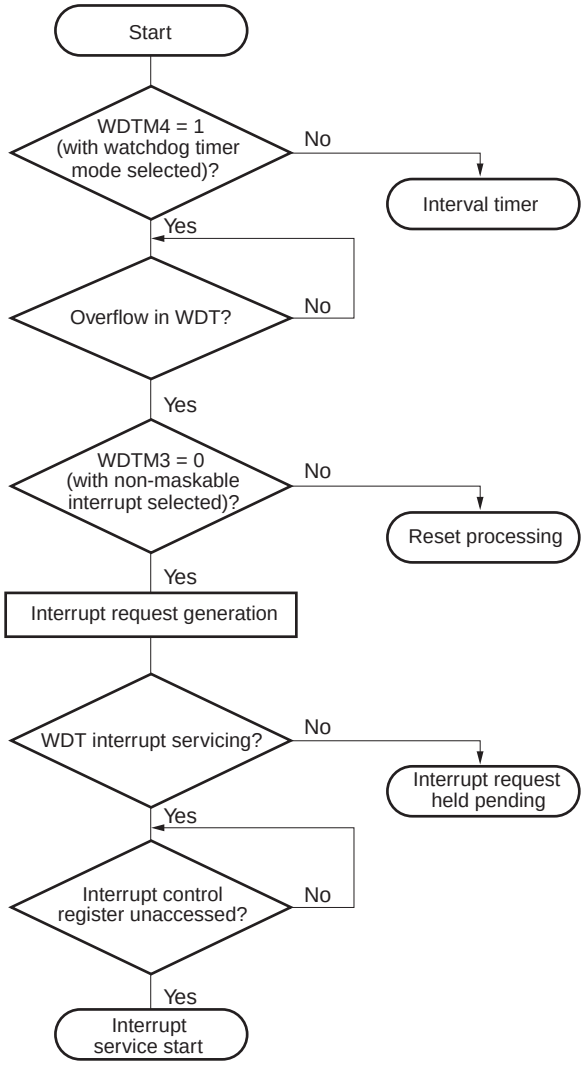
21.4.1 Non-maskable interrupt request acknowledge operation

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, the acknowledged interrupt is saved in the stacks, PSW and PC, in that order, the IE and ISP flags are reset to 0, and the vector table contents are loaded into PC and branched.

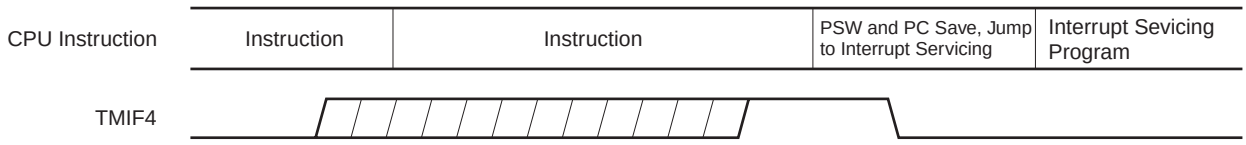
A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after the current execution of the non-maskable interrupt servicing program is terminated (following RETI instruction execution) and one main routine instruction is executed. If a new non-maskable interrupt request is generated twice or more during 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 21-7: Flowchart from Non-Maskable Interrupt Generation to Acknowledge



WDTM: Watchdog timer mode register
WDT: Watchdog timer

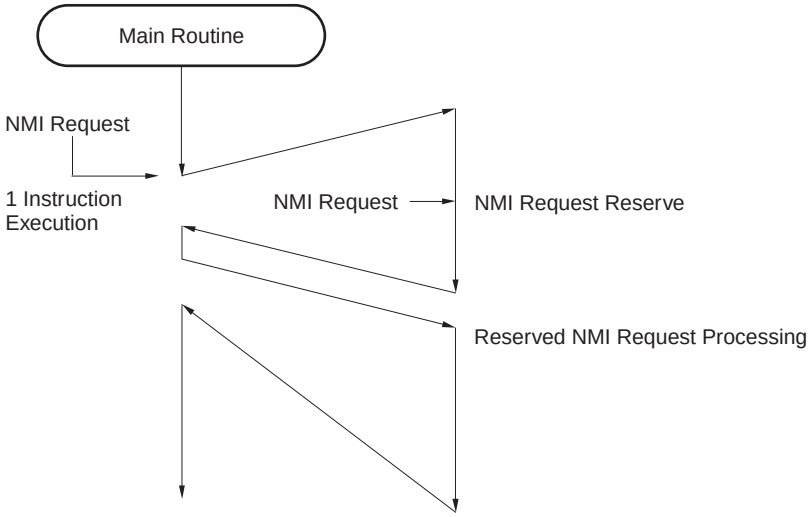
Figure 21-8: Non-Maskable Interrupt Request Acknowledge Timing



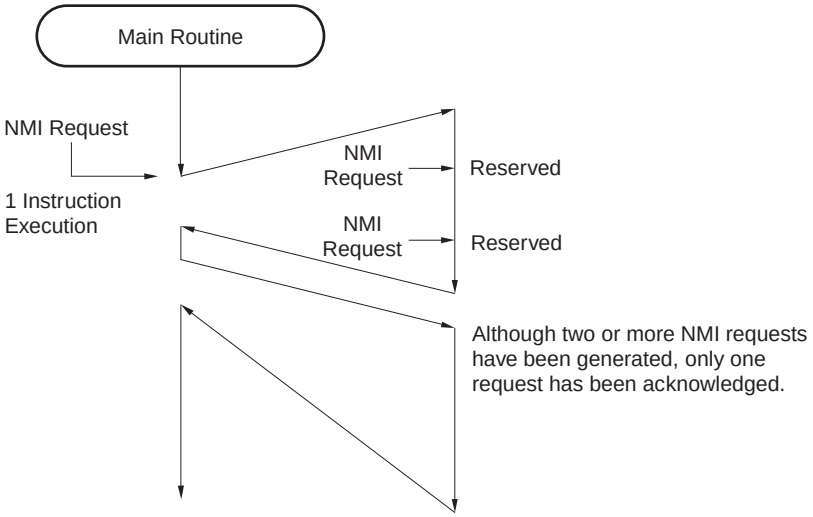
TMIF4: Watchdog timer interrupt request flag

Figure 21-9: 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



21.4.2 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 maskable interrupt request generation to interrupt servicing are as follows.

Table 21-3: 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: $\frac{1}{f_{CPU}}$ (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.

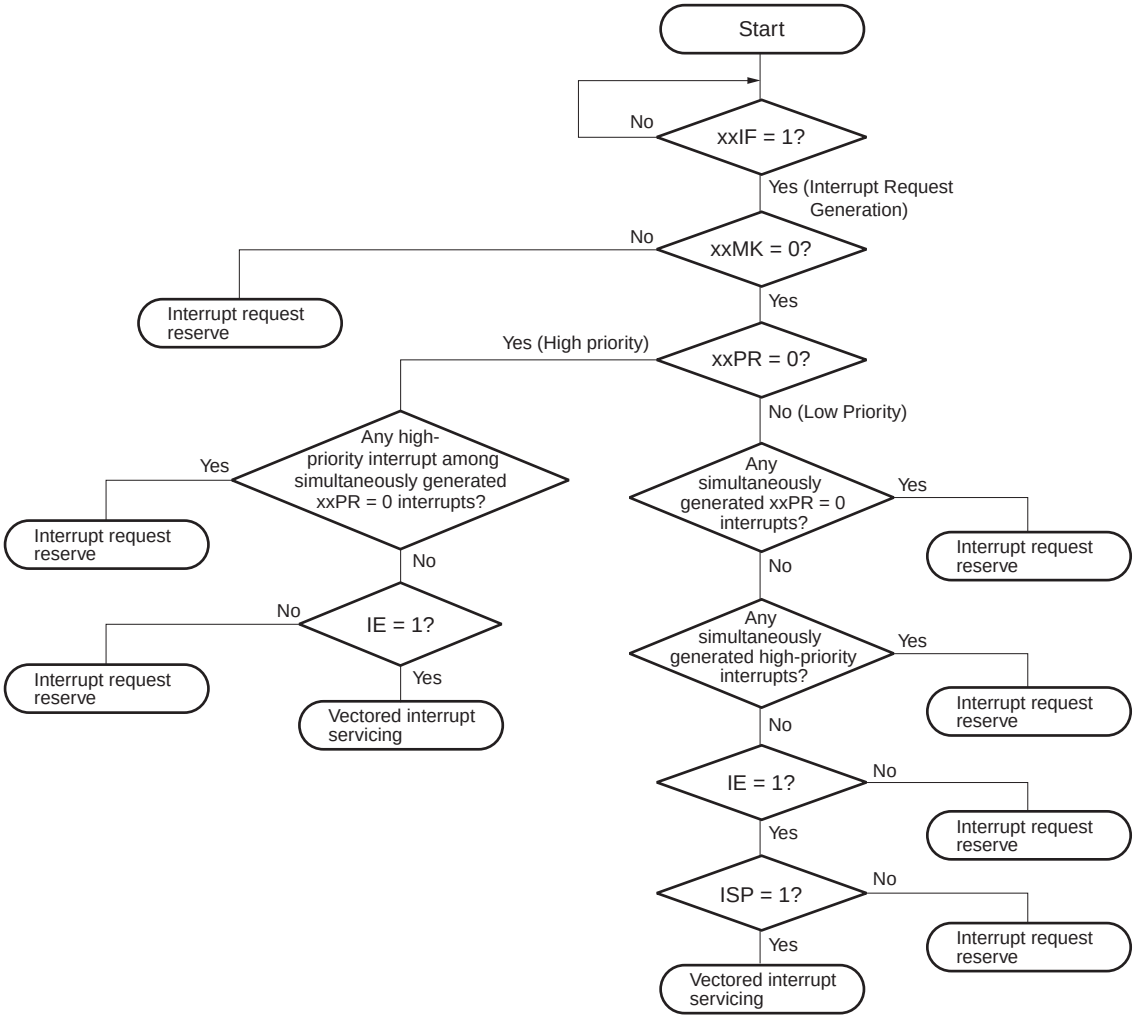
Any reserved interrupt requests are acknowledged when they become acknowledgeable.

Figure 21-10 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 to stacks, in this order. 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 branched.

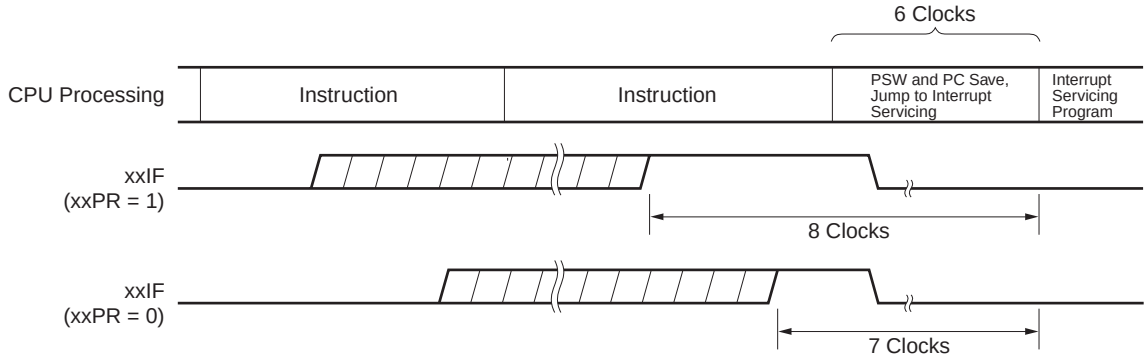
Return from the interrupt is possible with the RETI instruction.

Figure 21-10: Interrupt Request Acknowledge Processing Algorithm



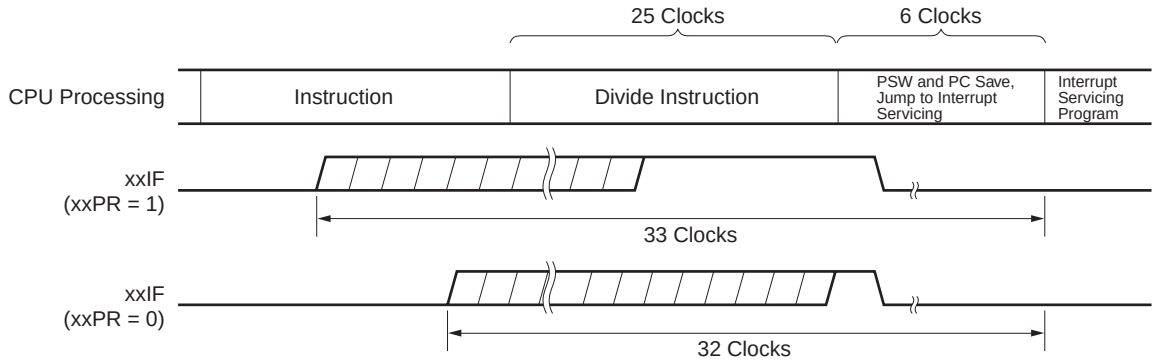
xxIF : Interrupt request flag
xxMK : Interrupt mask flag
xxPR : Priority specify flag
IE : Flag to control maskable interrupt request acknowledge
ISP : Flag to indicate the priority of interrupt being serviced (0 = an interrupt with higher priority is being serviced, 1 = interrupt request is not acknowledged or an interrupt with lower priority is being serviced)

Figure 21-11: Interrupt Request Acknowledge Timing (Minimum Time)



Remark: 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

Figure 21-12: Interrupt Request Acknowledge Timing (Maximum Time)



Remark: 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

21.4.3 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 branched.

Return from the software interrupt is possible with the RETB instruction.

Caution: Do not use the RETI instruction for returning from the software interrupt.

21.4.4 Multiple interrupt servicing

A multiple interrupt consists in acknowledging another interrupt during the execution of the interrupt.

A multiple interrupt 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 a multiple interrupt, 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.

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 21-4 shows an interrupt request enabled for multiple interrupt during interrupt servicing, and Figure 21-13 shows multiple interrupt examples.

