To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

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

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

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

User’s Manual

µPD780816 Subseries

8-bit Single-Chip Microcontroller

µPD780814
µPD780816
µPD78F0818

Document No. U15251EE4V0UD00
Date Published September 2005

NOTES FOR CMOS DEVICES

① **VOLTAGE APPLICATION WAVEFORM AT INPUT PIN**

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

② **HANDLING OF UNUSED INPUT PINS**

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

③ **PRECAUTION AGAINST ESD**

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

④ **STATUS BEFORE INITIALIZATION**

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

⑤ **POWER ON/OFF SEQUENCE**

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current.

The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

⑥ **INPUT OF SIGNAL DURING POWER OFF STATE**

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements.

Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

- **The information in this document is current as of September, 2005. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".
  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

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

  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).

  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

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

(Note)

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

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

M8E 02. 11-1

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

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 America Inc.
Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Europe) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 1101
Fax: 0211-65 03 1327

Sucursal en España
Madrid, Spain
Tel: 091- 504 27 87
Fax: 091- 504 28 60

Succursale Française
Vélizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

Filiale Italiana
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

Branch The Netherlands
Eindhoven, The Netherlands
Tel: 040-244 58 45
Fax: 040-244 45 80

Branch Sweden
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

United Kingdom Branch
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

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.
Singapore
Tel: 65-6253-8311
Fax: 65-6250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

Preface

Readers

This manual has been prepared for engineers who want to understand the functions of the µPD780816 Subseries and design and develop its application systems and programs.

µPD780816 Subseries:
µPD780814(A), µPD780816(A), µPD78F0818(A)
µPD780814(A1), µPD780816(A1)
µPD780814(A2), µPD780816(A2)

Purpose

This manual is intended for users to understand the functions of the µPD780816 Subseries.

Organization

The µPD780816 subseries manual is separated into two parts: this manual and the instruction edition (common to the 78K/0 series).

µPD780816
Subseries
This Manual

- Pin functions
- Internal block functions
- Interrupt
- Other on-chip peripheral functions

78K/0 series
User's Manual
Instruction

- CPU functions
- Instruction set
- Explanation of each instruction

How to Read This Manual

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

- When using this manual as the manual for (A) products, (A1) products, and (A2) products:
  → Only the quality grade differs between (A), (A1) and (A2) products.
  Read the part number as follows:
  µPD780814 → µPD780814(A), µPD780814(A1), µPD780814(A2)
  µPD780816 → µPD780816(A), µPD780816(A1), µPD780816(A2)
  µPD78F0818 → µPD78F0818(A)

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

Related Documents

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

- **Related documents for µPD780816 Subseries**

Document name	Document No.	
	Japanese	English
µPD780816 Subseries User’s Manual	Planned	This manual
78K/0 Series User’s Manual-Instruction	U12326J	U12326E
78K/0 Series Instruction Table	U10903J	-
78K/0 Series Instruction Set	U10904J	U12326E
µPD780816 Subseries Special Function Register Table	-	-

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

Document name		Document No.	
		Japanese	English
RA78K Series Assembler Package	Operation	EEU-809	EEU-1399
	Language	EEU-815	EEU-1404
RA78K Series Structured Assembler Preprocessor		EEU-817	EEU-1402
CC78K Series C Compiler	Operation	EEU-656	EEU-1280
	Language	EEU-655	EEU-1284
CC78K/0 C Compiler	Operation	U11517J	-
	Language	U11518J	-
CC78K/0 C Compiler Application Note	Programming Note	EEA-618	EEA-1208
CC78K Series Library Source File		EEU-777	-
IE-78K0-NS-A		U14889J	U14889E
IE-78K0-NS-P04		U14514J	U14514E
IE-780818-NS-EM4		U14514J	U14514E
NP-64GK		-	-
SM78K0 System Simulator Windows™ Base	Reference	U15373J	U15373E
SM78K0 Series System Simulator	External part user open Interface	U15802J	U15802E
ID78K0-NS Integrated Debugger		U15185J	U15185E

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

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

- **Other Documents**

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

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

Legend

Symbols and notation are used as follows:

Weight in data notation : Left is high-order column, right is low order column

Active low notation : $\overline{\text{xxx}}$ (pin or signal name is over-scored) or
/xxx (slash before signal name)

Memory map address: : High order at high stage and low order at low stage

Note : Explanation of (Note) in the text

Caution : Item deserving extra attention

Remark : Supplementary explanation to the text

Numeric notation : Binary . . . xxxx or xxxB
Decimal . . . xxxx
Hexadecimal . . . xxxxH or 0x xxxx

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

K (kilo): 2^{10} = 1024
M (mega): $2^{20} = 1024^2$ = 1,048,576
G (giga): $2^{30} = 1024^3$ = 1,073,741,824

Table of Contents

List of Figures

List of Tables

Chapter 1 Outline (µPD780816 Subseries)

1.1 Features

- Internal memory

Item Part Number	Program Memory (ROM)	Data Memory		Package
		Internal high-speed RAM	Internal Expansion RAM	
µPD780814	32 Kbytes	1024 bytes	480 bytes	64-pin plastic QFP (fine pitch)
µPD780816	48 Kbytes	1024 bytes	480 bytes	64-pin plastic QFP (fine pitch)
µPD78F0818	59.5 Kbytes	1024 bytes	2016 bytes	64-pin plastic QFP (fine pitch)

- Instruction execution time can be changed from high speed (0.25 µs) to ultra low speed
- I/O ports: 46
- 8-bit resolution A/D converter: 12 channels
- Main clock monitor
- CAN-Interface
- Serial interface : 3 channels
- 3-wire mode : 1 channel
- 2-wire/3-wire mode : 1 channel
- UART mode : 1 channel
- Timer : 6 channels
- Supply voltage : V_{DD} = 4.0 to 5.5 V

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

1.2 Application

Body electronics, industrial electronics, Security unit etc.

1.3 Ordering Information

Part Number	Package	Internal ROM
µPD780814GK(A)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Mask ROM
µPD780814GK(A1)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Mask ROM
µPD780814GK(A2)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Mask ROM
µPD780816GK(A)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Mask ROM
µPD780816GK(A1)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Mask ROM
µPD780816GK(A2)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Mask ROM
µPD78F0818GK(A)-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Flash Memory

Remark: xxx indicates ROM code suffix.

1.4 Quality Grade

Part Number	Package	Quality Grade
µPD780814GK(A)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special
µPD780814GK(A1)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special
µPD780814GK(A2)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special
µPD780816GK(A)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special
µPD780816GK(A1)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special
µPD780816GK(A2)-xxx-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special
µPD78F0818GK(A)-9ET	64-pin plastic QFP (12 × 12 mm, resin thickness 1.2 mm)	Special

Remark: xxx indicates ROM code suffix.

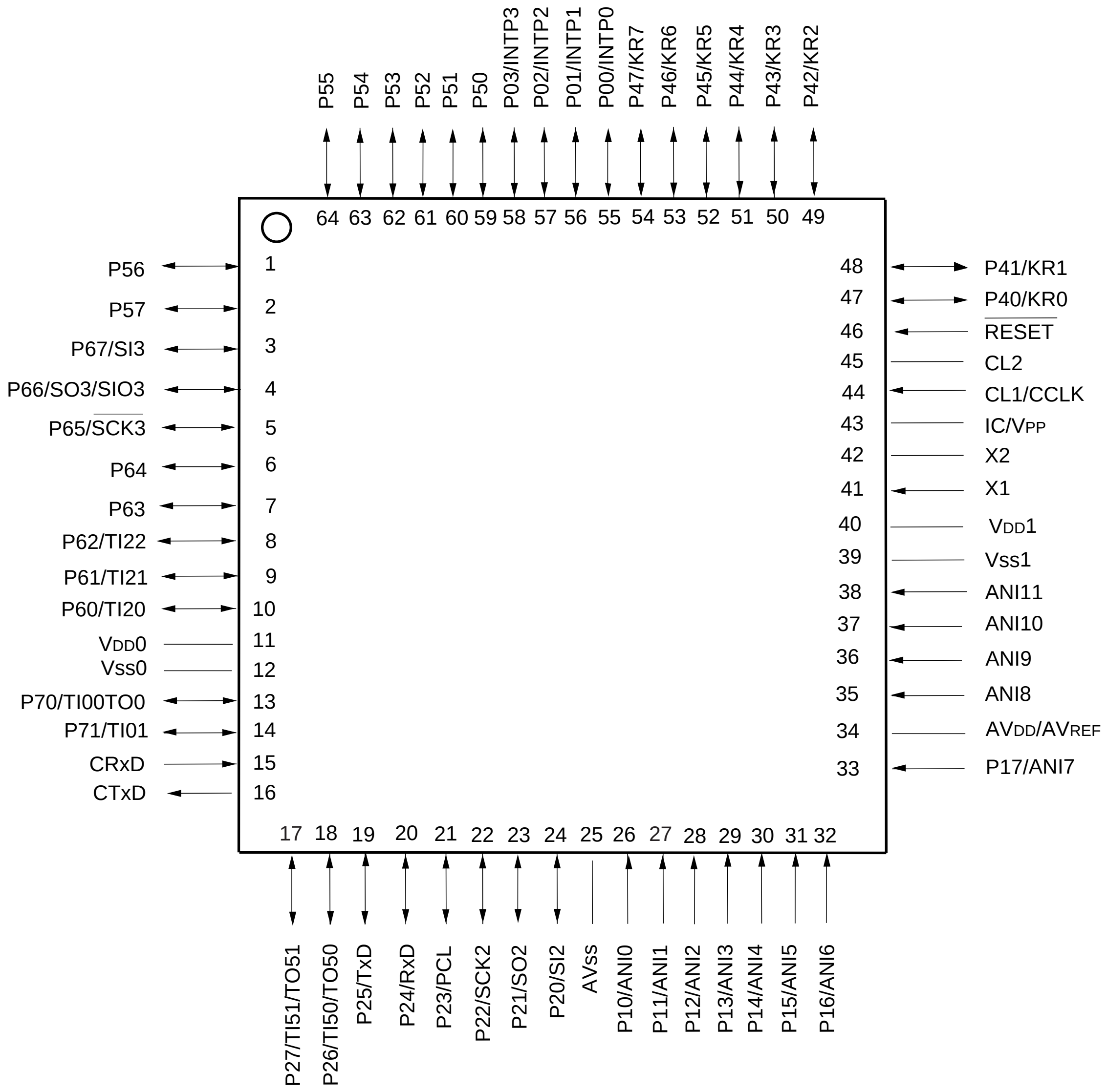
Please refer to "Quality Grades on NEC Semiconductor Device" (Document No. C11531E) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

1.5 Pin Configuration (Top View)

- 64-pin plastic QFP (12 × 12 mm)

µPD780814GK(A)- xxx - 9ET, µPD780814GK(A1)- xxx - 9ET, µPD780814GK(A2)- xxx - 9ET
µPD780816GK(A)- xxx - 9ET, µPD780816GK(A1)- xxx - 9ET, µPD780816GK(A2)- xxx - 9ET
µPD78F0818GK(A) - 9ET

Figure 1-1: Pin Configuration



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

2. AV_{DD}/AV_{REF} pin should be connected to V_{DD}.

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

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

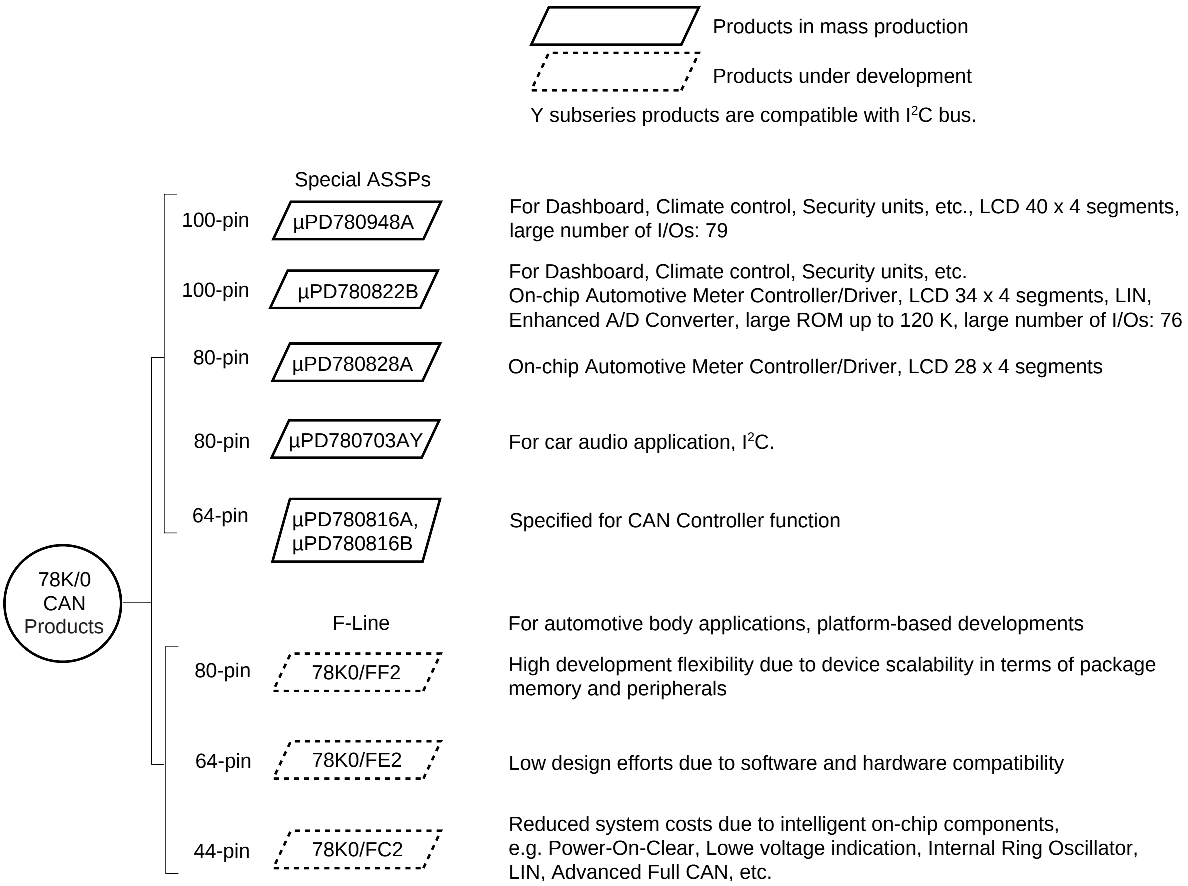
Pin Identifications

P00 to P03	:	Port 0	RXD	:	Receive Data
P10 to P17	:	Port 1	TXD	:	Transmit Data
P20 to P27	:	Port 2	PCL	:	Programmable Clock Output
P40 to P47	:	Port 4	X1, X2	:	Crystal (Main System Clock)
P50 to P57	:	Port 5	CL1, CL2		RC (Subsystem Clock)
P60 to P67	:	Port 6	$\overline{\text{RESET}}$	:	Reset
P70, P71	:	Port 7	ANI0 to ANI11	:	Analog Input
KR0 to KR7	:	Key Return Port	AV_{SS}	:	Analog Ground
INTP0 to INTP3	:	Interrupt from Peripherals	AV_{DD}/AV_{REF}	:	Power Supply and
TI00, TI01	:	Timer Input		:	Analog Reference Voltage
TI50, TI51	:	Timer Input	V_{DD0}, V_{DD1}	:	Power Supply
TI20 to TI22	:	Timer Input	V_{PP}	:	Programming Power Supply
TO00, TO51, TO52	:	Timer Output	V_{SS0}, V_{SS1}	:	Ground
CRXD	:	CAN Receive Data	IC	:	Internally Connected
CTXD	:	CAN Transmit Data			
CCLK	:	CAN Clock			
SI2, SI3	:	Serial Input			
SO2, SO3	:	Serial Output			
$\overline{\text{SCK2}}$, $\overline{\text{SCK3}}$	:	Serial Clock			
SIO3	:	Serial Input/Output			
SIO3	:	Serial Input/Output			

1.6 78K/0 CAN Products Expansion

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

Figure 1-2: 78K/0 CAN Products Expansion


Products in mass production
Products under development
Y subseries products are compatible with I²C bus.
78K/0 CAN Products
Special ASSPs
100-pin
µPD780948A
For Dashboard, Climate control, Security units, etc., LCD 40 x 4 segments, large number of I/Os: 79
100-pin
µPD780822B
For Dashboard, Climate control, Security units, etc.
On-chip Automotive Meter Controller/Driver, LCD 34 x 4 segments, LIN, Enhanced A/D Converter, large ROM up to 120 K, large number of I/Os: 76
80-pin
µPD780828A
On-chip Automotive Meter Controller/Driver, LCD 28 x 4 segments
80-pin
µPD780703AY
For car audio application, I²C.
64-pin
µPD780816A, µPD780816B
Specified for CAN Controller function
F-Line
For automotive body applications, platform-based developments
80-pin
78K0/FF2
High development flexibility due to device scalability in terms of package memory and peripherals
64-pin
78K0/FE2
Low design efforts due to software and hardware compatibility
44-pin
78K0/FC2
Reduced system costs due to intelligent on-chip components,
e.g. Power-On-Clear, Lowe voltage indication, Internal Ring Oscillator,
LIN, Advanced Full CAN, etc.

The major functional differences between the subseries are shown below.

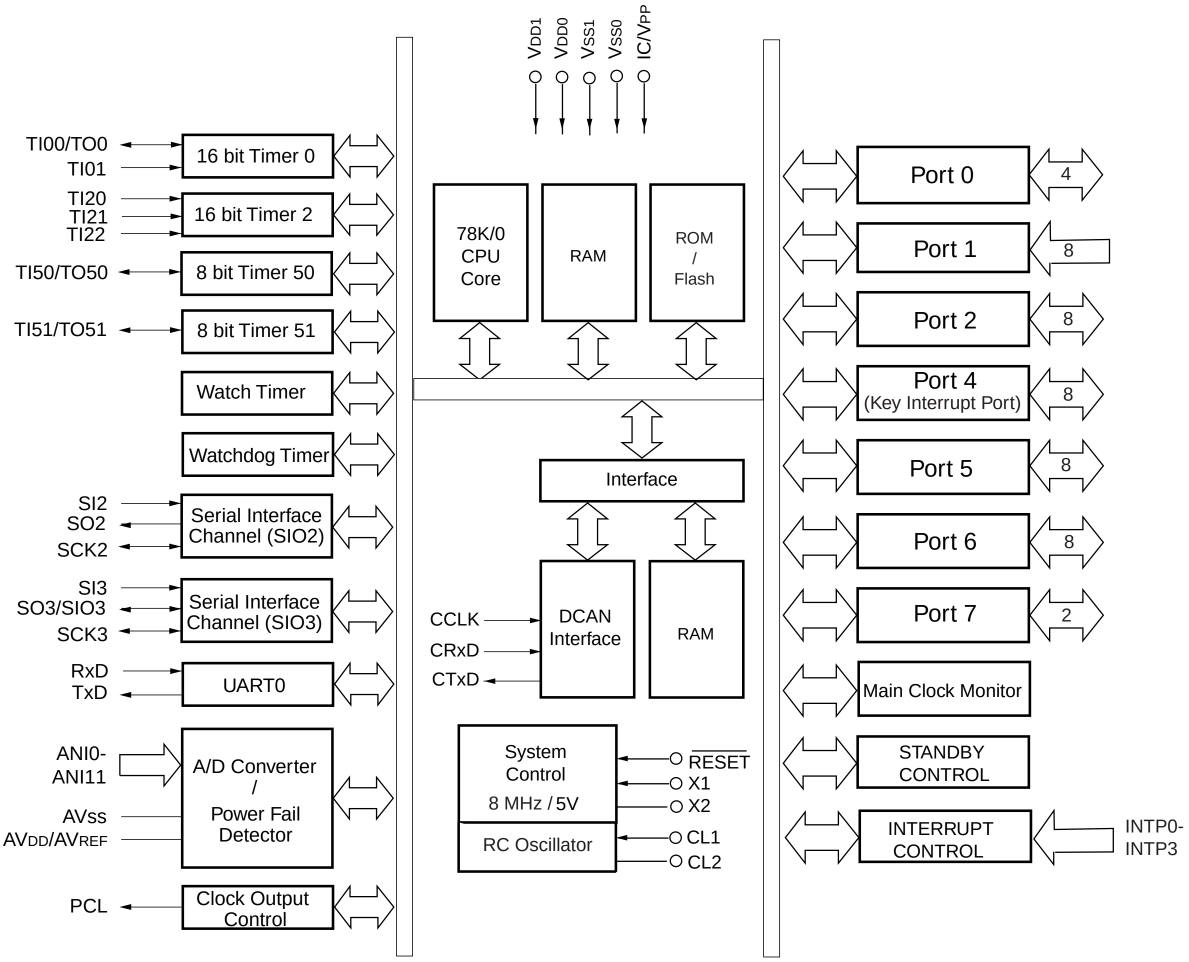
Table 1-1: The major functional differences between the subseries

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

Note: Under development (Target values)

1.7 Block Diagram

Figure 1-3: Block Diagram



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

1.8 Overview of Functions

Table 1-2: Overview of Functions

Item \ Part Number		µPD780814	µPD780816	µPD78F0818
Internal memory	ROM	32 Kbytes Mask ROM	48 Kbytes Mask ROM	59.5 Kbytes Flash EE
	Internal high-speed RAM	1024 bytes	1024 bytes	1024 bytes
	Internal expansion RAM	480 bytes	480 bytes	2016 bytes
Memory space		64 Kbytes		
General registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)		
Instruction cycle		On-chip instruction execution time selection 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)		
Main system clock		0.25 µs/0.5 µs/1 µs/2 µs/4 µs (at 8 MHz)		
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, boolean operation) • BCD adjustment, etc.		
I/O ports		Total: 46 • CMOS Input: 8 • CMOS I/O: 36		
A/D converter		8 bit resolution × 12 channels		
Serial Interface		• 3-wire mode: 1 channel • 2-wire/3-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 outputs (16-bit PWM × 1, 8-bit PWM output × 2)		
Main Clock Monitor		Main clock oscillation fail detection		
Clock output		8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz, 62.5 kHz at main system clock of 8 MHz		
CAN		1 channel		
Vectored interrupts		Maskable interrupts	Internal: 22 External: 4	
		Non-maskable interrupts	Internal: 1	
		Software interrupts	Internal: 1	
Supply voltage		V_{DD} = 4.0 V to 5.5 V		
Package		64-pin plastic QFP (12 × 12 mm)		

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 1-3: Differences between Flash and Mask ROM version

	Flash Version	Mask ROM Version
ROM	Flash EEPROM	Mask ROM
Internal ROM capacity	µPD78F0818: 59.5 Kbytes	µPD780814: 32 Kbytes µPD780816: 48 Kbytes
ICC Pin	None	Available
V_{PP} Pin	Available	None

[MEMO]

Chapter 2 Pin Function (µPD780816 Subseries)

2.1 Pin Function List

Normal Operating Mode Pins / Pin Input/Output Types

Table 2-1: Pin Input/Output Types

Input/Output	Pin Name	Function	Alternate Function	After Reset
Input/Output	P00	Port 0 4-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be connected by software bit-wise	INTP0	Input
	P01		INTP1	Input
	P02		INTP2	Input
	P03		INTP3	Input
Input	P10-P17	Port 1 8-bit input port	ANI0-ANI7	Input
Input/Output	P20	Port 2 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 specified bit-wise	SI2	Input
	P21		SO2	Input
	P22		SCK2	Input
	P23		PCL	Input
	P24		RXD	Input
	P25		TXD	Input
	P26		TI50/TO50	Input
	P27		TI51/TO51	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	KR0-KR7	Input
Input/Output	P50-P57	Port 5 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 specified bit-wise	-	Input
Input/Output	P60	Port 6 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	TI20	Input
	P61		TI21	
	P62		TI22	
	P63		-	
	P64		-	
	P65		SCK3	
	P66		SO3/SIO2	
	P67		SI3	
Input/Output	P70	Port 7 2-bit input / output port input / output mode can be specified bit-wise If used as an input port, a pull-up resistor can be specified bit-wise	TI00/TO00	Input
	P71		TI01	

2.2 Non-Port Pins

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

Pin Name	Input/Output	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
SI2	Input	Serial interface serial data input	Input	P20
SI3	Input	Serial interface serial data input	Input	P67
SO2	Output	Serial interface serial data output	Input	P21
SO3	Output	Serial interface serial data output	Input	P66
$\overline{\text{SCK2}}$	Input/Output	Serial interface serial clock input / output	Input	P22
$\overline{\text{SCK3}}$	Input/Output	Serial interface serial clock input / output	Input	P65
SIO3	Input/Output	Serial interface serial data input / output	Input	P66
RXD	Input	Asynchronous serial interface serial data input	Input	P24
TXD0	Output	Asynchronous serial interface serial data output	Input	P25
CRXD	Input	CAN serial data input	Input	-
CTXD	Output	CAN serial data output	Output	-
CCLK	Input	CAN serial clock input	-	CL1
TI00	Input	External signal input to 16-bit timer TMO	Input	P70/TO0
TI01				
TI20		Capture trigger input		P64
TI21		Capture trigger input		P65
TI22		Capture trigger input		P90/S23
TI50		External count clock input to 8-bit timer (TM50)		P34/TO50/S27
TI51		External count clock input to 8-bit timer (TM51)		P91/TO51/S22
TO0	Output	16-bit timer output	Input	P70/TI00
TO50		8-bit timer output (also used for PWM output)		P26/TI50
TO51		8-bit timer output (also used for PWM output)		P27/TI51
PCL	Output	Clock output (for main system clock trimming)	Input	P23
ANI0 - ANI7	Input	AD converter analog input	Input	P10-P17
ANI8 - ANI11	Input	AD converter analog input	Input	-
AV_{DD} /AV_{REF}	-	AD converter reference voltage input. Power supply of the AD converter.	-	-
AV_{SS}	-	AD converter ground potential. Connect to V_{SS}	-	-
$\overline{\text{RESET}}$	Input	System reset input	-	-
X1	-	Crystal connection for main system clock	-	-
X2	-	Crystal connection for main system clock	-	-
CL1	Input	RC connection for subsystem clock	-	CCLK
CL2	-	RC connection for subsystem clock	-	-

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

Pin Name	Input/Output	Function	After Reset	Alternate Function Pin
V_{DD0},V_{DD1}	-	Positive power supply	-	-
V_{SS0},V_{SS1}	-	Ground potential	-	-
V_{PP}	-	High voltage supply for flash programming (only flash version)	-	IC
IC	-	Internal connection. Connect directly to V_{SS} (only Mask ROM version)	-	V_{PP}

2.3 Description of Pin Functions

2.3.1 P00 to P03 (Port 0)

This is an 4-bit input/output port. Besides serving as input/output port the external interrupt input is implemented.

(1) Port mode

P00 to P03 function as input/output ports. P00 to P03 can be specified for input or output bit-wise with a port mode register. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistor option register 0.

(2) Control Mode

In this mode, this port functions as external interrupt input.

INTP0 to INTP3

INTP0 to INTP3 are external input pins which can specify valid edges (rising, falling or rising and falling) of this external interrupt pins.

2.3.2 P10 to P17 (Port 1)

These pins constitute ab 8-bit input only port. In addition, they are also used to input A/D converter analog signals. The following operating modes can be specified bit-wise.

(1) Port mode

In this mode, P10 to P17 function as an 8-bit input only port.

(2) (2) Control mode

In this mode, P10 to P17 function as A/D converter analog input pins (ANI0 to ANI7).

2.3.3 P20 to P27 (Port 2)

These are 8-bit input/output ports. Besides serving as input/output ports, they function as data input/ output to/from and clock input/out of the serial interface CSI or data transmit and receive of the UART. Additionally they function as timer input/output and processor clock output

(1) Port mode

In this mode, P20 to P27 function as an 8-bit input/output port. P20 to P27 can be specified for input or output bit wise with a port mode register. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistor option register 2.

(2) Control mode

These ports function as timer input/output, as serial interface data input/output, serial clock input/ output, UART transmit and receive and as processor clock output.

(a) SI2, SO2

Serial interface serial data input/output pins.

(b) SCK2

Serial interface serial clock input/output pins.

(c) TI50
Pin for external count clock input to 8-bit timer/event counter 50.

(d) TO50
Pin for output of the 8-bit timer/event counter 50.

(e) TI51
Pin for external count clock input to 8-bit timer/event counter 51.

(f) TO51
Pin for output of the 8-bit timer/event counter 51.

(g) RXD, TXD
Asynchronous serial interface data input/output pins.

(h) PCL
Clock output pin.

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

2.3.4 P40 to P47 (Port 4)

This is an 8-bit input/output port. Besides serving as input/output port, they function as key return signal.
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 4. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistor option register 4.

(2) Control mode

These ports function as key return signal by detection of a low level at this port pins.

2.3.5 P50 to P57 (Port 5)

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 5. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistor option register 5.

2.3.6 P60 to P65 (Port 6)

These are 6-bit input/output ports. Beside serving as input/output ports, they function as timer capture input.
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 or output ports with port mode register 6. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistor option register 6.

(2) Control mode

These ports function as timer capture input.

(a) TI20, TI21, TI22
Pins for external capture trigger input to the 16-bit timer capture registers of TM2.

(b) SI3, SO3, SIO3
Serial interface serial data input/output pins.

(c) SCK3
Serial interface serial clock input/output pin.

2.3.7 P70, P71 (Port 7)

These are 2-bit input/output ports. Besides serving as input/output ports, they function timer input/output.
The following operating modes can be specified bit-wise or byte-wise.

(1) Port mode

These ports function as 2-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 7. When they are used as input ports, pull-up resistors can be connected to them by defining the pull-up resistor option register 7.

(2) Control mode

In this mode, these port functions as external count clock and capture trigger signal input of the 16-bit timer and as timer signal output.

(a) TI00
Pin for external count clock input to the 16-bit timer/event counter and pin for capture trigger signal input to the 16-bit timer/event counter capture register.

(b) TI01
Pin for external count clock input to the 16-bit timer/event counter and pin for capture trigger signal input to the 16-bit timer/event counter capture register.

TO0
Pin for output of the 16-bit timer/event counter.

2.3.8 CTXD

This pin functions as CAN-controller transmit output.

2.3.9 CRXD

This pin functions as CAN-controller receive input.

2.3.10 CCLK

This pin functions as CAN-controller clock supply input.

2.3.11 ANI0 to ANI11

These pins constitute an analog input only port.

2.3.12 AV_{DD}/AV_{REF}

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}.

2.3.13 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.

2.3.14 $\overline{\text{RESET}}$

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

2.3.15 X1 and X2

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

2.3.16 CL1 and CL2

RC connect pins for sub system clock oscillation.

2.3.17 V_{DD0}, V_{DD1}

V_{DD0} is the positive power supply pin for ports.
V_{DD1} is the positive power supply pin for blocks other than ports.

2.3.18 V_{SS0}, V_{SS1}

V_{SS0} is the ground pin for ports.
V_{SS1} is the ground pin for blocks other than ports.

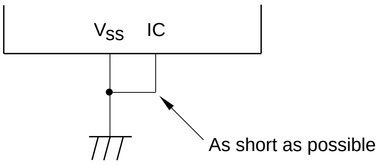
2.3.19 V_{PP} (µPD78F0818 only)

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

2.3.20 IC (Mask ROM version only)

The IC (Internally Connected) pin is provided to set the test mode to check the µPD780814 and µPD780816 at delivery. Connect it directly to the V_{SS} with the shortest possible wire in the normal operating mode. When a voltage difference is produced between the IC pin and V_{SS} 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



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

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

The input/output circuit type of each pin and recommended connection of unused pins are shown in the following table.
For the input/output circuit configuration of each type, see tTable 2-3, “Types of Pin Input/Output Circuits,” on page 43.

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

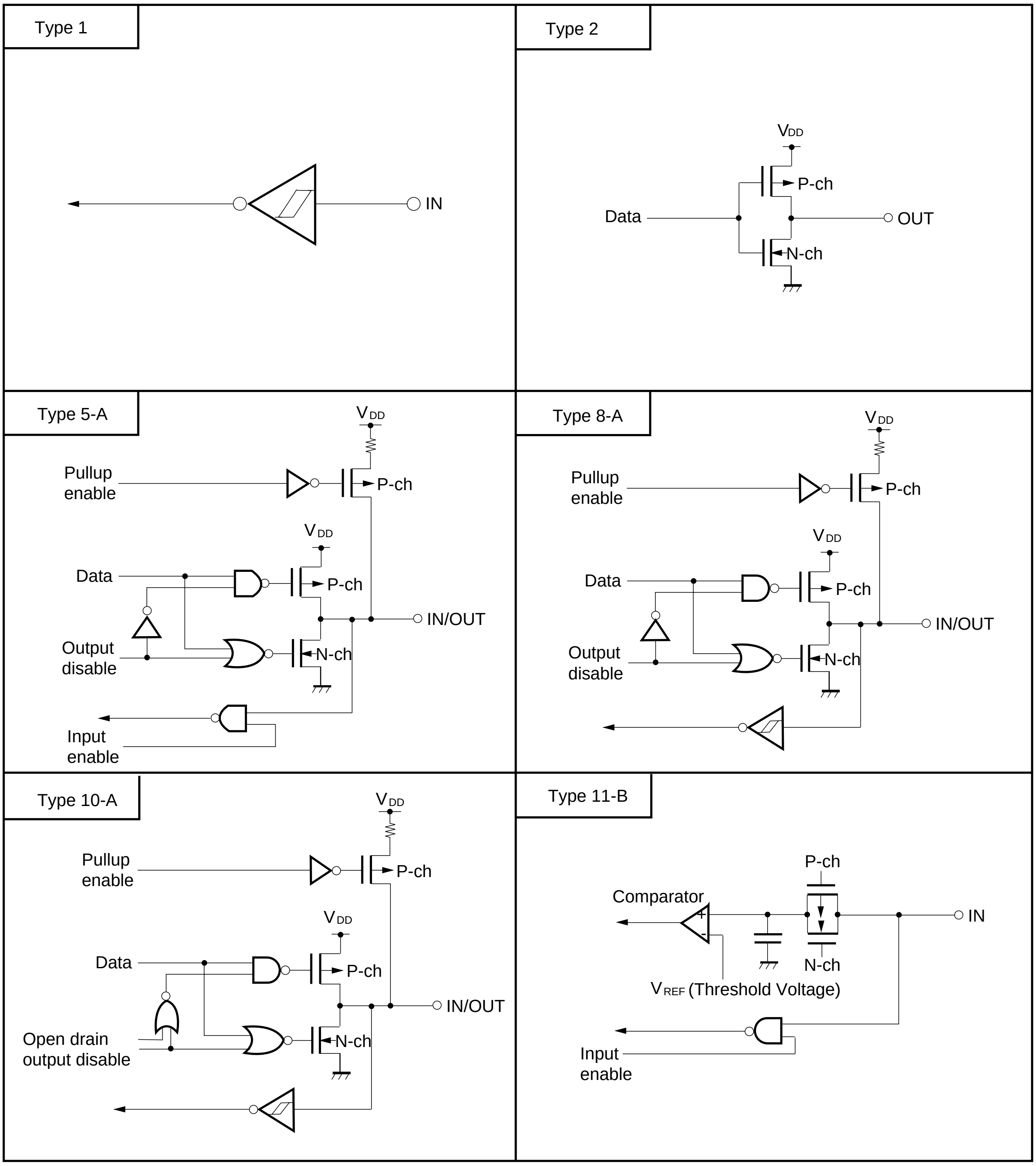
Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
P00/INTP0	8-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P01/INTP1			
P02/INTP2			
P03/INTP3			
P10/ANI0	11-B	I	Connect directly to V_{DD} or V_{SS}
P11/ANI1			
P12/ANI2			
P13/ANI3			
P14/ANI4			
P15/ANI5			
P16/ANI6			
P17/ANI7			
ANI8**Note**			
ANI9**Note**			
ANI10**Note**			
ANI11**Note**			
P20/SI2	10-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P21/SO2			
P22/SCK2			
P23/PCL	5-A		
P24/RXD	8-A		
P25/TXD	5-A		
P26/TI50/TO50	8-A		
P27/TI51/TO51	8-A		
P40/KR0	5-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P41/KR1			
P42/KR2			
P43/KR3			
P44/KR4			
P45/KR5			
P46/KR6			
P47/KR7			

Note: ANI8 to ANI11 have the same input/output circuit like P10/ANI0 to P17/ANI7, but the input port function of ANI8 to ANI11 is not implemented.