Table 21-4: Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing

Multiple 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 enable

2. D: Multiple interrupt disable

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

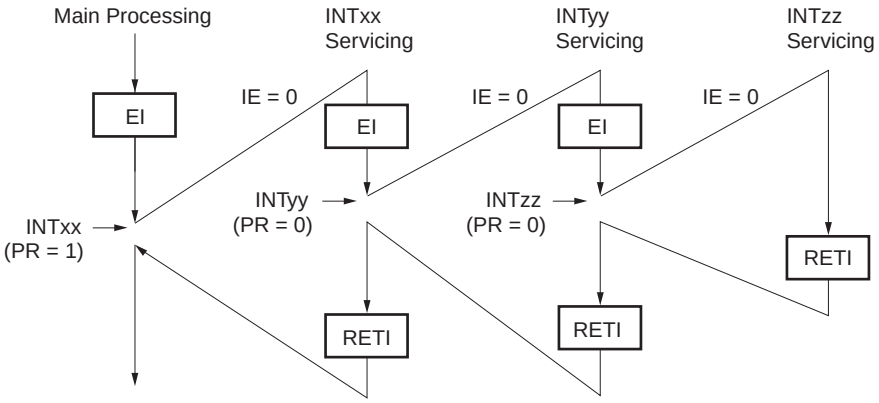
4. xxPR is a flag contained in PR0L, PR0H, and PRIL

xxPR = 0: Higher priority level

xxPR = 1: Lower priority level

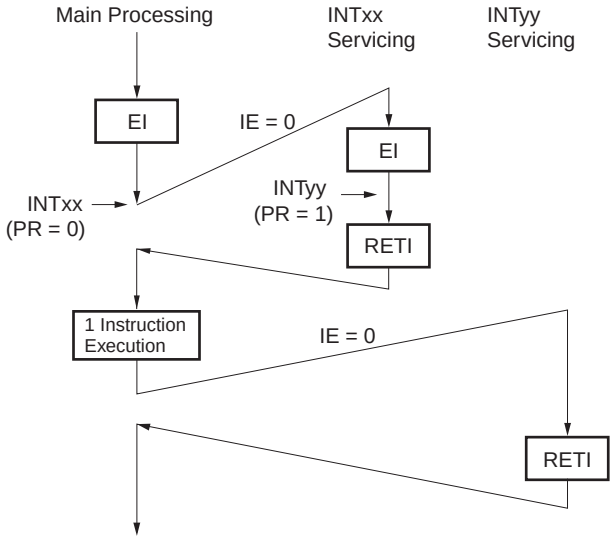
Figure 21-13: Multiple Interrupt Example (1/2)

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.

Example 2. Multiple interrupt is not generated by priority control

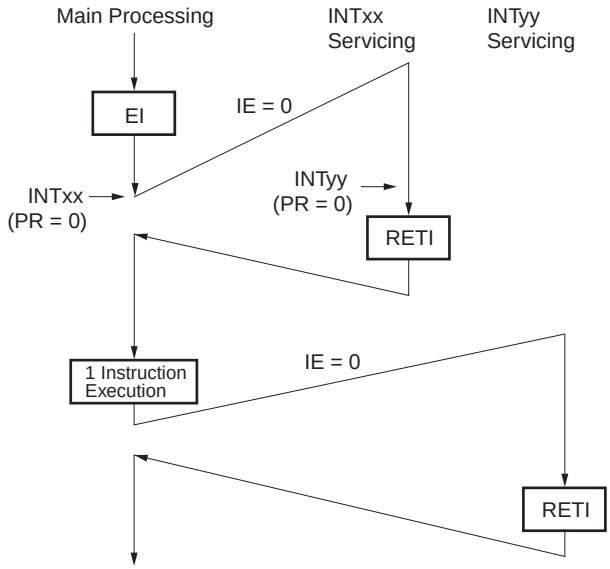


The interrupt request INTyy generated during interrupt INTxx servicing is not acknowledged because the interrupt priority is lower than that of INTxx, and a multiple interrupt is not generated. INTyy request is retained and acknowledged after execution of 1 instruction execution of the main processing.

PR = 0: Higher priority level
PR = 1: Lower priority level
IE = 0 : Interrupt request acknowledge disable

Figure 21-13: Multiple Interrupt Example (2/2)

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.

PR = 0: Higher priority level
IE = 0 : Interrupt request acknowledge disable

21.4.5 Interrupt request reserve

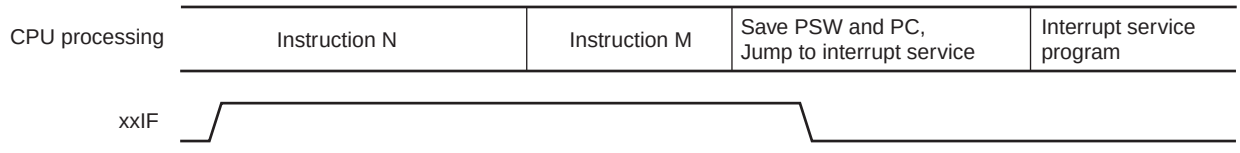
Some instructions may reserve the acknowledge of an instruction request until the completion of the execution of the next instruction even if the interupt request is generated during the execution. The following shows such instructions (interrupt request reserve instruction).

- 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
- XOR1 CY, PSW.bit
- SET1/CLR1 PSW.bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW.bit, $addr16
- BF PSW.bit, $addr16
- BTCLRPSW.bit, $addr16
- EI
- DI
- Manipulate instructions for IF0L, IF0H, IF1L, MK0L, MK0H, MK1L, PR0L, PR0H, PR1L, INTM0, INTM1 registers

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 21-14 shows the interrupt request hold timing.

Figure 21-14: 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 22 External Device Expansion

22.1 External Device Expansion Functions

The external device expansion functions connect external devices to areas other than the internal ROM, RAM, and SFR. Connection of external devices uses ports 4 to 6. Ports 4 to 6 control address/data, read/write strobe, wait, address strobe etc.

Table 22-1: Pin Functions in External Memory Expansion Mode

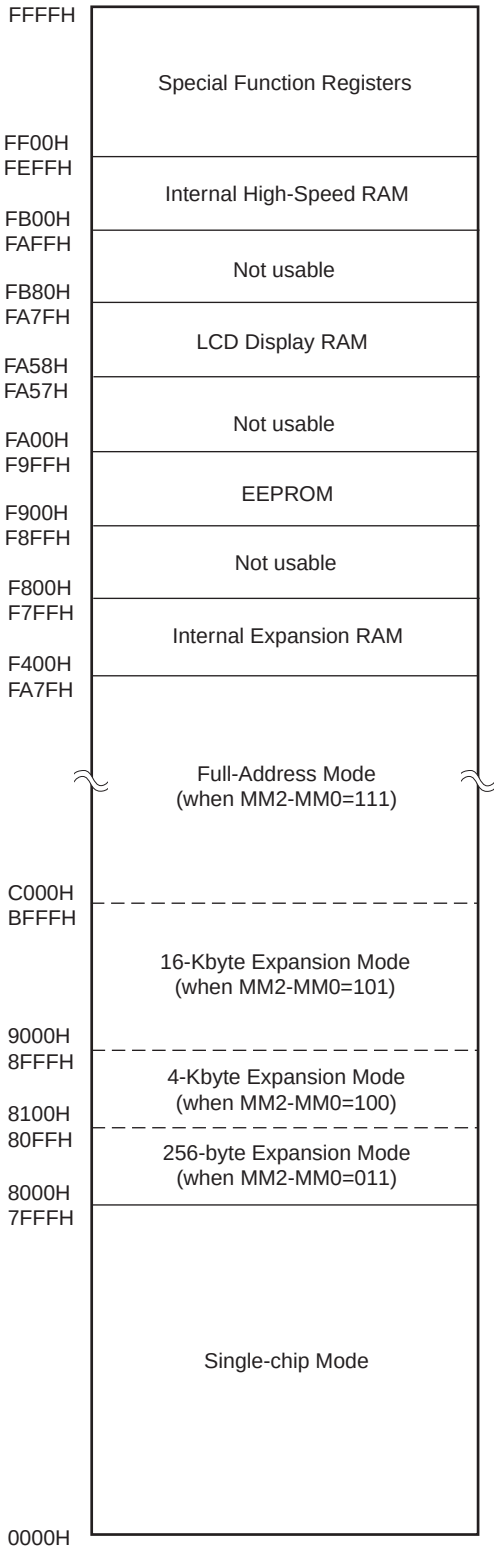
Pin function at external device connection		Alternate function
Name	Function	
AD0 to AD7	Multiplexed address/data bus	P40 to P47
A8 to A15	Address bus	P50 to P57
$\overline{RD}$	Read strobe signal	P64
$\overline{WR}$	Write strobe signal	P65
ASTB	Address strobe signal	P67

Table 22-2: State of Port 4 to Port 6 Pins in External Memory Expansion Mode

Ports and bits / Modes	Port 4	Port 5	Port 6
	0-7	0 1 2 3 4 5 6 7	45
Single-chip mode	Port	Port	Port
256-byte expansion mode	Address/data	Port	$\overline{RD}$, $\overline{WR}$, ASTB
4K-byte expansion mode	Address/data	Address / Port	$\overline{RD}$, $\overline{WR}$, ASTB
16K-byte expansion mode	Address/data	Address / Port	$\overline{RD}$, $\overline{WR}$, ASTB
Full address mode	Address/data	Address	$\overline{RD}$, $\overline{WR}$, ASTB

Figure 22-1: Memory Map when Using External Device Expansion Function (1/2)

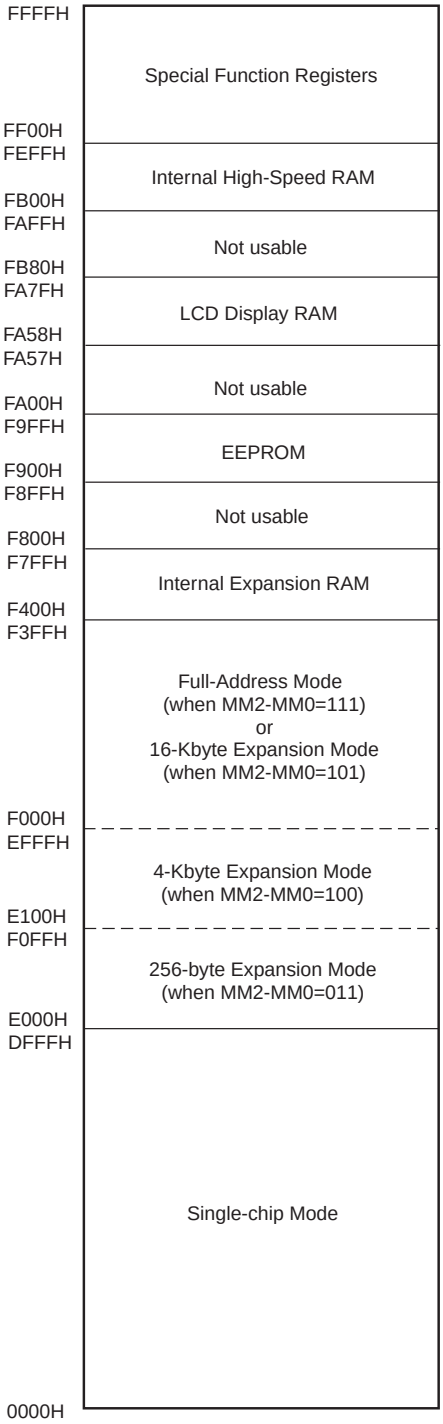
(a) μPD780948/78F0948, μPD780949/78F0949 Memory map when internal ROM size is 32 Kbytes



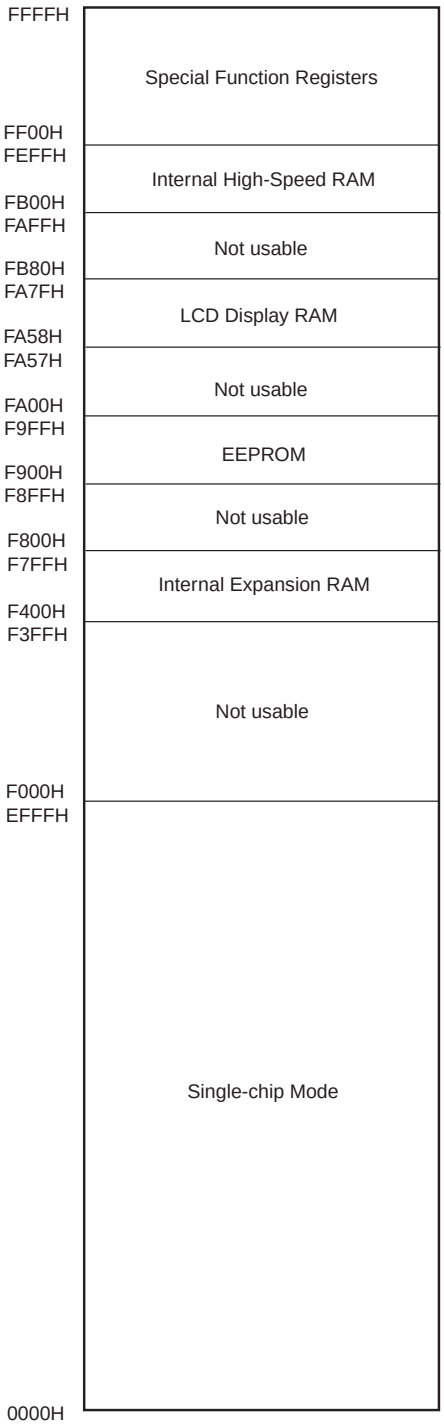
Note: The EEPROM is only available in the μPD78(F)0949.

Figure 22-1: Memory Map when Using External Device Expansion Function (2/2)

(b) μPD780948/78F0948, μPD780949/78F0949 Memory map when internal ROM (PROM) size is 56 Kbytes



(c) μPD780948/78F0948, μPD780949/78F0949 Memory map when internal ROM (PROM) size is 60 Kbytes



Caution: When the internal ROM (PROM) size is 60 Kbytes, the area from F000H to F3FFH cannot be used. F000H to F3FFH can be used as external memory by setting the internal ROM (PROM) size to less than 56 Kbytes by the memory size switching register (IMS).

Note: The EEPROM is only available in the μPD78(F)0949.

22.2 External Device Expansion Function Control Register

The external device expansion function is controlled by the memory expansion mode register (MEM), the memory expansion wait register (MM), and memory size switching register (IMS).

(1) Memory expansion mode register (MEM)

MM sets the wait count and external expansion area, and also sets the input/output of port 4.
MM is set with an 1-bit memory or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to 10H.