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

Pin Name	Input/Output Circuit Type	I/O	Recommended Connection for Unused Pins
P50	5-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P51			
P52			
P53			
P54			
P55			
P56			
P57			
P60/TI20	8-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P61TI21			
P62/TI22			
P63			
P64			
P65/$\overline{\text{SCK3}}$			
P66/SO3/SIO3			
P67/SI3			
P70/TI00/TO00	8-A	I/O	Input: Connect to V_{DD} or V_{SS} via a resistor individually. Output: Leave open.
P71/TI01			
CRXD	1	I	Connect to V_{DD} or V_{SS} via a resistor individually
CTXD	2	O	Leave open
CL1/CCLK	-	I	GND
CL2	-	-	Leave open
$\overline{\text{RESET}}$	1	-	-
AV_{DD} / AV_{REF}	-	-	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



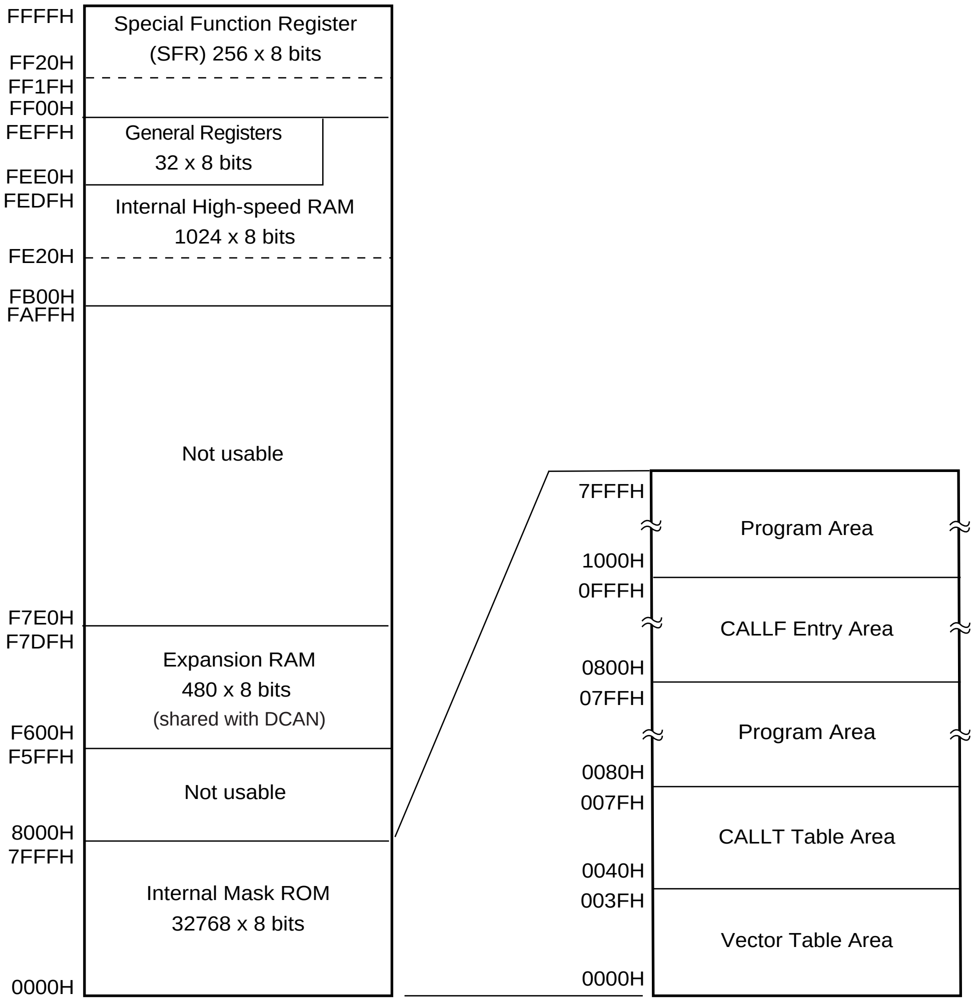
[MEMO]

Chapter 3 CPU Architecture

3.1 Memory Space

The memory map of the µPD780814 is shown in Figure 3-1.

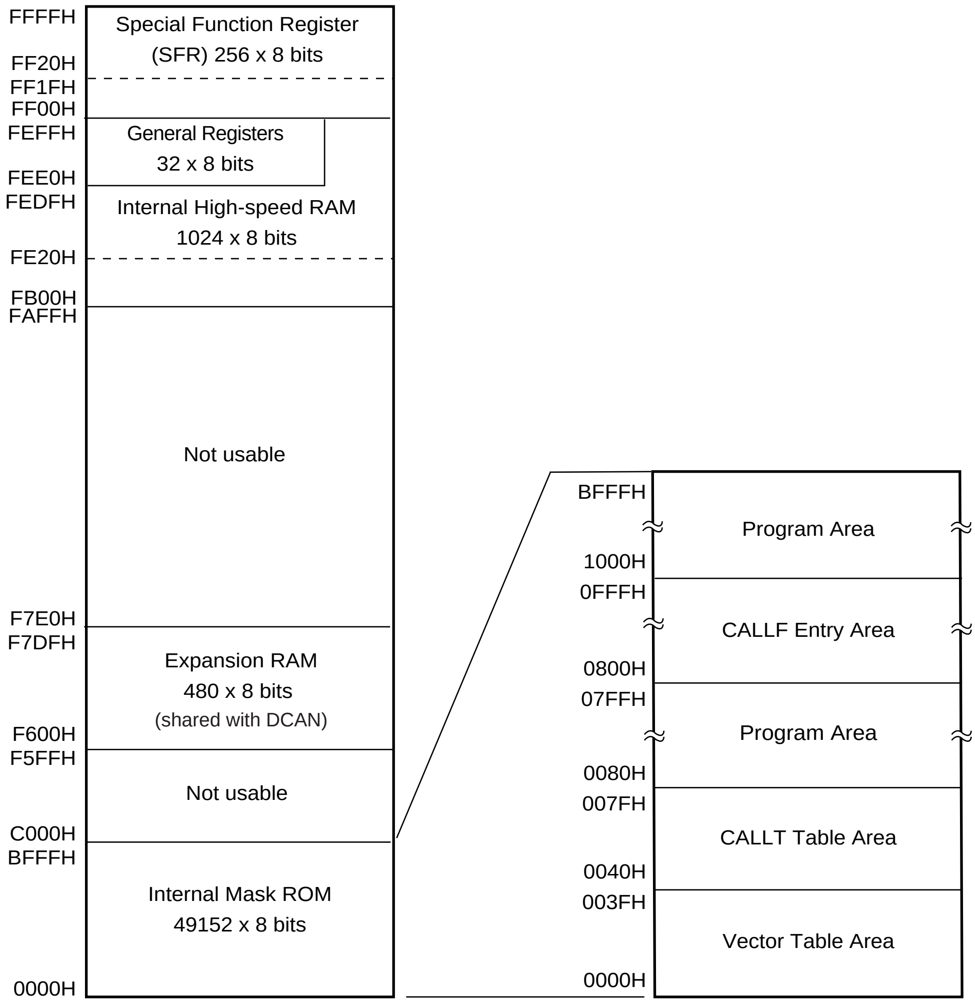
Figure 3-1: Memory Map 0f the µPD780814



Note: In the expansion RAM between F600H and F7DFH it is **not** possible to do code execution.

The memory map of the µPD780816 is shown in Figure 3-2.

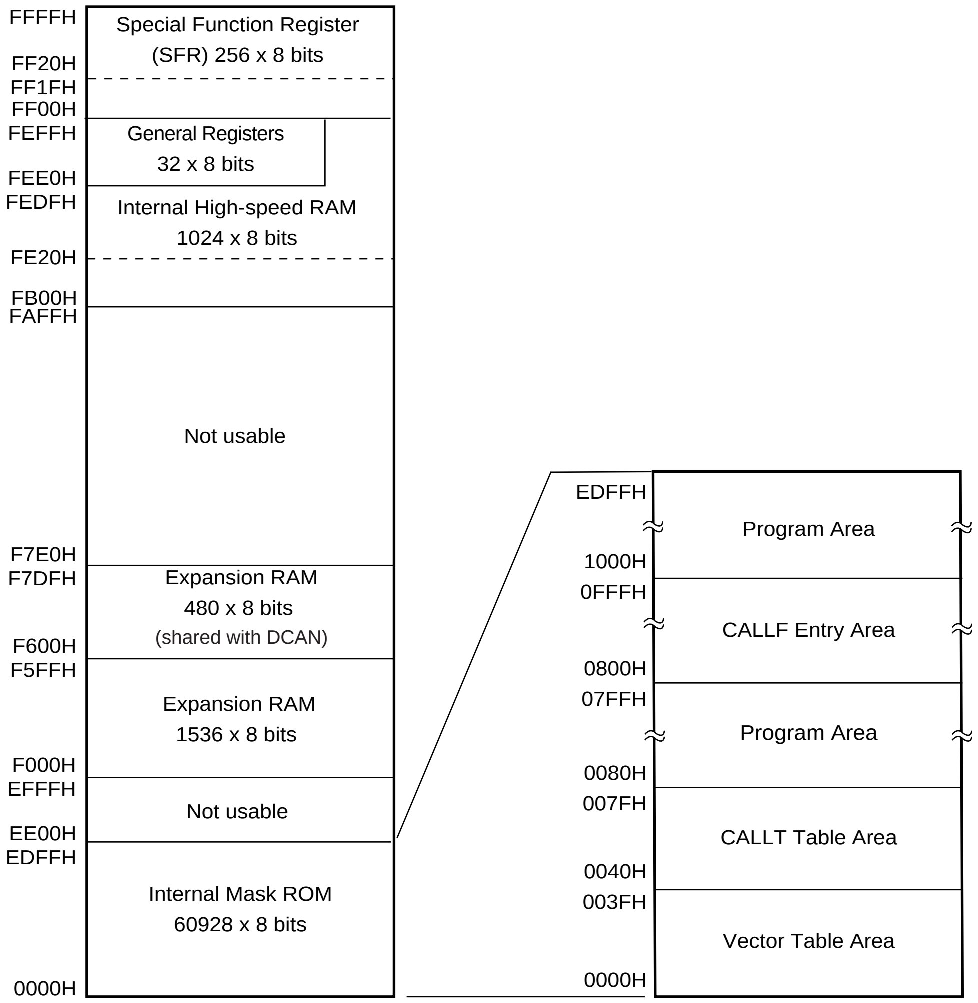
Figure 3-2: Memory Map of the µPD780816



Note: In the expansion RAM between F600H and F7DFH it is **not** possible to do code execution.

The memory map of the µPD78F0818 is shown in Figure 3-3.

Figure 3-3: Memory Map of the µPD78F0818



Notes: 1. In the expansion RAM between F000H and F5FFH it is possible to do code execution.

2. In the expansion RAM between F600H and F7DFH it is **not** possible to do code execution.

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 µPD780816 Subseries has various size of internal ROMs or Flash EPROM as shown below.

Table 3-1: Internal ROM Capacities

Part Number	Internal ROM	
	Type	Capacity
µPD780814	Mask ROM	32768 × 8-bits
µPD780816	Mask ROM	49152 × 8-bits
µPD78F0818	Flash EEPROM	60928 × 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	INTCE
001AH	INTCR
001CH	INTCT0
001EH	INTCT1
0020H	INTCSI20
0022H	INTSER
0024H	INTSR
0026H	INTST
0028H	INTTM00
002AH	INTTM01
002CH	INTTM50
002EH	INTTM51
0032H	INTWTI
0034H	INTWT
0036H	INTKR
0038H	INTCSI30
003EH	BRK

(2) CALLT instruction table area

The 64-byte area 0040H to 007FH can store the subroutine entry address of an 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 µPD780816 Subseries units incorporate the following RAMs.

(1) Internal high-speed RAM

Table 3-3: Internal high-speed RAM

Device	Internal High Speed RAM
µPD780814	1024 × 8 bits (FB00H to FEFFH)
µPD780816	1024 × 8 bits (FB00H to FEFFH)
µPD78F0818	1024 × 8 bits (FB00H to FEFFH)

The 32-byte area FEE0H to FEFF is allocated with four general purpose register banks composed of eight 8-bit registers.
The internal high-speed RAM has to be used as a stack memory.

(2) Internal expansion RAM

Table 3-4: Internal expansion RAM (including sharing with DCAN)

Device	Internal Expansion RAM
µPD780814	480 × 8 bits (F600H to F7DFH)
µPD780816	480 × 8 bits (F600H to F7DFH)
µPD78F0818	2016 × 8 bits (F000H to F7DFH)

3.1.3 Special function register (SFR) area

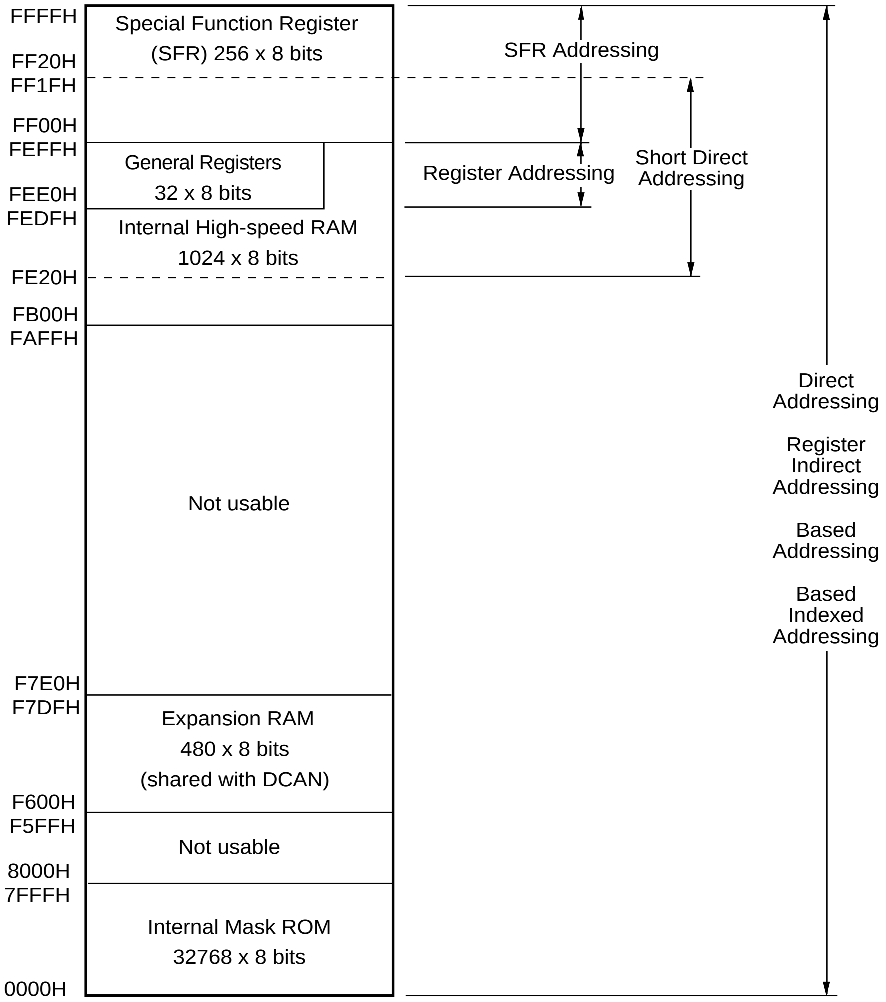
An on-chip peripheral hardware special function register (SFR) is allocated in the area FF00H to FFFFH. (Refer to **Table 3-5, “Special Function Register List,” on page 62**).

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

3.1.4 Data memory addressing

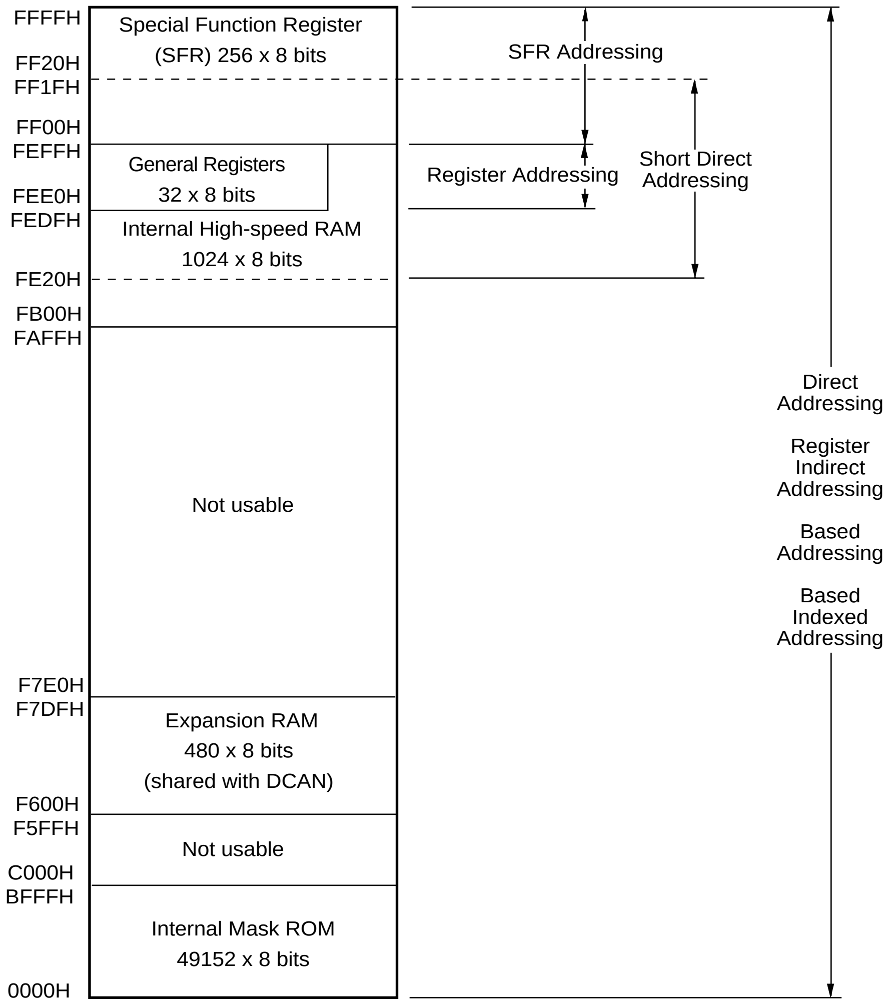
The µPD780816 Subseries is provided with a verity of addressing modes which take account of memory manipulability, etc. Special addressing methods are possible to meet the functions of the special function registers (SFRs) and general registers. The data memory space is the entire 64K-byte space (0000H to FFFFH).
Figures 3-4 to 3-6 show the data memory addressing modes.
For details of addressing, refer to **3.4 ”Operand Address Addressing” on page 69**.

Figure 3-4: Data Memory Addressing of µPD780814



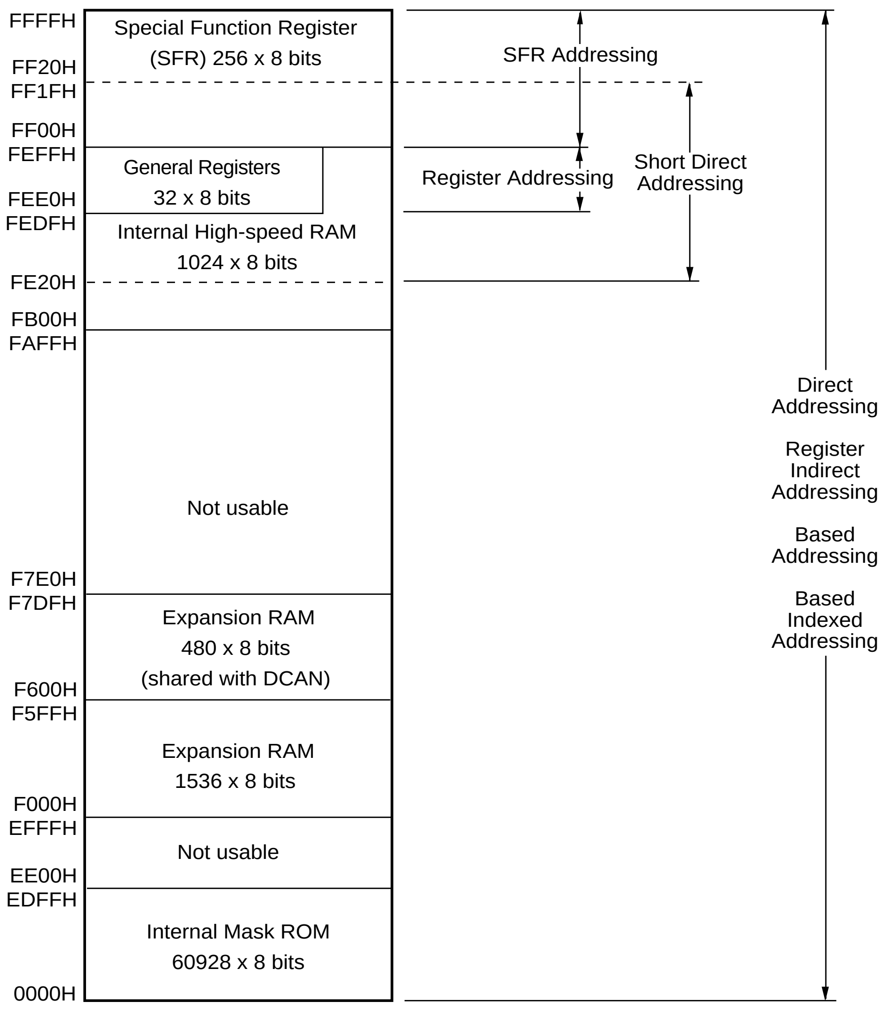
Note: In the expansion RAM between F600H and F7DFH it is **not** possible to do code execution.

Figure 3-5: Data Memory Addressing of µPD780816



Note: In the expansion RAM between F600H and F7DFH it is **not** possible to do code execution.

Figure 3-6: Data Memory Addressing of µPD78F0818



Notes: 1. In the expansion RAM between F000H and F5FFH it is possible to do code execution.

2. In the expansion RAM between F600H and F7DFH it is **not** possible to do code execution.

3.2 Processor Registers

The µPD780816 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.

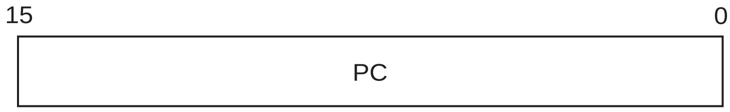
(1) Program counter (PC)

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 3-7: Program Counter Configuration



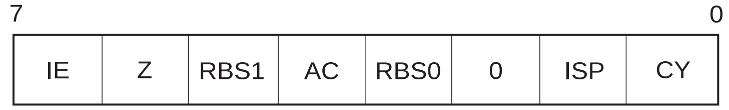
(2) Program status word (PSW)

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 3-8: Program Status Word Configuration



(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 acknowledge able maskable vectored interrupts. When 0, acknowledgment of the vectored interrupt request specified to low-order priority with the priority specify flag registers (PR0L, PR0H, PR1L and PR1H) is disabled. Whether an actual interrupt request is acknowledged or not is controlled with the interrupt enable flag (IE).

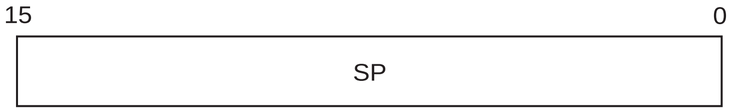
(f) Carry flag (CY)

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 3-9: 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 3-10 and 3-11.

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

Figure 3-10: Data to be Saved to Stack Memory

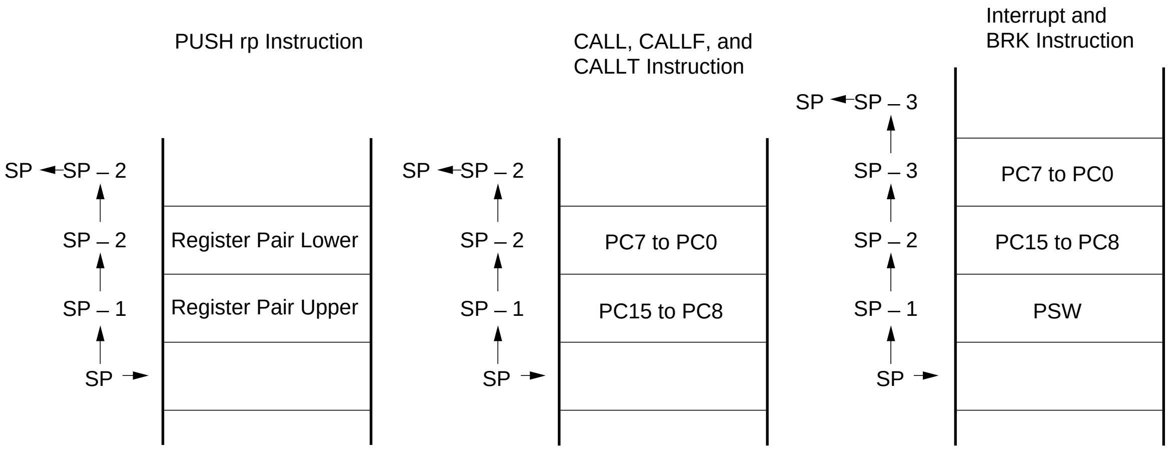
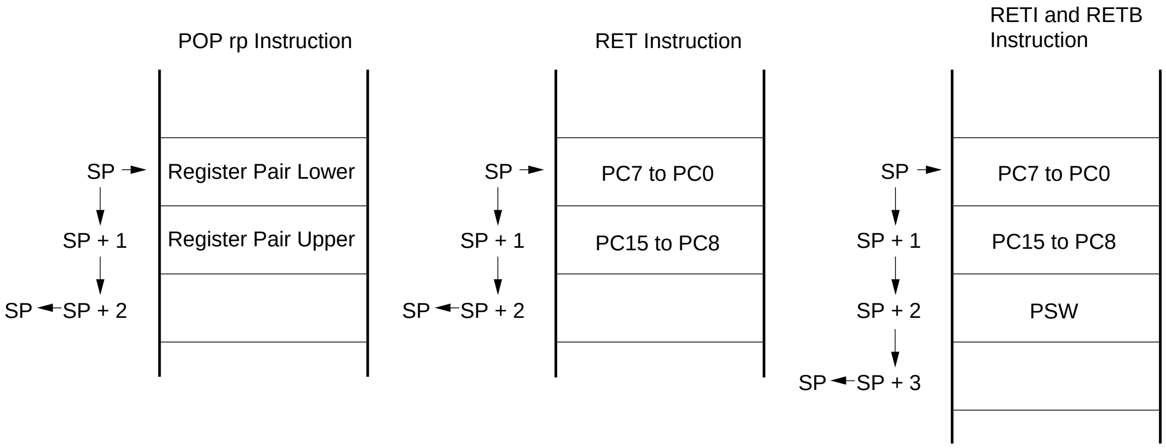


Figure 3-11: Data to be Reset to Stack Memory

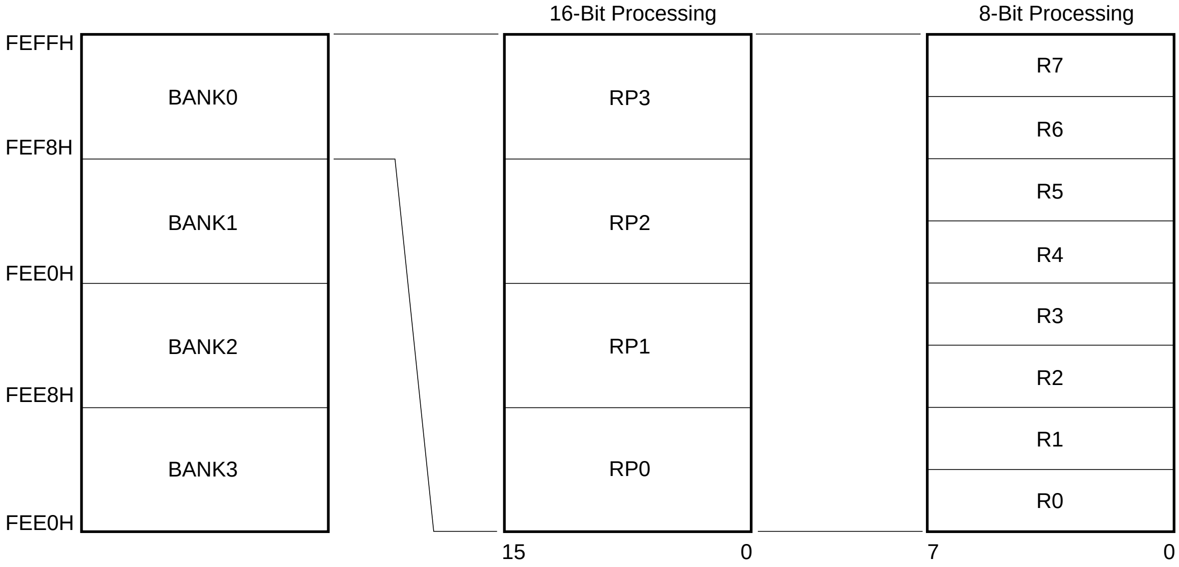


3.2.2 General registers

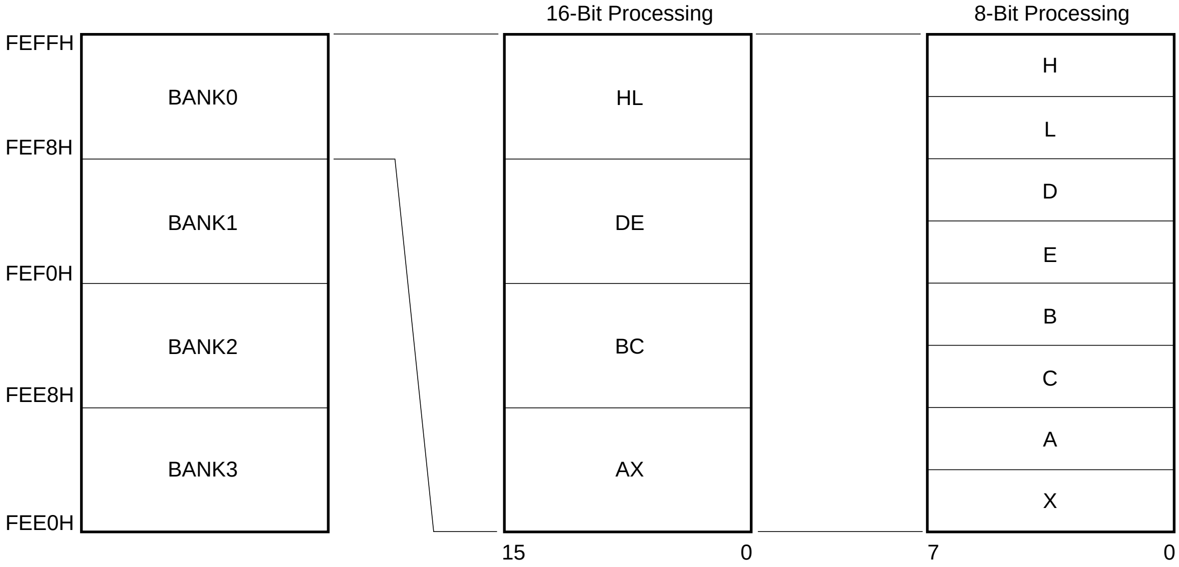
A general register is mapped at particular addresses (FEE0H to FEFFH) of the data memory. It consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).
Each register can also be used as an 8-bit register. Two 8-bit registers can be used in pairs as a 16-bit register (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 3-12: 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 3-5, “Special Function Register List,” on page 62 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 3-5: Special Function Register List (1/3)

Address	SFR Name	Symbol		R/W	Manipulation Bit Unit			After Reset
					1-bit	8-bit	16-bit	
FF00H	Port 0	P0		R/W	×	×	-	00H
FF01H	Port 1	P1		R	×	×	-	00H
FF02H	Port 2	P2		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
FF10H	16-bit timer/counter register 0	TM0	TM0L	R	-	-	×	00H
FF11H			TM0H					
FF12H	8-bit timer register 50	TM50		R	-	×	-	00H
FF13H	8-bit timer register 51	TM51		R	-	×	-	00H
FF14H	16-bit capture/compare register 00	CR00	CR00L	R/W	-	-	×	00H
FF15H			CR00H					
FF16H	16-bit capture/compare register 01	CR01	CR01L	R/W	-	-	×	00H
FF17H			CR01H					
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 20	SIO20		R/W	-	×	-	00H
FF20H	Port mode register 0	PM0		R/W	×	×	-	FFH
FF22H	Port mode register 2	PM2		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
FF30H	Pull-up resistor option register 0	PU0		R/W	×	×	-	00H
FF32H	Pull-up resistor option register 2	PU2		R/W	×	×	-	00H
FF34H	Pull-up resistor option register 4	PU4		R/W	×	×	-	00H
FF35H	Pull-up resistor option register 5	PU5		R/W	×	×	-	00H
FF36H	Pull-up resistor option register 6	PU6		R/W	×	×	-	00H
FF37H	Pull-up resistor option register 7	PU7		R/W	×	×	-	00H
FF40H	Clock output select register	CKS		R/W	×	×	-	00H
FF41H	Watch timer mode register	WTM		R/W	×	×	-	00H
FF42H	Watchdog timer clock selection register	WDCS		R/W	×	×	-	00H
FF47H	Key return mode register	KRM		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
FF50H	Flash programming mode control register	FLPMC		R/W	×	×	-	08H
FF51H	Self-programming and oscillation control register	SPOC		R/W	×	×	-	08H
FF52H	Port function register 2	PF2		R/W	×	×	-	00H
FF60H	16-bit timer mode control register 0	TMC0		R/W	×	×	-	00H

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

Address	SFR Name	Symbol		R/W	Manipulation Bit Unit			After Reset
					1-bit	8-bit	16-bit	
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	-	-	×	0000H
FF69H			TM2H					
FF6AH	16-bit capture register 20	CR20	CR20L	R	-	-	×	0000H
FF6BH			CR20H					
FF6CH	16-bit capture register 21	CR21	CR21L	R	-	-	×	0000H
FF6DH			CR21H					
FF6EH	16-bit capture register 22	CR22	CR22L	R	-	-	×	0000H
FF6FH			CR22H					
FF70H	8-bit timer mode control register 50	TMC50		R/W	×	×	-	00H
FF71H	Timer clock selection register 50	TCL50		R/W	-	×	-	00H
FF74H	8-bit timer mode control register 51	TMC51		R/W	×	×	-	00H
FF75H	Timer clock selection register 51	TCL51		R/W	-	×	-	00H
FF77H	Clock monitor mode register	CLM		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 register	PFM		R/W	×	×	-	00H
FF9BH	Power fail comparator threshold register	PFT		R/W	-	×	-	00H
FF9CH	D/A converter channel 0 mode register[Note]	DAM0		R/W	×	×	-	00H
FFA0H	UART operation mode register	ASIM0		R/W	×	×	-	00H
FFA1H	UART receive status register	ASIS0		R	-	×	-	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 20	CSIM20		R/W	×	×	-	00H
FFA9H	Serial receive data buffer	SIRB20		R	-	×	-	00H
FFAAH	Receive data buffer status	SRBS20		R	-	×	-	00H
FFADH	Serial I/O shift register 30	SIO30		R/W	-	×	-	00H
FFAEH	Serial I/O switch register	SIOSWI		R/W	×	×	-	00H
FFAFH	Serial I/O mode register 30	CSIM30		R/W	×	×	-	00H
FFB0H	CAN control register	CANC		R/W	×	×	-	01H
FFB1H	Transmit control register	TCR		R/W	-	×	-	00H
FFB2H	Received message register	RMES		R	-	×	-	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

Note: This register is needed for the emulation of power fail detect (PFD) function. DAM0 is not available in this product.

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

Address	SFR Name	Symbol		R/W	Manipulation Bit Unit			After Reset
					1-bit	8-bit	16-bit	
FFB6H	Receive error counter	REC		R	-	×	-	00H
FFB7H	Message count register	MCNT		R/W	-	×	-	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
FFE0H	Interrupt request flag register 0L	IF0	IF0L	R/W	×	×	×	00H
FFE1H	Interrupt request flag register 0H		IF0H	R/W	×	×		00H
FFE2H	Interrupt request flag register 1L	IF1	IF1L	R/W	×	×	×	00H
FFE3H	Interrupt request flag register 1H		IF1H	R/W	×	×	×	00H
FFE4H	Interrupt mask flag register 0L	MK0	MK0L	R/W	×	×	×	FFH
FFE5H	Interrupt mask flag register 0H		MK0H	R/W	×	×		FFH
FFE6H	Interrupt mask flag register 1L	MK1	MK1L	R/W	×	×	×	FFH
FFE7H	Interrupt mask flag register 1H		MK1H	R/W	×	×		FFH
FFE8H	Priority order specified flag 0L	PR0	PR0L	R/W	×	×	×	FFH
FFE9H	Priority order specified flag 0H		PR0H	R/W	×	×		FFH
FFEAH	Priority order specified flag 1L	PR1	PR1L	R/W	×	×	×	FFH
FFEBH	Priority order specified flag 1H		PR1H	R/W	×	×		FFH
FFF0H	Memory size switching register	IMS		R/W	-	×	-	CFH
FFF4H	Internal expansion RAM size switching register	IXS		R/W	-	×	-	**Note**
FFF9H	Watchdog timer mode register	WDTM		R/W	×	×	-	00H
FFFAH	Oscillation stabilisation time register	OSTS		R/W	-	×	-	04H
FFFBH	Processor clock control register	PCC		R/W	×	×	-	04H

Note: The values after reset depend on the product (see Table 22-4, “Values when the Internal Expansion RAM Size Switching Register is Reset,” on page 387).

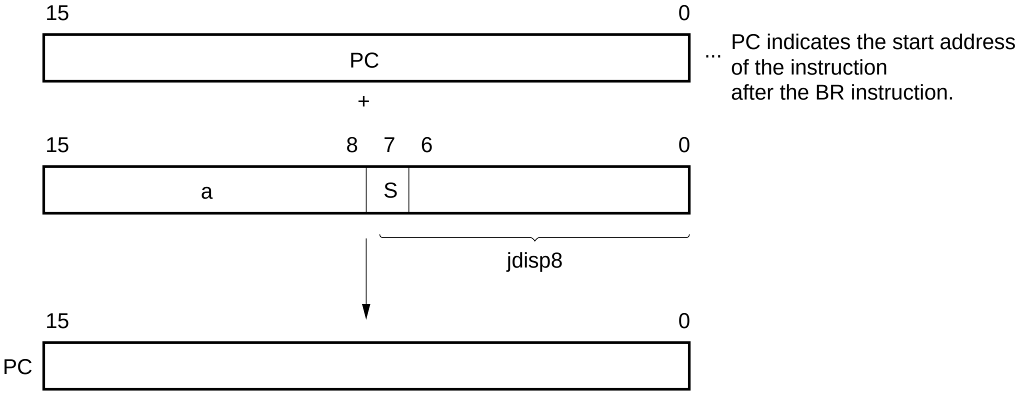
3.3 Instruction Address Addressing

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 (U12326E)**.

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 3-13: Relative Addressing


15
0
PC
... PC indicates the start address of the instruction after the BR instruction.
+
15
8
7
6
0
a
S
jdisp8
15
0
PC

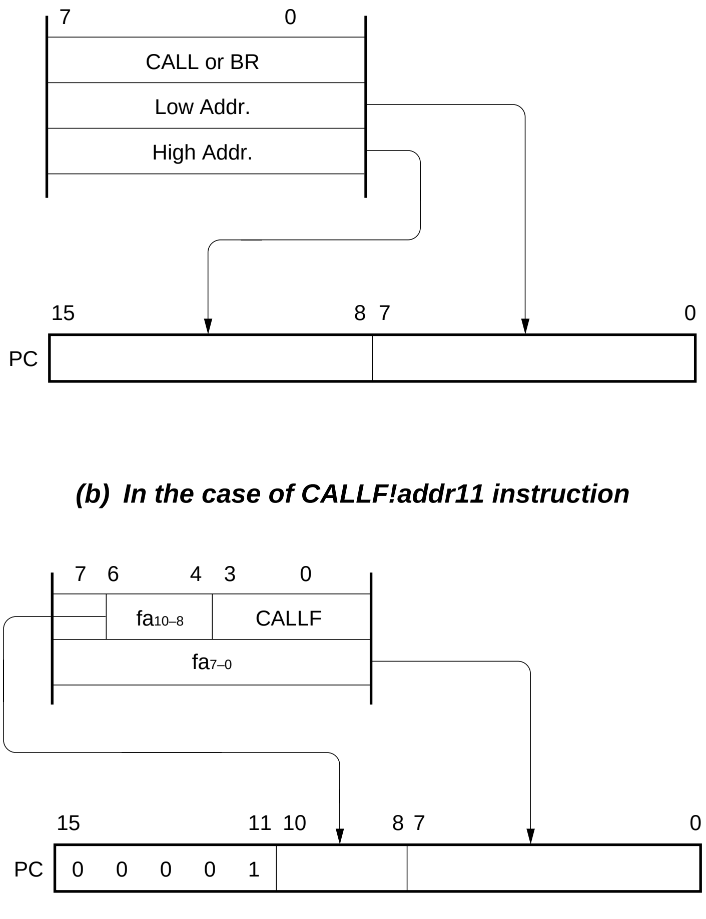
When S = 0, all bits of a are 0.
When S = 1, all bits of a are 1.

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 3-14: Immediate Addressing

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

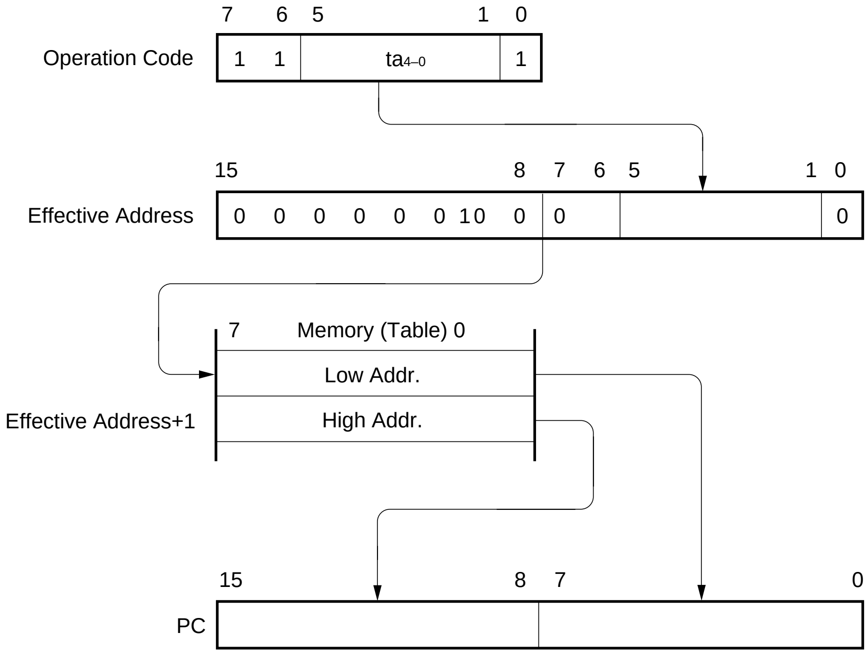


(b) In the case of CALLF!addr11 instruction

3.3.3 Table indirect addressing

Table contents (branch destination address) of the particular location to be addressed by bits 1 to 5 of the immediate data of an operation code are transferred to the program counter (PC) and branched. 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.

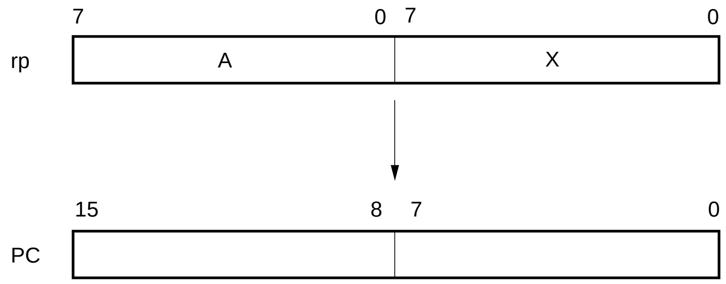
Figure 3-15: Table Indirect Addressing



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 3-16: 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.
Of the µPD780816 Subseries instruction words, the following instructions employ implied addressing.

Table 3-6: 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 3-7: Register Addressing

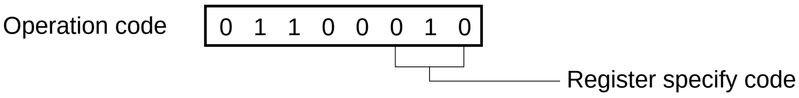
Identifier	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

'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).

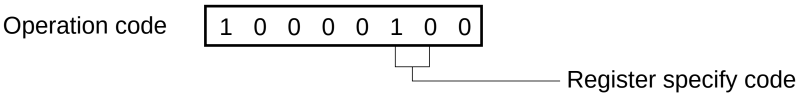
Description example

Figure 3-17: Register Addressing

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



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



3.4.3 Direct addressing

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

Operand format

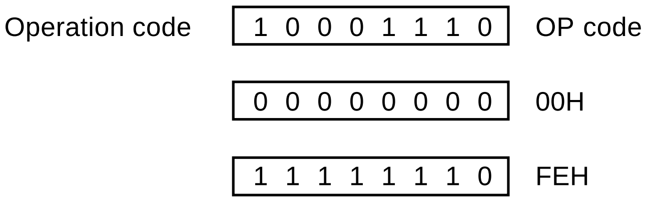
Table 3-8: Direct addressing

Identifier	Description
addr16	Label or 16-bit immediate data

Description example

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

Figure 3-18: Direct addressing



3.4.4 Short direct addressing

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word.
The fixed space to which this addressing is applied to is the 256-byte space, from FE20H to FF1FH. An internal high-speed RAM and 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 3-19 below.

Operand format

Table 3-9: Short direct addressing

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

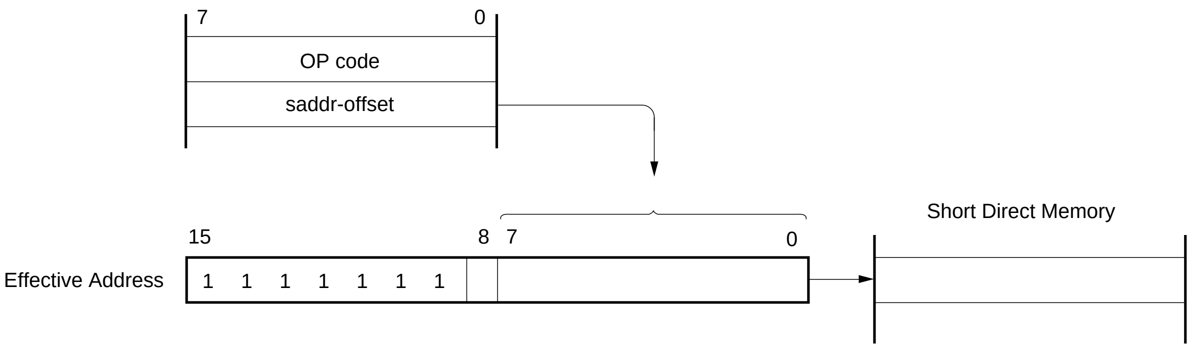
Figure 3-19: Short direct addressing

(a) 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)

(b) Illustration



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 3-10: Special-Function Register (SFR) Addressing

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

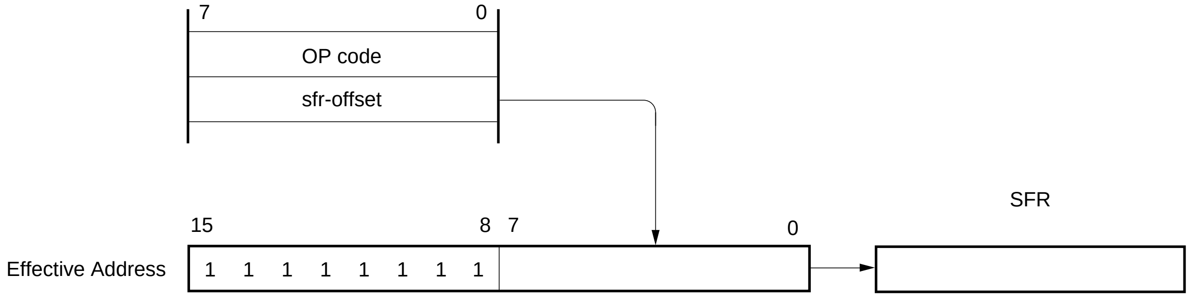
Figure 3-20: Special-Function Register (SFR) Addressing

(a) 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)

(b) Illustration



3.4.6 Register indirect addressing

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

Operand format

Table 3-11: Register indirect addressing

Identifier	Description
-	[DE], [HL]

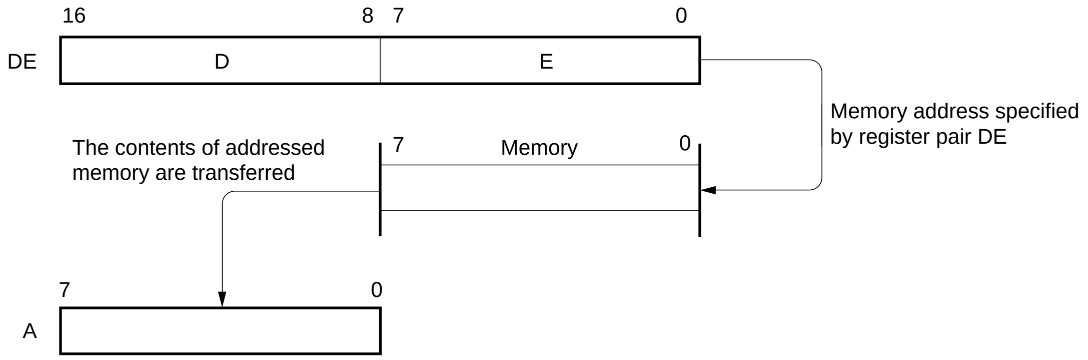
Figure 3-21: Register indirect addressing

(a) Description example

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

Operation code 1 0 0 0 0 1 0 1

(b) Illustration



3.4.7 Based addressing

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

Operand format

Table 3-12: Based addressing

Identifier	Description
	[HL + byte]

Figure 3-22: Based addressing description example

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

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

3.4.8 Based indexed addressing

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

Operand format

Table 3-13: Based indexed addressing

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

Figure 3-23: Based indexed addressing description example

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

Operation code 1 0 1 0 1 0 1 1

3.4.9 Stack addressing

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

Figure 3-24: Stack addressing description example

In the case of PUSH DE

Operation code 1 0 1 1 0 1 0 1

Chapter 4 Port Functions

4.1 Port Functions

The µPD780816 Subseries units incorporate five input ports and thirty-eight input/output ports. Figure 4-1 shows the port configuration. Every port is capable of 1-bit and 8-bit manipulations and can carry out considerably varied control operations. Besides port functions, the ports can also serve as on-chip hardware input/output pins.

Figure 4-1: Port Types

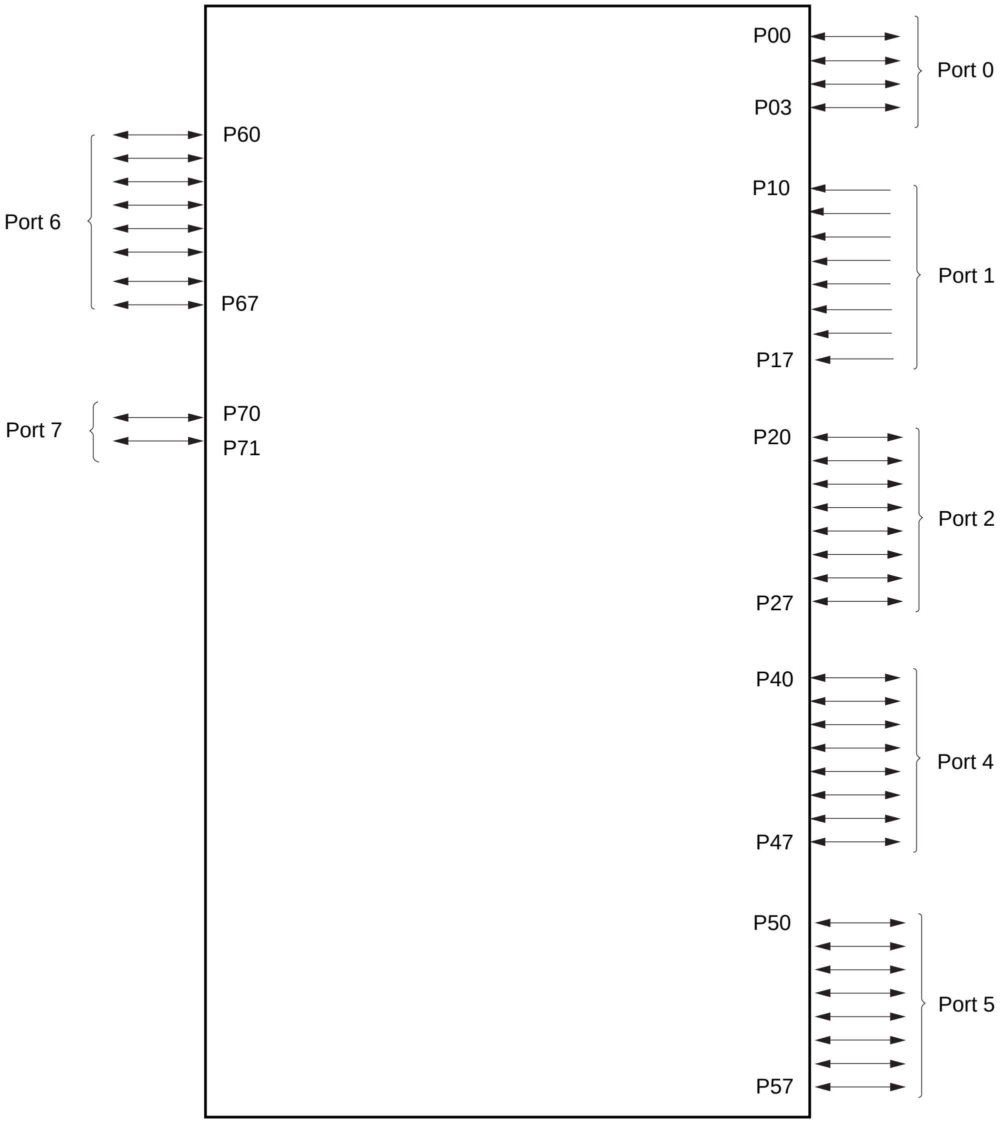


Table 4-1: Pin Input/Output Types

Input/ Output	Pin Name	Function	Alternate Function	After Reset
Input/ Output	P00	Port 0 4-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	INTP0	Input
	P01		INTP1	Input
	P02		INTP2	Input
	P03		INTP3	Input
Input	P10-P17	Port 1 8-bit input only port	ANI0-ANI7	Input
Output	P20	Port 2 8-bit output only port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	SI2	Input
	P21		SO2	Input
	P22		SCK2	Input
	P23		PC2	Input
	P24		RXD	Input
	P25		TXD	Input
	P26		TI50/TO50	Input
	P27		TI51/TO51	Input
Input/ Output	P40-P47	Port 4 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	KR0-KR7	Input
Input/ Output	P50-P57	Port 5 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	-	Input
Input/ Output	P60	Port 6 8-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	TI20	Input
	P61		TI21	
	P62		TI21	
	P63		-	
	P64		-	
	P65		SCK3	
	P66		SO3/SIO3	
	P67		SI3	
Input/ Output	P70	Port 7 2-bit input/output port Input/output mode can be specified bit-wise If used an input port, a pull-up resistor can be connected by software bit-wise	TI00/TO00	Input
	P71		TI01	

4.2 Port Configuration

A port consists of the following hardware:

Table 4-2: Port Configuration

Item	Configuration	
Control register	Port mode register (PMm: m = 0, 2, 4 to 7) Pull-up resistor option register (PUm: m = 0, 2, 4 to 7) Port function register (PFm: m = 2) Key return mode register (KRM)[Note]	
Port	Total: 46 ports	
Pull-up resistor	Mask ROM versions	Total: 38 pins
	(software-specifiable for 38 pins)	
	μPD78F0818	Total: 38 pins
	(software-specifiable for 38 pins)	

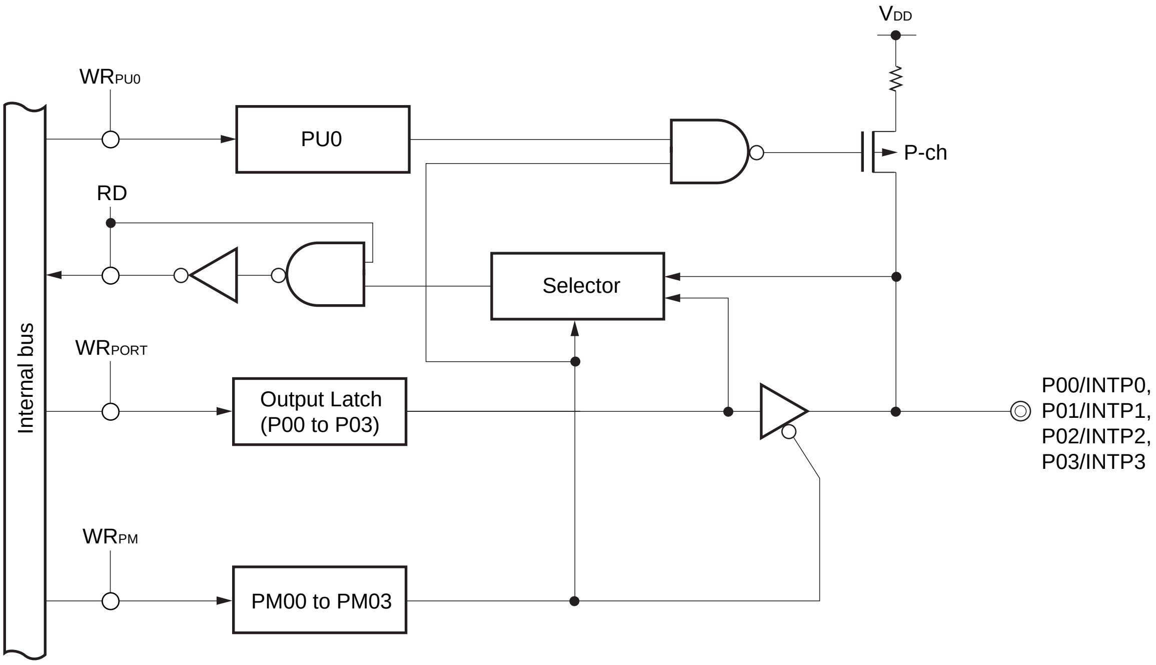
Note: Key return mode of Port 4

4.2.1 Port 0

Port 0 is an 4-bit input/output port with output latch. P00 to P03 pins can specify the input mode/output mode in 1-bit units with the port mode register 0 (PM0). When P00 to P03 pins are used as input pins, a pull-up resistor can be connected to them bit-wise with the pull-up resistor option register (PU0).
Dual-functions include external interrupt request input.
$\overline{\text{RESET}}$ input sets port 0 to input mode.
Figure 4-2 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 4-2: P00 to P03 Configurations



Remarks: 1. PU0 : Pull-up resistor option register

2. PM : Port mode register

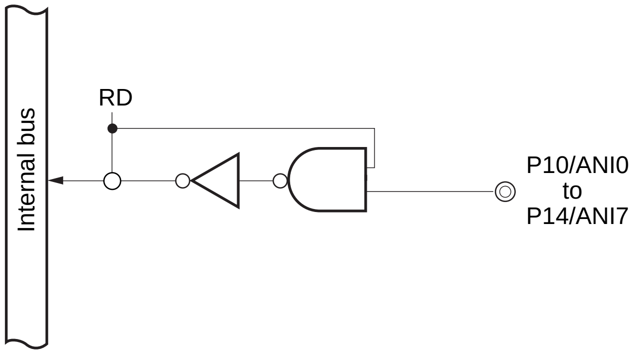
3. RD : Port 0 read signal

4. WR : Port 0 write signal

4.2.2 Port 1

Port 1 is an 8-bit input only port.
Dual-functions include an A/D converter analog input.
Figure 4-3 shows a block diagram of port 1.

Figure 4-3: P10 to P17 Configurations



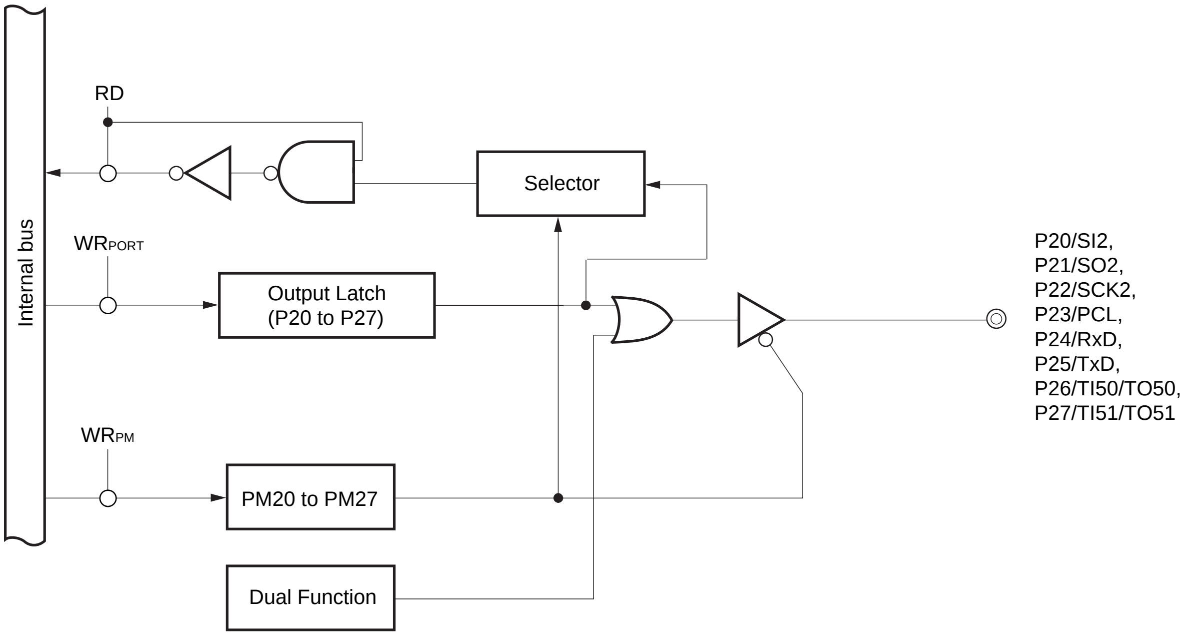
Remark: RD: Port 1 read signal

4.2.3 Port 2

Port 2 is an 8-bit output port with output latch. P20 to P27 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 P27 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 4-4 shows a block diagram of port 2.

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

Figure 4-4: P20 to P27 Configurations



Remarks: 1. PM : Port mode register

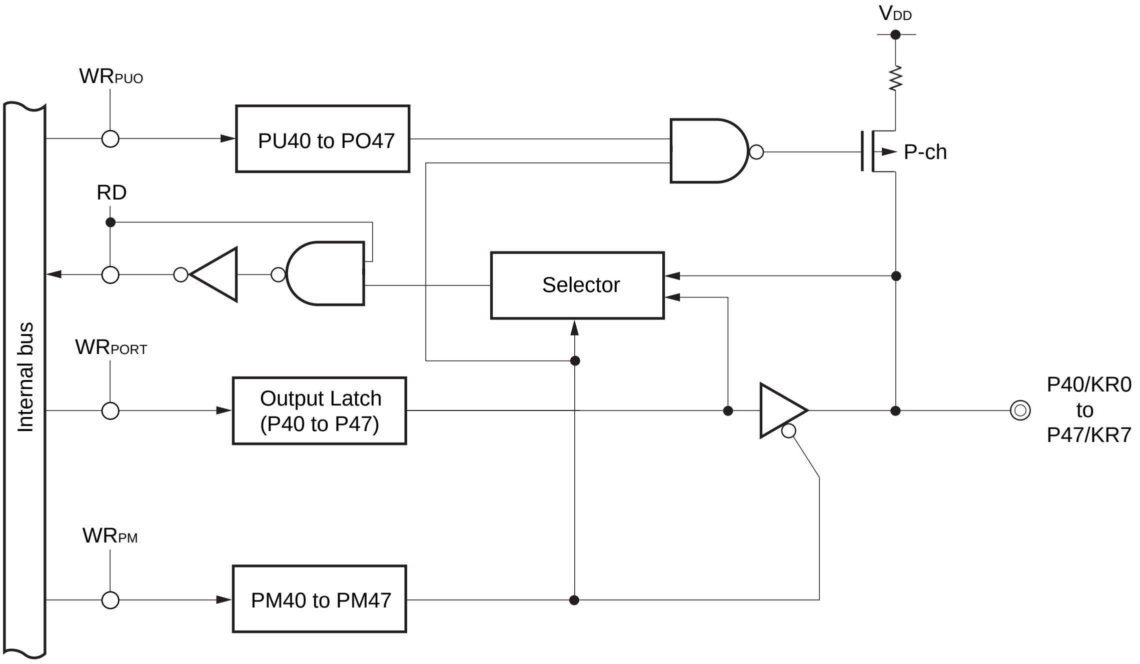
2. RD : Port 2 read signal

3. WR : Port 2 write signal

4.2.4 Port 4

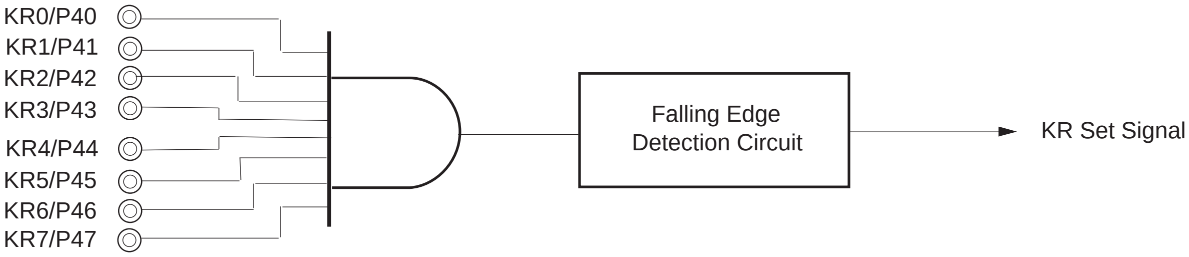
Port 4 an 8-bit input/output port with output latch. P40 to P47 pins can specify the input mode/output mode in 8-bit units with the memory expansion mode register (MM). When P40 to P47 pins are used as input pins, an on-chip pull-up resistor can be connected to them bit-wise with the pull-up resistor option register (PU4).
Dual-function includes the Key Return function.
$\overline{\text{RESET}}$ input sets the input mode.
The port 4 block diagram is shown in Figure 4-5.

Figure 4-5: P40 to P47 Configurations



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

Figure 4-6: Block Diagram of Falling Edge Detection Circuit

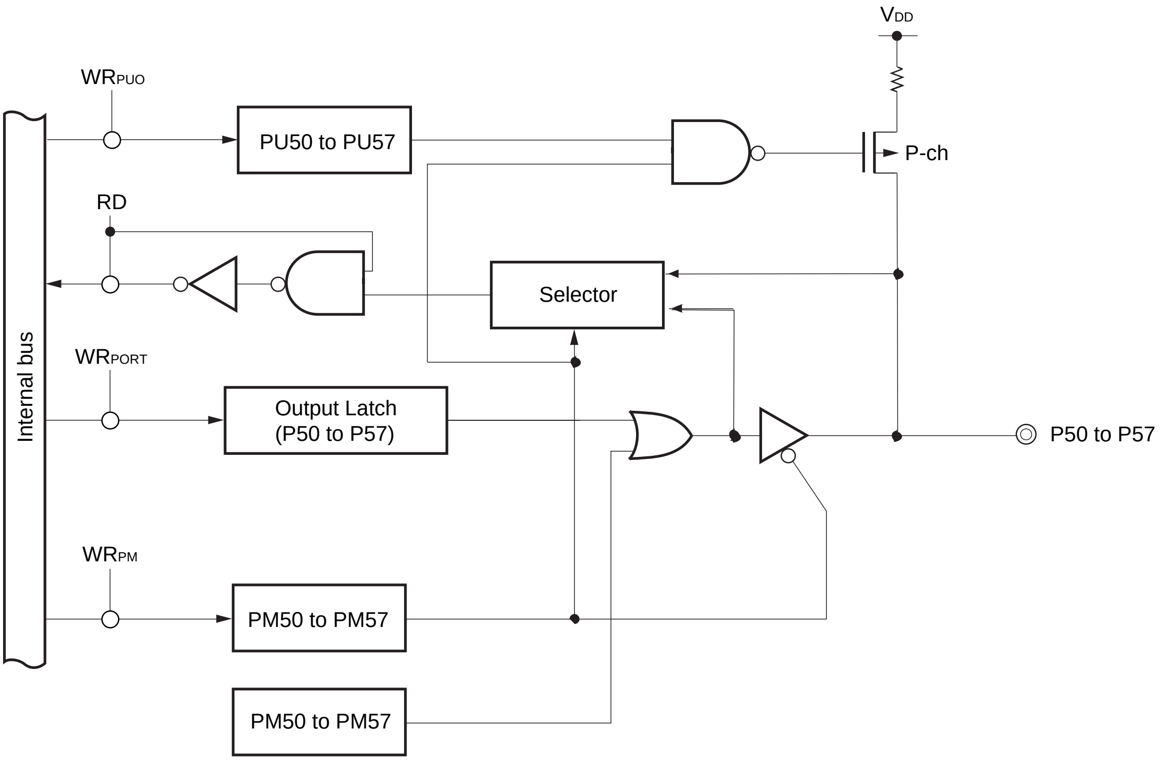


Remark: When Key Return Mode is enabled, a low level at any bit of Port 4 generates a Key Return Interrupt. Port pins that should not generate a Key return Interrupt can be disabled by switching the port pin to output.

4.2.5 Port 5

Port 5 is an 8-bit 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).
$\overline{\text{RESET}}$ input sets port 5 to input mode.
Figure 4-7 shows a block diagram of port 5.

Figure 4-7: P50 to P57 Configurations



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

2. PM : Port mode register

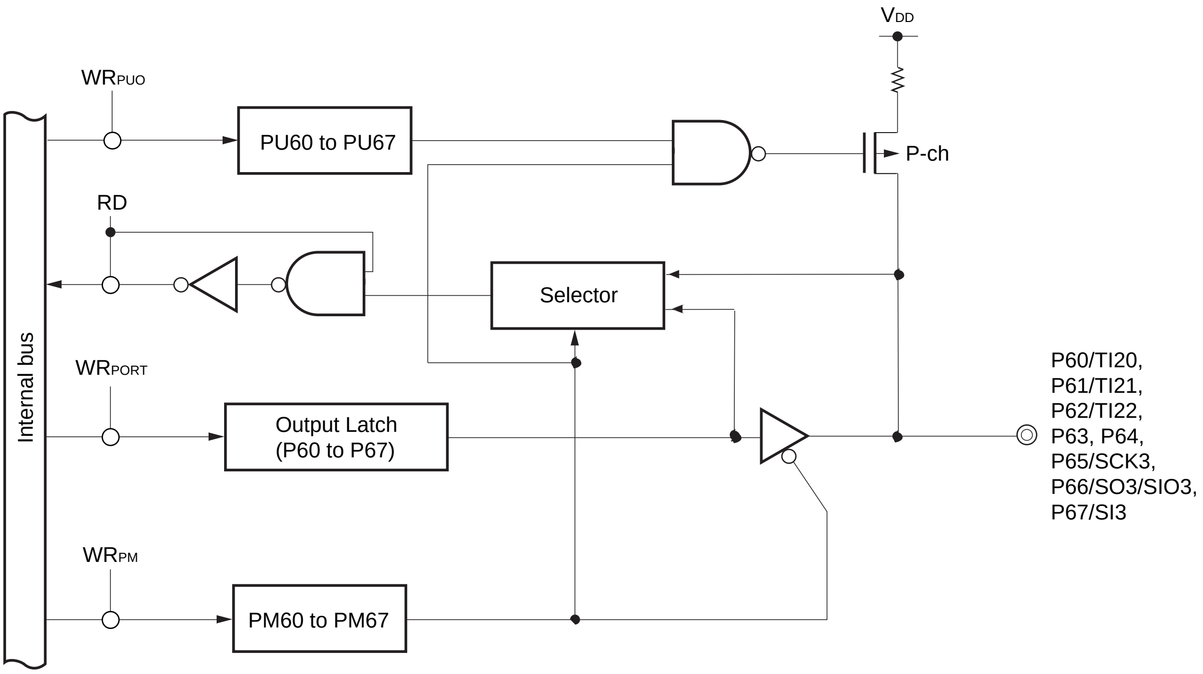
3. RD : Port 5 read signal

4. WR : Port 5 write signal

4.2.6 Port 6

Port 6 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 6 (PM6). When pins P60 to P67 are used as input port pins, an on-chip pull-up resistor can be connected bit-wise with the pull-up resistor option register (PU6).
The dual-functions include the timer capture input signal.
$\overline{\text{RESET}}$ input sets port 6 to input mode.
Figure 4-8 shows block diagram of port 6.