Figure 22-2: Memory Expansion Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When Reset	R/W
MEM	0	0	0	0	0	MM2	MM1	MM0	FF47H	10H	R/W

MM2	MM1	MM0	Single-chip/ Memory Expansion Mode Selection		P40-P47, P50-P57, P64, P65, P67 Pin state				
					P40-P47	P50-P53	P54, P55	P56, P57	P64, P65,P67
0	0	0	Single-chip mode		Port mode				
0	0	1							
0	1	1	Memory expansion mode	256-byte mode	AD0-AD7	Port mode			P64=$\overline{\text{RD}}$ P65=$\overline{\text{WR}}$ P67=ASTB
1	0	0		4K-byte mode		A8-A11	Port mode		
1	0	1		16K-byte mode			A12, A13	Port mode	
1	1	1		Full address mode[Note]				A14, A15	
Other than above			Setting prohibited						

Note: The full address mode allows external expansion to the entire 64-Kbyte address space except for the internal ROM, RAM, and SFR areas and the reserved areas.

(2) Memory expansion wait register (MM)

MM sets the wait count.

MM is set with an 1-bit memory or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to 10H.

Figure 22-3: Memory Expansion Wait Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When Reset	R/W
MM	0	0	0	PW0	0	0	0	0	FFF8H	10H	R/W

PW0	Wait Control
0	No wait
1	Wait (one wait state insertion)

(3) Memory size switching register (IMS)

This register specifies the internal memory size. In principle, use IMS in a default status. However, when using the external device expansion function with the μPD780949, set IMS so that the internal ROM capacity is 56 Kbytes or lower.

IMS is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to the value indicated in Table 22-3.

Figure 22-4: Memory Size Switching Register 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	Note	R/W

ROM3	ROM2	ROM1	ROM0	Internal ROM size selection
1	0	0	0	32 Kbytes
1	1	1	0	56 Kbytes
1	1	1	1	60 Kbytes
Other than above				Setting prohibited

RAM2	RAM1	RAM0	Internal high-speed RAM size selection
1	1	0	1024 bytes
Other than above			Setting prohibited

Note: The values after reset depend on the product (See Table 22-3).

Table 22-3: Values when the Memory Size Switching Register is Reset

Part Number	Reset Value
μPD780948, μPD78F0948	CFH
μPD780949, μPD78F0949	CFH

22.3 External Device Expansion Function Timing

Timing control signal output pins in the external memory expansion mode are as follows.

(1) $\overline{RD}$ pin (Alternate function: P64)

Read strobe signal output pin. The read strobe signal is output in data accesses and instruction fetches from external memory.

During internal memory access, the read strobe signal is not output (maintains high level).

(2) $\overline{WR}$ pin (Alternate function: P65)

Write strobe signal output pin. The write strobe signal is output in data access to external memory.

During internal memory access, the write strobe signal is not output (maintains high level).

(3) ASTB pin (Alternate function: P67)

Address strobe signal output pin. Timing signal is output without regard to the data accesses and instruction fetches from external memory. The ASTB signal is also output when the internal memory is accessed.

(4) AD0 to AD7, A8 to A15 pins (Alternate function : P40 to P47, P50 to P57)

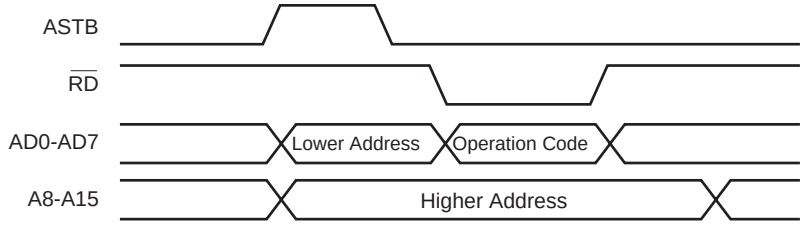
Address/data signal output pin. Valid signal is output or input during data accesses and instruction fetches from external memory.

These signals change when the internal memory is accessed (output values are undefined).

Timing charts are shown in Figure 22-5 to 22-8.

Figure 22-5: Instruction Fetch from External Memory

(a) No wait (PW0 = 0) setting



(b) Wait (PW0 = 1) setting

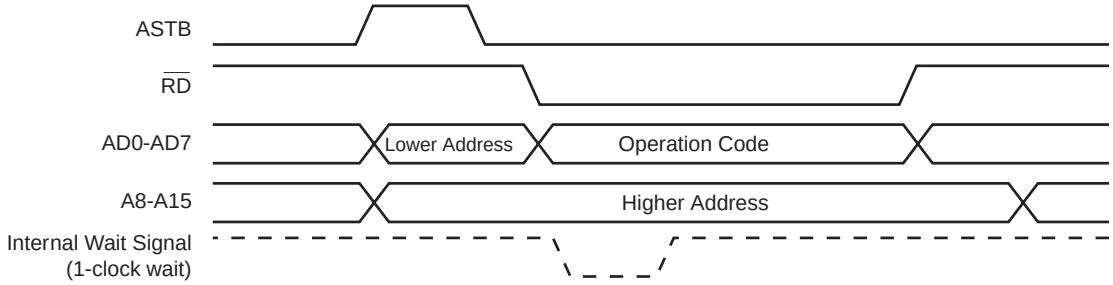
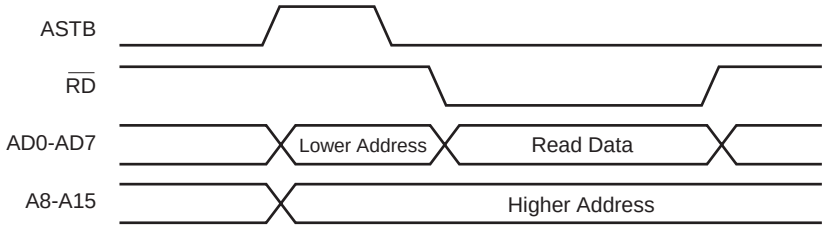


Figure 22-6: External Memory Read Timing

(a) No wait (PW0 = 0) setting



(b) Wait (PW0 = 1) setting

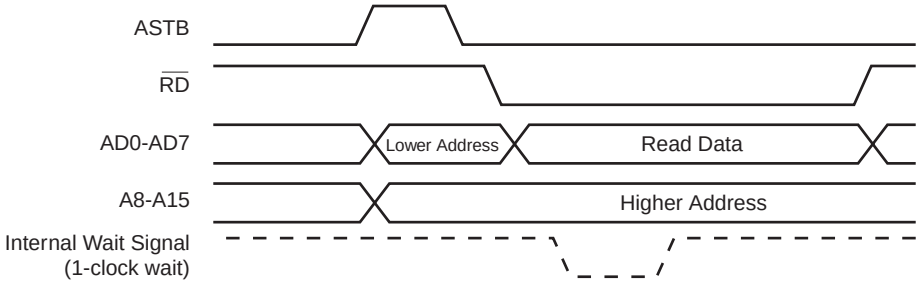
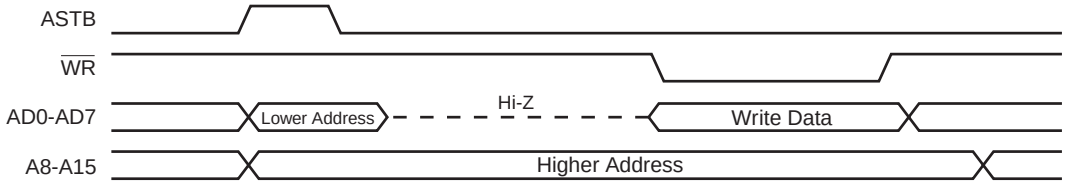


Figure 22-7: External Memory Write Timing

(a) No wait (PW0 = 0) setting



(b) Wait (PW0 = 1) setting

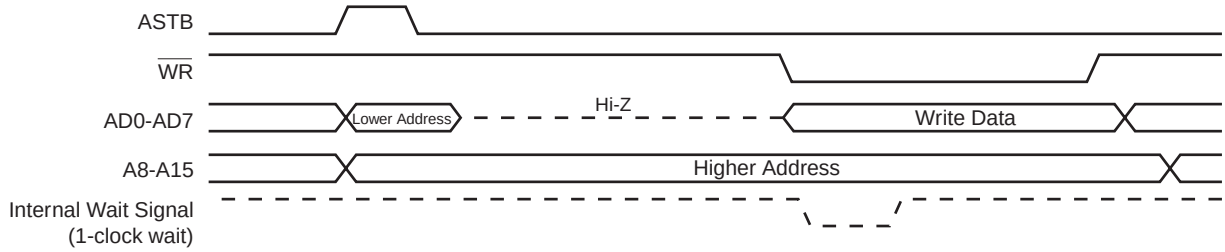
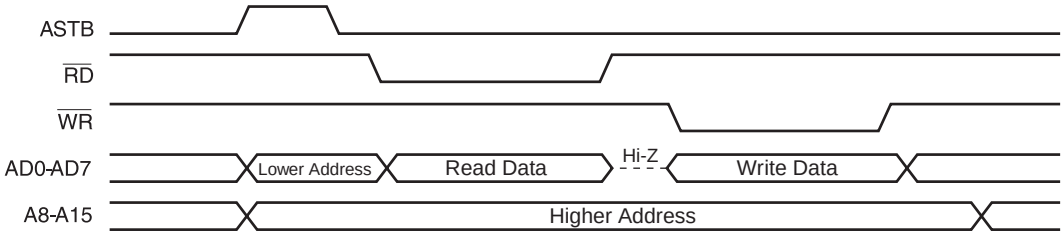
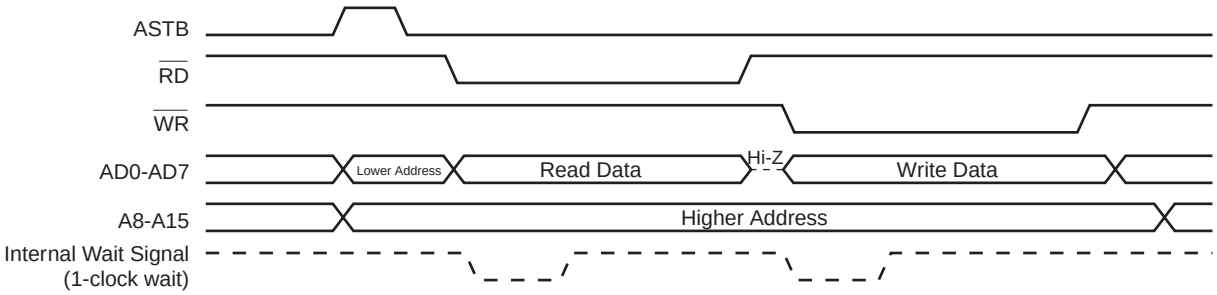


Figure 22-8: External Memory Read Modify Write Timing

(a) No wait (PW0 = 0) setting



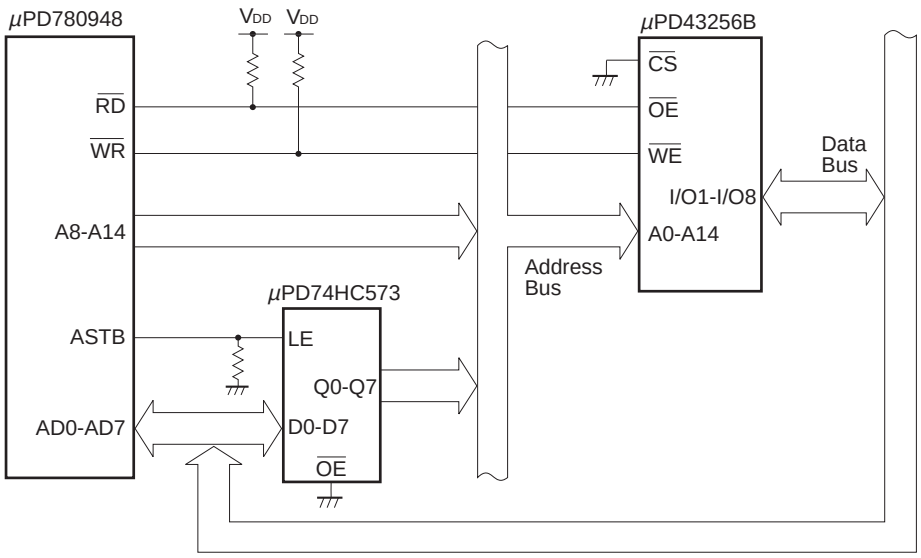
(b) Wait (PW0 = 1) setting



22.4 Example of Connection with Memory

This section provides μPD780948 and external memory connection examples in Figure 22-9. SRAMs are used as the external memory in these diagrams. In addition, the external device expansion function is used in the full-address mode, and the address from 0000H to 7FFFH (32 Kbytes) are allocated for internal ROM, and the addresses after 8000H for SRAM.

Figure 22-9: Connection Example of μPD780948 and Memory



[Memo]

Chapter 23 Standby Function

23.1 Standby Function and Configuration

23.1.1 Standby function

The standby function is designed to decrease power consumption of the system. The following two modes are available.

(1) HALT mode

HALT instruction execution sets the HALT mode. The HALT mode is intended to stop the CPU operation clock. System clock oscillator continues oscillation. In this mode, current consumption cannot be decreased as in the STOP mode. The HALT mode is valid to restart immediately upon interrupt request and to carry out intermittent operations such as watch applications.

(2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the main system clock oscillator stops and the whole system stops. CPU current consumption can be considerably decreased.

Data memory low-voltage hold (down to V_{DD} = 4.0 V) is possible. Thus, the STOP mode is effective to hold data memory contents with ultra-low current consumption. Because this mode can be cleared upon interrupt request, it enables intermittent operations to be carried out.

However, because a wait time is necessary to secure an oscillation stabilization time after the STOP mode is cleared, select the HALT mode if it is necessary to start processing immediately upon interrupt request.

In any mode, all the contents of the register, flag, and data memory just before standby mode setting are held. The input/output port output latch and output buffer statuses are also held.

Cautions: 1. The STOP mode can be used only when the system operates with the main system clock (subsystem clock oscillation cannot be stopped). The HALT mode can be used with either the main system clock or the subsystem clock.

2. When proceeding to the STOP mode, be sure to stop the peripheral hardware operation and execute the STOP instruction.

3. The following sequence is recommended for power consumption reduction of the A/D converter when the standby function is used: first clear bit 7 (CS) to 0 to stop the A/D conversion operation, and then execute the HALT or STOP instruction.

23.1.2 Standby function control register

A wait time after the STOP mode is cleared upon interrupt request till the oscillation stabilizes is controlled with the oscillation stabilization time select register (OSTS).

OSTS is set with an 8-bit memory manipulation instruction.