Figure 4-8: P60 to P67 Configurations



Remarks:

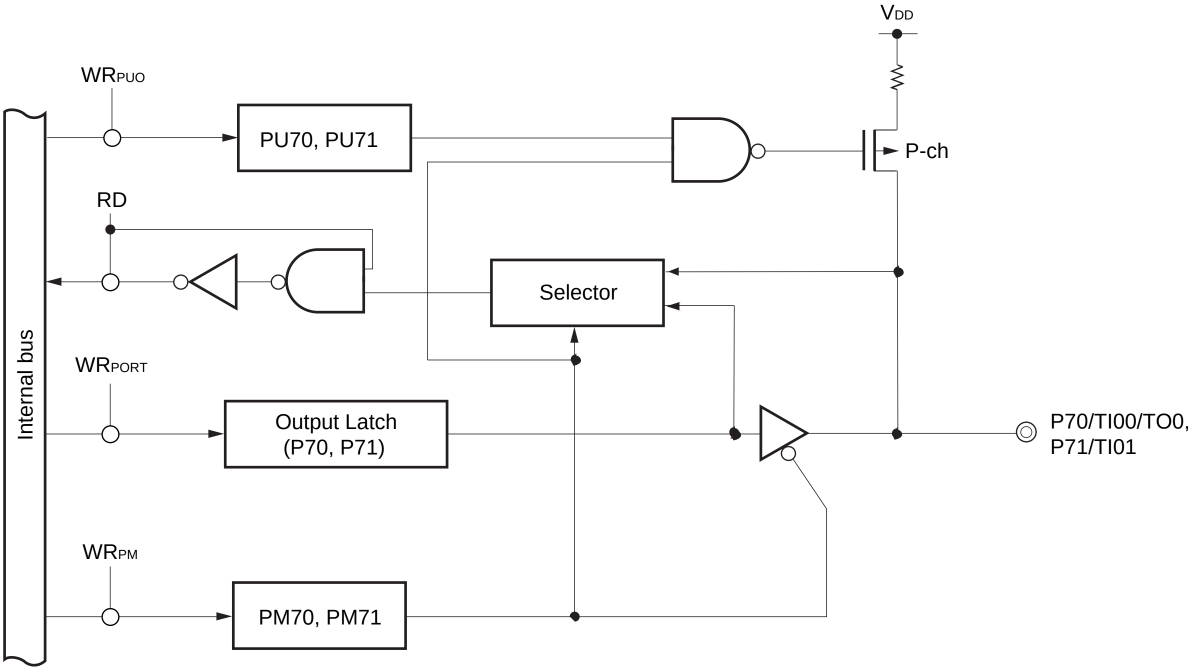
1.	PUO	: Pull-up resistor option register
2.	PM	: Port mode register
3.	RD	: Port 6 read signal
4.	WR	: Port 6 write signal

4.2.7 Port 7

This is a 2-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 and P71 are used as input port pins, an on-chip pull-up resistor can be connected bit-wise with the pull-up resistor option register (PU7).
These pins are dual-function pins and they are used as timer input/output signal.
$\overline{\text{RESET}}$ input sets the input mode.
Port 7 block diagram is shown in Figure 4-9.

Caution: When used as segment lines, set the port function PF8 according to its functions.

Figure 4-9: P70, P71 Configurations



Remarks:

1.	PUO	: Pull-up resistor option register
2.	PM	: Port mode register
3.	RD	: Port 7 read signal
4.	WR	: Port 7 write signal

4.3 Port Function Control Registers

The following four types of registers control the ports.

- Port mode registers (PM0, PM2, PM4 to PM7)
- Pull-up resistor option register (PU0, PU2, PU4 to PU7)
- Port function register (PF2)
- Key Return Mode register (KRM)

(1) Port mode registers (PM0, PM2, PM4 to PM7)

These registers are used to set port input/output in 1-bit units.
PM0, PM2, PM4 to PM7 are independently set with an 1-bit or an 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 P14 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 Key Return mode register specifies P40 to p47 as Key Return port pins.

Figure 4-10: Port Mode Register Format

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

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	R/W	FF22H	FFH

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	R/W	FF24H	FFH

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	R/W	FF25H	FFH

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	R/W	FF26H	FFH

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM7	1	1	1	1	1	1	PM71	PM70	R/W	FF27H	FFH

PMmn	PMmn Pin Input/Output Mode Selection (m = 0, 2, 4 - 7; n = 0 - 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option register (PU0, PU2, PU4 to PU7)

These registers are 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, PU2, PU4 to PU7. No on-chip pull-up resistors can be used to the bits set to the output mode, irrespective of PU0, PU2, PU4 to PU7 setting.
PU0, PU2, PU4 to PU7 are set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets these registers to 00H.

Caution: When ports PU0, PU2, PU4 to PU7 pins are used as dual-functions pins, an on-chip pull-up resistor cannot be used even if 1 is set in PUm (m = 0, 2, 4 to 7.

Figure 4-11: Pull-Up Resistor Option Register (PU0, PU2, PU4 to PU7) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU0	0	0	0	0	PU03	PU02	PU01	PU00	R/W	FF30H	00H

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU2	PU27	PU26	PU25	PU24	PU23	PU22	PU21	PU20	R/W	FF32H	00H

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU4	PU47	PU46	PU45	PU44	PU43	PU42	PU41	PU40	R/W	FF34H	00H

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU5	PU57	PU56	PU55	PU54	PU53	PU52	PU51	PU50	R/W	FF35H	00H

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU6	PU67	PU66	PU65	PU64	PU63	PU62	PU61	PU60	R/W	FF36H	00H

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PU7	0	0	0	0	0	0	PU71	PU70	R/W	FF37H	00H

PUmn	PUmn Pin Internal Pull-up Resistor Selection (m = 0, 2, 4 - 7; n = 0 - 7)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(3) Port function register (PF2)

This register is used to set the LCD segment function of port 2.
PF2 is set with an 1-bit or 8-bit manipulation instruction.
RESET input sets this register to 00H.

Figure 4-12: Port Function Register (PF2) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PF2	0	0	0	0	0	PF22	PF21	PF20	R/W	FF52H	00H

PF2n	PF2n Port Function Selection (n = 0 - 2)
0	Push pull output buffer
1	N-channel open drain output buffer

Caution: Bits 3 to 7 have to be set always to 0.

(4) Key return mode register (KRM)

This register is used to enable/disable the key return signalling.
KRM is set with an 1-bit or 8-bit manipulation instruction.
$\overline{\text{RESET}}$ input sets this register to 00H.

Figure 4-13: Key Return Mode Register (KRM) Format

	7	6	5	4	3	2	1	<0>	R/W	Address	After Reset
KRM	0	0	0	0	0	0	0	KR	R/W	FF47H	00H

KR	Key Return Mode Signalling
0	Key Return Mode disable
1	Key Return Mode enable

4.4 Port Function Operations

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

4.4.1 Writing to input/output port

(1) Output mode

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the 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.

4.4.2 Reading from input/output port

(1) Output mode

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

(2) Input mode

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

4.4.3 Operations on input/output port

(1) Output mode

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

(2) Input mode

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

Caution: In the case of 1-bit memory manipulation instruction, although a single bit is manipulated the port is accessed as an 8-bit unit. Therefore, on a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined, even for bits other than the manipulated bit.

Chapter 5 Clock Generator

5.1 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 is available.

(1) Main system clock oscillator

This circuit oscillates at frequencies of 4 to 8.38 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.

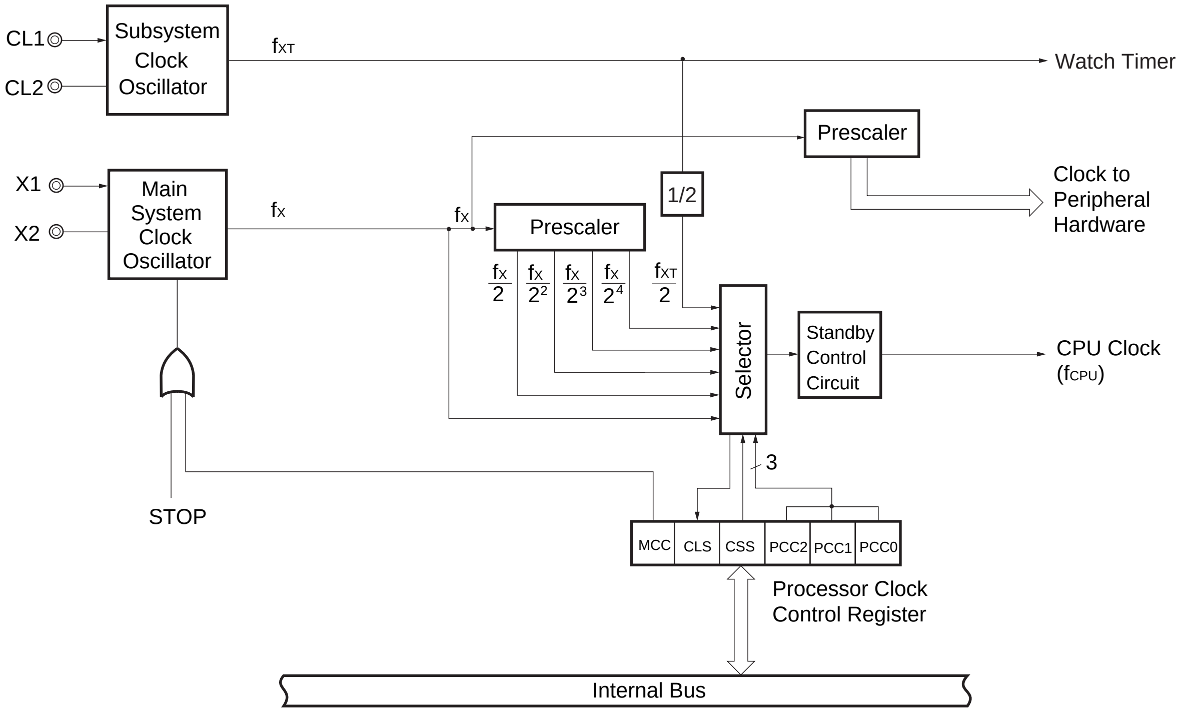
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1: Clock Generator Configuration

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

Figure 5-1: Block Diagram of Clock Generator



5.3 Clock Generator Control Register

The clock generator is controlled by the 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 an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the PCC to 04H.

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

	<7>	6	<5>	<4>	3	2	1	0	R/W	Address	After Reset
PCC	MCC	0	CLS[Note]	CCS	0	PCC2	PCC1	PCC0	R/W	FFFBH	04H
	R/W	R	R	R/W	R	R/W	R/W	R/W			

CSS	PCC2	PCC1	PCC0	CPU Clock Selection (f_{CPU})
0	0	0	0	f_X (0.25 µs)
	0	0	1	f_X /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

CLS	CPU Clock Status
0	Main system clock
1	Subsystem clock

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

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

Note: Bit 5 is a read-only bit.

Remark: 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:
2 f_{CPU} when operating at f_X = 8.0 MHz or f_{XT} = 32.768 kHz.

5.4 System Clock Oscillator

5.4.1 Main system clock oscillator

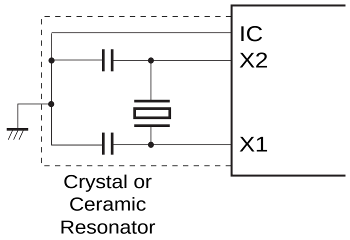
The main system clock oscillator oscillates with a crystal resonator or a ceramic resonator (standard: 8.0 MHz) connected to the X1 and X2 pins.

External clocks can be input to the main system clock oscillator. In this case, input the clock signal to the X1 pin and leave open he X2 pin.

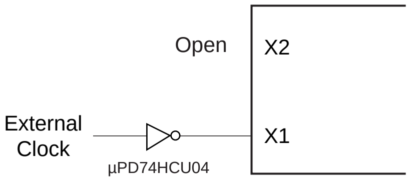
Figure 5-3 shows an external circuit of the main system clock oscillator.

Figure 5-3: External Circuit of Main System Clock Oscillator

(a) Crystal and ceramic oscillation



(b) External clock



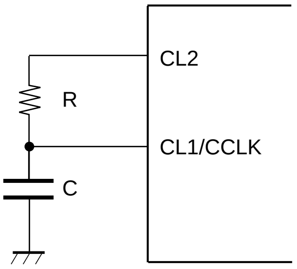
Caution: Do not execute the STOP instruction and do not set MCC (bit 7 of processor clock control register PCC) to 1 if an external clock is input. This is because when the STOP instruction or MCC is set to 1, the main system clock operation stops and the X2 pin is connected to V_{DD1} via a pull-up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

5.4.2 Subsystem clock oscillator

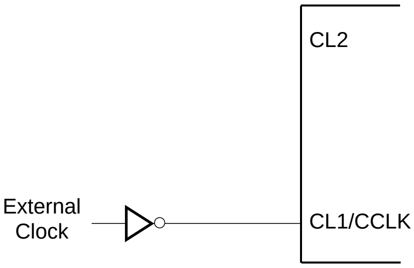
The subsystem clock oscillator oscillates with a RC-resonator (standard: 40 kHz) connected to the CL1 and CL2 pins.
External clocks can be input to the subsystem clock oscillator. In this case, input the clock signal to the CL1 pin and leave open he CL2 pin.
Figure 5-4 shows an external circuit of the subsystem clock oscillator.

Figure 5-4: 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 is prohibited.
The setting of the CSS-bit (PCC-register) and the WTM 7-bit (WTM-register) to 1 is prohibited.

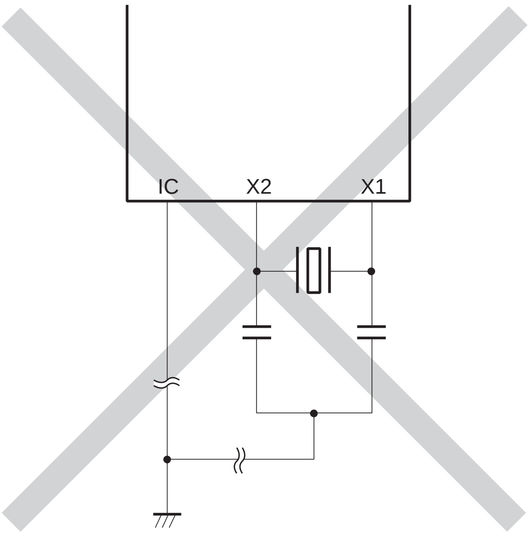
2. When using a main system clock oscillator and a subsystem clock oscillator, carry out wiring in the broken-line area in Figures 5-3 and 5-4 as follows to prevent any effects from wiring capacities.

- **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 5-5 shows examples of oscillator having bad connection.

Figure 5-5: 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

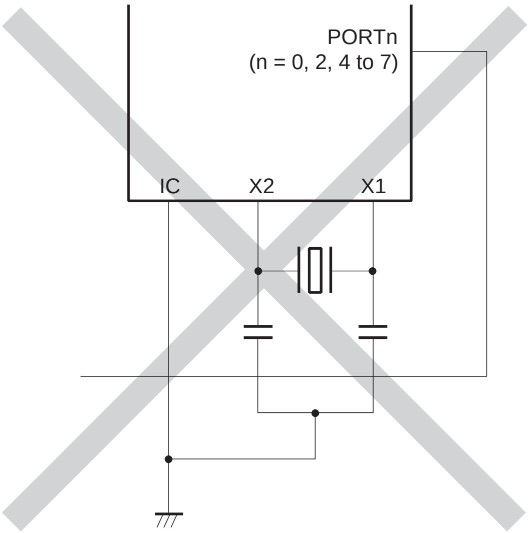
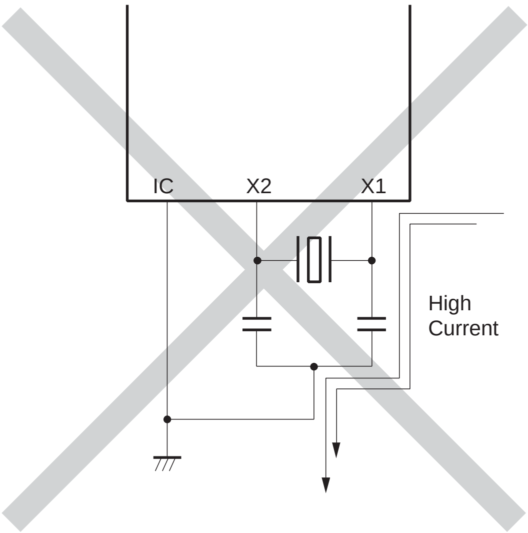


Figure 5-5: Examples of Oscillator with Bad Connection (2/3)

(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)

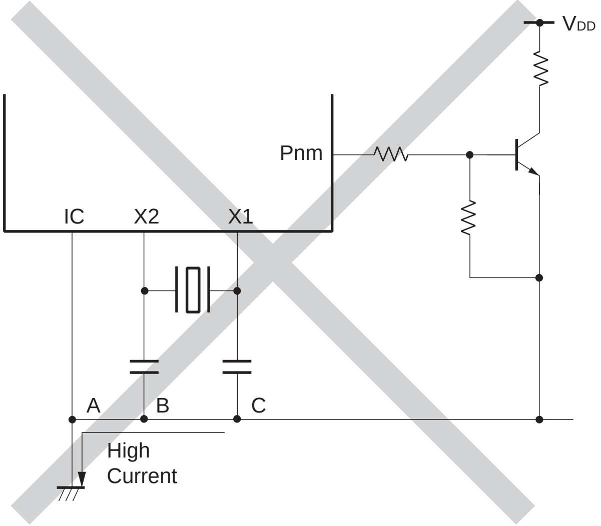
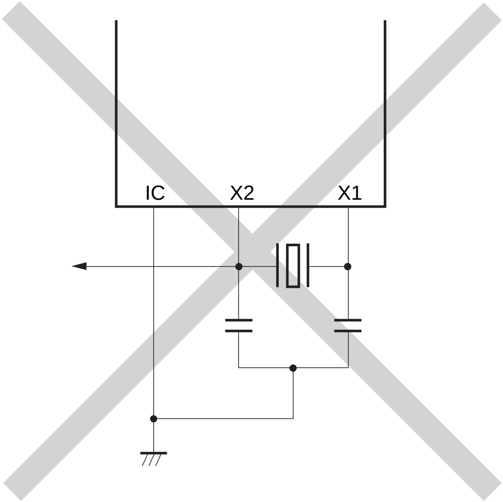
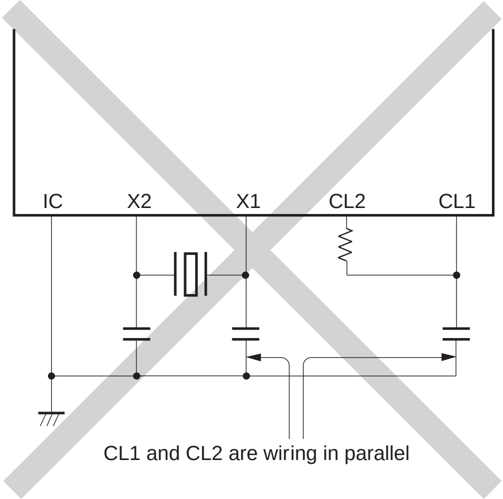


Figure 5-5: Examples of Oscillator with Bad Connection (3/3)

(e) Signals are fetched



(f) Signal conductors of the main and subsystem 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 5-5 (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}.

5.4.3 When no subsystem clock is 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

5.5 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.)

5.5.1 Main system clock operations

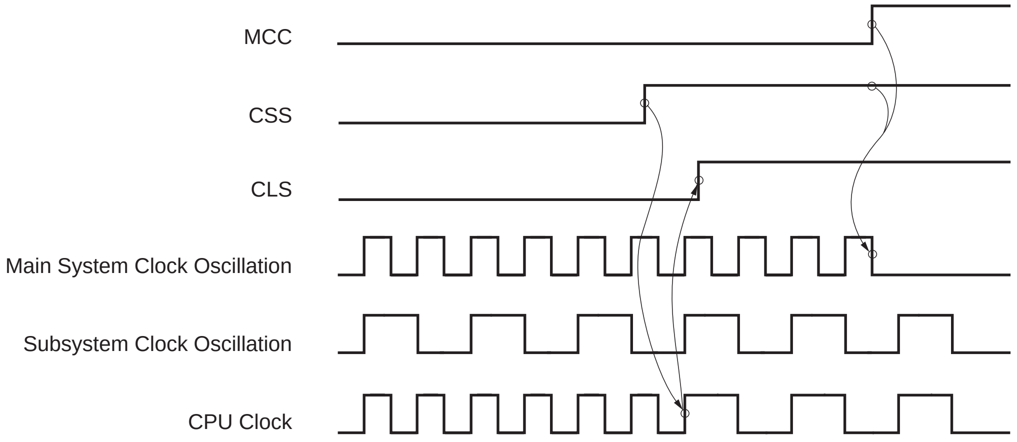
When operated with the main system clock (with bit 5 (CLS) of the processor clock control register (PCC) set to 0), the following operations are carried out by PCC setting.

(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 5-6).

Figure 5-6: Main System Clock Stop Function

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



5.5.2 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.

5.6 Changing System Clock and CPU Clock Settings

5.6.1 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 5-2).
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 5-2: Maximum Time Required for CPU Clock Switchover

Set Values after Switchover				Set Values before Switchover					
CSS	PCC2	PCC1	PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0
				0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	1 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	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)	

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).

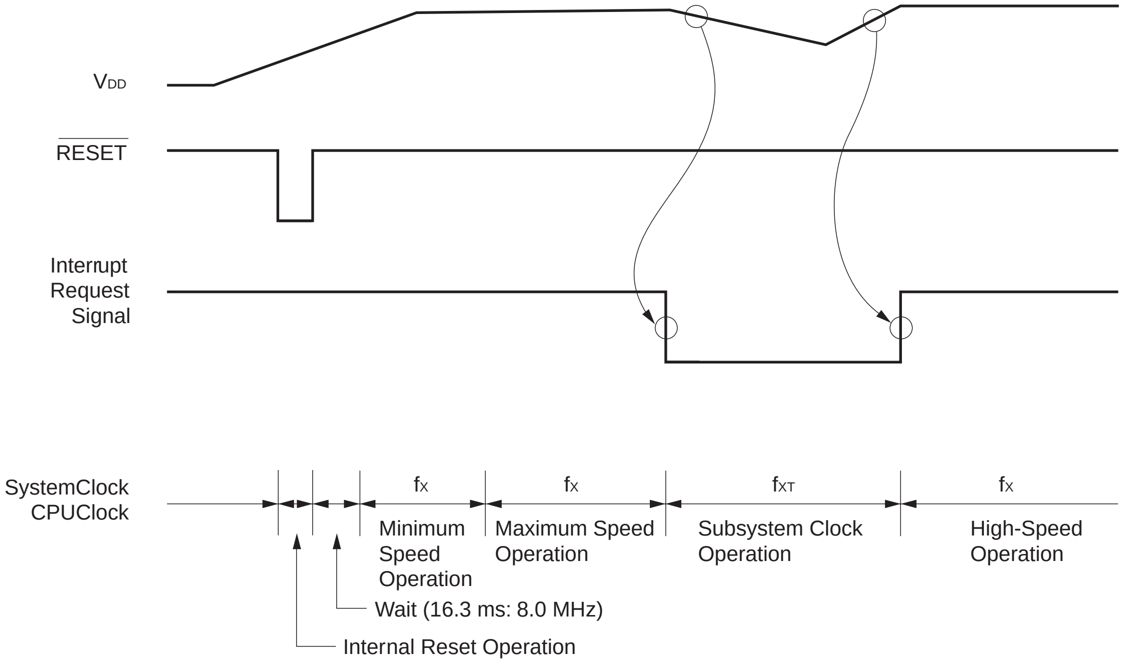
Remarks: 1. One instruction is the minimum instruction execution time with the pre-switchover CPU clock.

2. Figures in parentheses apply to operation with f_X = 8.0 MHz and f_{XT} = 32.768 kHz.

5.6.2 System clock and CPU clock switching procedure

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

Figure 5-7: 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 6 Main Clock Monitor

6.1 Main Clock Monitor Function

The main clock monitor task is, to watch the activities of the main system clock by using the subsystem clock. If the main clock fails for more than three sub clock cycles, the main clock monitor detects the fault condition and triggers the chip reset.
The following procedure allows the using of the main clock monitor:

(1) Main system clock is working.

(2) Subsystem clock is working.

(3) Enable the main system clock monitor by setting the CLME bit to 1.

Cautions: 1. The main clock monitor is automatically disabled, if the CPU is in STOP mode, or the CPU is clocked by the subsystem clock and MCC is set to 1.

2. Once the main clock monitor has been enabled, it can only disabled by triggering the external reset.

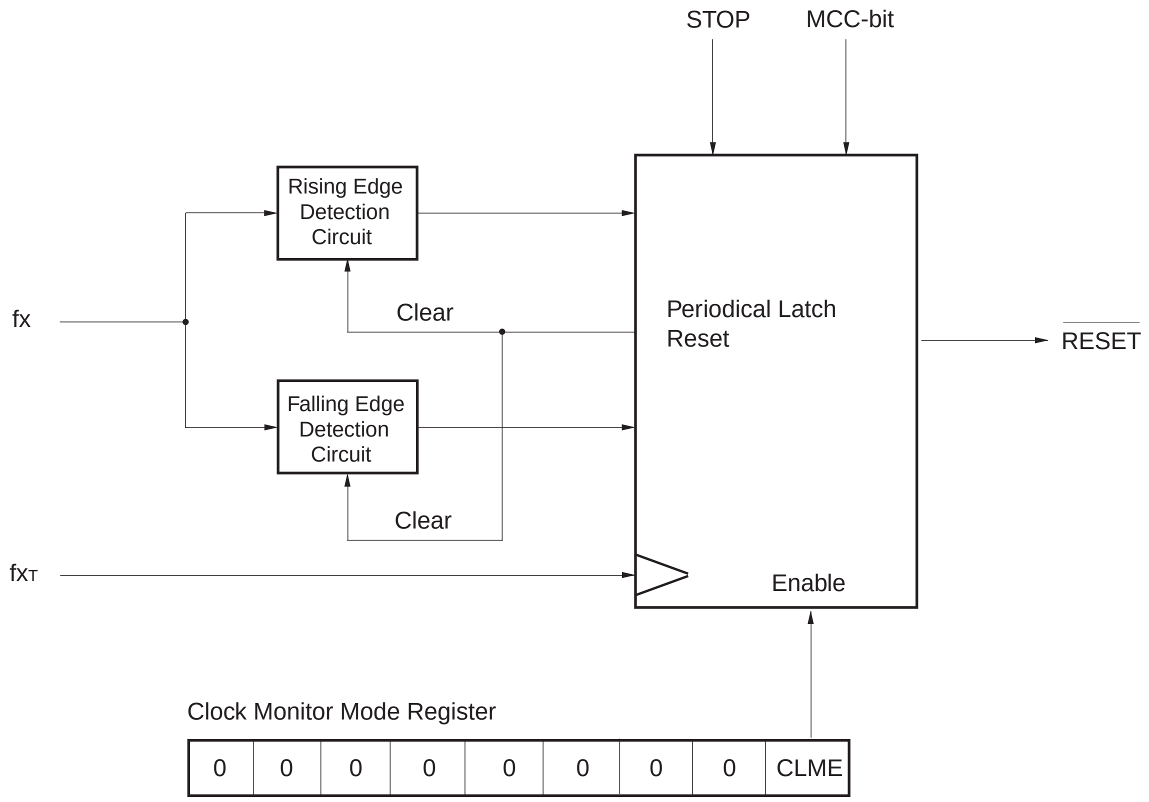
6.2 Main Clock Monitor Circuit Configuration

The main clock monitor consists of the following hardware.

Table 6-1: Main Clock Monitor Configuration

Item	Configuration
Control register	Clock monitor mode register (CLM)

Figure 6-1: Main Clock Monitor Circuit Block Diagram


STOP
MCC-bit
Rising Edge Detection Circuit
Clear
fx
Periodical Latch Reset
RESET
Falling Edge Detection Circuit
Clear
fxT
Enable
Clock Monitor Mode Register
0
0
0
0
0
0
0
0
CLME

6.3 Main Clock Monitor Control Register

The following register is used to control the main clock monitor.

- Clock monitor mode register.

(1) Clock monitor mode register (CLM)

This register ends the main clock monitor.
CLM is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CLM to 00H.

Figure 6-2: Format Clock Monitor Mode Register (CLM)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
CLM	0	0	0	0	0	0	0	CLME	R/W	FF77H	00H

CLME	Clock Monitor Operation Selection
0	Clock Monitor disable
1	Clock Monitor enable

6.4 Operating Modes

(1) STOP Mode Release by Interrupt

When the CPU works on the main system clock and the main system clock oscillator is stopped by the MCC-bit or the STOP instruction the clock monitor is disabled. An interrupt wakes up the oscillator and after the finished oscillation stabilization time (based on the setting of the OSTS register) the clock monitor is enabled and the CPU starts working again.
The sub clock keeps on running.

(2) Release by Reset

When the CPU works on the main system clock and the main system clock oscillator is stopped by the MCC-bit or the STOP instruction the clock monitor is disabled. An external reset wakes up the oscillator and clears the clock monitor enable bit. After the finished oscillation stabilization time (based on the setting of the OSTS register) the CPU starts working again.
The sub clock keeps on running.

(3) Main clock stopped, Sub clock operating, CPU works on sub clock

When the CPU works on the subsystem clock and the main oscillator is stopped by the MCC-bit or the STOP instruction the clock monitor is disabled. When the main oscillator is restarted by CPU (via clear of the MCC-bit in PCC register) the clock monitor will be enabled again when the CPU switches from the sub clock to the main system clock after the CPU has got at least 83 main system clocks in minimum. The application software has to take care of a delay time before switching to the main system clock.
The sub clock keeps on running.

(4) Main clock stopped, Sub clock operating, CPU in STOP mode on sub clock

When the main system clock oscillator is stopped, the CPU works on the sub system clock and the CPU is then set to STOP mode by the STOP instruction the clock monitor is disabled. This mode can only be released by Reset, where a release by an interrupt is not possible.
The sub clock keeps on running.

(5) Main clock operating, Sub clock operating, CPU in HALT mode on sub clock

When the CPU works on the subsystem clock, the main oscillator is running and the CPU is set to HALT mode and woken up via interrupt the clock monitor continues working the whole time.

(6) Main clock stopped, Sub clock operating, CPU in HALT mode on Sub clock

The clock monitor is disabled when the main clock oscillator is switched off and will be enabled when the CPU is running again on main system clock oscillator after the CPU has got at least 83 main system clocks in minimum. The following conditions are valid:

- The CPU works on the subsystem clock.
- The main system clock oscillator is stopped via the MCC-bit of the PCC register.
- The CPU is finally stopped via the halt instruction.
- Wake up is done via an interrupt.

The application software has to take care of a delay time before switching to the main system clock.

Chapter 7 16-Bit Timer 0

7.1 16-bit Timer/Event Counter Function

16-bit timer/event counter 0 (TM0) has the following functions:

- Interval timer
- PPG output
- Pulse width measurement
- External event counter
- Square wave 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.

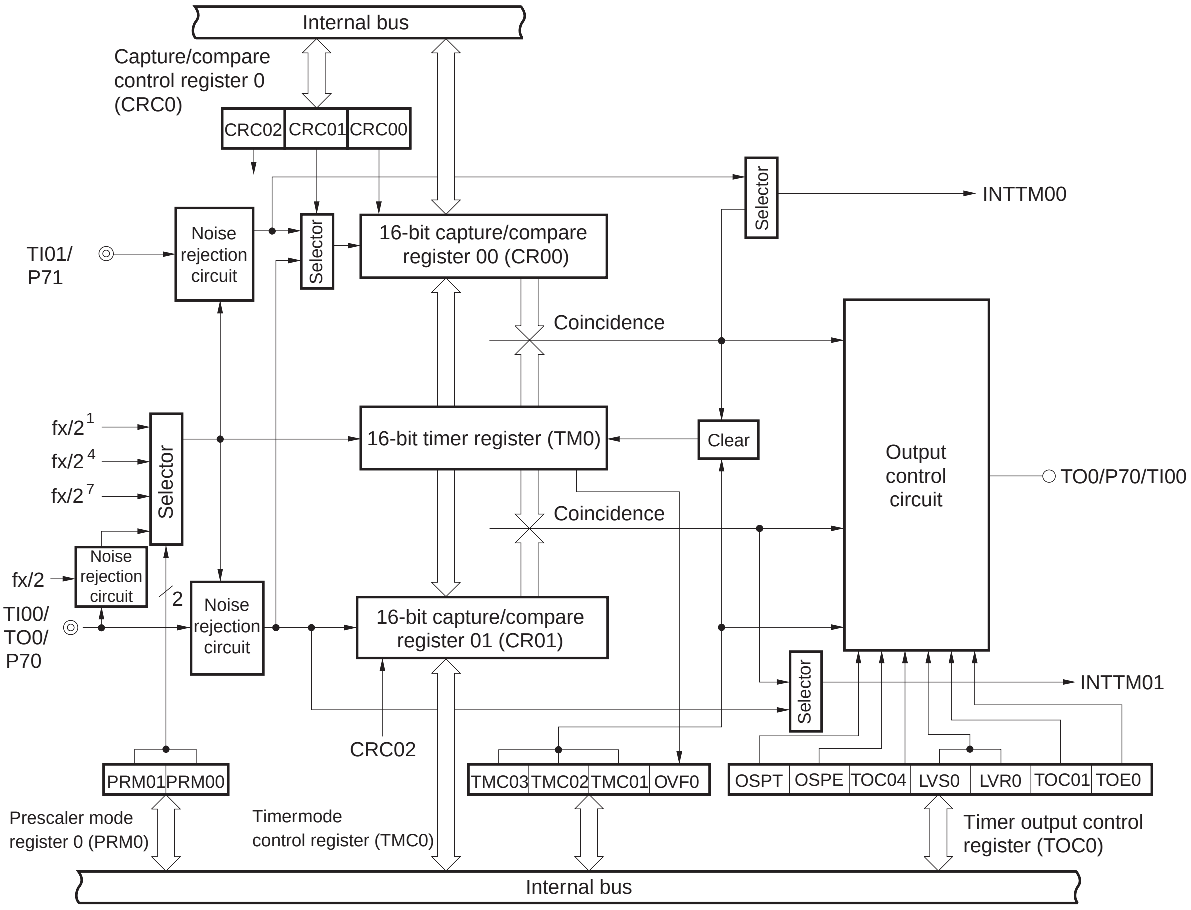
7.2 16-bit Timer/Event Counter 0 Configuration

16-bit timer/event counter 0 (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 × 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 7 (PM7)

Figure 7-1: Block Diagram of 16-Bit Timer/Event Counter 0 (TM0)



(1) 16-bit timer register (TM0)

TM0 is a 16-bit read-only register that counts 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> 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.

(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.

(a) 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.

(b) 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	Capture Trigger of CR01
0	0	Falling edge	Rising edge	Falling edge
0	1	Rising edge	Falling edge	Rising edge
1	0	Setting prohibited	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	No capture operation	Both rising and falling edges

Table 7-3: Valid edge of TI01 Pin and valid edge of capture trigger of capture/compare register

ES01	ES00	Valid Edge of TI01 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 another value than 0000H to 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.

(a) 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.

(b) 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).
R01 is set by a 16-bit memory manipulation instruction.
After $\overline{\text{RESET}}$ input, the value of CR00 is undefined.

Caution: Set another value than 0000H to CR01. This means, that an 1-pulse count operation cannot be performed when CR01 is used as an event counter.

7.3 16-Bit Timer/Event Counter 0 Control Register

The following four types of registers control 16-bit timer/event counter 0.