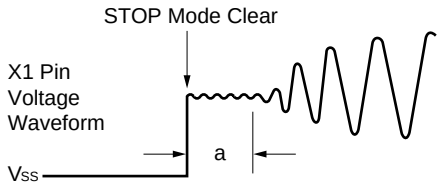
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, it takes 2^{17}/fx until the STOP mode is cleared by $\overline{\text{RESET}}$ input.

Figure 23-1: Oscillation Stabilization Time Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Selection of Oscillation Stabilization Time when STOP Mode is Released
0	0	0	$2^{12}/f_x$ (512 μs)
0	0	1	$2^{14}/f_x$ (2 ms)
0	1	0	$2^{15}/f_x$ (4.1 ms)
0	1	1	$2^{16}/f_x$ (8.19 ms)
1	0	0	$2^{17}/f_x$ (16.38 ms)
Other than above			Setting prohibited

Caution: The wait time after STOP mode clear does not include the time (see "a" in the illustration below) from STOP mode clear to clock oscillation start, regardless of clearance by $\overline{\text{RESET}}$ input or by interrupt generation.



Remarks: **1.** fx: Main system clock oscillation frequency
2. Values in parentheses apply to operating at fx = 8.00 MHz

23.2 Standby Function Operations

23.2.1 HALT mode

(1) HALT mode set and operating status

The HALT mode is set by executing the HALT instruction. It can be set with the main system clock or the subsystem clock. The operating status in the HALT mode is described below.

Table 23-1: HALT Mode Operating Status

HALT mode setting / Item		HALT execution during main system clock operation	HALT execution during subsystem clock operation (Main system clock stops)
Clock generator		Both main and subsystem clocks can be oscillated / Clock supply to the CPU stops	
CPU		Operation stops	
Port (output latch)		Status before HALT mode setting is held	
16-bit timer /event counter (TM0)		Operable	Operable when TI is selected as count clock
16-bit timer (TM2)		Operable	Operation stops
8-bit timer event counter (TM50/TM51)		Operable	Operable when TI is selected as count clock
Watch timer		Operable	Operable when fxt is selected as count clock
Watchdog timer		Operable	Operation stops
A/D converter		Operation stops	
Serial I/F		Operable	Operable at external SCK
CAN		Operation stops	
Sound generator		Operable	Operation stops
External interrupt (INTP0 to INTP4)		Operable	
LCD		Operable	Operation stops
Bus lines in external expansion			
	AD0 to AD7	High impedance	
	A8 to A15	Status before HALT mode is held	
	PSTB	Low level	
	$\overline{WR}$, $\overline{RD}$	High level	

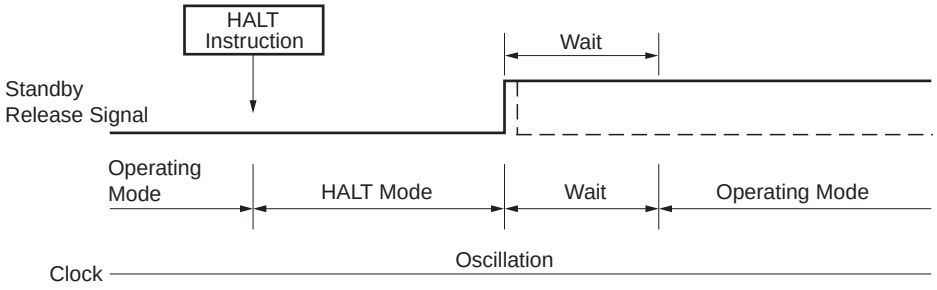
(2) HALT mode clear

The HALT mode can be cleared with the following four types of sources.

(a) Clear upon unmasked interrupt request

An unmasked interrupt request is used to clear the HALT mode. If interrupt acknowledge is enabled, vectored interrupt service is carried out. If disabled, the next address instruction is executed.

Figure 23-2: HALT Mode Clear upon Interrupt Generation



Remarks: **1.** The broken line indicates the case when the interrupt request which has cleared the standby status is acknowledged.

2. Wait time will be as follows:

- When vectored interrupt service is carried out: 8 to 9 clocks
- When vectored interrupt service is not carried out: 2 to 3 clocks

(b) Clear upon non-maskable interrupt request

The HALT mode is cleared and vectored interrupt service is carried out whether interrupt acknowledge is enabled or disabled.

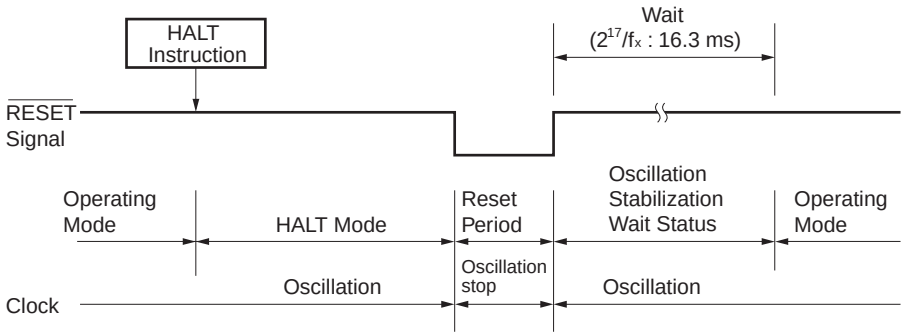
(c) Clear upon unmasked test input

The HALT mode is cleared by unmasked test input and the next address instruction of the HALT instruction is executed.

(d) Clear upon $\overline{\text{RESET}}$ input

As is the case with normal reset operation, a program is executed after branch to the reset vector address.

Figure 23-3: HALT Mode Release by $\overline{\text{RESET}}$ Input



Remarks: **1.** fx: Main system clock oscillation frequency

2. Values in parentheses apply to operation at fx = 8.0 MHz

Table 23-2: Operation after HALT Mode Release

Release Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	x	Next address instruction execution
	0	0	1	x	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	x	0	
	0	1	1	1	Interrupt service execution
	1	x	x	x	HALT mode hold
Non-maskable interrupt request	-	-	x	x	Interrupt service execution
$\overline{\text{RESET}}$ input	-	-	x	x	Reset processing

x: Don't care.

23.2.2 STOP mode

(1) STOP mode set and operating status

The STOP mode is set by executing the STOP instruction. It can be set only with the main system clock.

Cautions: 1. When the STOP mode is set, the X2 pin is internally connected to V_{DD} via a pull-up resistor to minimize leakage current at the crystal oscillator. Thus, do not use the STOP mode in a system where an external clock is used for the main system clock.

2. 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. After the wait set using the oscillation stabilization time select register (OSTS), the operating mode is set.

The operating status in the STOP mode is described below.

Table 23-3: STOP Mode Operating Status

STOP mode setting / Item		With subsystem clock	Without subsystem clock
Clock generator		Only main system clock stops oscillation	
CPU		Operation stops	
Port (output latch)		Status before STOP mode setting is held	
16-bit timer /event counter (TM0)		Operable when TI is selected as count clock	
16-bit timer (TM2)		Operation stops	
8-bit timer event counter 5 and 6		Operable when TI50 or TI51 are selected as count clock	
Watch timer		Operable when fxt is selected as count clock	Operation stops
Watchdog timer		Operation stops	
A/D converter		Operation stops	
Serial I/F		Operable at external SCK	
CAN		Operation stops	
Sound generator		Operation stops	
External interrupt (INTP0 to INTP4)		Operable	
LCD		Operation stops	
Bus lines in external expansion			
	AD0 to AD7	High impedance	
	A8 to A15	Status before STOP mode is held	
	ASTB	Low level	
	$\overline{WR}$, $\overline{RD}$	High level	

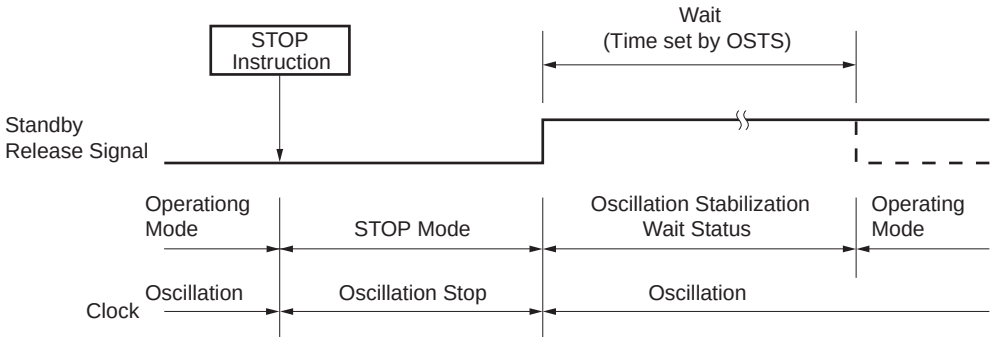
(2) STOP mode release

The STOP mode can be cleared with the following three types of sources.

(a) Release by unmasked interrupt request

An unmasked interrupt request is used to release the STOP mode. If interrupt acknowledge is enabled after the lapse of oscillation stabilization time, vectored interrupt service is carried out. If interrupt acknowledge is disabled, the next address instruction is executed.

Figure 23-4: STOP Mode Release by Interrupt Generation



Remark: The broken line indicates the case when the interrupt request which has cleared the standby status is acknowledged.

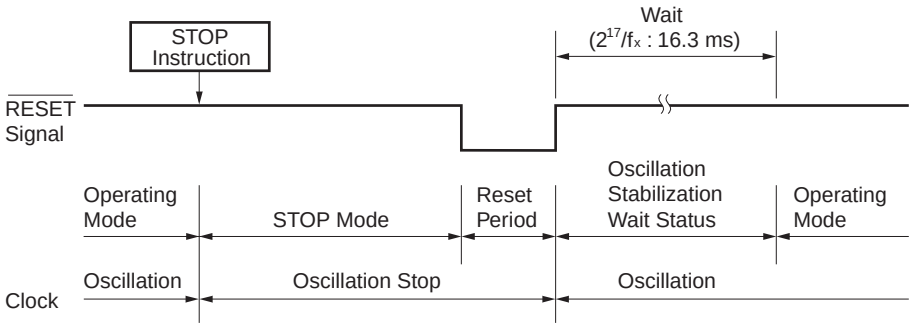
(b) Release by unmasked test input

The STOP mode is cleared by unmasked test input. After the lapse of oscillation stabilization time, the instruction at the next address of the STOP instruction is executed.

(c) Release by $\overline{\text{RESET}}$ input

The STOP mode is cleared and after the lapse of oscillation stabilization time, reset operation is carried out.

Figure 23-5: Release by STOP Mode $\overline{\text{RESET}}$ Input



Remarks **1.** fx: Main system clock oscillation frequency

2. Values in parentheses apply to operation at fx = 5.0 MHz

Table 23-4: Operation after STOP Mode Release

Release Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	x	Next address instruction execution
	0	0	1	x	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	x	0	
	0	1	1	1	Interrupt service execution
	1	x	x	x	STOP mode hold
Non-maskable interrupt request	-	-	x	x	Interrupt service execution
$\overline{\text{RESET}}$ input	-	-	x	x	Reset processing

x: Don't care.

[Memo]

Chapter 24 Reset Function

24.1 Reset Function

The following two operations are available to generate the reset signal.

(1) External reset input with $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer overrun time detection

External reset and internal reset have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H by $\overline{\text{RESET}}$ input.

When a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, a reset is applied and each hardware is set to the status as shown in Table 24-1. Each pin has high impedance during reset input or during oscillation stabilization time just after reset clear.

When a high level is input to the $\overline{\text{RESET}}$ input, the reset is cleared and program execution starts after the lapse of oscillation stabilization time (2^{17}/fx). The reset applied by watchdog timer overflow is automatically cleared after a reset and program execution starts after the lapse of oscillation stabilization time (2^{17}/fx) (see Figure 24-2 to 24-4).

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, main system clock oscillation remains stopped but subsystem clock oscillation continues.
3. When the STOP mode is cleared by reset, the STOP mode contents are held during reset input. However, the port pin becomes high-impedance.

Figure 24-1: Block Diagram of Reset Function

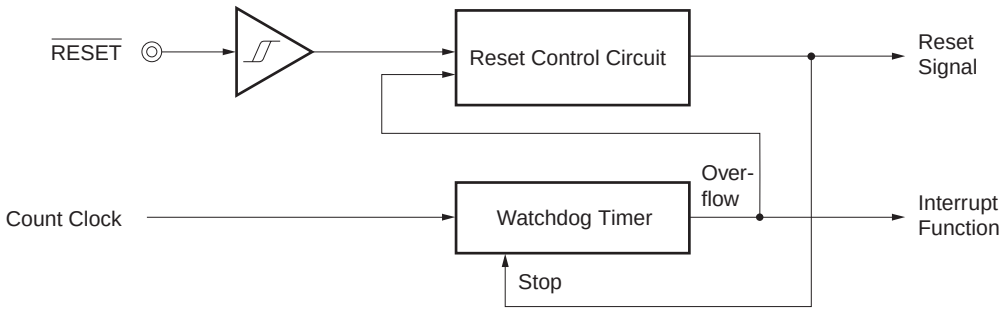


Figure 24-2: Timing of Reset Input by $\overline{\text{RESET}}$ Input

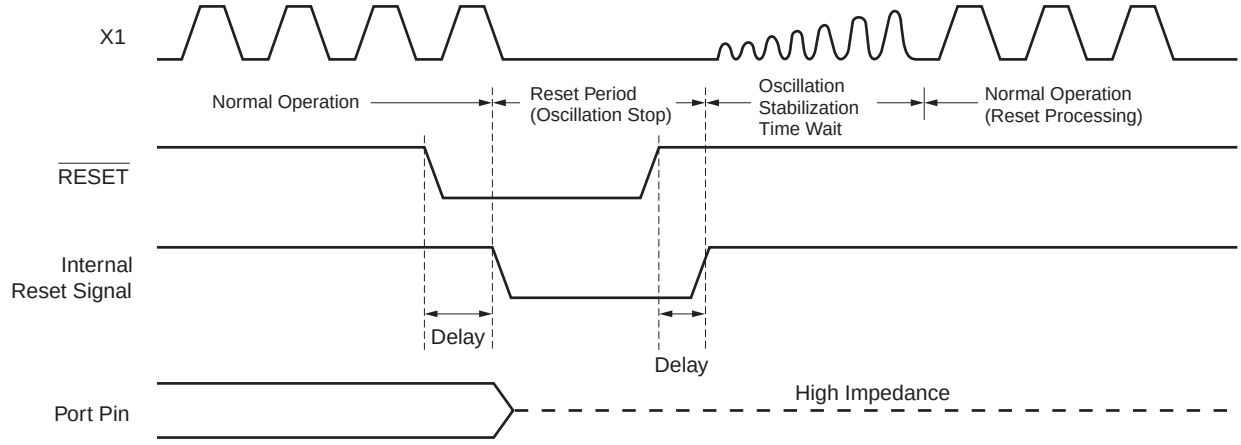


Figure 24-3: Timing of Reset due to Watchdog Timer Overflow

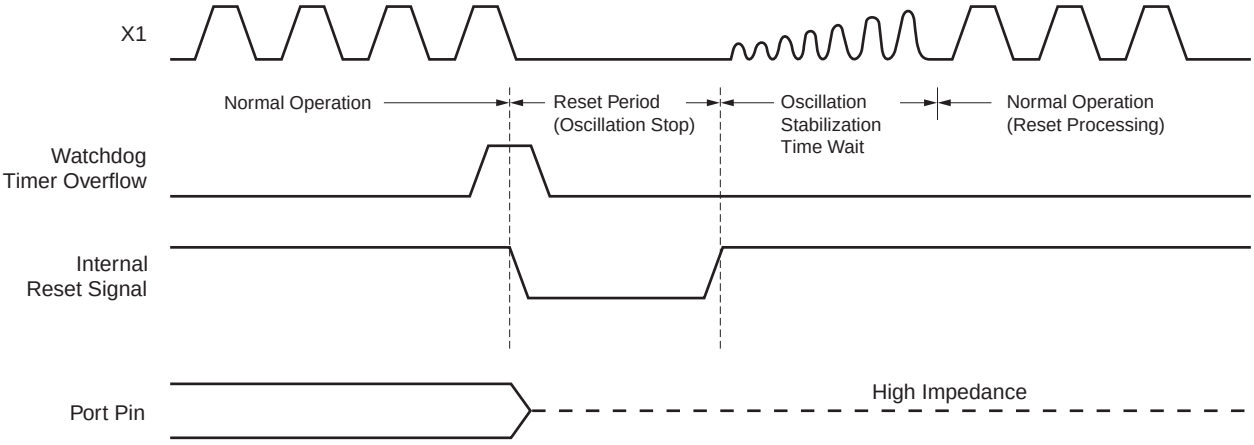


Figure 24-4: Timing of Reset Input in STOP Mode by $\overline{\text{RESET}}$ Input

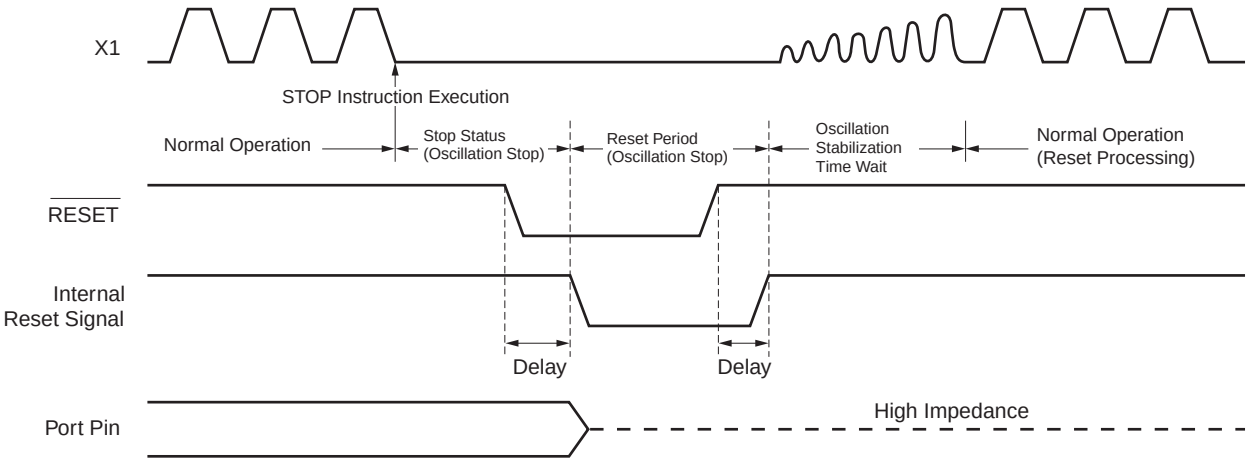


Table 24-1: Hardware Status after Reset (1/3)

Hardware		Status after Reset
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 register	Undefined[Note 2]
Port (Output latch)	Ports 0 to 7, Port 12, 13, 14 (P0 to P7, P12, P13, P14)	00H
Port mode register (PM0 to PM7, PM12, PM13, PM14)		FFH
Pull-up resistor option register (PU0, PU4, PU7, PU13)		00H
Port function selection (PF2, PF5, PF7, PF12 - PF14)		00H
Processor clock control register (PCC)		04H
Memory size switching register (IMS)		CFH
Internal expansion RAM size switching register (IXS)		0CH
Memory expansion mode register (MEM)		00H
Expansion wait register (MM)		10H
EEPROM	Write control register (EEWC)	00H
Oscillation stabilization time select register (OSTS)		04H
16-bit timer/event counter 0	Timer register (TM0)	00H
	Capture/compare register (CR00, CR01)	00H
	Prescaler selection register (PRM0)	00H
	Mode control register (TMC0)	00H
	Capture/compare control register 0 (CRC0)	04H
	Output control register (TOC0)	00H
16-bit timer/event counter 2	Timer register (TM0)	00H
	Capture control register (CR20, CR21, CR22)	00H
	Prescaler mode register (PRM2)	00H
	Mode control register (TMC2)	00H

Notes: **1.** During reset input or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remains unchanged after reset.

2. The post-reset status is held in the standby mode.

Table 24-1: Hardware Status after Reset (2/3)

Hardware		Status after Reset
8-bit timer/event counters 50 and 51	Timer register (TM50, TM51)	00H
	Compare register (CR50, CR51)	00H
	Clock select register (TCL50, TCL51)	00H
	Mode control register (TMC50, TMC51)	04H
Watch timer	Mode register (WTM)	00H
Watchdog timer	Clock selection register (WDCS)	00H
	Mode register (WDTM)	00H
PCL clock output	Clock output selection register (CKS)	00H
Sound generator	Control register (SGCR)	04H
	Amplitude control (SGAM)	00H
	Buzzer control (SGBC)	00H
Serial interface	Operating mode register 0 (CSIM30)	00H
	Shift register 0 (SIO30)	00H
	Operating mode register 1 (CSIM31)	00H
	Shift register 1 (SIO31)	00H
	Asynchronous mode register (ASIM0)	00H
	Asynchronous status register (ASIS0)	00H
	Baudrate generator control register (BRGC0)	00H
	Transmit shift register (TXS0) Receive buffer register (RXB0)	FFH
A/D converter	Mode register (ADM1)	00H
	Conversion result register (ADCR1)	00H
	Input select register (ADS1)	00H
	Power fail comparator mode (PFM)	00H
	Power fail threshold register (PFT)	00H
LCD-controller/driver	Mode register (LCDM)	00H
	Control register (LCDC)	00H
Interrupt	Request flag register (IF0L, IF0H, IF1L, IF1H)	00H
	Mask flag register (MK0L, MK0H, MK1L, MK1H)	FFH
	Priority specify flag register (PR0L, PR0H, PR1L, PR1H)	FFH
	External interrupt rising edge register (EGP)	00H
	External interrupt falling edge register (EGN)	00H

Table 24-1: Hardware Status after Reset (3/3)

Hardware		Status after Reset
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)	0CH
	Bit rate prescaler register (BRPRS)	00H
	Synchronous control register (SYNC0)	18H
	Synchronous control register (SYNC1)	0EH
	Mark control register (MASKC)	00H

[Memo]

Chapter 25 μPD78F0948, μPD78F0949

The flash memory versions of the μPD780949 Subseries includes the μPD78F0948 and the μPD78F0949.

The μPD78F0948/μPD78F0949 replaces the internal mask ROM of the μPD780948/μPD780949 with flash memory to which a program can be written, deleted and overwritten while mounted on the substrate. Table 25-1 lists the differences among the μPD78F0948/μPD78F0949 and the mask ROM versions.

Table 25-1: Differences among μPD78F0948/μPD78F0949 and Mask ROM Versions

Item	μPD78F0948/μPD78F0949	Mask ROM Versions
IC pin	None	Available
V_{PP} pin	Available	None
Electrical characteristics	See data sheet of each product.	

Caution: Flash memory versions and mask ROM versions differ in their noise tolerance and noise emission. If replacing flash memory versions with mask ROM versions when changing from test production to mass production, be sure to perform sufficient evaluation with CS versions (not ES versions) of mask ROM versions.

25.1 Memory Size Switching Register (IMS)

This register specifies the internal memory size by using the memory size switching register (IMS), so that the same memory map as on the mask ROM version can be achieved.

IMS is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets this register to the value indicated in Table 25-2.

Figure 25-1: Memory Size Switching Register 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	Note	R/W

ROM3	ROM2	ROM1	ROM0	Internal ROM size selection
1	0	0	0	32 Kbytes
1	1	1	0	56 Kbytes
1	1	1	1	60 Kbytes
Other than above				Setting prohibited

RAM2	RAM1	RAM0	Internal high-speed RAM size selection
1	1	0	1024 bytes
Other than above			Setting prohibited

Note: The values after reset depend on the product (See Table 25-2).

Table 25-2: Values when the Memory Size Switching Register is Reset

Part Number	Reset Value
μPD780948, 78F0948	CFH
μPD780949, 78F0949	CFH

25.2 Internal Extension RAM Size Switching Register

The μPD78F0948 und μPD78F0949 allow users to define its internal extension RAM size by using the internal extension RAM size switching register (IXS), so that the same memory mapping as that of a mask ROM version with a different internal extension RAM is possible.

The IXS is set by an 8-bit memory manipulation instruction.

RESET signal input sets IXS to 0AH.

Caution: When the μPD780948/μPD78F0948 and the μPD780949/μPD78F0949 are used, be sure to set the value specified in the Table 25-3 to IXS. Other settings are prohibited.

Figure 25-2: Internal Extension RAM Size Switching Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IXS	0	0	0	0	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	0AH	W

IXRAM3	IXRAM2	IXRAM1	IXRAM0	Internal extension RAM capacity selection
1	0	1	0	1024 bytes
Other than above				Setting prohibited

The value whitch is set in the IXS that has the identical memory map to the mask ROM versions is given in the Table 24-3.

Table 25-3: Examples of internal Extension RAM Size Switching Register Settings

Relevant Mask ROM Version	IXS Setting
μPD780948, μPD78F0948	0AH
μPD780949, μPD78F0949	

25.3 Flash memory programming

On-board writing of flash memory (with device mounted on target system) is supported.

On-board writing is done after connecting a dedicated flash writer (Flashpro) to the host machine and target system.

Moreover, writing to flash memory can also be performed using a flash memory writing adapter connected to Flashpro.

Remark: Flashpro is a product of Naitoudensei Machida Seisakusho, Co., Ltd.

25.3.1 Selection of transmission method

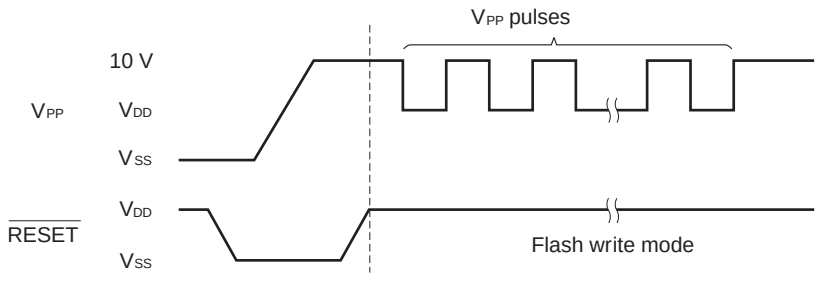
Writing to flash memory is performed using Flashpro and serial communication. Select the transmission method for writing from Table 25-4. For the selection of the transmission method, a format like the one shown in Figure 25-3 is used. The transmission methods are selected with the V_{PP} pulse numbers shown in Table 25-4.

Table 25-4 Transmission Method List

Transmission Method	Number of Channels	Pin Used	Number of V_{PP} Pulses
3-wire serial I/O	1	SI0/P20 SO0/P21 $\overline{SCK0}$/P22	1
Pseudo 3-wire serial I/O	1	P30 (Serial clock input) P31(Serial data input) P32(Serial data input)	12
UART	1	RxD0/P25 TxD0/P26	8

Cautions:
1. **Be sure to select the number of Vpp pulses shown in Table 25-3 for the transmission method.**
2. **If performing write operations to flash memory with the UART transmission method, set the main system clock oscillation frequency to 3 MHz or higher.**

Figure 25-3: Transmission Method Selection Format



25.3.2 Initialization of the programming mode

When V_{PP} reaches up to 10 V with $\overline{RESET}$ terminal activated, on-board programming mode becomes available.

After release of $\overline{RESET}$, the programming mode is selected by the number of V_{PP} pulses.

25.3.3 Flash memory programming function

Flash memory writing is performed through command and data transmit/receive operations using the selected transmission method. The main functions are listed in Table 25-5.

Table 25-5: Main Functions of Flash Memory Programming

Function	Description
Reset	Detects write stop and transmission synchronization.
Batch verify	Compares entire memory contents and input data.
Batch delete	Deletes the entire memory contents.
Batch blank check	Checks the deletion status of the entire memory.
High-speed write	Performs writing to flash memory according to write start address and number of write data (bytes).
Continuous write	Performs successive write operations using the data input with high-speed write operation.
Status	Checks the current operation mode and operation end.
Oscillation frequency setting	Inputs the resonator oscillation frequency information.
Delete time setting	Inputs the memory delete time.
Baud rate setting	Sets the transmission rate when the UART method is used.
Silicon signature read	Outputs the device name, memory capacity, and device block information.

25.3.4 Flashpro connection

Connection of Flashpro and μPD78F0948/μPD78F0949 differs depending on communication method (3-wire serial I/O, UART). Each case of connection shows in Figures 25-4, 25-5 and 25-6.