- 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 7 (PM7)

(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 an 1-bit or an 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)

	7	6	5	4	3	2	1	<0>	R/W	Address	After Reset
TMC0	0	0	0	0	TMC03	TMC02	TMC01	OVF0	R/W	FF60H	00H

TMC03	TMC02	TMC01	Operating mode, 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 or 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	-	
1	0	1		-	
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	

OVF0	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 an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CRC0 to 00H.

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

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
CRC0	0	0	0	0	0	CRC02	CRC01	CRC00	R/W	FF62H	00H

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 CRC00 as a capture register.

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

4. To surely perform the capture operation, the capture trigger requires a pulse two times longer than the count clock selected by prescaler mode register 0 (PRM0).

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

This register controls the operation of the 16-bit timer/event counter 0 output control circuit by setting or resetting the R-S flip-flop, 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 an one-shot pulse by software.

TOC0 is set by an 1-bit or an 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)

	7	6	5	4	<3>	<2>	1	<0>	R/W	Address	After Reset
TOC0	0	0	0	TOC04	LVS0	LVR0	TOC01	TOE0	R/W	FF63H	00H

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 0 and the valid edge of TI00, TI01 input. PRM0 is set by an 1-bit or an 8-bit memory manipulation instruction.
RESET input sets PRM0 to 00H.

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

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PRM0	ES11	ES10	ES01	ES00	0	0	PRM01	PRM00	R/W	FF61H	00H

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 7 (PM7)

This register sets port 7 input/output in 1-bit units.
When using the P70/TO0/TI00 pin for timer output, set PM70 and the output latch of P70 to 0.
PM7 is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets PM7 value to FFH.

Figure 7-6: Port Mode Register 7 (PM7) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM7	1	1	1	1	1	1	PM71	PM70	R/W	FF27H	FFH

PM7n	P0n pin input/output mode selection (n = 0, 1)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

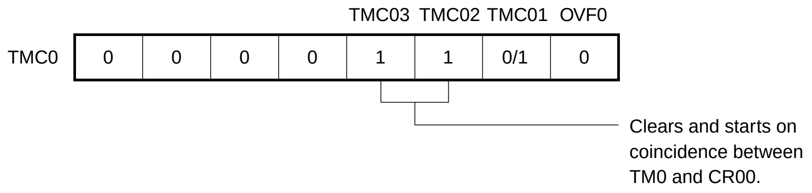
7.4 16-Bit Timer/Event Counter 0 Operations

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 0 (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 0 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

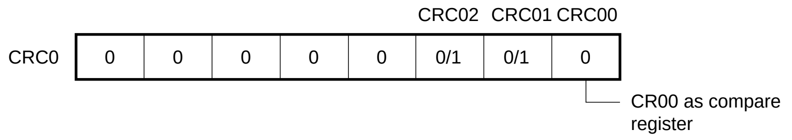
(a) 16-bit timer mode control register (TMC0)



					TMC03	TMC02	TMC01	OVF0
TMC0	0	0	0	0	1	1	0/1	0

Clears and starts on coincidence between TM0 and CR00.

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



						CRC02	CRC01	CRC00
CRC0	0	0	0	0	0	0/1	0/1	0

CR00 as compare register

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

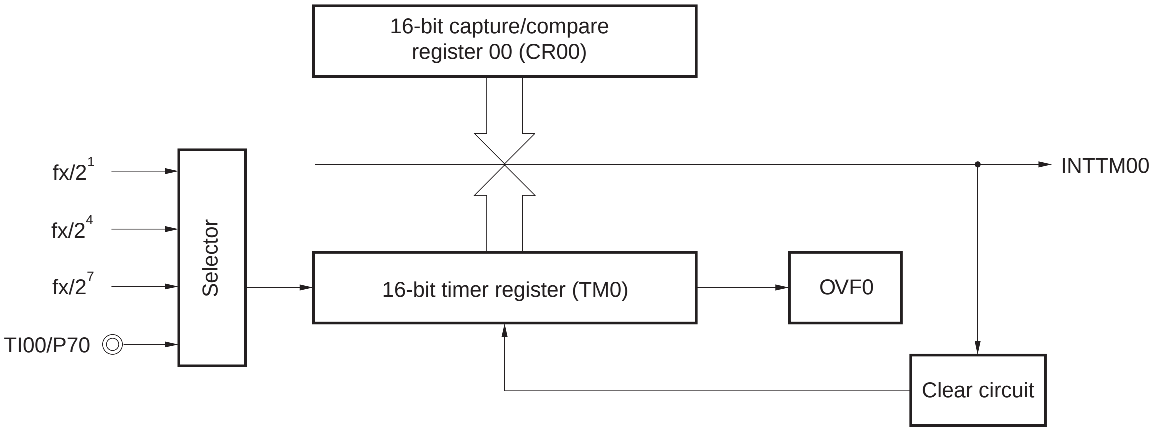
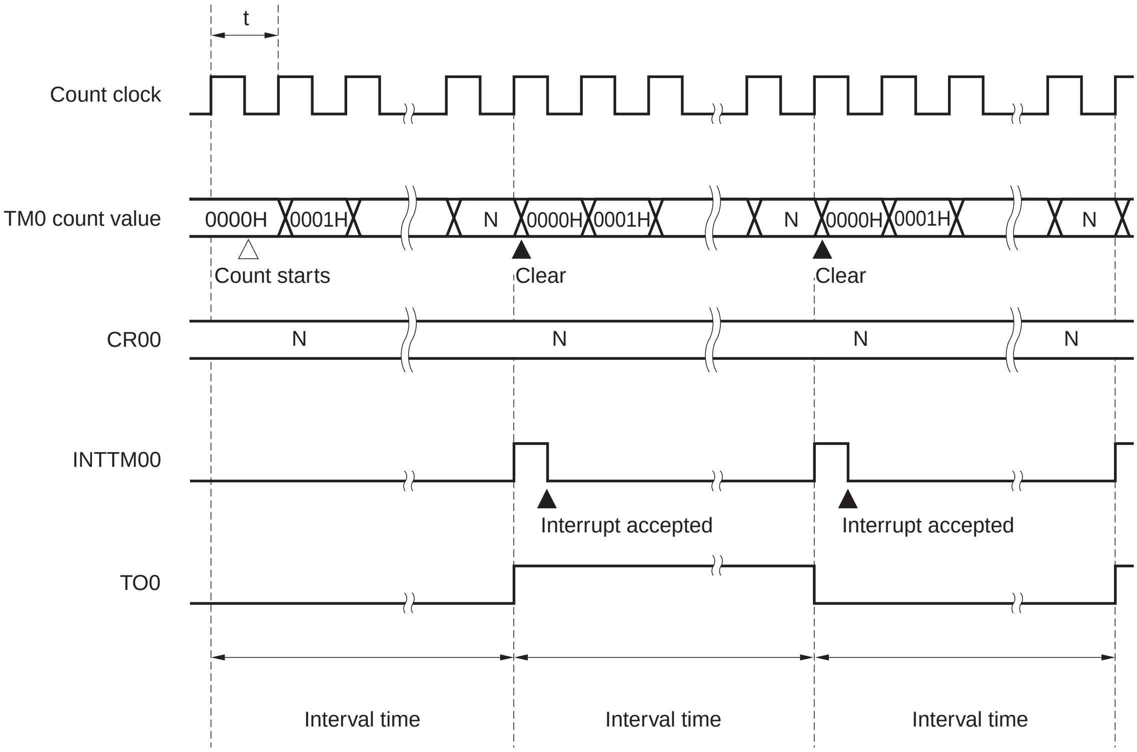

16-bit capture/compare register 00 (CR00)
fx/2^1
fx/2^4
fx/2^7
TI00/P70
Selector
16-bit timer register (TM0)
OVF0
INTTM00
Clear circuit

Figure 7-9:   Timing of Interval Timer Operation



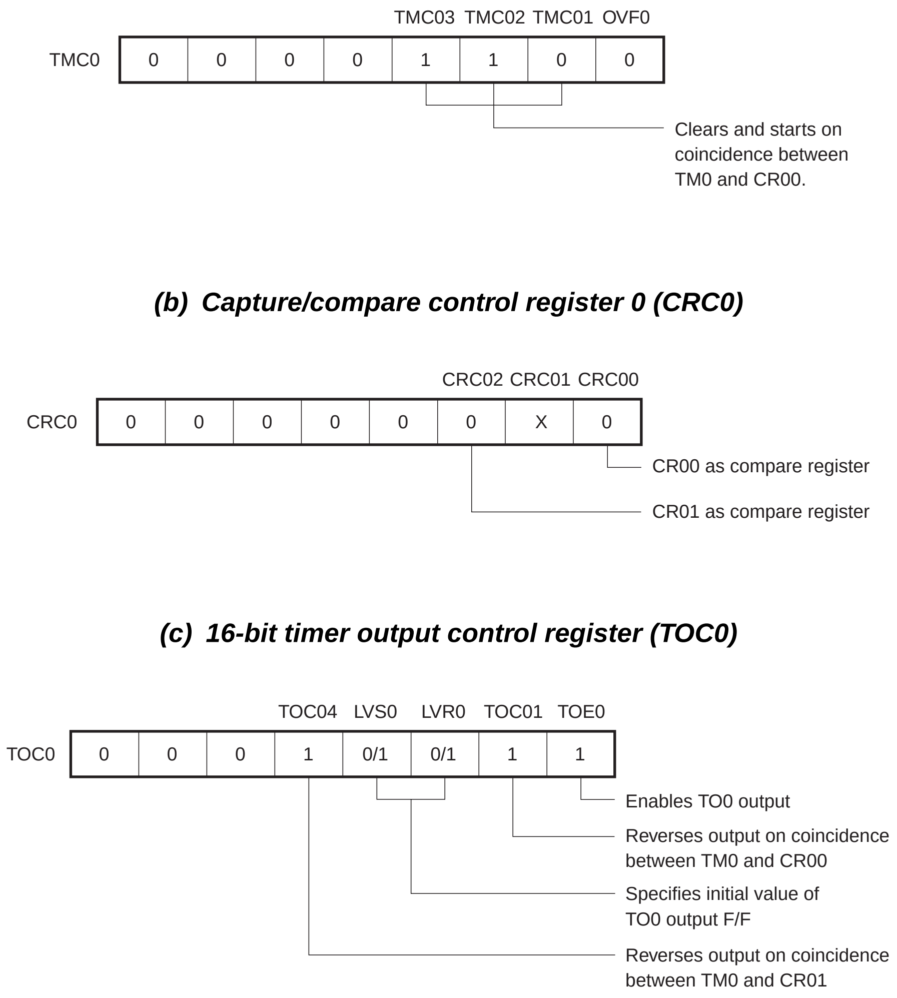
Remark: Interval time = (N+1) x t: N = 0000H to FFFFH

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)

TMC03 TMC02 TMC01 OVF0
TMC0 0 0 0 0 1 1 0 0
Clears and starts on coincidence between TM0 and CR00.
CRC02 CRC01 CRC00
CRC0 0 0 0 0 0 0 X 0
CR00 as compare register
CR01 as compare register
TOC04 LVS0 LVR0 TOC01 TOE0
TOC0 0 0 0 1 0/1 0/1 1 1
Enables TO0 output
Reverses output on coincidence between TM0 and CR00
Specifies initial value of TO0 output F/F
Reverses output on coincidence between TM0 and CR01

Remark: x: don’t care

Cautions: 1. Make sure that 0000H ≤ CR01 < CR00 ≤ FFFFH is set to CR00 and CR01.

2. The cycle of the pulse generator through PPG output (CR00 setting value +1) has a duty of (CR01 setting value + 1) / (CR00 setting value + 1)

7.4.3 Pulse width measurement

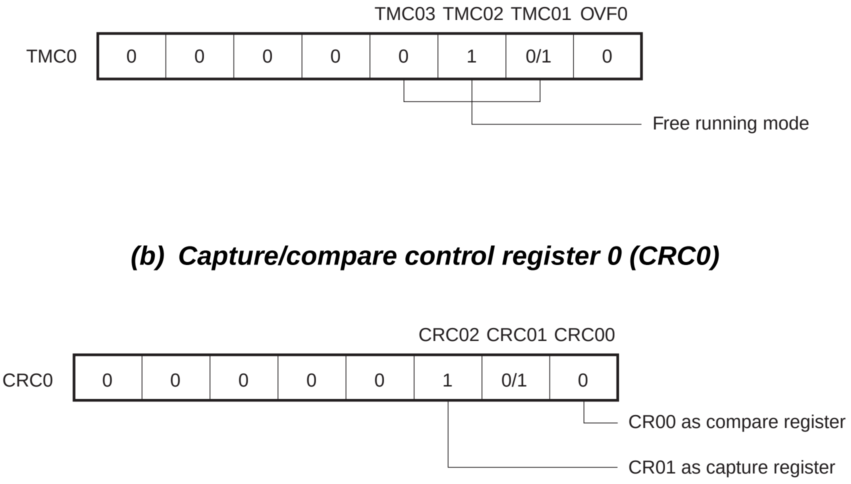
The 16-bit timer register (TM0) can be used to measure the pulse widths of the signals input to the TI00 and TI01 pins.
Measurement can be carried out with TM0 used as a free running counter or by restarting the timer in synchronization with the edge of the signal input to the TI00 pin.

(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

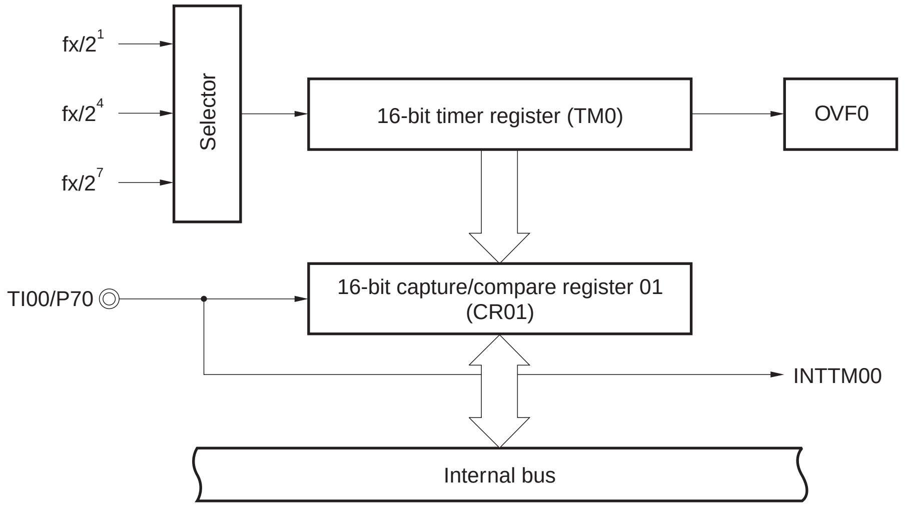
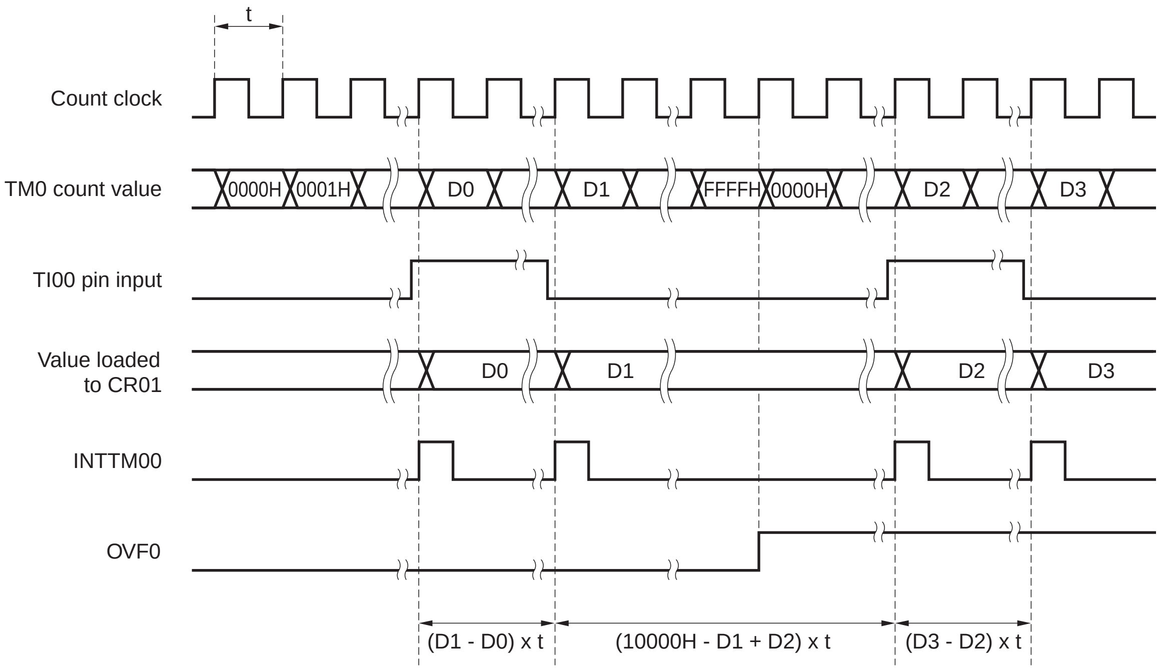

fx/2^1
fx/2^4
fx/2^7
Selector
16-bit timer register (TM0)
OVF0
TI00/P70
16-bit capture/compare register 01 (CR01)
INTTM00
Internal bus

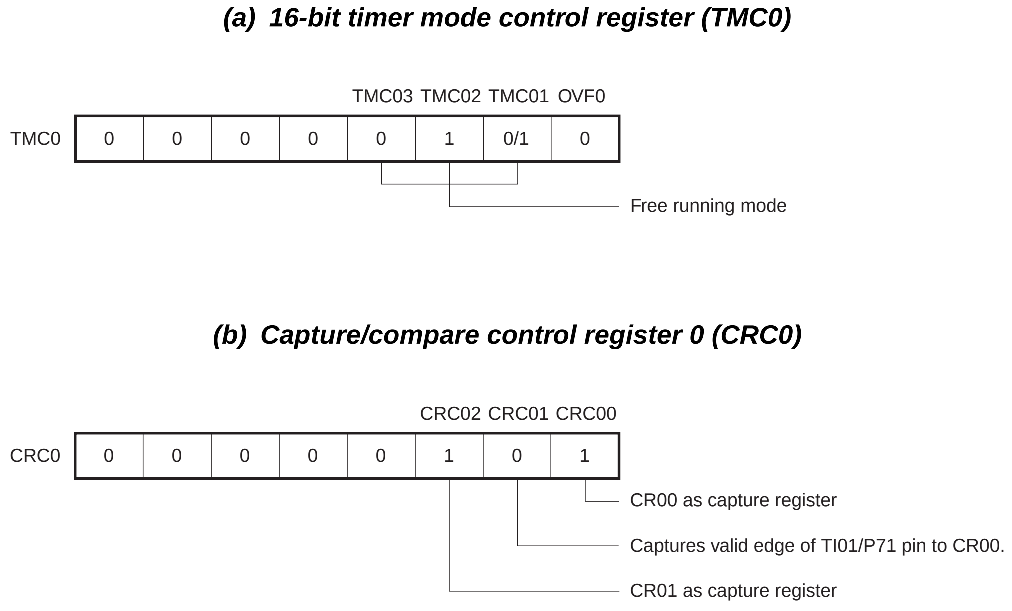
Figure 7-13: Timing of Pulse Width Measurement with Free Running Counter and One Capture Register (with both edges specified)



(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 pin and TI01 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)

					TMC03	TMC02	TMC01	OVF0
TMC0	0	0	0	0	0	1	0/1	0

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

						CRC02	CRC01	CRC00
CRC0	0	0	0	0	0	1	0	1

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.

(a) 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

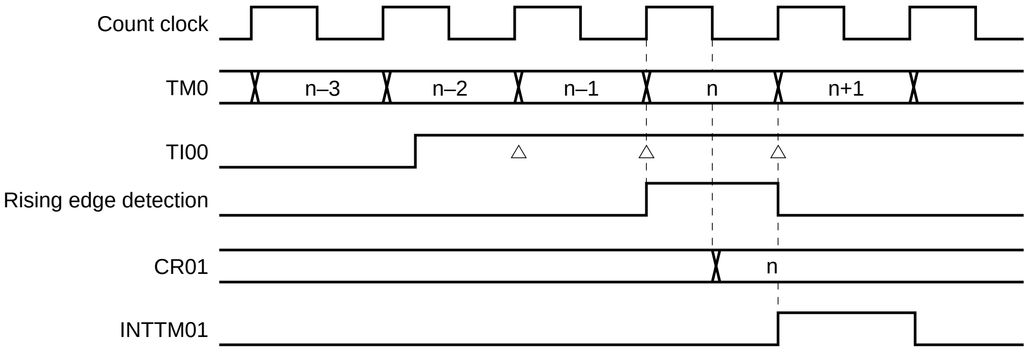
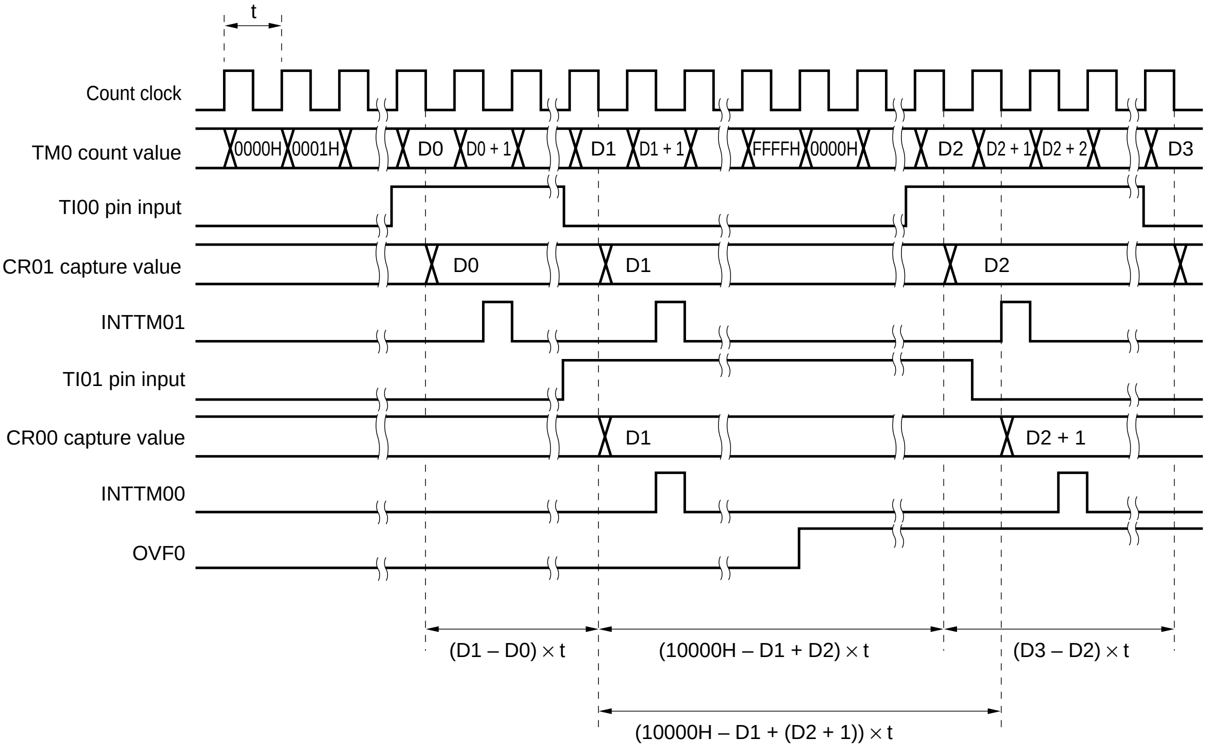


Figure 7-16: Timing of Pulse Width Measurement Operation with Free Running Counter (with both edges specified)



(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.

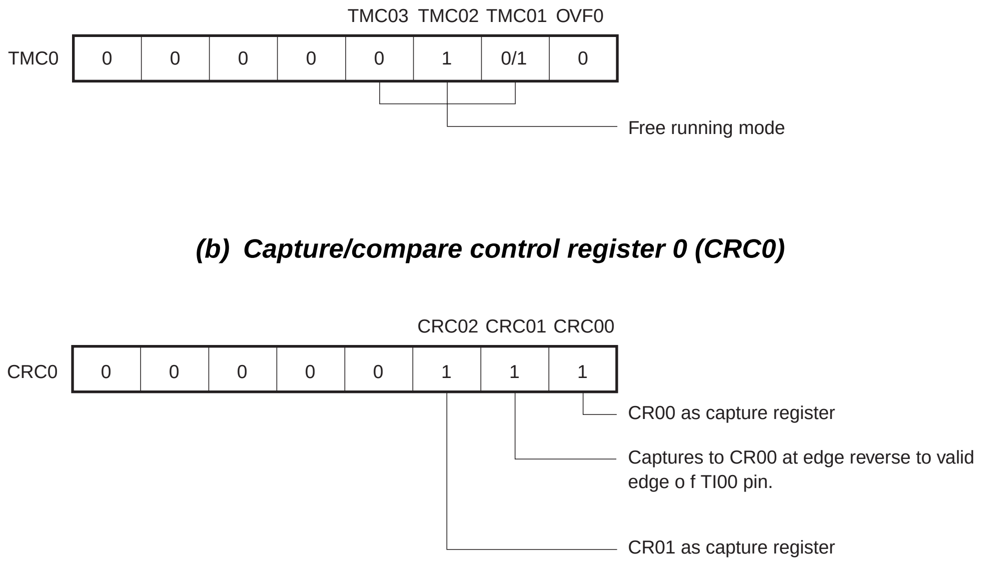
The edge of the TI00 pin is specified by bits 4 and 5 (ES00 and ES01) of the prescaler mode register 0 (PRM0). The rising or falling edge can be specified.

The valid edge of TI00 pin and TI01 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

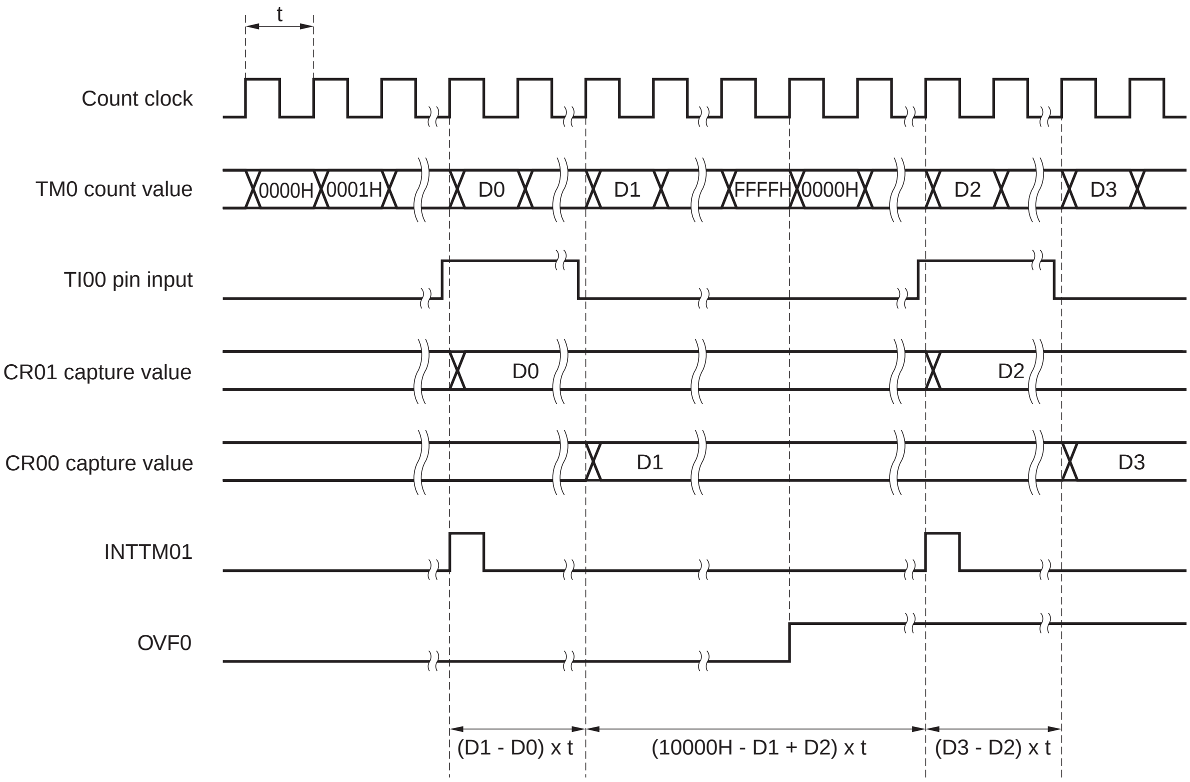
(a) 16-bit timer mode control register (TMC0)



(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)

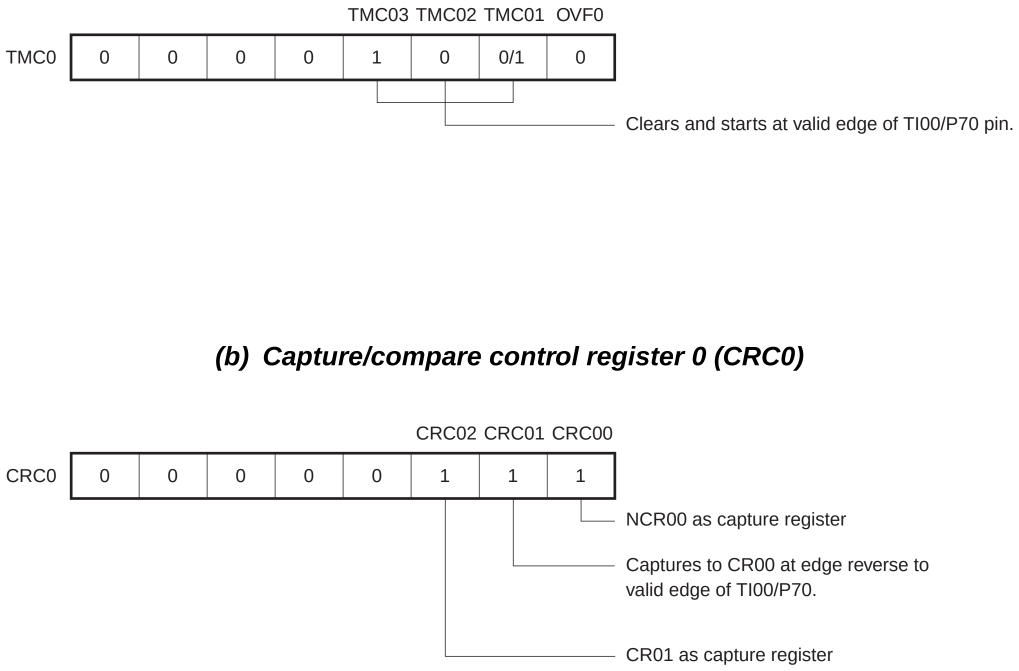


(4) Pulse width measurement by restarting

When the valid edge of the TI00 pin is detected, the pulse width of the signal input to the TI00 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



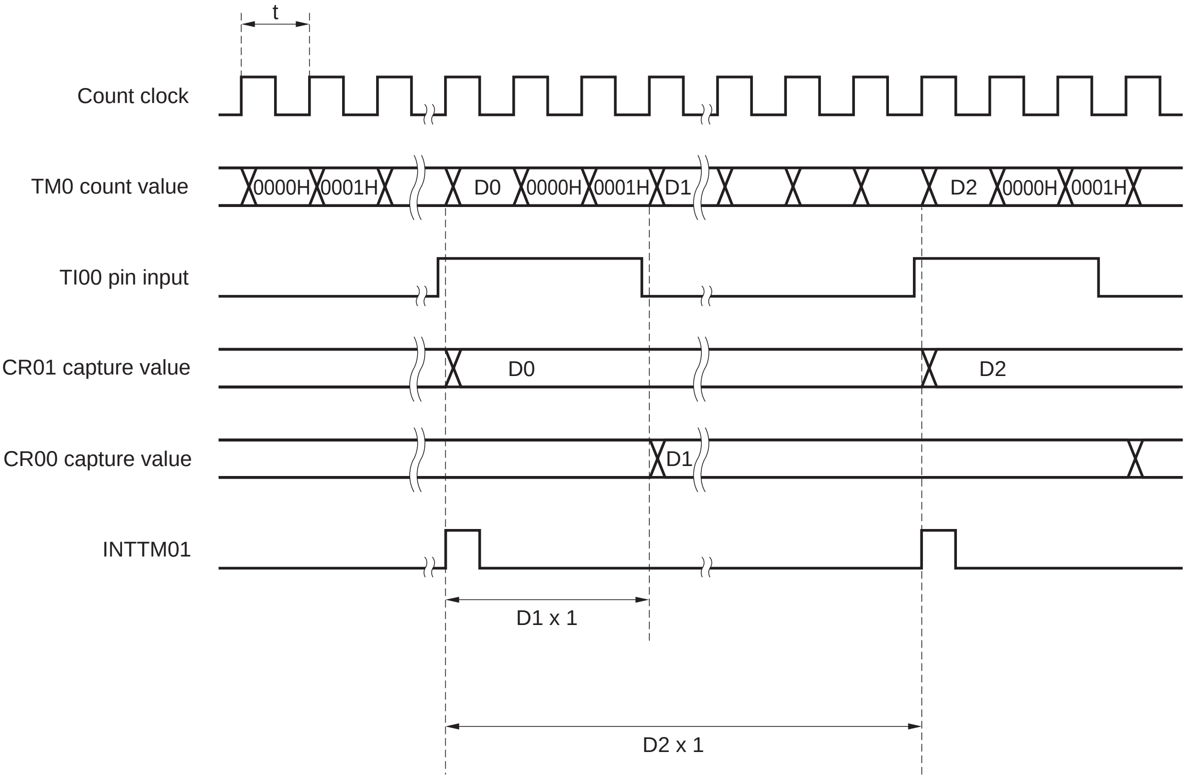
(a) 16-bit timer mode control register (TMC0)

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

TMC03 TMC02 TMC01 OVF0
TMC0
0
0
0
0
1
0
0/1
0
Clears and starts at valid edge of TI00/P70 pin.
CRC02 CRC01 CRC00
CRC0
0
0
0
0
0
1
1
1
NCR00 as capture register
Captures to CR00 at edge reverse to valid edge of TI00/P70.
CR01 as capture register

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.