Figure 25-4: Connection of Flashpro Using 3-Wire Serial I/O Method

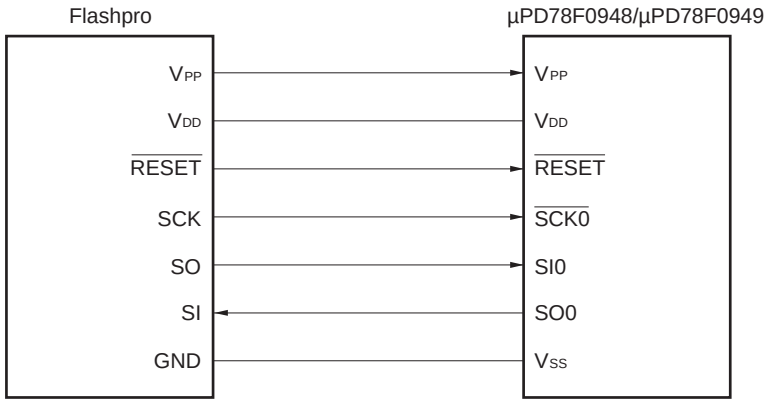


Figure 25-5: Flashpro Connection Using UART Method

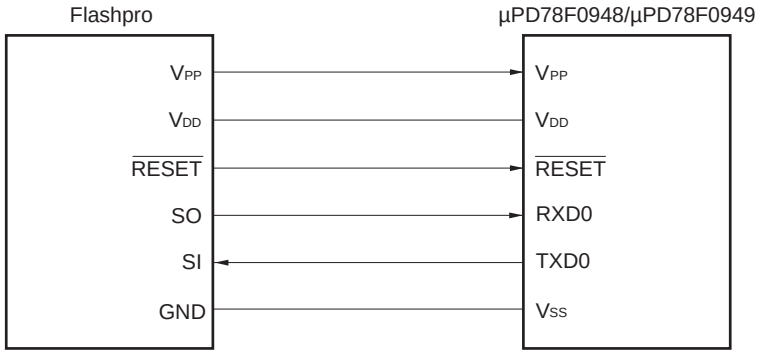
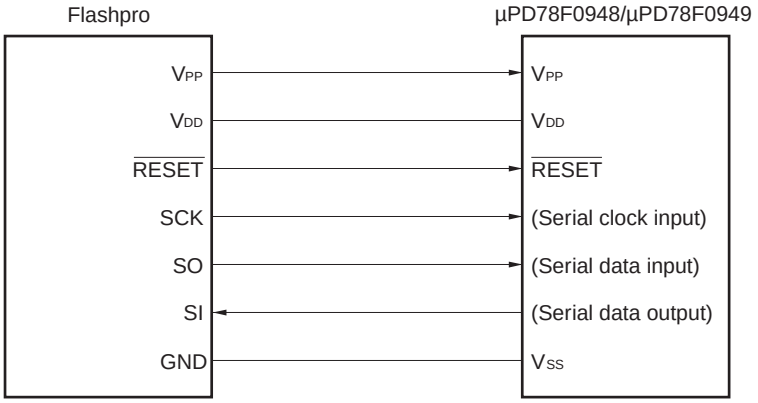


Figure 25-6: Flashpro Connection Using Pseudo 3-wire Serial I/O



VPP: 10.3 V applied from the o-board programming tool.
RESET: A RESET is generated and the device is set to the on-board programming mode.
System clock: The CPU clock for the device may be supplied by the on-board program tool. Altenatively the crystal or ceramic oscillator on the target H/W can be used in the on-board programming mode. The external system clock has to be connected with the X1 pin on the device.
VDD: The power supply for the device may be supplied by the on-board program tool. Alternatively the power supply on the target H/W can be used in the on-board programming mode.
GND: Ground level VSS.
SCK/PSCK: Serial clock generated by the on-board programming tool.
SI/PSI: Serial data sent by the on-board programming tool.
SO/PSO: Serial data sent by the device.
RxD: Serial data sent by the on-board programming tool.
TxD: Serial data sent by the device.

25.3.5 Flash programming precautions

- Please make sure that the signals used by the on-board programming tool do not conflict with other devices on the target H/W.
- A read functionality is not supported because of software protection. Only a verify operation of the whole Flash EPROM is supported. In verify mode data from start address to final address (EFFFH) has to be supplied by the programming tool. The device compares each data with on-chip flash content and replies with a signal for O.K. or not O.K.

[Memo]

Chapter 26 Instruction Set

This chapter describes each instruction set of the μPD780949 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)**."

26.1 Legends Used in Operation List

26.1.1 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 26-1: 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".

26.1.2 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)

26.1.3 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

26.2 Operation List

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	sfr ← 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	x x x
		A, PSW	2	–	5	A ← PSW	
		PSW, A	2	–	5	PSW ← A	x x x
		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”

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.

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 3	1	4	–	AX ← rp	
		rp, AX Note 3	1	4	–	rp ← AX	
		AX, !addr16	3	10	12 + 2n	AX ← (addr16)	
		!addr16, AX	3	10	12 + 2m	(addr16) ← AX	
	XCHW	AX, rp Note 3	1	4	–	AX × rp	
8-bit operation	ADD	A, #byte	2	4	–	A, CY ← A + byte	x x x
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	x x x
		A, r Note 4	2	4	–	A, CY ← A + r	x x x
		r, A	2	4	–	r, CY ← r + A	x x x
		A, saddr	2	4	5	A, CY ← A + (saddr)	x x x
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16)	x x x
		A, [HL]	1	4	5 + n	A, CY ← A + (HL)	x x x
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte)	x x x
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B)	x x x
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C)	x x x
	ADDC	A, #byte	2	4	–	A, CY ← A + byte + CY	x x x
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	x x x
		A, r Note 4	2	4	–	A, CY ← A + r + CY	x x x
		r, A	2	4	–	r, CY ← r + A + CY	x x x
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	x x x
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16) + CY	x x x
		A, [HL]	1	4	5 + n	A, CY ← A + (HL) + CY	x x x
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte) + CY	x x x
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B) + CY	x x x
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C) + CY	x x x

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. Only when rp = BC, DE or HL
4. Except “r = A”

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.

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	x x x
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte	x x x
		A, r Note 3	2	4	–	A, CY ← A – r	x x x
		r, A	2	4	–	r, CY ← r – A	x x x
		A, saddr	2	4	5	A, CY ← A – (saddr)	x x x
		A, !addr16	3	8	9 + n	A, CY ← A – (addr16)	x x x
		A, [HL]	1	4	5 + n	A, CY ← A – (HL)	x x x
		A, [HL + byte]	2	8	9 + n	A, CY ← A – (HL + byte)	x x x
		A, [HL + B]	2	8	9 + n	A, CY ← A – (HL + B)	x x x
		A, [HL + C]	2	8	9 + n	A, CY ← A – (HL + C)	x x x
	SUBC	A, #byte	2	4	–	A, CY ← A – byte – CY	x x x
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte – CY	x x x
		A, r Note 3	2	4	–	A, CY ← A – r – CY	x x x
		r, A	2	4	–	r, CY ← r – A – CY	x x x
		A, saddr	2	4	5	A, CY ← A – (saddr) – CY	x x x
		A, !addr16	3	8	9 + n	A, CY ← A – (addr16) – CY	x x x
		A, [HL]	1	4	5 + n	A, CY ← A – (HL) – CY	x x x
		A, [HL + byte]	2	8	9 + n	A, CY ← A – (HL + byte) – CY	x x x
		A, [HL + B]	2	8	9 + n	A, CY ← A – (HL + B) – CY	x x x
		A, [HL + C]	2	8	9 + n	A, CY ← A – (HL + C) – CY	x x x
	AND	A, #byte	2	4	–	A ← A ∧ byte	x
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	x
		A, r Note 3	2	4	–	A ← A ∧ r	x
		r, A	2	4	–	r ← r ∧ A	x
		A, saddr	2	4	5	A ← A ∧ (saddr)	x
		A, !addr16	3	8	9 + n	A ← A ∧ (addr16)	x
		A, [HL]	1	4	5 + n	A ← A ∧ [HL]	x
		A, [HL + byte]	2	8	9 + n	A ← A ∧ [HL + byte]	x
		A, [HL + B]	2	8	9 + n	A ← A ∧ [HL + B]	x
		A, [HL + C]	2	8	9 + n	A ← A ∧ [HL + C]	x

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”

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.

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	x
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	x
		A, r Note 3	2	4	–	A ← A ∨ r	x
		r, A	2	4	–	r ← r ∨ A	x
		A, saddr	2	4	5	A ← A ∨ (saddr)	x
		A, !addr16	3	8	9 + n	A ← A ∨ (addr16)	x
		A, [HL]	1	4	5 + n	A ← A ∨ (HL)	x
		A, [HL + byte]	2	8	9 + n	A ← A ∨ (HL + byte)	x
		A, [HL + B]	2	8	9 + n	A ← A ∨ (HL + B)	x
		A, [HL + C]	2	8	9 + n	A ← A ∨ (HL + C)	x
	XOR	A, #byte	2	4	–	A ← A ⊻ byte	x
		saddr, #byte	3	6	8	(saddr) ← (saddr) ⊻ byte	x
		A, r Note 3	2	4	–	A ← A ⊻ r	x
		r, A	2	4	–	r ← r ⊻ A	x
		A, saddr	2	4	5	A ← A ⊻ (saddr)	x
		A, !addr16	3	8	9 + n	A ← A ⊻ (addr16)	x
		A, [HL]	1	4	5 + n	A ← A ⊻ (HL)	x
		A, [HL + byte]	2	8	9 + n	A ← A ⊻ (HL + byte)	x
		A, [HL + B]	2	8	9 + n	A ← A ⊻ (HL + B)	x
		A, [HL + C]	2	8	9 + n	A ← A ⊻ (HL + C)	x
	CMP	A, #byte	2	4	–	A – byte	x x x
		saddr, #byte	3	6	8	(saddr) – byte	x x x
		A, r Note 3	2	4	–	A – r	x x x
		r, A	2	4	–	r – A	x x x
		A, saddr	2	4	5	A – (saddr)	x x x
		A, !addr16	3	8	9 + n	A – (addr16)	x x x
		A, [HL]	1	4	5 + n	A – (HL)	x x x
		A, [HL + byte]	2	8	9 + n	A – (HL + byte)	x x x
		A, [HL + B]	2	8	9 + n	A – (HL + B)	x x x
		A, [HL + C]	2	8	9 + n	A – (HL + C)	x x x

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”

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.

Instruction Group	Mnemonic	Operands	Byte			Operation	
16-bit operation	**ADDW**	AX, #word	3	6	–	AX, CY ← AX + word	x x x
	SUBW	AX, #word	3	6	–	AX, CY ← AX – word	x x x
	CMPW	AX, #word	3	6	–	AX – word	x x x
Multiply/ divide	**MULU**	X	2	16	–	AX ← A x X	
	DIVUW	C	2	25	–	AX (Quotient), C (Remainder) ← AX ÷ C	
Increment/ decrement	**INC**	r	1	2	–	r ← r + 1	x x
		saddr	2	4	6	(saddr) ← (saddr) + 1	x x
	DEC	r	1	2	–	r ← r – 1	x x
		saddr	2	4	6	(saddr) ← (saddr) – 1	x x
	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	x
	ROL	A, 1	1	2	–	(CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$) x 1 time	x
	RORC	A, 1	1	2	–	(CY ← A_0, $A_7 \leftarrow$ CY, $A_{m-1} \leftarrow A_m$) x 1 time	x
	ROLC	A, 1	1	2	–	(CY ← A_7, $A_0 \leftarrow$ CY, $A_{m+1} \leftarrow A_m$) x 1 time	x
	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]	2	10	12+n+m	$A_{3-0} \leftarrow (HL)_{7-4}$, $(HL)_{3-0} \leftarrow A_{3-0}$, $(HL)_{7-4} \leftarrow (HL)_{3-0}$	
BCD adjust	**ADJBA**		2	4	–	Decimal Adjust Accumulator after Addition	x x x
	ADJBS		2	4	–	Decimal Adjust Accumulator after Subtract	x x x
Bit manipulate	**MOV1**	CY, saddr.bit	3	6	7	CY ← (saddr.bit)	x
		CY, sfr.bit	3	–	7	CY ← sfr.bit	x
		CY, A.bit	2	4	–	CY ← A.bit	x
		CY, PSW.bit	3	–	7	CY ← PSW.bit	x
		CY, [HL].bit	2	6	7 + n	CY ← (HL).bit	x
		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	x x
		[HL].bit, CY	2	6	8+n+m	(HL).bit ← 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

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.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulate	**AND1**	CY, saddr.bit	3	6	7	CY ← CY ∧ (saddr.bit)			x
		CY, sfr.bit	3	–	7	CY ← CY ∧ sfr.bit			x
		CY, A.bit	2	4	–	CY ← CY ∧ A.bit			x
		CY, PSW.bit	3	–	7	CY ← CY ∧ PSW.bit			x
		CY, [HL].bit	2	6	7 + n	CY ← CY ∧ (HL).bit			x
	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ (saddr.bit)			x
		CY, sfr.bit	3	–	7	CY ← CY ∨ sfr.bit			x
		CY, A.bit	2	4	–	CY ← CY ∨ A.bit			x
		CY, PSW.bit	3	–	7	CY ← CY ∨ PSW.bit			x
		CY, [HL].bit	2	6	7 + n	CY ← CY ∨ (HL).bit			x
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ⊻ (saddr.bit)			x
		CY, sfr.bit	3	–	7	CY ← CY ⊻ sfr.bit			x
		CY, A.bit	2	4	–	CY ← CY ⊻ A.bit			x
		CY, PSW. bit	3	–	7	CY ← CY ⊻ PSW.bit			x
		CY, [HL].bit	2	6	7 + n	CY ← CY ⊻ (HL).bit			x
	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	x	x	x
		[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	x	x	x
		[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 ← $\overline{CY}$			x

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

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.

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Call/return	**CALL**	!addr16	3	7	–	(SP – 1) ← (PC + 3)H, (SP – 2) ← (PC + 3)L, PC ← addr16, SP ← SP – 2	
	CALLF	!addr11	2	5	–	(SP – 1) ← (PC + 2)H, (SP – 2) ← (PC + 2)L, PC15 – 11 ← 00001, PC10 – 0 ← addr11, SP ← SP – 2	
	CALLT	[addr5]	1	6	–	(SP – 1) ← (PC + 1)H, (SP – 2) ← (PC + 1)L, PCH ← (00000000, addr5 + 1), PCL ← (00000000, addr5), SP ← SP – 2	
	BRK		1	6	–	(SP – 1) ← PSW, (SP – 2) ← (PC + 1)H, (SP – 3) ← (PC + 1)L, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0	
	RET		1	6	–	PCH ← (SP + 1), PCL ← (SP), SP ← SP + 2	
	RETI		1	6	–	PCH ← (SP + 1), PCL ← (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) ← PSW, SP ← SP – 1	
		rp	1	4	–	(SP – 1) ← rpH, (SP – 2) ← rpL, SP ← SP – 2	
	POP	PSW	1	2	–	PSW ← (SP), SP ← SP + 1	R R R
		rp	1	4	–	rpH ← (SP + 1), rpL ← (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	–	PCH ← 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

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.

Instruction Group	Mnemonic	Operands	Byte			Operation	
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	x x x
		[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

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.

26.3 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

First Operand \ Second 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	MOV									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

Second 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

Second Operand / First 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

First Operand \ Second Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

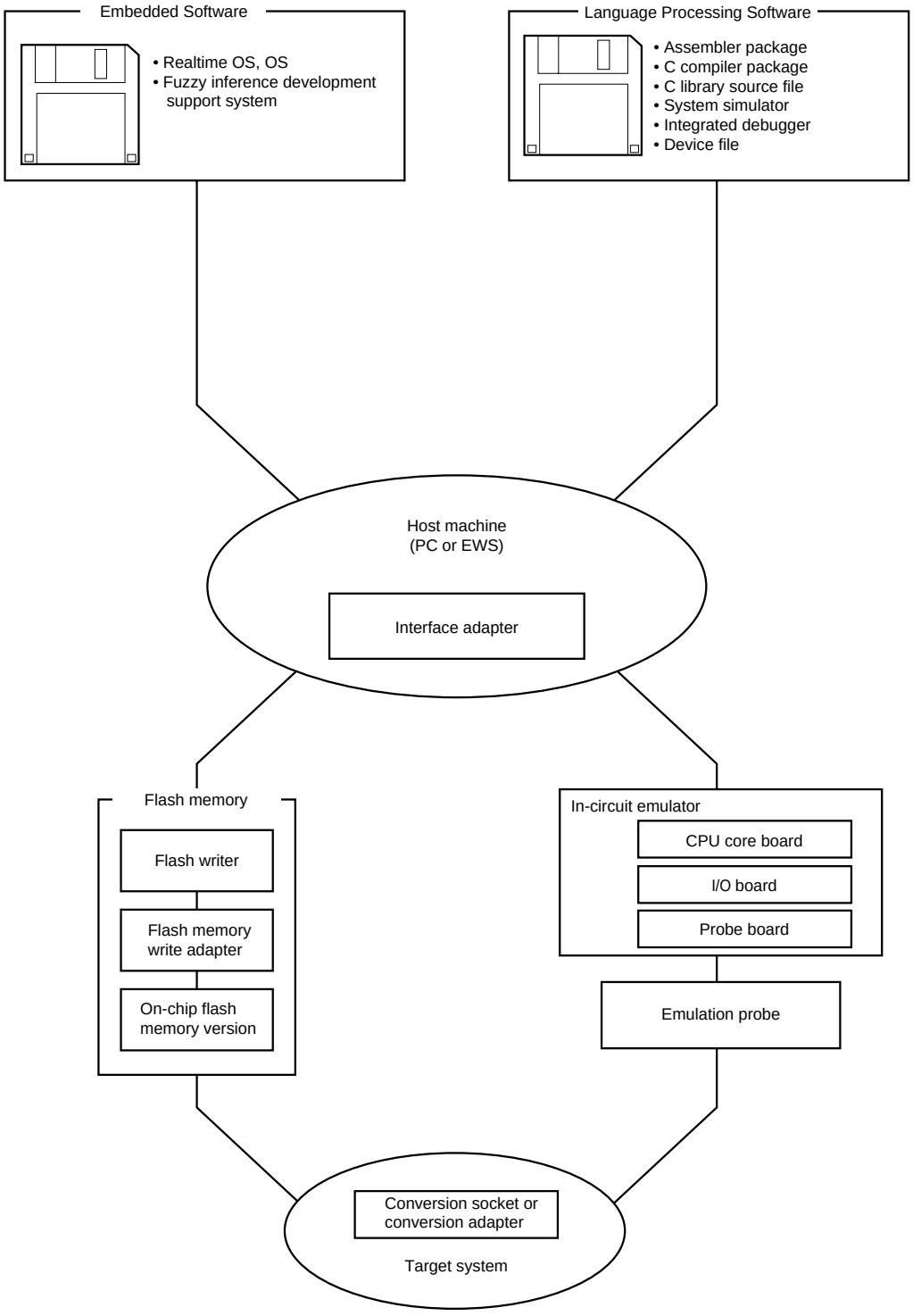
[Memo]

Appendix A Development Tools

The following development tools are available for the development of systems that employ the μPD780949 Subseries.

Figure A-1 shows the development tool configuration.

Figure A-1: Development Tool Configuration



A.1 Language Processing Software

RA78K/0 Assembler Package	This assembler converts programs written in mnemonics into an object code executable with a microcomputer. Further, this assembler is provided with functions capable of automatically creating symbol tables and branch instruction optimization. This assembler is used in combination with an optional device file (DF780949).
	Part Number: μSxxxxRA78K0
CC78K/0 C Compiler Package	This compiler converts programs written in C language into object code executable with a microcomputer. This compiler is used in combination with an optional assembler package (RA78K/0) and device file (780949).
	Part Number: μSxxxxCC78K0
DF780948[Note 1, 2] Device File	This file contains information peculiar to the device. This file is used in combination with the RA78K/0, CC78K/0, SM78K0, and ID78K0.
	Part Number: μSxxxxDF780949
CC78K/0-L C Library Source File	This is a source program of functions configuring the object library included in the C compiler package (CC78K/0). It is required for matching the object library included in the CC78K/0 with to the customer's specifications.
	Part Number: μSxxxxCC78K0-L

Notes: **1.** Used in common with DF780948, RA78K/0, CC78K0, SM78K0 and ID78K0.
2. Under development

Remark: xxxx in the part number differs depending on the host machine and OS used.

μSxxxxRA78K0
μSxxxxCC78K0
μSxxxxDF780948
μSxxxxCC78K0-L

xxxx	Host Machine	OS	Supply Medium
5A13	PC-9800 Series	MS-DOS (Ver.3.30 to Ver.6.2[Note])	3.5-inch 2HD
5A10			5-inch 2HD
7B13	IBM PC/AT or compatible	See A.4	3.5-INCH 2HD
7B10			5-inch 2HD
3H15	HP9000 Series 300™	HP-UX™ (rel.7.05B)	Cartrige tape (QIC-24)
3P16	HP9000 Series 700™	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation™	SunOS™ (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 Series (RISC)	EWS-UX/V (rel.4.0)	

Note: The task swap function is not supported by the software listed above, although it is provided in MS-DOS version 5.0 and later.

A.2 Flash Memory Writing Tools

Flashpro Flash Writer	Dedicated flash writer for microcontrollers with on-chip flash memory. Flashpro is a product of Naitoudensei Machida Seisakusho, Co., Ltd.
Flash memory writing adapter FA-100GF-SL	μPD780949 Subseries flash memory writing adapter used connected to Flashpro. These are products of Naitoudensei Machida Seisakusho, Co., Ltd. • FA-100GF-SL: 100-pin plastic QFP (14 x 20 mm)

A.3 Debugging Tools

A.3.1 Hardware

IE-78001-R-A **Note** In-Circuit Emulator	This in-circuit emulator serves to debug hardware and software when developing application systems using the 78K/0 Series. It corresponds to integrated debugger ID78K0. This emulator is used in combination with an emulation probe an interface adapter for connection to a host machine.
IE-70000-PC-IF-C Interface Adapter	This adapter is required when using an IBM PC/AT or compatible as the IE-780000-SL host machine.
IE-780948-Sl-EM1**Note** I/O Board	This board is used to perform emulation of device specific peripheral hardware. This board is used in combination with an in-circuit emulator, CPU core board, and probe board.
IE-780948-SL-EM4**Note** Probe Board	This board is used to perform mask option settings and pin connection changes.
EP-100GF-SL**Note** Emulation Probe	This probe is used to connect the in-circuit emulator and the target system. It is for 100-pin plastic QFP . A 100-pin conversion socket is included to facilitate target system development.
N0 PACK 100 KB Y0 PACK 100 KB	This conversion socket is used to connect a target system substrate designed to allow mounting of a 100-pin plastic QFP and the EP-100GF-SL.

Note: Under development.

A.3.2 Software (1/2)

SM78K0 System Simulator	This system simulator is used to perform debugging at C source level or assembler level while simulating the operatin of the target system on a host machine. The SM78K0 operates on Windows. Use of the SM78K0 allows the execution of application logical testing and performance testing on an independent basis from hardware development without having to use an in-circuit emulator, thereby providing higher development efficiency and software quality. The SM78K0 is used in combination with the optional device file DF78F0949.
	Part Number: μSxxxxSM78K0-L

Remark: xxxx in the part number differs depending on the host machine and OS used.

μSxxxxSM78K0

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	MS-DOS (Ver.3.30 to Ver.6.2[Note]) + Windows (Ver. 3.0 to Ver. 3.1)	3.5-inch 2HD
AB13	IBM PC/AT and compatible (Windows Japanese version)	See A.4	3.5-inch 2HC
BB13	IBM PC/AT and compatible (Windows English version)		3.5-inch 2HC

Note: The task swap funtion is not supported by the software listed above, although it is provided in MS-DOS version 5.0 and later.

A.3.2 Software (2/2)

ID78K0 Integrated Debugger	This is a control program used to debug the 78K/0 Series. The graphical user interfaces employed are Windows for personal computers and OSF/Motif for EWSs, offering the standard appearance and operability typical of these interfaces. Further, debugging functions supporting C language are reinforced, and the trace result can be displayed in C language level by using a window integrating function that associates the source program, disassemble display, and memory display with the trace result. In addition, it can enhance the debugging efficiency of a program using a real-time OS by incorporating function expansion modules such as a task debugger and system performance analyzer. This debugger is used in combination with an optional device file.
	Part Number: μSXXXXID78K0
DF780949**Notes 1, 2** Device File	File containing information peculiar to the device. Used in combination with optional RA78K/0, CC78K/0, SM78K0, or ID78K0.
	Part Number: μSXXXXDF780949.

Notes: **1.** The DF780948 can be used in conjunction with the RA78K/0, CC78K/0, SM78K0, and ID78K0.
2. Under development.

Remark: xxxx in the part number differs depending on the host machine and OS used.

μSxxxxID78K0
μSxxxxDF780948

XXXX	Host Machine	OS	Supply Medium
AA13	PC-9800 Series	MS-DOS (Ver.3.30 to Ver.6.2**Note**) + Windows (Ver. 3.1)	3.5-inch 2HD
AB13	IBM PC/AT or compatible (Japanese Windows)	See A.4	3.5-inch 2HC
BB13	IBM PC/AT or compatible (English Windows)		3.5-inch 2HC
3P16	HP9000 Series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3K13			3.5-inch 2HC
3R16	NEWS™ (RISC)	NEWS-OS™ (6.1x)	1/4 inch CGMT
3R13			3.5-inch 2HC
3M15	EWS4800 Series (RISC)	EWS-UX/V (rel.4.0)	Cartridge tape (QIC-24)

Note: The task swap function is not supported by the software listed above, although it is provided in MS-DOS version 5.0 and later.

A.4 OS for IBM PC

The following OSs for IBM PCs are supported.

To operate SM78K0, ID78K0, and FE9200 (see **B.2 Fuzzy Inference Development Support System**), Windows (Ver. 3.0 to Ver. 3.1) is necessary.

OS	Version
PC DOS	Ver. 5.02 to Ver. 6.3
	J6.1/V[Note] to J6.3V[Note]
IBM DOS™	J5.02/V[Note]
MS-DOS	Ver. 5.0 to Ver. 6.22
	5.0/V[Note] to 6.2/V[Note]

Note: Only English mode is supported.

Caution: Although Ver. 5.0 and above have a task swapping function, this function cannot be used with this software.

A.5 Development Environment when Using IE-78001-R-A

When using the IE-78001-R-A as the in-circuit emulator, the following debugging tools are required.

IE-78001-R-A In-Circuit Emulator	This in-circuit emulator is used to debug hardware and software when an application system using the 78K/0 Series is developed. It supports the integrated debugger (ID78K0). This emulator is used in combination with an emulation probe and an interface adapter that connects the emulator with the host machine.
IE-70000-98-IF-B IE-70000-98N-IN IE-70000-PC-IF-C Interface adapter	See **A.3.1 Hardware**.
E-78000-R-SV3 Interface Adapter	Adapter cable necessary when using an EWS as the host machine of the IE-780000-R-A. This cable is connected to the board in the IE-780000-R-A. As Ethernet™, 10Base-5 is supported. If other methods are used, a commercially available conversion adapter is necessary.
IE-780948-SL-EM1**Note** I/O Board	This board is used to emulate peripheral hardware peculiar to the device. It is used in combination with an in-circuit emulator, interface board, and probe board. See also **A.3.1 Hardware**.
IE-780948-SL-EM4**Note** Probe Board	See **A.3.1 Hardware**.
EP-100GF-SL**Note** Emulation Probe Conversion Adapter	

Note: Under development.

[Memo]

Appendix B Embedded Software

For efficient development and maintenance of the μPD780949 Subseries, the following embedded software products are available.

B.1 Real-Time OS (1/2)

RX78K/0 Real-time OS	RX78K/0 is a real-time OS conforming with the μITRON specifications. Tool (configurator) for generating nucleus of RX78K/0 and plural information tables is supplied. Used in combination with an optional assembler package (RA78K/0) and device file.
	Part number: μSxxxxRX78013-ΔΔΔΔ

Caution: When purchasing the RX78K/0, fill in the purchase application form in advance and sign the User Agreement.

Remark: xxxx and ΔΔΔ in the part number differ depending on the host machine and OS used.