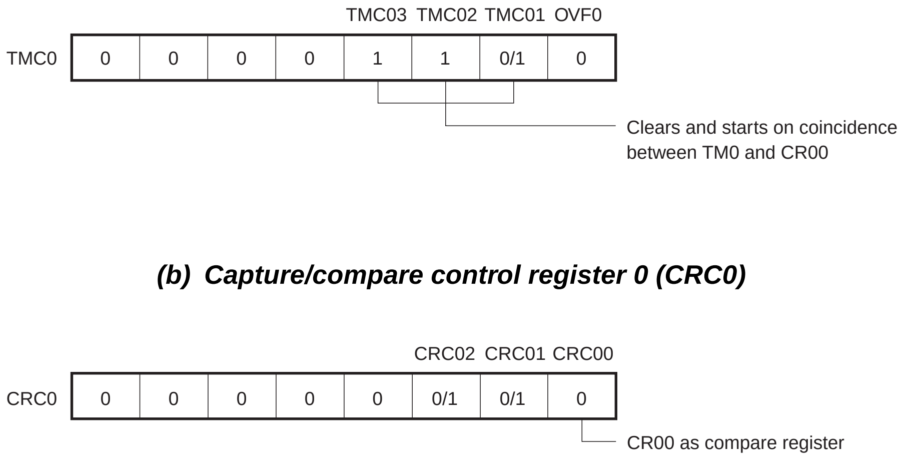
Figure 7-20: Timing of Pulse Width Measurement by Restarting (with rising edge specified)



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)

					TMC03	TMC02	TMC01	OVF0
TMC0	0	0	0	0	1	1	0/1	0

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

						CRC02	CRC01	CRC00
CRC0	0	0	0	0	0	0/1	0/1	0

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

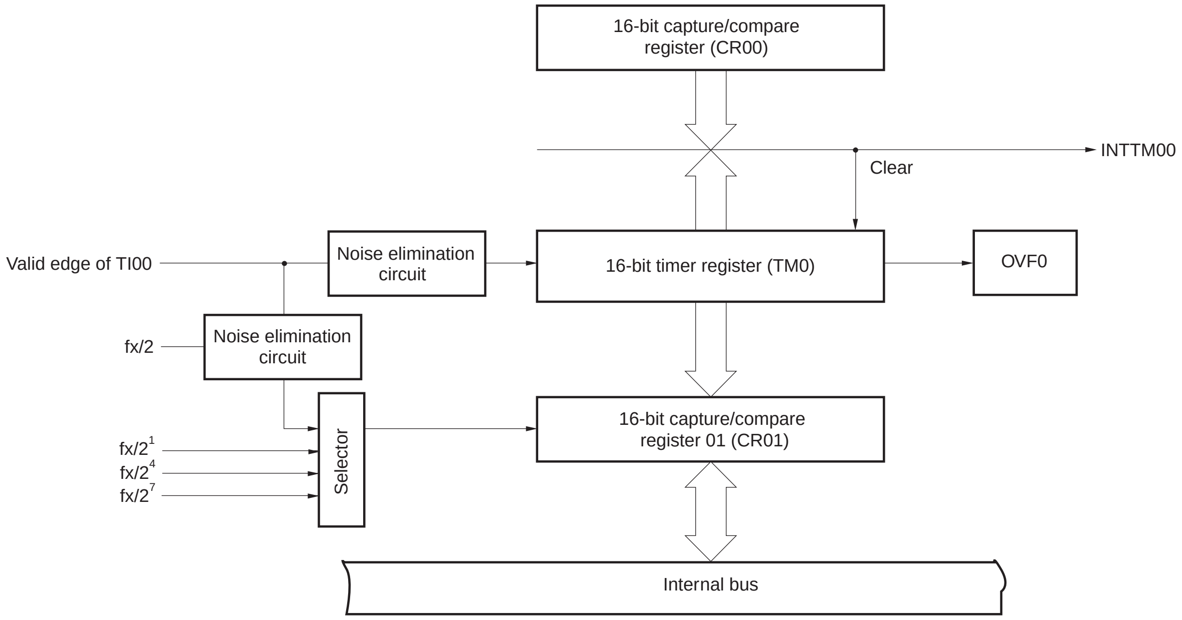
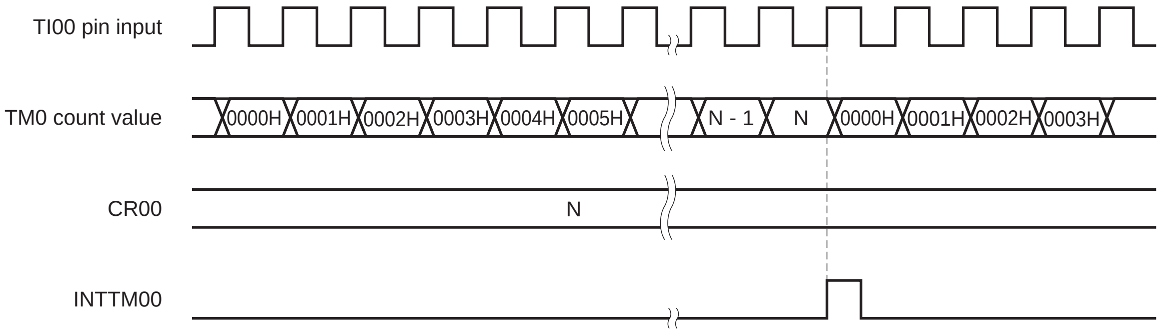

16-bit capture/compare register (CR00)
INTTM00
Clear
Valid edge of TI00
Noise elimination circuit
16-bit timer register (TM0)
OVF0
fx/2
Noise elimination circuit
16-bit capture/compare register 01 (CR01)
fx/2^1
fx/2^4
fx/2^7
Selector
Internal bus

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



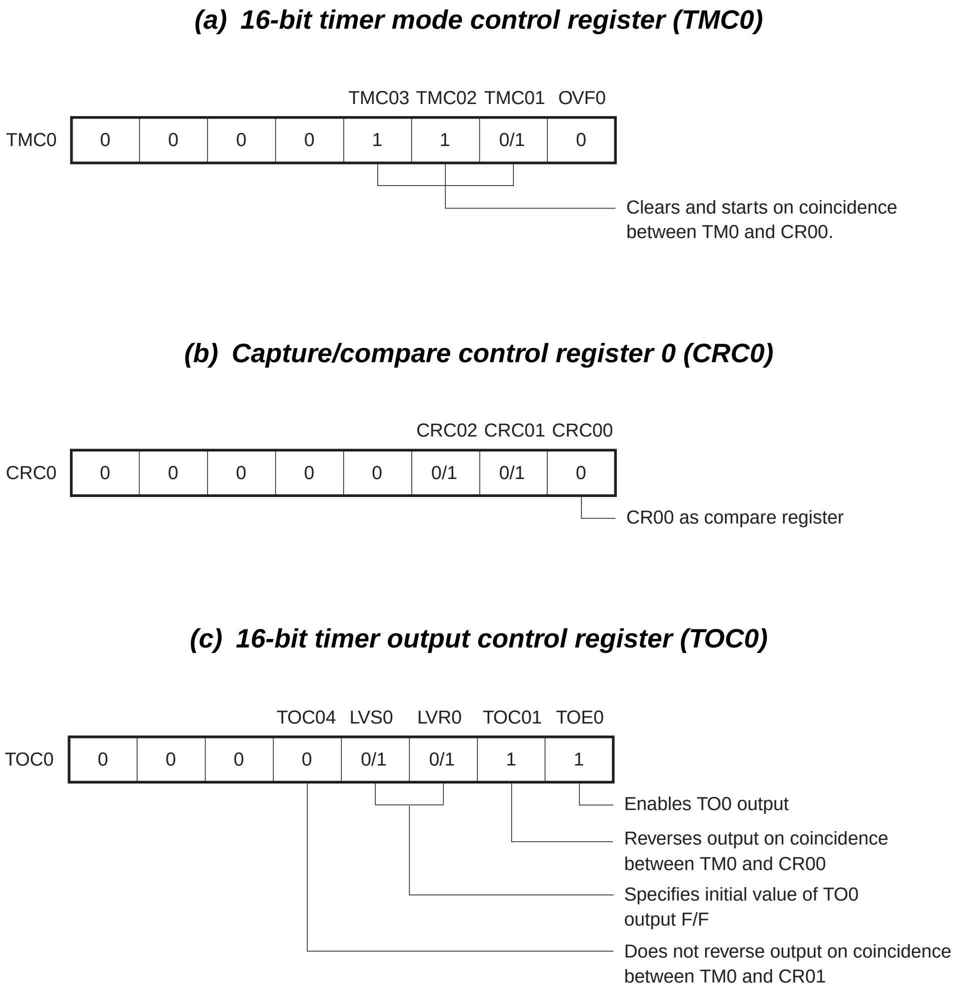
Caution: Read TM0 when reading the count value of the external event counter.

7.4.5 Operation to output square wave

The 16-bit timer/event counter 0 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 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)

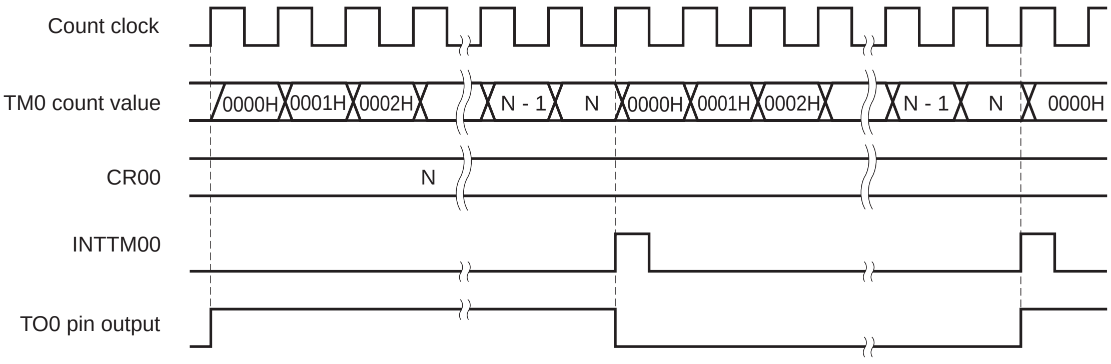
(b) Capture/compare control register 0 (CRC0)

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

TMC03 TMC02 TMC01 OVF0
TMC0 0 0 0 0 1 1 0/1 0
Clears and starts on coincidence between TM0 and CR00.
CRC02 CRC01 CRC00
CRC0 0 0 0 0 0 0/1 0/1 0
CR00 as compare register
TOC04 LVS0 LVR0 TOC01 TOE0
TOC0 0 0 0 0 0/1 0/1 1 1
Enables TO0 output
Reverses output on coincidence between TM0 and CR00
Specifies initial value of TO0 output F/F
Does not reverse output on coincidence between TM0 and CR01

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

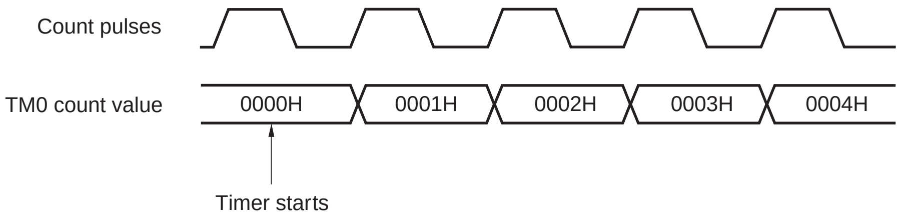


7.5 16-Bit Timer/Event Counter 0 Operating Precautions

(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 7-26: Start Timing of 16-Bit Timer Register



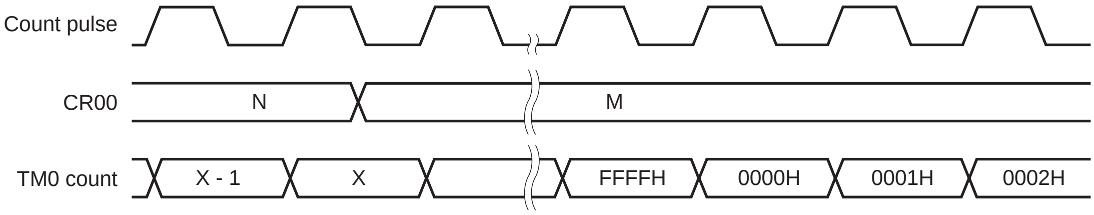
(2) 16-bit compare register setting

Set another value than 0000H to the 16-bit captured compare register CR00, CR01. This means, that an 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 7-27: Timing after Changing Compare Register during Timer Count Operation

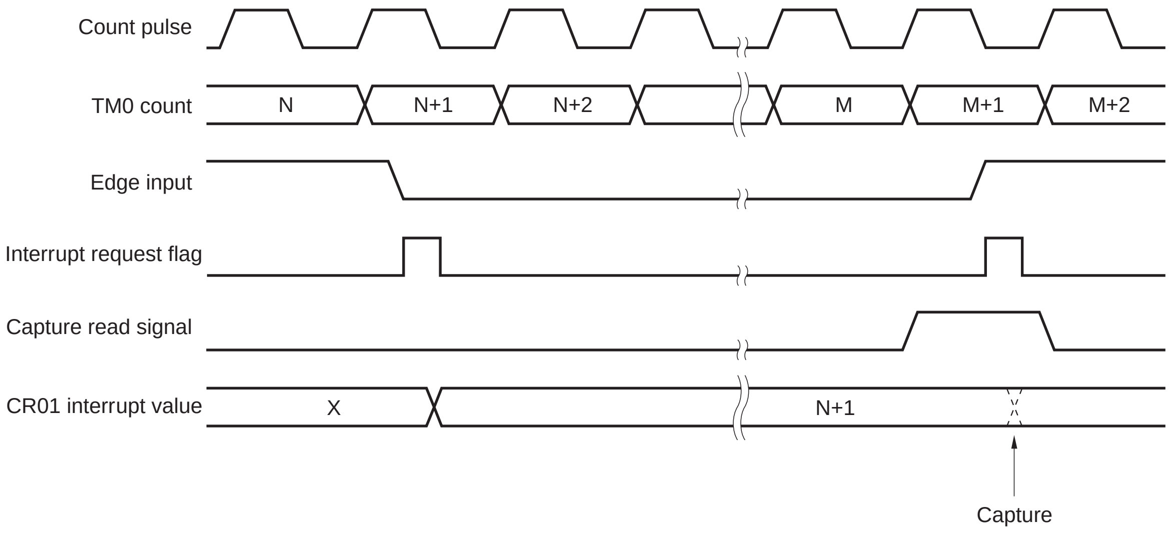


Remark: N > X > M

(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 7-28: Data Hold Timing of Capture Register



(5) Setting valid edge

Before setting the valid edge of the TI00/TO0/P70 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) 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.

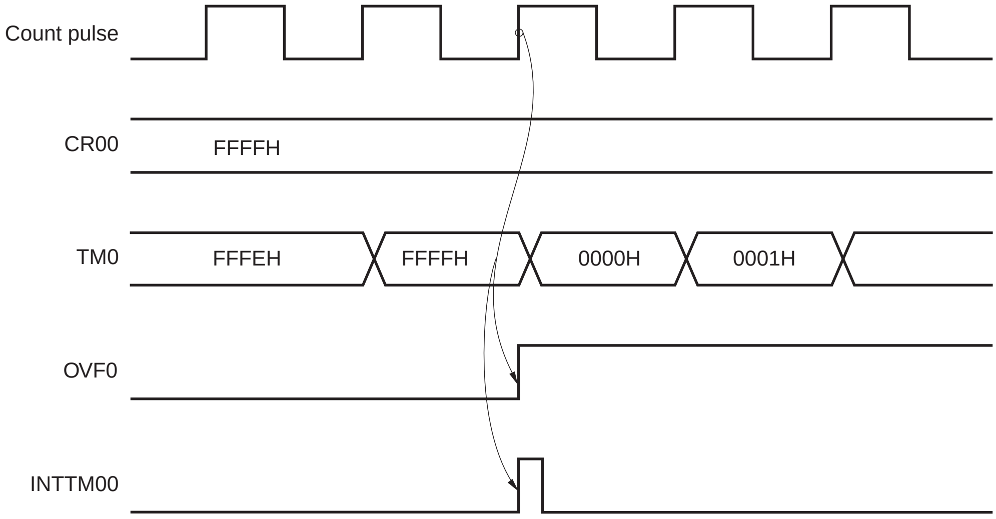
↓

Set CR00 to FFFFH

↓

When TM0 counts up from FFFFH to 0000H

Figure 7-29: Operation Timing of OVF0 Flag



(7) 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.

(8) Timer operation

(a) Even if the 16-bit timer counter 0 (TM0) is read, the value is not captured by 16-bit timer capture/compare register 01 (CR01).

(b) Regardless of the CPU's operation mode, when the timer stops, the input signals to pins TI00/TI01 are not acknowledged.

(9) Capture operation

(a) If TI00 is specified as the valid edge of the count clock, capture operation by the capture register specified as the trigger for TI00 is not possible.

(b) If both the rising and falling edges are selected as the valid edges of TI00, capture is not performed.

(c) To ensure the reliability of the capture operation, the capture trigger requires a pulse two times longer than the count clock selected by prescaler mode register 0 (PRM0).

(d) The capture operation is performed at the fall of the count clock. An interrupt request input (INTTM0n), however, is generated at the rise of the next count clock.

(10) Compare operation

(a) The INTTM0n may not be generated if the set value of 16-bit timer capture registers 00, 01 (CR00, CR01) and the count value of 16-bit timer counter (TM0) match and CR00 and CR01 are overwritten at the timing of INTTM0n generation. Therefore, do not overwrite CR00 and CR01 frequently even if overwriting the same value.

(b) Capture operation may not be performed for CR00/CR01 set in compare mode even if a capture trigger has been input.

Chapter 8   16-Bit Timer 2

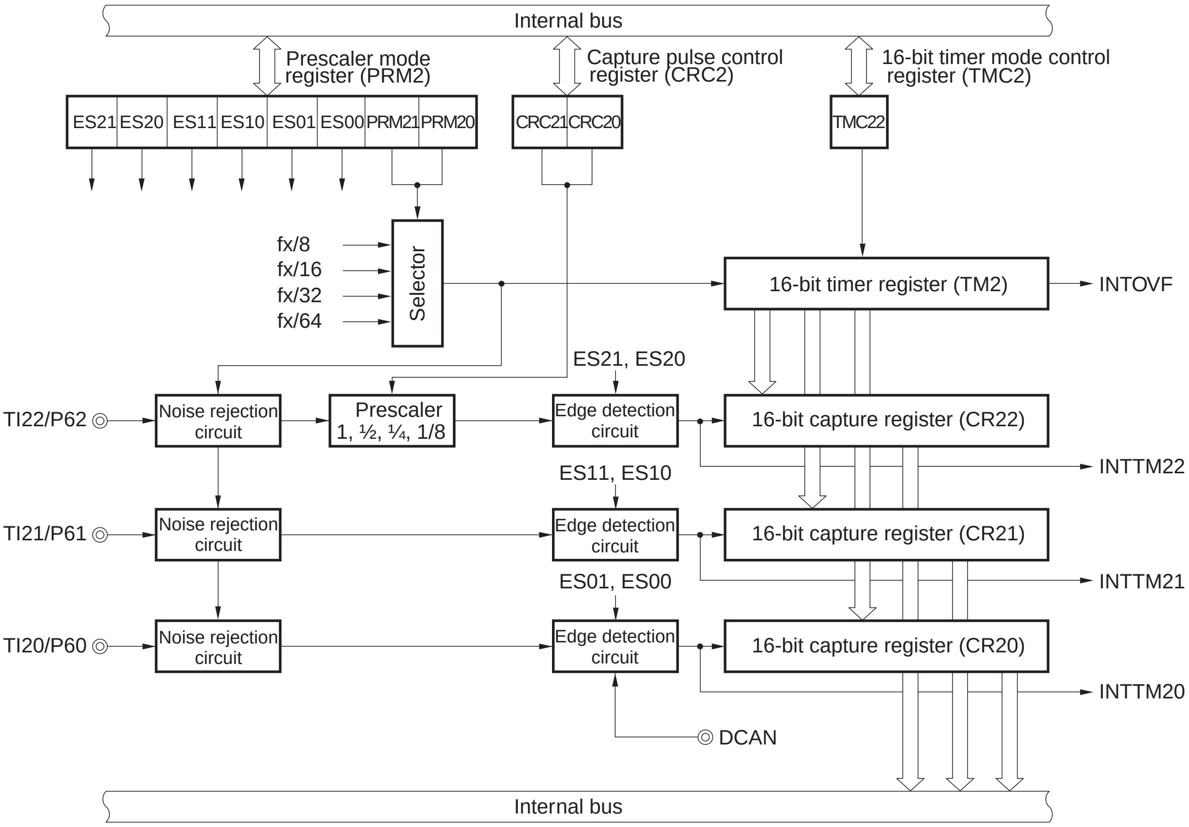
8.1   16-Bit Timer 2 Functions

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 2 Block Diagram



(1)   Pulse width measurement

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

(2)   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.
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/P60 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/P61 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/P62 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 an 1-bit or an 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

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

TMC22	Timer 2 Operating Mode Selection
0	Operation stop (TM2 cleared to 0)
1	Operation enabled

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

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

(2) Capture pulse control register (CRC2)

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

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

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

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

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

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

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

(3) Prescaler mode register (PRM2)

This register is used to set 16-bit timer (TM2) count clock and valid edge of 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

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PRM2	ES21	ES20	ES11	ES10	ES01	ES00	PRM21	PRM20	R/W	FF61H	00H

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

ES11	ES10	Selection of valid edge of TI21
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 TI20
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling and rising edges

PRM21	PRM20	Selection of count clock
0	0	$f_X/2^3$
0	1	$f_X/2^4$
1	0	$f_X/2^5$
1	1	$f_X/2^6$

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/P60 to TI22/P62 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/P60 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

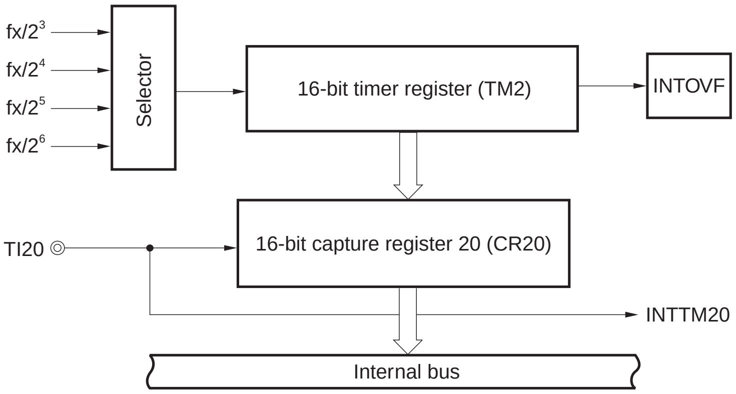
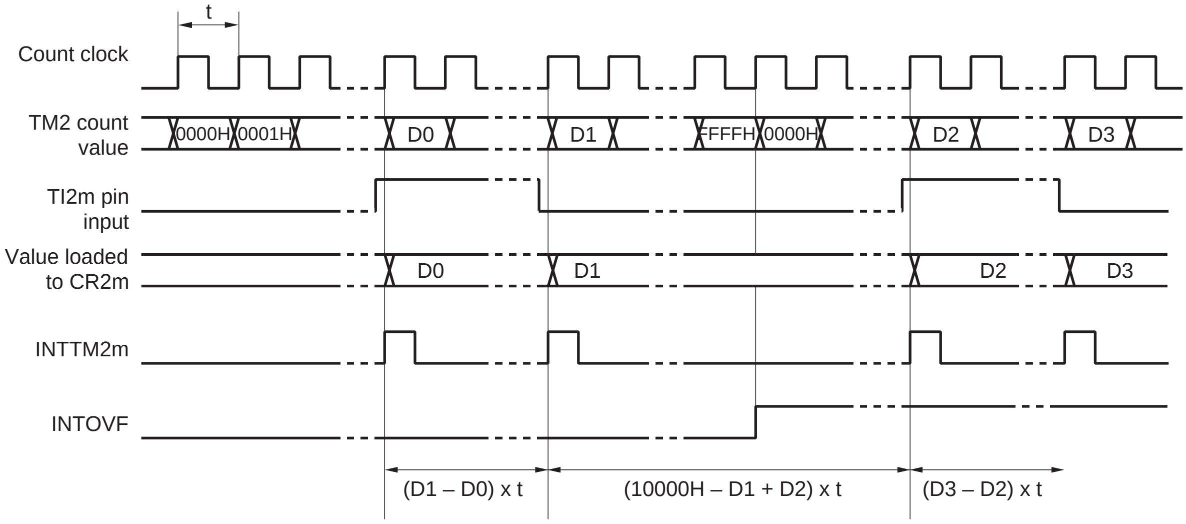


Figure 8-6: Timing of Pulse Width Measurement Operation by Using the Free Running Counter and One Capture Register (with Both Edges Specified)



Remark: m = 0 to 2

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

The 16-bit timer register (TM2) allows simultaneous measurement of the pulse widths of the three signals input to the TI20/P60 to TI22/P62 pins.

When the edge specified by bits 2 and 3 (ES00 and ES01) of prescaler mode register (PRM2) is input to the TI20/P60 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/P61 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/P62 pin, the value of TM2 is taken into 16-bit capture register 22 (CR22) and external interrupt request signal (INTTM22) is set.

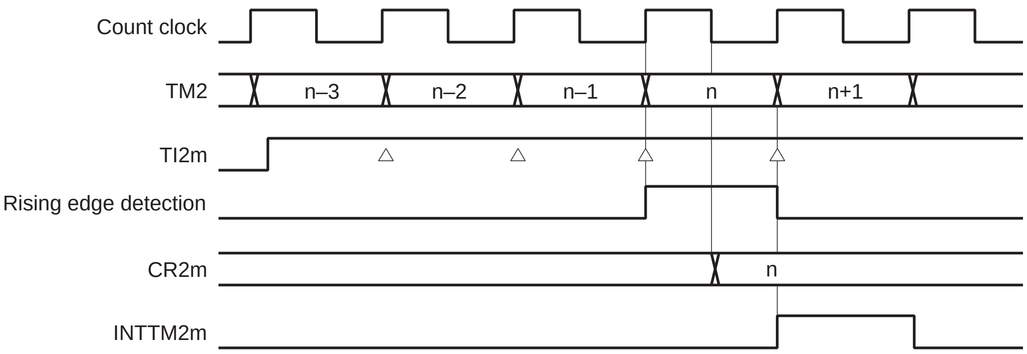
Any of three edge specifications can be selected - rising, falling, or both edges - as the valid edges for the TI20/P60 to TI22/P62 pins by means of bits 2 and 3 (ES00 and ES01), bits 4 and 5 (ES10 and ES11), and bits 6 and 7 (ES06 and ES07) of PRM2, respectively.

For TI20/P60 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**

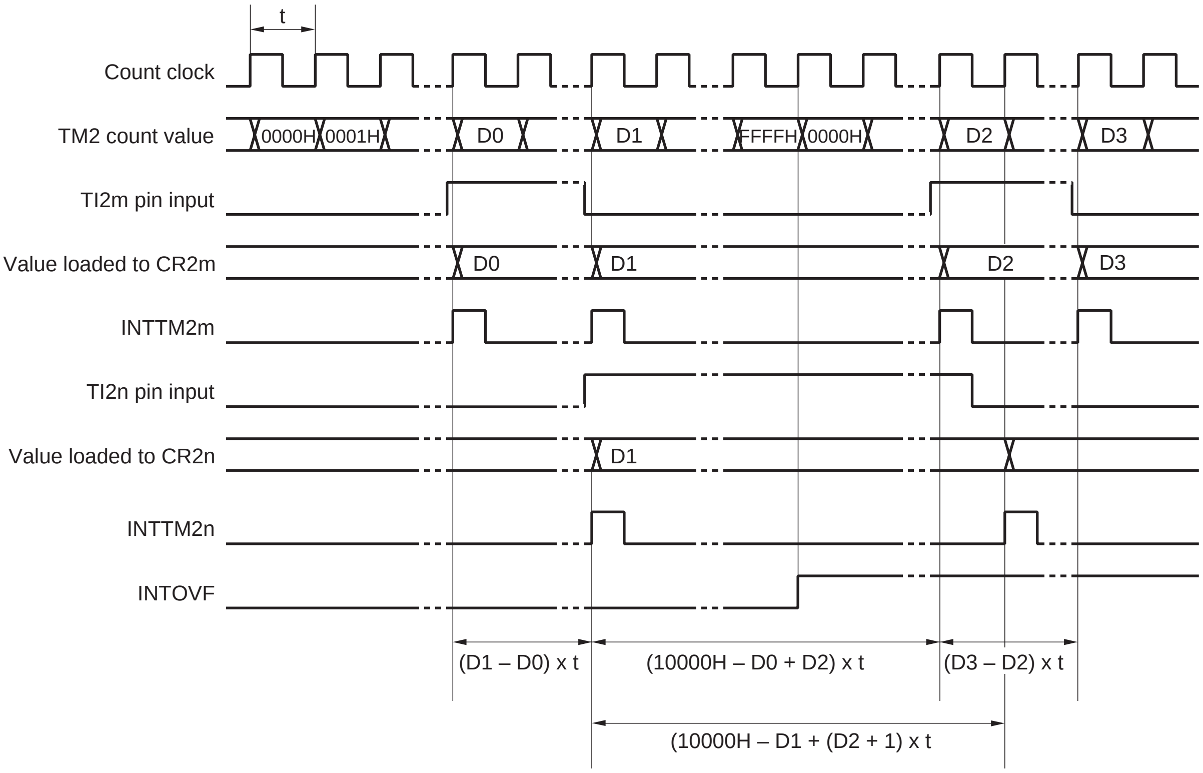
Capture register operation in capture trigger input is shown.

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)



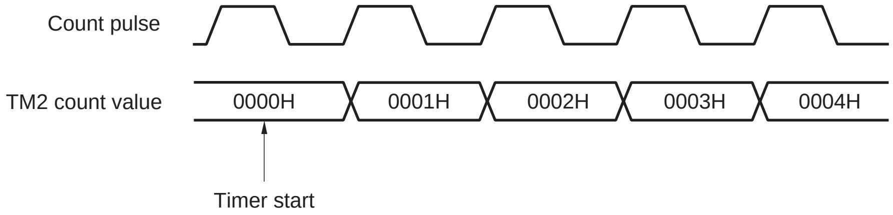
Remark: m = 0 to 2
n = 1, 2

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.

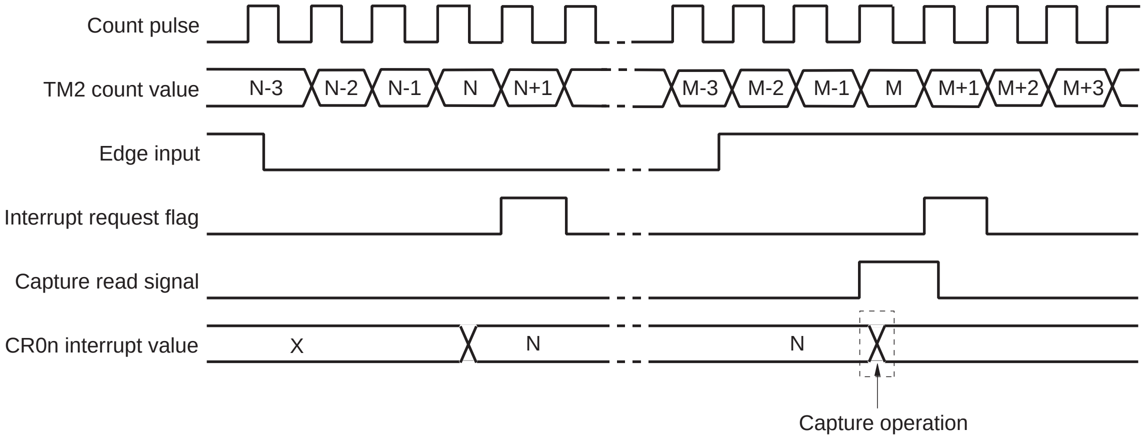
Figure 8-9: 16-Bit Timer Register Start Timing



(2) Capture register data retention timings

If the valid edge of the TI2m/P6m 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: n = 0 to 2

(3) Valid edge setting

Set the valid edge of the TI2m/P6m 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.

(5) The value of the timer register

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

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

9.1 8-Bit Timer/Event Counters 50 and 51 Functions

The timer 50 and 51 have the following two modes:

- Mode using TM50 and TM51 alone (individual mode)
- Mode using the cascade connection (16-bit cascade mode connection).

(1) Mode using TM50 and TM51 alone

The timer operate as 8-bit timer/event counters.
They have the following functions:

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

(2) Mode using the cascade connection

The timer operates as 16-bit timer/event counter.
It has the following functions:

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

9.1.1 8-bit operation modes

(1) 8-bit interval timer

Interrupts are generated at the present time intervals.

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

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

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

Minimum Interval Width	Maximum Interval Width	Resolution
$1/f_X$ (125 ns)	$2^8 \times 1/f_X$ (32 µs)	$1/f_X$ (125 ns)
$2^1 \times 1/f_X$ (250 ns)	$2^9 \times 1/f_X$ (64 µs)	$2^1 \times 1/f_X$ (250 ns)
$2^3 \times 1/f_X$ (1 µs)	$2^{11} \times 1/f_X$ (256 µs)	$2^3 \times 1/f_X$ (1 µs)
$2^5 \times 1/f_X$ (4 µs)	$2^{13} \times 1/f_X$ (1 ms)	$2^5 \times 1/f_X$ (4 µs)
$2^8 \times 1/f_X$ (32 µs)	$2^{16} \times 1/f_X$ (8 ms)	$2^8 \times 1/f_X$ (32 µs)
$2^9 \times 1/f_X$ (64 µs)	$2^{17} \times 1/f_X$ (16 ms)	$2^9 \times 1/f_X$ (64 µs)

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

2. Values in parentheses when operated at f_X = 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 Interval Width	Maximum Interval Width	Resolution
$1/f_X$ (125 ns)	$2^8 \times 1/f_X$ (32 µs)	$1/f_X$ (125 ns)
$2^1 \times 1/f_X$ (250 ns)	$2^9 \times 1/f_X$ (64 µs)	$2^1 \times 1/f_X$ (250 ns)
$2^3 \times 1/f_X$ (1 µs)	$2^{11} \times 1/f_X$ (256 µs)	$2^3 \times 1/f_X$ (1 µs)
$2^5 \times 1/f_X$ (4 µs)	$2^{13} \times 1/f_X$ (1 ms)	$2^5 \times 1/f_X$ (4 µs)
$2^7 \times 1/f_X$ (16 µs)	$2^{15} \times 1/f_X$ (4 ms)	$2^7 \times 1/f_X$ (16 µs)
$2^{12} \times 1/f_X$ (512 µs)	$2^{20} \times 1/f_X$ (131 ms)	$2^{12} \times 1/f_X$ (512 µs)

Table 9-4: 8-Bit Timer/Event Counter 51 Square-Wave Output Ranges

Minimum Interval Width	Maximum Interval Width	Resolution
$1/f_X$ (125 ns)	$2^8 \times 1/f_X$ (32 µs)	$1/f_X$ (125 ns)
$2^1 \times 1/f_X$ (250 ns)	$2^9 \times 1/f_X$ (64 µs)	$2^1 \times 1/f_X$ (250 ns)
$2^3 \times 1/f_X$ (1 µs)	$2^{11} \times 1/f_X$ (256 µs)	$2^3 \times 1/f_X$ (1 µs)
$2^5 \times 1/f_X$ (4 µs)	$2^{13} \times 1/f_X$ (1 ms)	$2^5 \times 1/f_X$ (4 µs)
$2^8 \times 1/f_X$ (32 µs)	$2^{16} \times 1/f_X$ (8 ms)	$2^8 \times 1/f_X$ (32 µs)
$2^9 \times 1/f_X$ (64 µs)	$2^{17} \times 1/f_X$ (16 ms)	$2^9 \times 1/f_X$ (64 µs)

Remarks: 1. f_X: Main system clock oscillation frequency

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

(4) PWM output

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

9.1.2 16-bit operation modes

(1) Interval timer

Interrupts are generated at the present interval time.

Table 9-5: 16-Bit Timer/Event Counter TM50/TM51 Interval Times

Minimum Interval Width	Maximum Interval Width	Resolution
$1/f_X$ (125 ns)	$2^{16} \times 1/f_X$ (8 ms)	$1/f_X$ (125 ns)
$2^1 \times 1/f_X$ (250 ns)	$2^{17} \times 1/f_X$ (16 ms)	$2^1 \times 1/f_X$ (250 ns)
$2^3 \times 1/f_X$ (1 µs)	$2^{19} \times 1/f_X$ (65.5 ms)	$2^3 \times 1/f_X$ (1 µs)
$2^5 \times 1/f_X$ (4 µs)	$2^{21} \times 1/f_X$ (262 ms)	$2^5 \times 1/f_X$ (4 µs)
$2^7 \times 1/f_X$ (16 µs)	$2^{23} \times 1/f_X$ (1.05 s)	$2^7 \times 1/f_X$ (16 µs)
$2^{12} \times 1/f_X$ (512 µs)	$2^{28} \times 1/f_X$ (33.6 s)	$2^{12} \times 1/f_X$ (512 µs)

(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 9-6: 16-Bit Timer/Event Counter TM50/TM51 Square-Wave Output Ranges

Minimum Interval Width	Maximum Interval Width	Resolution
$1/f_X$ (125 ns)	$2^{16} \times 1/f_X$ (8 ms)	$1/f_X$ (125 ns)
$2^1 \times 1/f_X$ (250 ns)	$2^{17} \times 1/f_X$ (16 ms)	$2^1 \times 1/f_X$ (250 ns)
$2^3 \times 1/f_X$ (1 µs)	$2^{19} \times 1/f_X$ (65.5 ms)	$2^3 \times 1/f_X$ (1 µs)
$2^5 \times 1/f_X$ (4 µs)	$2^{21} \times 1/f_X$ (262 ms)	$2^5 \times 1/f_X$ (4 µs)
$2^7 \times 1/f_X$ (16 µs)	$2^{23} \times 1/f_X$ (1.05 s)	$2^7 \times 1/f_X$ (16 µs)
$2^{12} \times 1/f_X$ (512 µs)	$2^{28} \times 1/f_X$ (33.6 s)	$2^{12} \times 1/f_X$ (512 µs)

Remarks: 1. f_X: Main system clock oscillation frequency

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

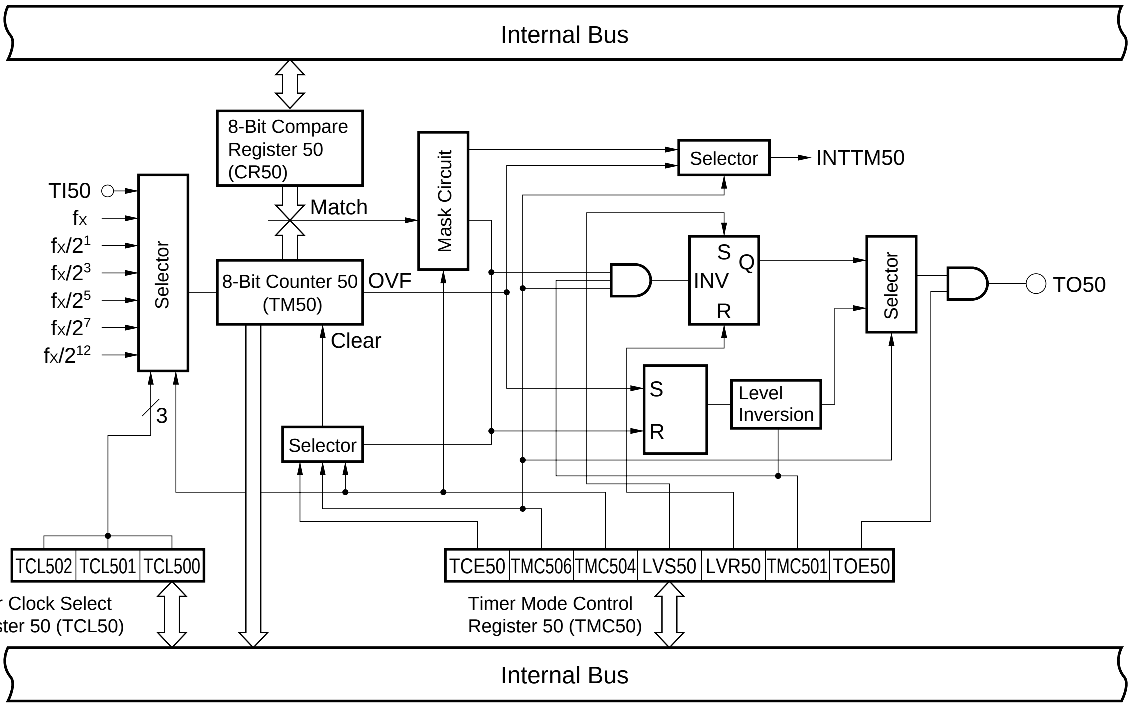
9.2 8-Bit Timer/Event Counters 50 and 51 Configurations

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

Table 9-7: 8-Bit Timer/Event Counters 50 and 51 Configurations

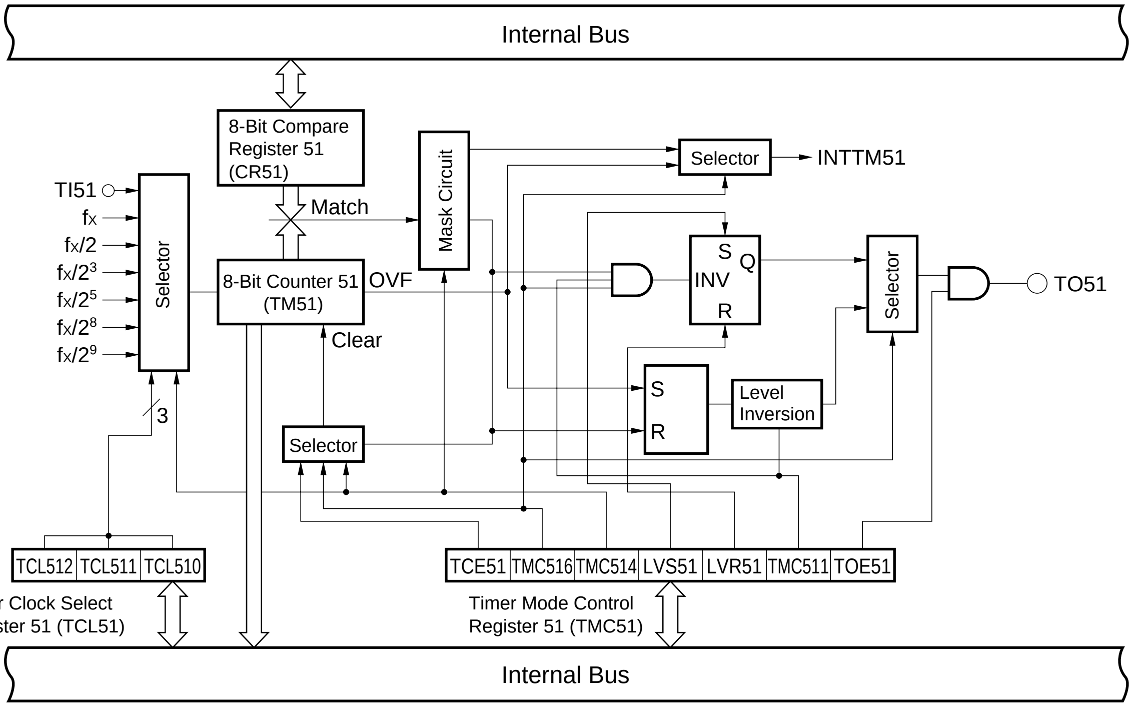
Item	Configuration
Timer register	8 bits x 2 (TM50, TM51)
Register	Compare register 8 bits x 2 (CR50, CR51)
Timer output	2 (TO50, TO51)
Control register	Timer clock select register 50 and 51 (TCL50, TCL51)
	8-bit timer mode control registers 50 and 51 (TMC50, TMC51)
	Port mode register 2 (PM2)

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



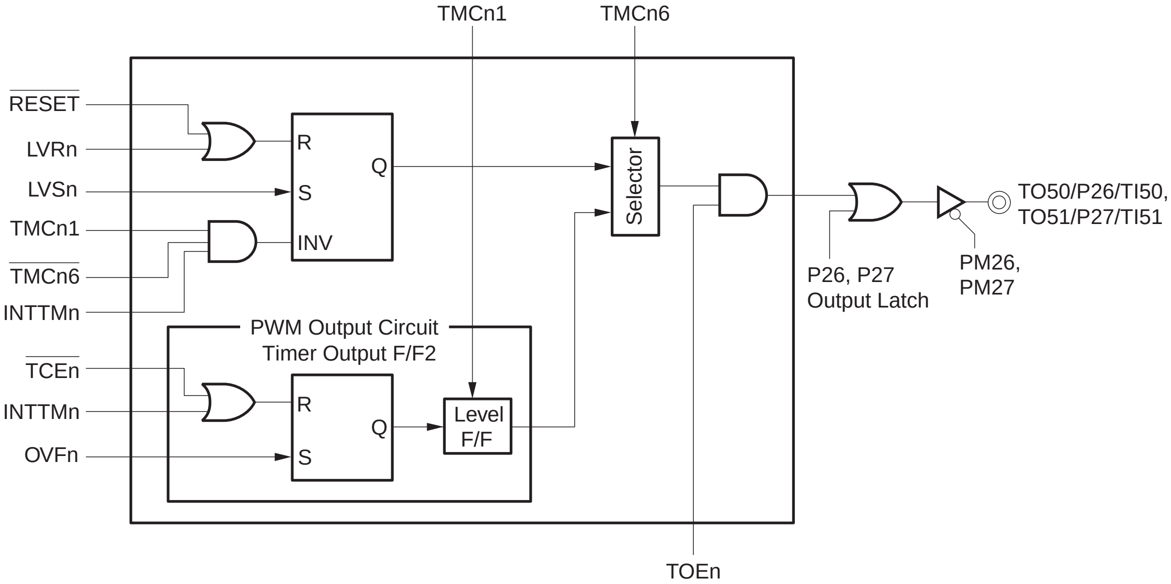
Note: Refer to Figure 9-2 for details of configurations of 8-bit timer/event counters 50 and 51 output control circuits.

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, CR51)

These 8-bit registers compare the value set to CR50 to 8-bit timer register 5 (TM50) count value, and the value set to CR51 to the 8-bit timer register 51 (TM51) count value, and, if they match, generate interrupts request (INTTM50 and INTTM51, respectively).
CR50 and CR51 are set with an 8-bit memory manipulation instruction. They cannot be set with a 16-bit memory manipulation instruction. The 00H to FFH values can be set.

$\overline{\text{RESET}}$ input sets CR50 and CR51 values to 00H.

Cautions: 1. To use PWM mode, set CRn value before setting TMCn (n = 50, 51) to PWM mode.

2. If the data is set in cascade mode, always set it after stopping the timer.

(2) 8-bit timer registers 50 and 51 (TM50, TM51)

These 8-bit registers count pulses.
TM50 and TM51 are read with an 8-bit memory manipulation instruction.

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

Caution: The cascade connection time becomes 00H even when the bit TCE50 of the timer TM50 is cleared.

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 2 (PM2)

(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

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

TCL502	TCL501	TCL500	8-bit Timer Register 50 Count Clock Selection
0	0	0	TI50 falling edge Note
0	0	1	TI50 rising edge Note
0	1	0	f_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	$f_X/2^{12}$ (1.65 kHz)
Other than above			Setting prohibited

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

Cautions: 1. When rewriting TCL50 to other data, stop the timer operation beforehand.

2. Set always bits 3 to 7 to "0".

Remarks: 1. f_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

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

TCL512	TCL511	TCL510	8-bit Timer Register 51 Count Clock Selection
0	0	0	TI51 falling edge Note
0	0	1	TI51 rising edge Note
0	1	0	f_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	$f_X/2^8$ (31.25 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.

Cautions: 1. When rewriting TCL51 to other data, stop the timer operation beforehand.

2. Set always bits 3 to 7 to "0".

Remarks: 1. f_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 an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC50 to 00H.

Figure 9-6: 8-Bit Timer Mode Control Register 50 Format

	<7>	6	5	4	<3>	<2>	1	<0>	R/W	Address	After Reset
TMC50	TCE50	TMC506	0	0	LVS50	LVR50	TMC501	TOE50	R/W	FF70H	00H

TOE50	8-Bit Timer/Event Counter 50 Output Control
0	Output disabled (Port mode)
1	Output enabled

TMC501	In PWM Mode	In Other Mode
	Active level selection	Timer output F/F1 control
0	Active high	Inversion operation disabled
1	Active low	Inversion operation enabled

LVS50	LVR50	8-Bit Timer/Event Counter 50 Timer Output F/F1 Status Setting
0	0	No change
0	1	Timer output F/F1 reset (0)
1	0	Timer output F/F1 set (1)
1	1	Setting prohibited

TMC506	8-Bit Timer/Event Counter 50 Operating Mode Selection
0	Clear & start mode on match of TM50 and CR50
1	PWM mode (free-running)

TCE50	8-Bit Timer Register 50 Operation Control
0	Operation Stop (TM50 clear to 0)
1	Operation Enable

Cautions: 1. Timer operation must be stopped before setting TMC50.

2. If LVS50 and LVR50 are read after data are set, they will be 0.

3. Be sure to set bit 4 and bit 5 to 0.

(4) 8-bit timer mode control register 51 (TMC51)

This register enables/stops operation of 8-bit timer register 51, sets the operating mode of 8-bit timer register 51 and controls operation of 8-bit timer/event counter 51 output control circuit.

It selects the R-S flip-flop (timer output F/F 1, 2) setting/resetting, active level in PWM mode, inversion enabling/disabling in modes other than PWM mode and 8-bit timer/event counter 51 timer output enabling/disabling.

TMC51 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC51 to 00H.

Figure 9-7: 8-Bit Timer Mode Control Register 51 Format (1/2)

	<7>	6	5	4	<3>	<2>	1	<0>	R/W	Address	After Reset
TMC51	TCE51	TMC516	0	TMC514	LVS51	LVR51	TMC511	TOE51	R/W	FF74H	00H

TOE51	8-Bit Timer/Event Counter 51 Output Control
0	Output disabled (Port mode)
1	Output enabled

TMC511	In PWM Mode	In Other Mode
	Active level selection	Timer output F/F1 control
0	Active high	Inversion operation disabled
1	Active low	Inversion operation enabled

LVS51	LVR50	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

TMC514	Individual of cascade mode connection
0	Individual mode (8-bit timer/counter mode)
1	Cascade connection mode (16-bit timer/counter mode)

Figure 9-7: 8-Bit Timer Mode Control Register 51 Format (2/2)

TMC516	8-Bit Timer/Event Counter 51 Operating Mode Selection
0	Clear & start mode on match of TM51 and CR51
1	PWM mode (free-running)

TCE51	8-Bit Timer Register 51 Operation Control
0	Operation Stop (TM51 clear to 0)
1	Operation Enable

Cautions: 1. Timer operation must be stopped before setting TMC51.

2. If LVS51 and LVR51 are read after data are set, they will be 0.

3. Be sure to set bit 5 to 0.

(5) Port mode register 2 (PM2)

This register sets port 2 input/output in 1-bit units.
When using the P26/TI50/TO50 and P27/TI51/TO51 pins for timer output, set PM26, PM27 and the output latches of P26 and P27 to 0.
PM2 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM2 to FFH.

Figure 9-8: Port Mode Register 2 Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	R/W	FF22H	FFH

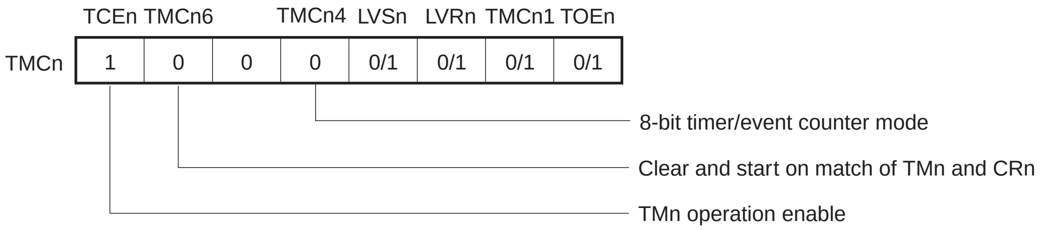
PM2n	PM2n Input/Output mode Selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

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

9.4.1 Interval timer operations (8-bit timer/event counter mode)

Setting the 8-bit timer mode control registers (TMC50 and TMC51) as shown in Figure 9-9 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



Setting Method

(1) Set each register

TCL5n	:	Selects the count clock
CR5n	:	Compare value
TMC5n	:	Selects the clear and start mode when TM5n and CR5n match. (TMC5n = 0000xxxx0B, x is not done care).

(2) When TCE5n = 1 is set, counting starts.

(3) When the values of TM5n and CR5n match, INTTM5n is generated (TM5n is cleared to 00H).

(4) Then, INTTM5n is repeatedly generated during the same interval. When counting stops, set TCE5n = 0.

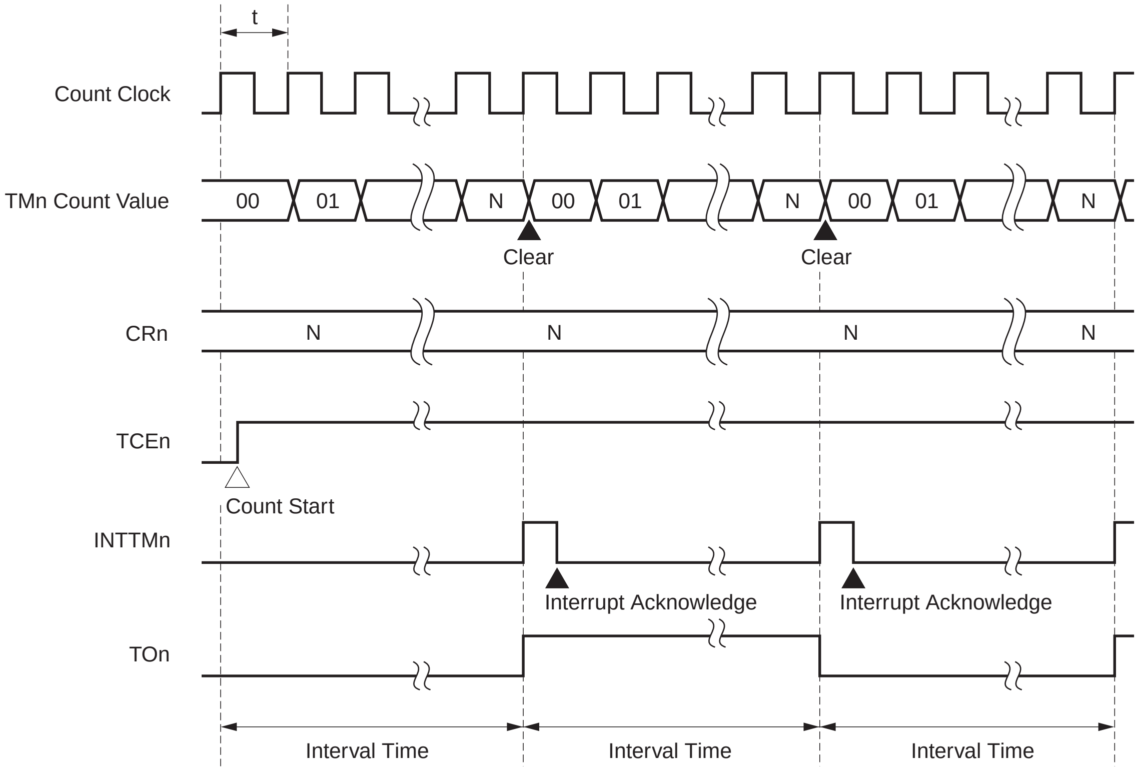
Remarks: **1.** 0/1: Setting 0 or 1 allows another function to be used simultaneously with the interval timer. See 9.3 (3)”8-bit timer mode control register 50 (TMC50)” on page 160 and (4)”8-bit timer mode control register 51 (TMC51)” on page 161 for details.

2. n = 50, 51

3. TMC5n4 is only available at TM51.

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

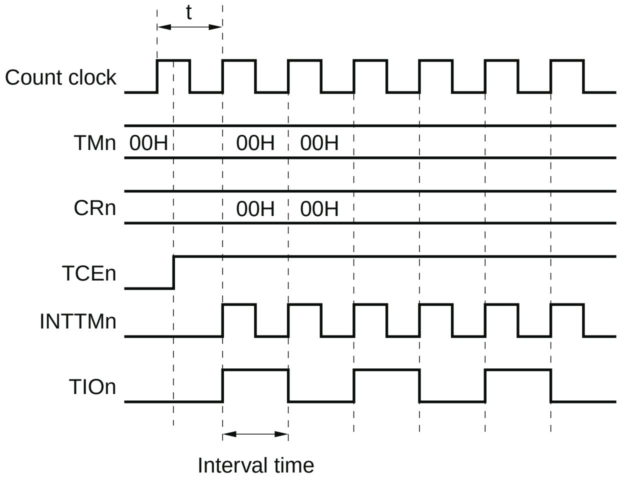
(a) When N = 00H to FFH



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

2. n = 50, 51

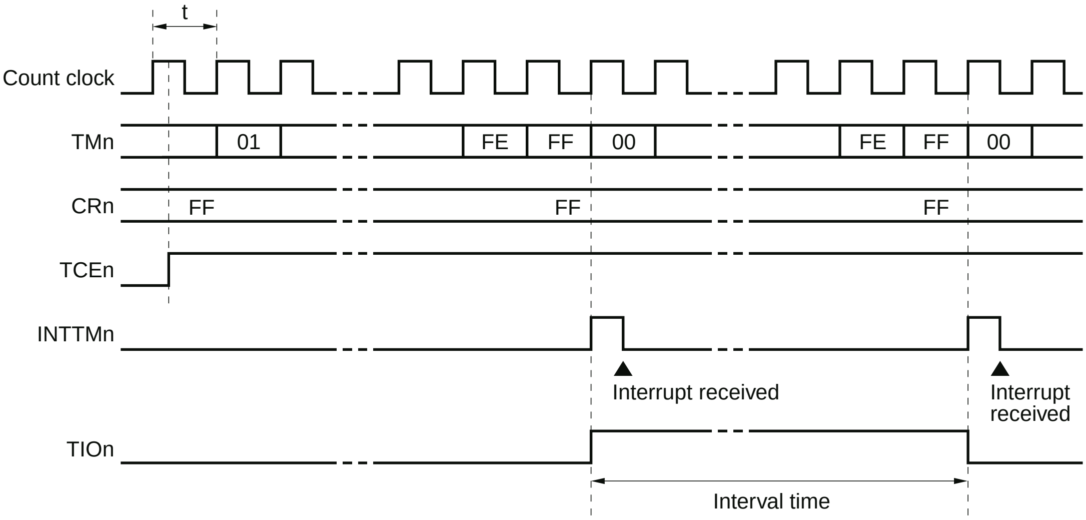
(b) When CRn = 00H



Remark: n = 50, 51

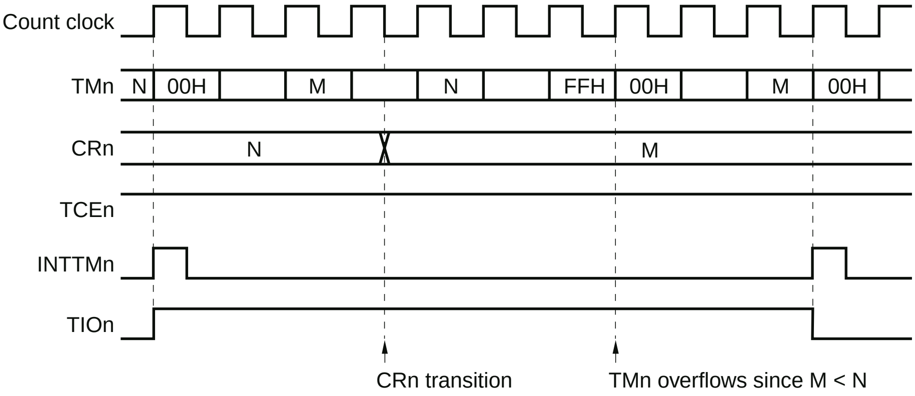
Figure 9-10: Interval Timer Operation Timings (2/3)

(c) When CRn = FFH



Remark: n = 50, 51

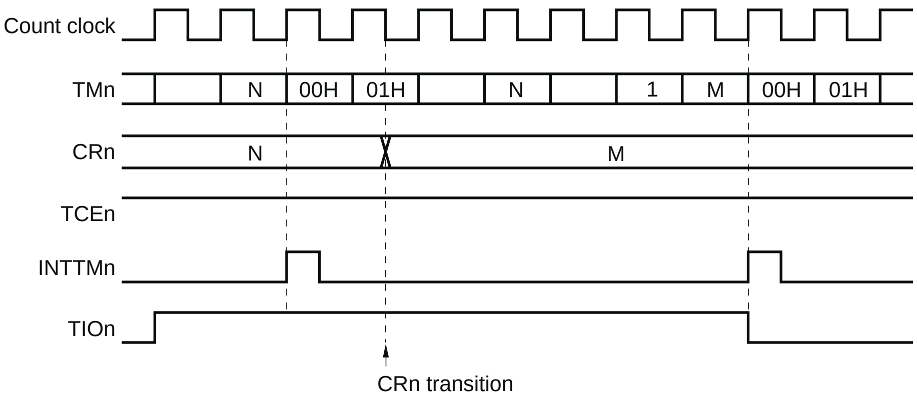
(d) Operated by CR5n transition (M < N)



Remark: n = 50, 51

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

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



Remark: n = 50, 51

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

TCLn2	TCLn1	TCLn0	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	T/n input cycle	2^8 × T/n input cycle	T/n input edge input cycle
0	0	1	T/n input cycle	2^8 × T/n input cycle	T/n input edge input cycle
0	1	0	$1/f_X$ (125 ns)	2^8 × $1/f_X$ (32 µs)	$1/f_X$ (125 ns)
0	1	1	2^1 × $1/f_X$ ((250 ns)	2^9 × $1/f_X$ (64 ms)	2^1 × $1/f_X$ ((250 ns)
1	0	0	2^3 × $1/f_X$ (1 µs)	2^{11} × $1/f_X$ (256 ms)	2^3 × $1/f_X$ (1 µs)
1	0	1	2^5 × $1/f_X$ (4 µs)	2^{13} × $1/f_X$ (1 ms)	2^5 × $1/f_X$ (4 µs)
1	1	0	2^7 × $1/f_X$ (16 µs)	2^{15} × $1/f_X$ (4 ms)	2^7 × $1/f_X$ (16 µs)
1	1	1	2^{12} × $1/f_X$ (512 µs)	2^{20} × $1/f_X$ (131 ms)	2^{12} × $1/f_X$ (512 µs)
Other than above			Setting prohibited		

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

TCLn2	TCLn1	TCLn0	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	T/n input cycle	2^8 × T/n input cycle	T/n input edge input cycle
0	0	1	T/n input cycle	2^8 × T/n input cycle	T/n input edge input cycle
0	1	0	$1/f_X$ (125 ns)	2^8 × $1/f_X$ (32 µs)	$1/f_X$ (125 ns)
0	1	1	2^1 × $1/f_X$ ((250 ns)	2^9 × $1/f_X$ (64 ms)	2^1 × $1/f_X$ ((250 ns)
1	0	0	2^3 × $1/f_X$ (1 µs)	2^{11} × $1/f_X$ (256 ms)	2^3 × $1/f_X$ (1 µs)
1	0	1	2^5 × $1/f_X$ (4 µs)	2^{13} × $1/f_X$ (1 ms)	2^5 × $1/f_X$ (4 µs)
1	1	0	2^8 × $1/f_X$ (32 µs)	2^{16} × $1/f_X$ (8 ms)	2^8 × $1/f_X$ (32 µs)
1	1	1	2^9 × $1/f_X$ (64 µs)	2^{17} × $1/f_X$ (16 ms)	2^9 × $1/f_X$ (64 µs)
Other than above			Setting prohibited		

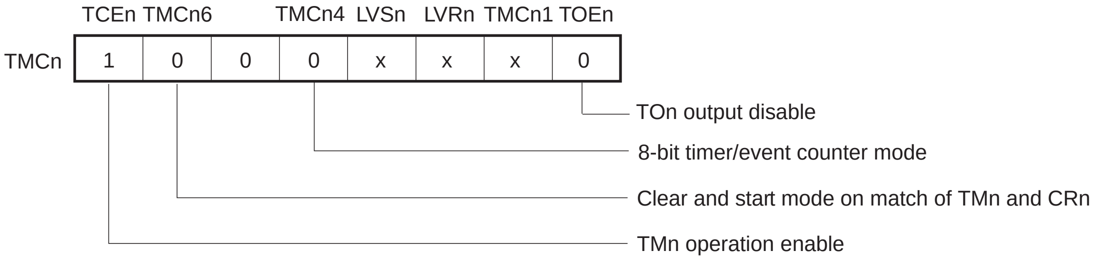
Remarks: 1. f_X: Main system clock oscillation frequency

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

9.4.2 External event counter operation

The external event counter counts the number of external clock pulses to be input to the TI50/P26/TO50 and TI51/P27/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.

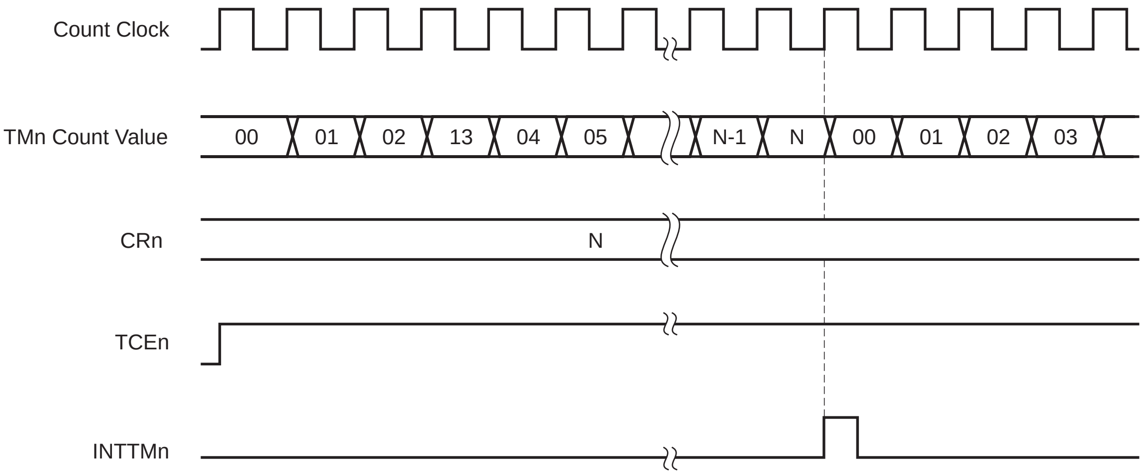
Figure 9-11: 8-Bit Timer Mode Control Register Setting for External Event Counter Operation



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

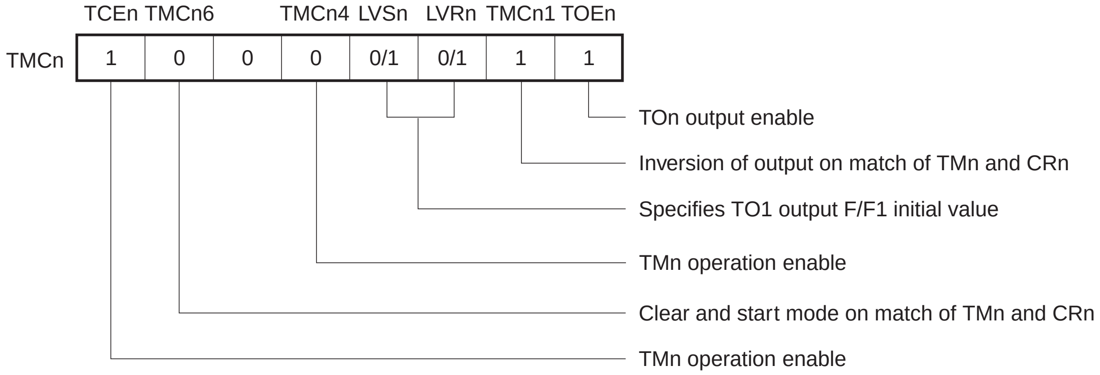
2. n = 50, 51

3. The bit TMCn4 is just valid for timer TM51.

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/P26/TI50 or TO51/P27/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 a selected frequency to be output.

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


TCEn
TMCn6
TMCn4
LVSn
LVRn
TMCn1
TOEn
TMCn
1
0
0
0
0/1
0/1
1
1
TOn output enable
Inversion of output on match of TMn and CRn
Specifies TO1 output F/F1 initial value
TMn operation enable
Clear and start mode on match of TMn and CRn
TMn operation enable

Setting Method

(1) Set the registers

Set the port latch and port mode register to 0.

TCL5n	:	Selects the count clock
CR5n	:	Compare value
TMC5n	:	Selects the clear and start mode when TM5n and CR5n match.

LVS5n	LVR5n	Setting State of Timer Output flip-flop
1	0	High level output
0	1	Low level output

Inversion of timer output flip-flop enabled
Timer output enabled → TOE5n = 1

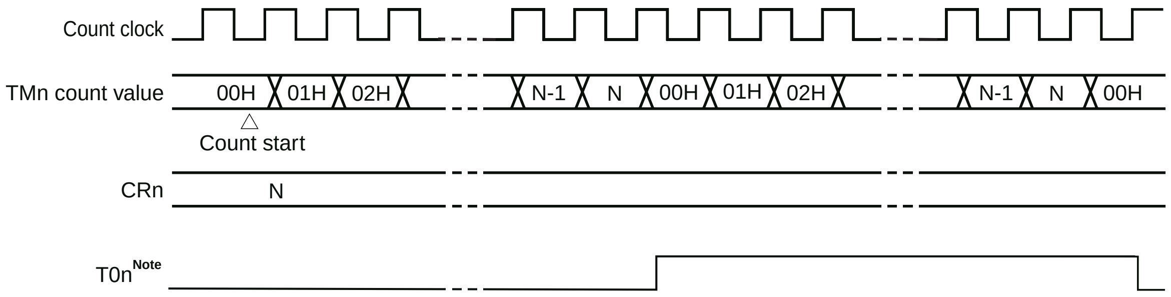
(2) When TCE5n = 1 is set, the counter starts operating.
(3) When the values of TM5n and CR5n match, the timer output flip-flop inverts. Also, INTTM5n is generated and TM5n is cleared to 00H.
(4) Then, the timer output flip-flop is inverted for the same interval to output a square wave from TO5n.

Caution: When TI50/P26/TO50 or TI51/P27/TO51 pin is used as the timer output, set port mode register (PM26 or PM27), and output latch to 0.

Remarks: 1. n = 50, 51

2. The bit TMC5n4 is just valid for timer TM51.

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 9-10: 8-Bit Timer/Event Counters 50 Square-Wave Output Ranges (8-Bit Timer/Event Counter Mode)

TCL502	TCL501	TCL500	Minimum Pulse Time	Maximum Pulse Time	Resolution
0	1	0	$1/f_X$ (125 ns)	$2^8 \times 1/f_X$ (32 µs)	$1/f_X$ (125 ns)
0	1	1	$2^1 \times 1/f_X$ ((250 ns)	$2^9 \times 1/f_X$ (64 ms)	$2^1 \times 1/f_X$ ((250 ns)
1	0	0	$2^3 \times 1/f_X$ (1 µs)	$2^{11} \times 1/f_X$ (256 ms)	$2^3 \times 1/f_X$ (1 µs)
1	0	1	$2^5 \times 1/f_X$ (4 µs)	$2^{13} \times 1/f_X$ (1 ms)	$2^5 \times 1/f_X$ (4 µs)
1	1	0	$2^7 \times 1/f_X$ (16 µs)	$2^{15} \times 1/f_X$ (4 ms)	$2^7 \times 1/f_X$ (16 µs)
1	1	1	$2^{12} \times 1/f_X$ (512 µs)	$2^{20} \times 1/f_X$ (131 ms)	$2^{12} \times 1/f_X$ (512 µs)

Table 9-11: 8-Bit Timer/Event Counters 51 Square-Wave Output Ranges (8-Bit Timer/Event Counter Mode)

TCL502	TCL501	TCL500	Minimum Pulse Time	Maximum Pulse Time	Resolution
0	1	0	$1/f_X$ (125 ns)	$2^8 \times 1/f_X$ (32 µs)	$1/f_X$ (125 ns)
0	1	1	$2^1 \times 1/f_X$ ((250 ns)	$2^9 \times 1/f_X$ (64 ms)	$2^1 \times 1/f_X$ ((250 ns)
1	0	0	$2^3 \times 1/f_X$ (1 µs)	$2^{11} \times 1/f_X$ (256 ms)	$2^3 \times 1/f_X$ (1 µs)
1	0	1	$2^5 \times 1/f_X$ (4 µs)	$2^{13} \times 1/f_X$ (1 ms)	$2^5 \times 1/f_X$ (4 µs)
1	1	0	$2^8 \times 1/f_X$ (32 µs)	$2^{16} \times 1/f_X$ (8 ms)	$2^8 \times 1/f_X$ (32 µs)
1	1	1	$2^9 \times 1/f_X$ (64 µs)	$2^{17} \times 1/f_X$ (16 ms)	$2^9 \times 1/f_X$ (64 µs)

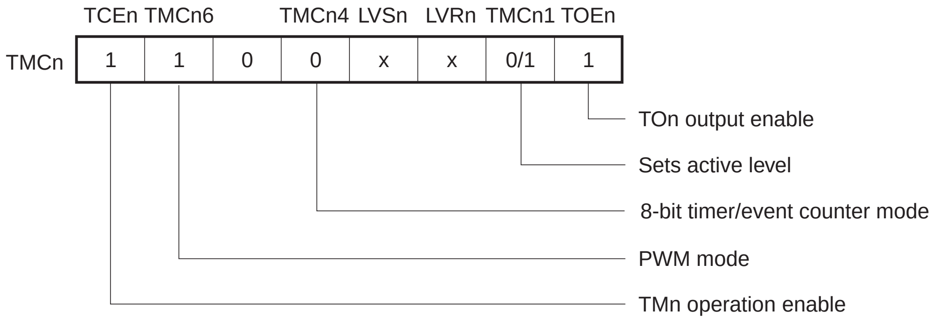
Remarks: **1.** f_X: Main system clock oscillation frequency

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

9.4.4 PWM output operations

Setting the 8-bit timer mode control registers (TMC50 and TMC51) as shown in Figure 9-15 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/P26/TI50 or TO51/P27/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



Setting Method

(1) Set the port latch and port mode register to "0".
(2) Set the active level width in the 8-bit compare register n (CR5n).
(3) Select the count clock in the timer clock selection register n (TCL5n).
(4) Set the active level in bit 1 (TMC5n1) of TMC5n.
(5) If bit 7 (TCE5n) of TMC5n is set to "1", counting starts.
When counting starts, set TCE5n to "0".