μSxxxxMX78013-ΔΔΔΔ

ΔΔΔΔ	Product Outline	Upper limit of mass-production quantity
001	Evaluation object	Do not use for mass-produced products.
100K	Object for mass-produced product	0.1 million units
001M		1 million units
010M		10 million units
S01	Source program	Source program for mass-produced object

xxxx	Host Machine	OS	Supply Medium
5A13	PC-9800 Series	MS-DOS (Ver. 3.30 to Ver.6.2[Note])	3.5-inch 2HD
5A10			5-inch 2HD
7B13	IBM PC/AT and compatible	See A.4.	3.5-inch 2HC
7B10			5-inch 2HC
3H15	HP9000 Series 300	HP-UX (rel.7.05B)	Cartrige tape (QIC-24)
3P16	HP9000 Series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 Series (RISC)	EWS-UX/V (rel.4.0)	

Note: The task swap function is not supported by the software listed above, although it is provided in MS-DOS version 5.0 and later.

B.1 Real-Time OS (2/2)

MX78K0 OS	μTRON specification subset OS. Nucleus of MX78K0 is supplied. This OS performs task management, event management, and time management. It controls the task execution sequence for task management and selects the task to be executed next.
	Part number: μSxxxxMX78K0-ΔΔΔ

Remark: xxxx and ΔΔΔ in the part number differ depending on the host machine and OS used.

μSxxxxMX78K0-ΔΔΔ

ΔΔΔ	Product Outline	Note
001	Evaluation object	Use for trial product.
xx	Object for mass-produced product	Use for mass-produced product.
S01	Source program	Can be purchased only when object for mass-produced product is purchased.

xxxx	Host Machine	OS	Supply Medium
5A13	PC-9800 Series	MS-DOS (Ver. 3.30 to Ver.6.2[Note])	3.5-inch 2HD
5A10			5-inch 2HD
7B13	IBM PC/AT and compatible	See A.4.	3.5-inch 2HC
7B10			5-inch 2HC
3H15	HP9000 Series 300	HP-UX (rel.7.05B)	Cartrige tape (QIC-24)
3P16	HP9000 Series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 Series (RISC)	EWS-UX/V (rel.4.0)	

Note: The task swap function is not supported by the software listed above, although it is provided in MS-DOS version 5.0 and later.

B.2 Fuzzy Inference Development Support System

FE9000/FE9200 Fuzzy knowledge data creation tool	Program that supports input, edit, and evaluation (simulation) of fuzzy knowledge data (fuzzy rule and membership function). FE9200 works on Windows.
	Part number: μSxxxxFE9000 (PC-9800 Series) μSxxxxFE9200 (IBM PC/AT and compatible machines)
FT9080/FT9085	Program that translates fuzzy knowledge data obtained by using fuzzy knowledge Translator data creation tool into assembler source program for RA78K0.
	Part number: μSxxxxFT9080 (PC-9800 Series) μSxxxxFT9085 (IBM PC/AT and compatible machines)
FI78K0	Program that executes fuzzy inference. Executes fuzzy inference when linked with Fuzzy inference module fuzzy knowledge data translated by translator.
	Part number: μSxxxxFI78K0 (PC-9800 Series, IBM PC/AT and compatible machines)
FD78K0	Support software for evaluation and adjustment of fuzzy knowledge data by using Fuzzy inference debugger in-circuit emulator and at hardware level.
	Part number: μSxxxxFD78K0 (PC-9800 Series, PC/AT and compatible machines)

Remark: xxxx in the part number differs depending on the host machine and the OS used.

μSxxxxFE9000
μSxxxxFT9080
μSxxxxFI78K0
μSxxxxFD78K0

xxxx	Host machine	OS	Supply media
5A13	PC-9800 Series	MS-DOS	3.5" 2HD
5A10		(Ver. 3.30 to Ver. 6.2[Note])	5" 2HD

Note: MS-DOS Ver. 5.0 and later have the task swap function, but this function cannot be used for the above software.

μSxxxxFE9200
μSxxxxFT9085
μSxxxxFI78K0
μSxxxxFD78K0

xxxx	Host machine	OS	Supply media
7B13	IBM PC/AT and compatible machines	See A.4.	3.5" 2HC
7B10			5" 2HC

[Memo]

Appendix C Register Index

C.1 Register Index (In Alphabetical Order with Respect to Register Names)

[A]
A/D conversion result register 1 (ADCR1) ... 210, 222
A/D converter mode register (ADM1) ... 212
Analog input channel specification register (ADS1) ... 213
Asynchronous serial interface mode register (ASIM0) ... 244, 245, 249
Asynchronous serial interface status register (ASIS0) ... 246, 250

[B]
Baud rate generator control register (BRGC0) ... 247, 251
Bit rate prescaler (BRPRS) ... 310

[C]
CAN control register (CANC) ... 301
CAN error status register (CANES) ... 304
Capture/compare control register (CRC0) ... 130
Capture/compare register 00 (CR00) ... 125
Capture/compare register 01 (CR01) ... 126
Capture pulse control register (CRC2) ... 160
Capture register 20 (CR20) ... 158
Capture register 21 (CR21) ... 158
Capture register 22 (CR22) ... 158
Clock output selection register (CKS) ... 206

[D]
D/A converter mode register (DAM0) ... 223

[E]
EEPROM write control register (EEWC) ... 83
8-bit compare register 50 (CR50) ... 172
8-bit compare register 51 (CR51) ... 172
8-bit counter 50 (TM50) ... 172
8-bit counter 51 (TM51) ... 172
8-bit timer mode control register 50 (TMC50) ... 161, 175
8-bit timer mode control register 51 (TMC51) ... 161, 176
External interrupt falling edge enable register (EGN) ... 364
External interrupt rising edge enable register (EGP) ... 364

[I]
Internal extension RAM size switching register (IXS) ... 407
Interrupt mask flag register 0H (MK0H) ... 362
Interrupt mask flag register 0L (MK0L) ... 362
Interrupt mask flag register 1H (MK1H) ... 362
Interrupt mask flag register 1L (MK1L) ... 362
Interrupt request flag register 0H (IF0H) ... 361
Interrupt request flag register 0L (IF0L) ... 361
Interrupt request flag register 1H (IF1H) ... 361
Interrupt request flag register 1L (IF1L) ... 361

[L]
LCD display mode register (LCDM) ... 334, 345
LCD display control register (LCDC) ... 335

[M]
Mask control register (MASKC) ... 318
Message count register (MCNT) ... 309
Memory expansion wait setting register (MM) ... 381
Memory size switching register (IMS) ... 382, 406
Memory expansion mode register (MEM) ... 106, 380

[O]
Oscillation stabilization time selection register (OSTS) ... 391

[P]
Port 0 (P0) ... 91
Port 1 (P1) ... 92
Port 2 (P2) ... 93
Port 3 (P3) ... 94
Port 4 (P4) ... 95
Port 5 (P5) ... 96
Port 6 (P6) ... 97
Port 7 (P7) ... 98
Port 12 (P12) ... 99
Port 13 (P13) ... 100
Port 14 (P14) ... 101
Port function register 2 (PF2) ... 105
Port function register 5 (PF5) ... 105
Port function register 7 (PF7) ... 105
Port function register 12 (PF12) ... 105
Port function register 13 (PF13) ... 105
Port function register 14 (PF14) ... 105
Port mode register 0 (PM0) ... 102, 103, 133, 177
Port mode register 2 (PM2) ... 102, 103
Port mode register 3 (PM3) ... 102, 103, 207
Port mode register 4 (PM4) ... 102, 103
Port mode register 5 (PM5) ... 102, 103
Port mode register 6 (PM6) ... 102, 103
Port mode register 7 (PM7) ... 102, 103
Port mode register 12 (PM12) ... 102, 103
Port mode register 13 (PM13) ... 102, 103
Port mode register 14 (PM14) ... 102, 103
Power-fail compare mode register (PFM) ... 214
Power-fail compare threshold value register (PFT) ... 214

[P]
Prescaler mode register (PRM0) … 132
Prescaler mode register (PRM2) … 161
Priority specify flag register 0H (PR0H) … 363
Pirority specify flag register 0L (PR0L) … 363
Priority specify flag register 1H (PR1H) … 363
Priority specify flag register 1L (PR1L) … 363
Processor clock control register (PCC) … 111
Program status word (PSW) … 365
Pull-up resistor option register 0 (PU0) … 104
Pull-up resistor option register 4 (PU4) … 104
Pull-up resistor option register 7 (PU7) … 104
Pull-up resistor option register 13 (PU13) … 104

[R]
Receive buffer register (RXB0) … 243
Receive error counter (REC) ... 308
Receive message register (RMES) ... 317
Receive shift register (RXS0) … 243
Redefinition control register (REDEF) ... 320

[S]
Serial I/O shift register 30 (SIO30) … 228, 229, 230, 231
Serial I/O shift register 31 (SIO31) … 235
Serial operation mode register 30 (CSIM30) … 228
Serial operation mode register 31 (CSIM31) … 235, 236, 237, 238
16-bit timer mode control register (TMC0) … 127
16-bit timer mode control register (TMC2) … 159
16-bit timer output control register (TOC0) … 131
16-bit timer register (TM0) … 124
16-bit timer register (TM2) … 158
Sound generator control register (SGCR) ... 350
Sound generator buzzer control register (SGBR) ... 351
Sound generator amplitude register (SGAM) ... 352
Successive approximation register (SAR) ... 210
Synchronisation control register 0 (SYNC0) ... 311, 313
Synchronisation control register 1 (SYNC1) ... 311

[T]
Timer clock selection register 50 (TCL50) … 173
Timer clock selection register 51 (TCL51) … 174
Transmit control register (TCR) ... 315
Transmit error counter (TEC) ... 307
Transmit shift register (TXS0) … 243

[W]
Watch timer mode control register (WTM) … 193
Watchdog timer clock selection register (WDCS) … 199
Watchdog timer mode register (WDTM) … 200

C.2 Register Index (In Alphabetical Order with Respect to Register Symbol)

ADCR1 : A/D conversion result register 1
ADM1 : A/D converter mode register
ADS1 : Analog input channel specification register
ASIM0 : Asynchronous serial interface mode register
ASIS0 : Asynchronous serial interface status register
BRGC0 : Baud rate generator control register
BRPRS : Bit rate prescaler
CANC : CAN control register
CANES : CAN error status register
CKS : Clock output selection register
CR00 : Capture/compare register 00
CR01 : Capture/compare register 01
CR20 : Capture register 20
CR21 : Capture register 21
CR22 : Capture register 22
CR50 : 8-bit compare register 50
CR51 : 8-bit compare register 51
CRC0 : Capture/compare control register
CRC2 : Capture pulse control register
CSIM30 : Serial operation mode register 30
CSIM31 : Serial operation mode register 31
DAM0 : D/A converter mode register
EEWC : EEPROM write control register
EGN : External interrupt falling edge enable register
EGP : External interrupt rising edge enable register
IF0H : Interrupt request flag register 0H
IF0L : Interrupt request flag register 0L
IF1H : Interrupt request flag register 1H
IF1L : Interrupt request flag register 1L
IMS : Memory size switching register
IXS : Internal extension RAM size switching register
LCDC : LCD display control register
LCDM : LCD display mode register
MASKC : Mask control register
MCNT : Message count register
MEM : Memory expansion mode register
MK0H : Interrupt mask flag register 0H
MK0L : Interrupt mask flag register 0L
MK1L : Interrupt mask flag register 1L
MK1H : Interrupt mask flag register 1H
MM : Memory expansion wait setting register
OSTS : Oscillation stabilization time selection register

P0 : Port 0
P1 : Port 1
P2 : Port 2
P3 : Port 3
P4 : Port 4
P5 : Port 5
P6 : Port 6
P7 : Port 7
P12 : Port 12
P13 : Port 13
P14 : Port 14
PCC : Processor clock contrtol register
PF2 : Port function register 2
PF5 : Port function register 5
PF7 : Port function register 7
PF12 : Port function register 12
PF13 : Port function register 13
PF14 : Port function register 14
PFM : Power-fail compare mode register
PFT : Power-fail compare threshold value register
PM0 : Port mode register 0
PM2 : Port mode register 2
PM3 : Port mode register 3
PM4 : Port mode register 4
PM5 : Port mode register 5
PM6 : Port mode register 6
PM7 : Port mode register 7
PM12 : Port mode register 12
PM13 : Port mode register 13
PM14 : Port mode register 14
PR0H : Priority specify flag register 0H
PR0L : Priority specify flag register 0L
PR1H : Priority specify flag register 1H
PR1L : Priority specify flag register 1L
PRM0 : Prescaler mode register 0
PRM2 : Prescaler mode register 2
PSW : Program status word
PU0 : Pull-up resistor option register 0
PU4 : Pull-up resistor option register 4
PU7 : Pull-up resistor option register 7
PU13 : Pull-up resistor option register 13
REC : Receive error counter
REDEF : Redefinition control register
RMES : Receive message register
RXB0 : Receive buffer register
RXS0 : Receive shift register

SAR : Successive approximation register
SGAM : Sound generator amplitude register
SGBR : Sound generator buzzer control register
SGCR : Sound generator control register
SIO30 : Serial I/O shift register 30
SIO31 : Serial I/O shift register 31
SYNC0 : Synchronisation control register 0
SYNC1 : Synchronisation control register 1
TCR : Transmit control register
TEC : Transmit error counter
TCL50 : Timer clock selection register 50
TCL51 : Timer clock selection register 51
TM0 : 16-bit timer register 0
TM2 : 16-bit timer register 2
TM50 : 8-bit counter 50
TM51 : 8-bit counter 51
TMC0 : 16-bit timer mode control register 0
TMC2 : 16-bit timer mode control register 2
TMC50 : 8-bit timer mode control register 50
TMC51 : 8-bit timer mode control register 51
TOC0 : 16-bit timer output control register
TXS0 : Transmit shift register
WDCS : Watchdog timer clock selection register
WDTM : Watchdog timer mode register
WTM : Watch timer mode control register

[Memo]

Appendix D Revision History

The following shows the revision history up to present. Application portions signifies the chapter of each edition.

Edition No.	Major items revised	Revised Sections

Appendix D Revision History

Edition No.	Major items revised	Revised Sections

[Memo]

NEC

Facsimile Message

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

From:

Name

Company

Tel. FAX

Address

Thank you for your kind support.

North America
NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

Korea
NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-551-0451

Japan
NEC Corporation
Semiconductor Solution Engineering Division
Technical Information Support Dept.
Fax: 044-548-7900

South America
NEC do Brasil S.A.
Fax: +55-11-889-1689

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-719-5951

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	❑	❑	❑	❑
Technical Accuracy	❑	❑	❑	❑
Organization	❑	❑	❑	❑

CS 95.12