Remarks: 1. n = 50, 51

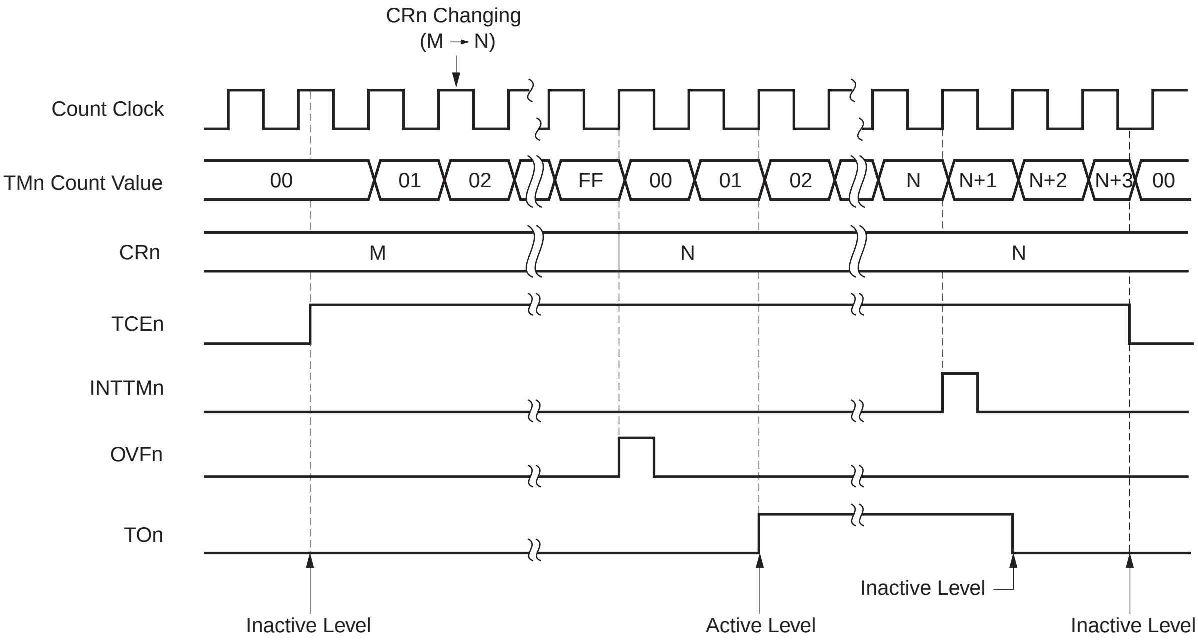
2. x: don’t care

PWM Output Operation

(1) When counting starts, the PWM output (output from TO5n) outputs the inactive level until an overflow occurs.
(2) When the overflow occurs, the active level specified in step (1) in the setting method is output. The active level is output until CR5n and the count of the 8-bit counter n (TM5n) match.
(3) The PWM output after CR5n and the count match is the inactive level until an overflow occurs again.
(4) Steps (2) and (3) repeat until counting stops.
(5) If counting is stopped by TCE5n = 0, the PWM output goes to the inactive level.

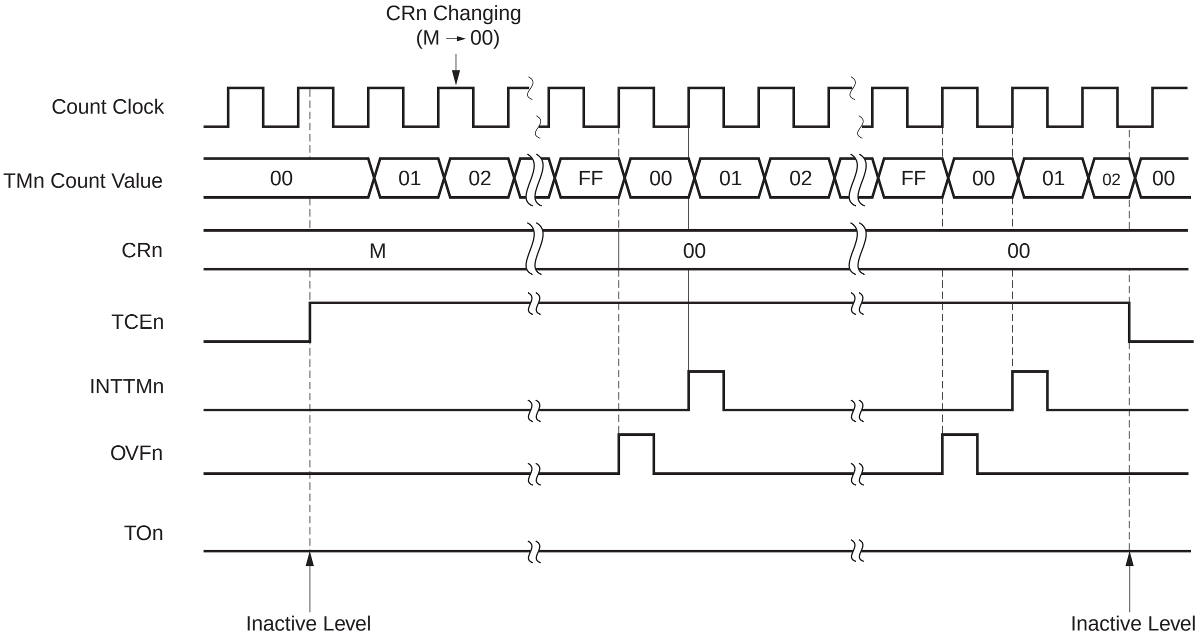
Remark: n = 50, 51

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



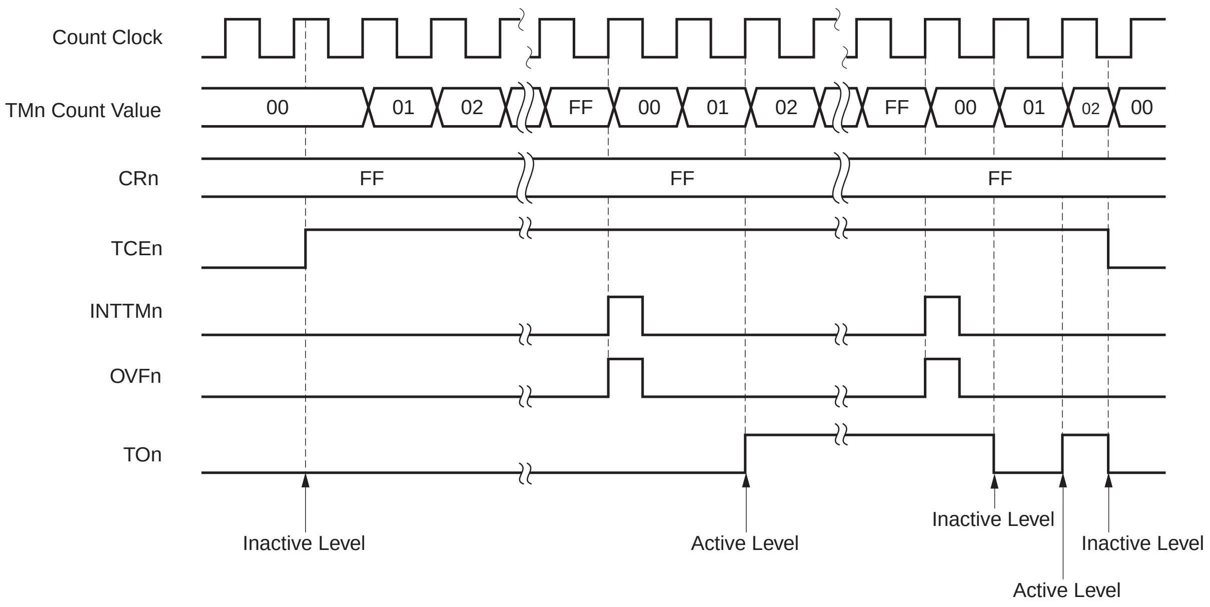
Remark: n = 50, 51

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



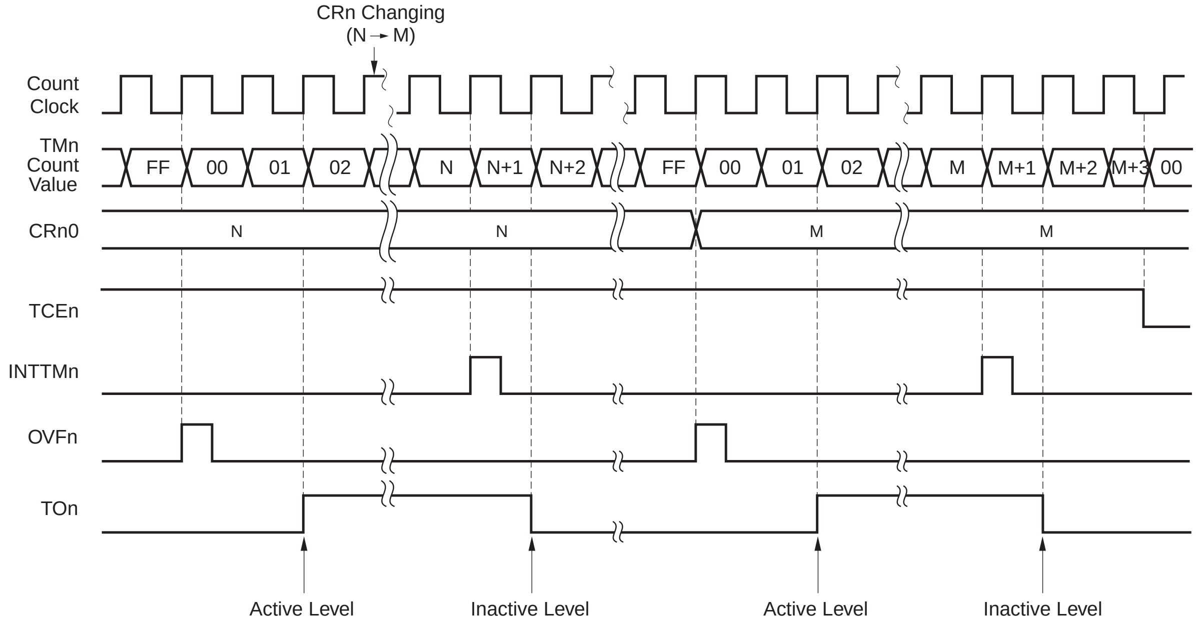
Remark: n = 50, 51

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.**

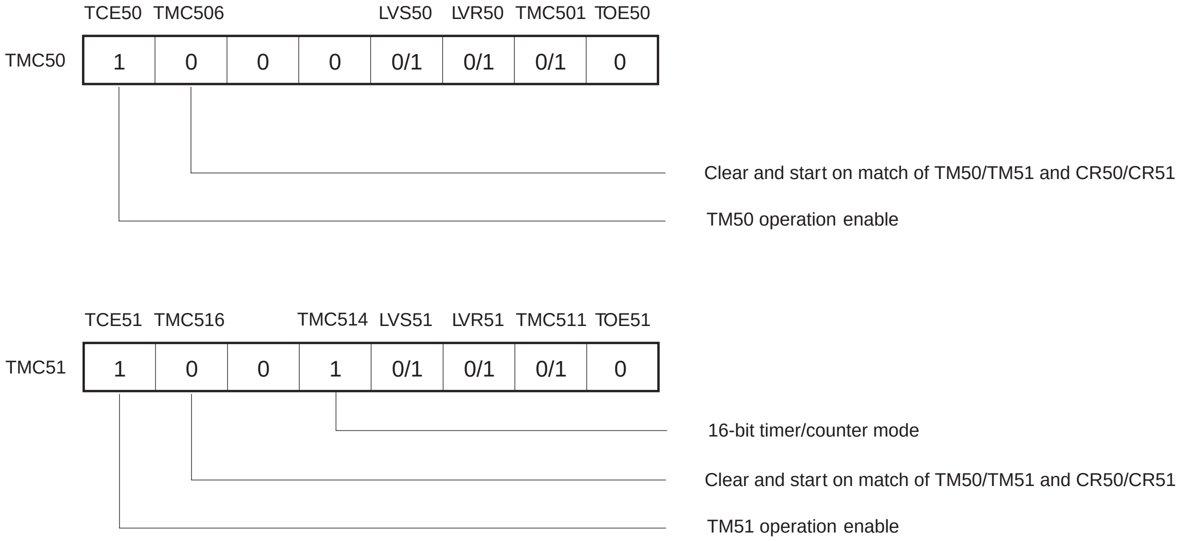
9.5 Operation as interval timer (16-bit operation)

(1) Cascade (16-bit timer) mode (TM50 and TM51)

The 16-bit resolution timer/counter mode is set by setting bit 4 (TMC514) of the 8-bit timer mode control register 51 (TMC51) to “1”.

In this mode, TM50 and TM51 operate as a 16-bit interval timer that repeatedly generates an interrupt request at intervals specified by the count value set in advance to 8-bit compare registers 50 and 51 (CR50 and CR51).

Figure 9-20: 8-Bit Timer Mode Control Register Settings for 16-Bit Interval Timer Operation



	TCE50	TMC506			LVS50	LVR50	TMC501	TOE50
TMC50	1	0	0	0	0/1	0/1	0/1	0

Clear and start on match of TM50/TM51 and CR50/CR51
TM50 operation enable

	TCE51	TMC516		TMC514	LVS51	LVR51	TMC511	TOE51
TMC51	1	0	0	1	0/1	0/1	0/1	0

16-bit timer/counter mode
Clear and start on match of TM50/TM51 and CR50/CR51
TM51 operation enable

Remark: **0/1:** Setting 0 or 1 allows another function to be used simultaneously with the interval timer.

<Setting>

<1> Set each register.

- TCL50: TM50 selects a count clock.
  TM51, which is connected in cascade, does not have to be set.
- CR50 and CR51: Compare values (Each compare value can be set in a range of 00H to FFH).
- TMC50 and TMC51: Select the mode that clears and starts the timer on coincidence between TM50 and CR50 (TM51 and CR51).

  TM50 → TMC50 = 0000xxx0B x: don’t care

  TM51 → TMC51 = 0001xxx0B x: don’t care

<2> By setting TCE51 to 1 for TMC51 first, and then setting TCE50 to 1 for TMC50, the count operation is started.

<3> When the value of TM50 connected in cascade coincides with the value of CR50, TM50 generates INTTM50 (TM50 and TM51 are cleared to 00H).

<4> After that, INTTM50 is repeatedly generated at the same interval.

Cautions: 1. Be sure to set the compare registers (CR50 and CR51) after stopping the timer operation.

2. Even if the timers are connected in cascade, TM51 generates INTTM51 when the count value of TM51 coincides with the value of CR51. Be sure to mask TM51 to disable it from generating an interrupt.

3. Set TCE50 and TCE51 in the order of TM51, then TM50.

4. Counting can be started or stopped by setting or clearing only TCE50 of TM50 to 1 or 0.

Figure 9-21 shows an example of timing in the 16-bit resolution cascade mode.

Figure 9-21: 16-Bit Resolution Cascade Mode (with TM50 and TM51)

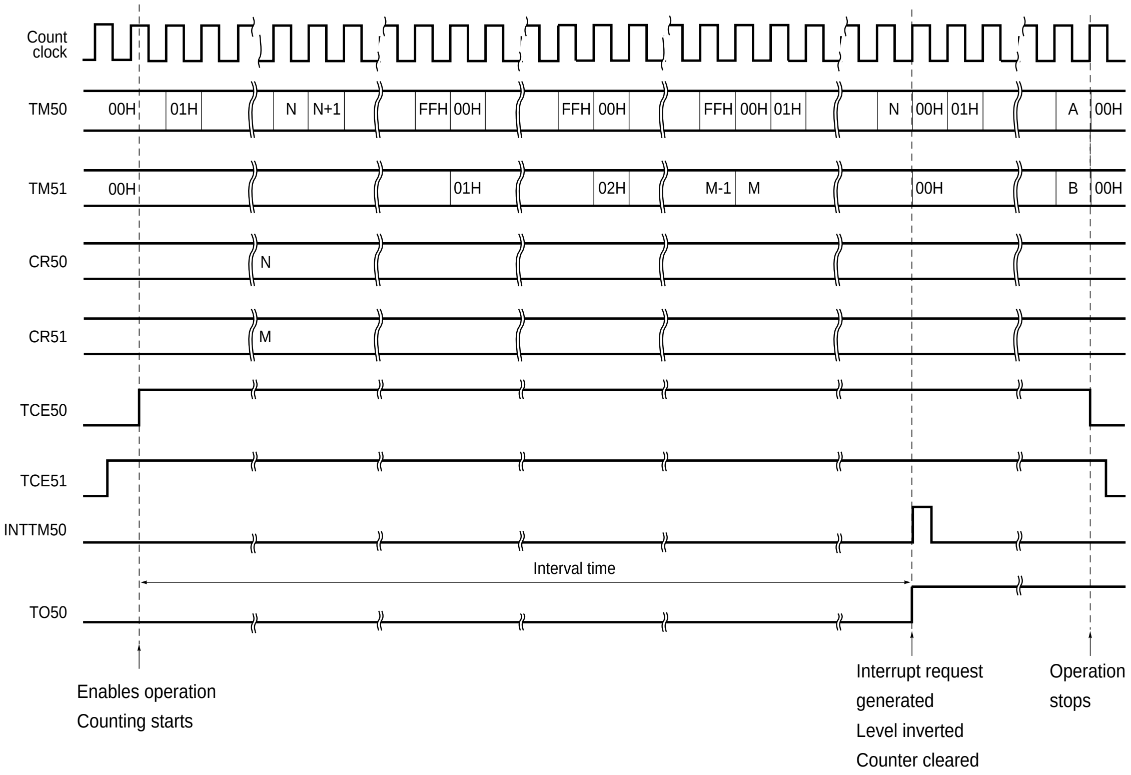


Table 9-12: 8-Bit Timer/Event Counters Interval Times (16-Bit Timer/Event Counter Mode)

TCL502	TCL501	TCL500	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	TI50 input cycle	2^{16} × TIn input cycle	TIn input cycle
0	0	1	TI50 input cycle	2^{16} × TIn input cycle	TIn input cycle
0	1	0	$1/f_X$ (125 ns)	2^{16} × $1/f_X$ (8 ms)	$1/f_X$ (125 ns)
0	1	1	2^1 × $1/f_X$ ((250 ns)	2^{17} × $1/f_X$ (16 ms)	2^1 × $1/f_X$ ((250 ns)
1	0	0	2^3 × $1/f_X$ (1 µs)	2^{19} × $1/f_X$ (65.5 ms)	2^3 × $1/f_X$ (1 µs)
1	0	1	2^5 × $1/f_X$ (4 µs)	2^{21} × $1/f_X$ (262 ms)	2^5 × $1/f_X$ (4 µs)
1	1	0	2^7 × $1/f_X$ (16 µs)	2^{23} × $1/f_X$ (1.05 s)	2^7 × $1/f_X$ (16 µs)
1	1	1	2^{12} × $1/f_X$ (512 µs)	2^{28} × $1/f_X$ (33.6 s)	2^{12} × $1/f_X$ (512 µs)

Table 9-13: 8-Bit Timer/Event Counter Square-Wave Output Ranges (16-Bit Timer/Event Counter Mode)

TCL502	TCL501	TCL500	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	1	0	$1/f_X$ (125 ns)	2^{16} × $1/f_X$ (8 ms)	$1/f_X$ (125 ns)
0	1	1	2^1 × $1/f_X$ ((250 ns)	2^{17} × $1/f_X$ (16 ms)	2^1 × $1/f_X$ ((250 ns)
1	0	0	2^3 × $1/f_X$ (1 µs)	2^{19} × $1/f_X$ (65.5 ms)	2^3 × $1/f_X$ (1 µs)
1	0	1	2^5 × $1/f_X$ (4 µs)	2^{21} × $1/f_X$ (262 ms)	2^5 × $1/f_X$ (4 µs)
1	1	0	2^7 × $1/f_X$ (16 µs)	2^{23} × $1/f_X$ (1.05 s)	2^7 × $1/f_X$ (16 µs)
1	1	1	2^{12} × $1/f_X$ (512 µs)	2^{28} × $1/f_X$ (33.6 s)	2^{12} × $1/f_X$ (512 µs)

Caution: The clock selection in the cascade mode (16-bit timer/event counter mode) is done by the register TCL50.

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

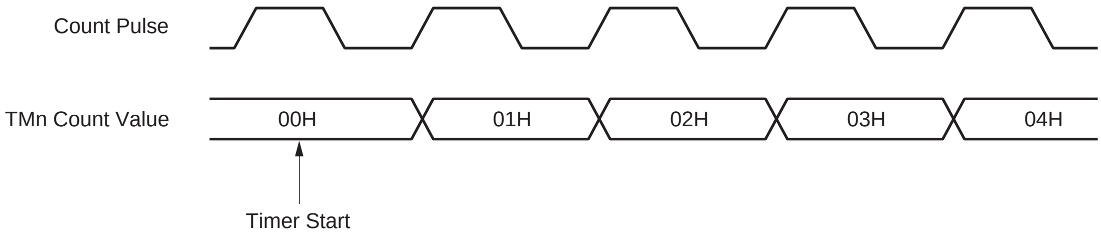
2. Values in parentheses when operated at f_X = 8.0 MHz.

9.6 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 9-22: 8-bit Timer Registers 50 and 51 Start Timings

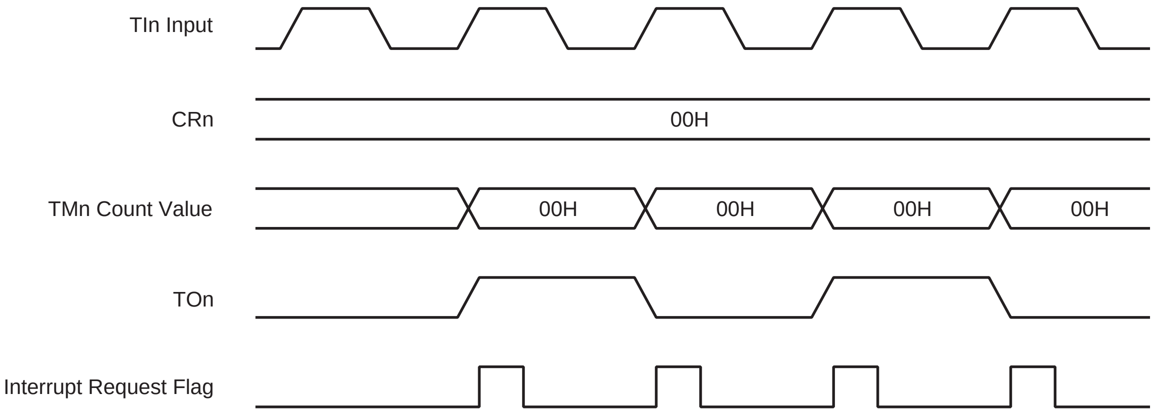


Remark: n = 50, 51

(2) Compare registers 50 and 51 sets

The 8-bit compare registers (CR50 and CR51) can be set to 00H.
Thus, when an 8-bit compare register is used as an event counter, one-pulse count operation can be carried out.

Figure 9-23: External Event Counter Operation Timings

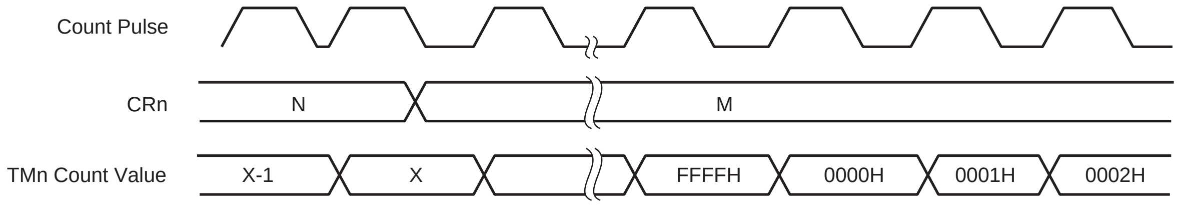


Remark: n = 50, 51

(3) Operation after compare register change during timer count operation

If the values after the 8-bit compare registers (CR50 and CR51) are changed are smaller than those of 8-bit timer registers (TM50 and TM51), TM50 and TM51 continue counting, overflow and then restarts counting from 0. Thus, if the value (M) after CR50 and CR51 change is smaller than that (N) before change it is necessary to restart the timer after changing CR50 and CR51.

Figure 9-24: Timings after Compare Register Change during Timer Count Operation



Remark: n = 50, 51

[MEMO]

Chapter 10 Watch Timer

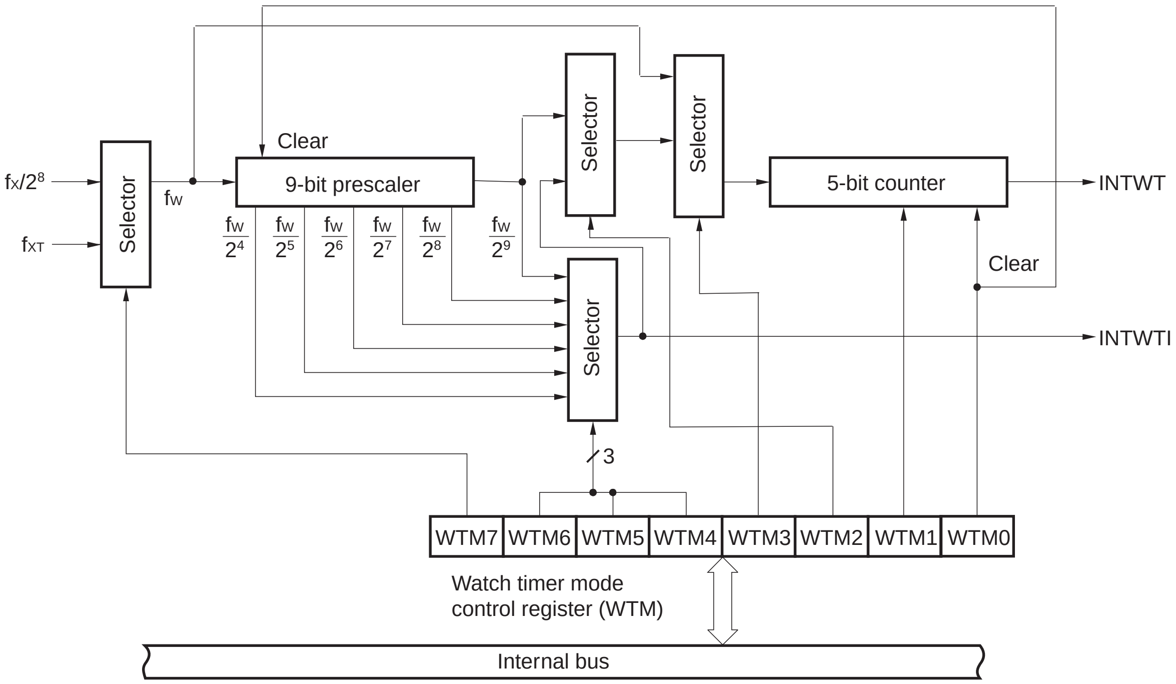
10.1 Watch Timer Functions

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 the following time intervals.

Table 10-1: Watch Timer Interval Time

Watch Timer Interval Time	$f_W = f_X/2^8$		$f_W = f_{XT}$
	When operated at f_X = 8.00 MHz	When operated at f_X = 5.00 MHz	When operated at f_{XT} = 32.768 kHz
$2^5/f_W$	1.02 ms	1.64 ms	976.56 µs
$2^9/f_W$	16.38 ms	26.21 ms	15.63 ms
$2^{10}/f_W$	32.77 ms	52.43 ms	31.25 ms
$2^{11}/f_W$	65.54 ms	104.86 ms	62.5 ms
$2^{12}/f_W$	131.07 ms	209.72 ms	125 ms
$2^{13}/f_W$	262.14 ms	419.43 ms	250 ms
$2^{14}/f_W$	524.29 ms	838.86 ms	500 ms

Remarks:

1. f_X: Main system clock oscillation frequency

2. f_{XT}: Subsystem clock oscillation frequency

3. f_W: Watch timer clock frequency

(2) Interval timer

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

Table 10-2: Interval Timer Interval Time

Interval Time	$f_W = f_X/2^8$		$f_W = f_{XT}$
	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

Remarks:

1. f_X: Main system clock oscillation frequency

2. f_{XT}: Subsystem clock oscillation frequency

3. f_W: Watch timer clock frequency

10.2 Watch Timer Configuration

The watch timer consists of the following hardware.

Table 10-3: Watch Timer Configuration

Item	Configuration
Counter	5 bits × 1
Prescaler	9 bits × 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 an 1-bit or an 8-bit memory manipulation instruction.

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

Figure 10-2: Watch Timer Mode Control Register (WTM) Format (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
WTM	WTM7	WTM6	WTM5	WTM4	WTM3	WTM2	WTM1	WTM0	R/W	FF41H	00H

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

WTM6	WTM5	WTM4	Prescaler Interval Time Selection		
			f_X = 8.00 MHz Operation	f_X = 5.00 MHz Operation	f_{XT} = 32.768 kHz Operation
0	0	0	$2^4/f_W$ (512 µs)	$2^4/f_W$ (819 µs)	$2^4/f_W$ (488 µs)
0	0	1	$2^5/f_W$ (1 ms)	$2^5/f_W$ (1.6 ms)	$2^5/f_W$ (977 µs)
0	1	0	$2^6/f_W$ (2 ms)	$2^6/f_W$ (3.2 ms)	$2^6/f_W$ (1.95 ms)
0	1	1	$2^7/f_W$ (4 ms)	$2^7/f_W$ (6.55 ms)	$2^7/f_W$ (3.91 ms)
1	0	0	$2^8/f_W$ (8.19 ms)	$2^8/f_W$ (13.1 ms)	$2^8/f_W$ (7.81 ms)
1	0	1	$2^9/f_W$ (16.38 ms)	$2^9/f_W$ (26.2 ms)	$2^9/f_W$ (15.6 ms)
Other than above			Setting prohibited		

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

WTM2	Interrupt Generation Mode Select for normal Operating Mode
0	Standard mode (interrupt generation fixed at $2^{14}/f_W$)
1	Selectable mode (interrupt generation at $2^9/f_W$ to $2^{14}/f_W$)

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

WTM1	5-Bit Counter Operation Control
0	Clear after operation stop
1	Operation enable

WTM0	Prescaler Operation Control
0	Clear after operation stop
1	Operation enable

Caution: When the watch timer is used, the prescaler should not be cleared frequently. When rewriting WTM4 to WTM6 to other data, stop the timer operation beforehand.

Remarks: **1.** f_W: Watch timer clock frequency ($f_X/2^8$ or f_{XT})

2. f_X: Main system clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

10.4 Watch Timer Operations

10.4.1 Watch timer operation

The watch timer operates as internal timer and generates interrupt requests repeatedly at a defined interval.
When the 32.768 kHz subsystem clock is used, the watch timer generates also 0.25 -second and 0.5-second intervals which can be used for clocks etc.
The interval time can be selected with the bits 2, 3, 4 to 6 and 7 of the watch timer mode control register.

Table 10-4: Watch Timer Operation

WTM6	WTM5	WTM4	WTM3	WTM2	Interval Time	$f_W = f_X/2^8$		$f_W = f_{XT}$
						f_X = 8.00 MHz Operation	f_X = 5.00 MHz Operation	f_{XT} = 32.768 MHz Operation
x	x	x	1	x	$2^5 /f_W$	1 ms	1.63 ms	976.56 ms
0	0	0	0	1	$2^9 /f_W$	16.38 ms	26.21 ms	15.62 ms
0	0	1	0	1	$2^{10} /f_W$	32.76 ms	52.42 ms	31.25 ms
0	1	0	0	1	$2^{11} /f_W$	65.54 ms	104.85 ms	62.5 ms
0	1	1	0	1	$2^{12} /f_W$	131.07 ms	209.71 ms	125 ms
1	0	0	0	1	$2^{13} /f_W$	262.14 ms	413.43 ms	250 ms
1	0	1	0	x	$2^{14} /f_W$	524.29 ms	838.86 ms	500 ms
Other than above					Setting prohibited			

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

2. f_{XT} : Subsystem clock oscillation frequency

3. f_W : Watch timer clock frequency

4. x : Don't care

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 10-5: Interval Timer Operation

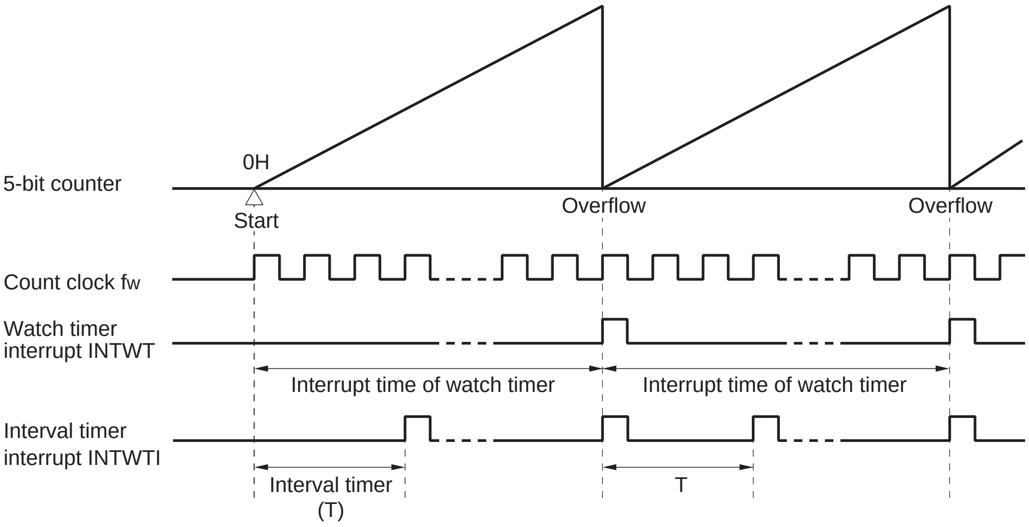
WTM6	WTM5	WTM4	Interval Time	$f_W = f_X/2^8$		$f_W = f_{XT}$
				f_X = 8.00 MHz Operation	f_X = 5.00 MHz Operation	f_{XT} = 32.768 MHz Operation
0	0	0	$2^4 \times 1/f_W$	512 µs	819 µs	488 µs
0	0	1	$2^5 \times 1/f_W$	1 ms	1.6 ms	977 µs
0	1	0	$2^6 \times 1/f_W$	2 ms	3.2 ms	1.95 ms
0	1	1	$2^7 \times 1/f_W$	4 ms	6.55 ms	3.91 ms
1	0	0	$2^8 \times 1/f_W$	8.19 ms	13.1 ms	7.81 ms
1	0	1	$2^9 \times 1/f_W$	16.38 ms	26.2 ms	15.6 ms
Other than above			Setting prohibited			

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

2. f_{XT} : Subsystem clock oscillation frequency

3. f_W : Watch timer clock frequency

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


5-bit counter
0H
Start
Overflow
Overflow
Count clock fw
Watch timer interrupt INTWT
Interrupt time of watch timer
Interrupt time of watch timer
Interval timer interrupt INTWTI
Interval timer (T)
T

Remark: f_W : Watch timer clock frequency

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	
$2^{12} \times 1/f_X$	$2^{12} \times 1/f_X$ (512 µs)
$2^{13} \times 1/f_X$	$2^{13} \times 1/f_X$ (1 ms)
$2^{14} \times 1/f_X$	$2^{14} \times 1/f_X$ (2 ms)
$2^{15} \times 1/f_X$	$2^{15} \times 1/f_X$ (4 ms)
$2^{16} \times 1/f_X$	$2^{16} \times 1/f_X$ (8.19 ms)
$2^{17} \times 1/f_X$	$2^{17} \times 1/f_X$ (16.38 ms)
$2^{18} \times 1/f_X$	$2^{18} \times 1/f_X$ (32.76 ms)
$2^{20} \times 1/f_X$	$2^{20} \times 1/f_X$ (131 ms)

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

(2) Interval timer mode

Interrupts are generated at the preset time intervals.

Table 11-2: Interval Times

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

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

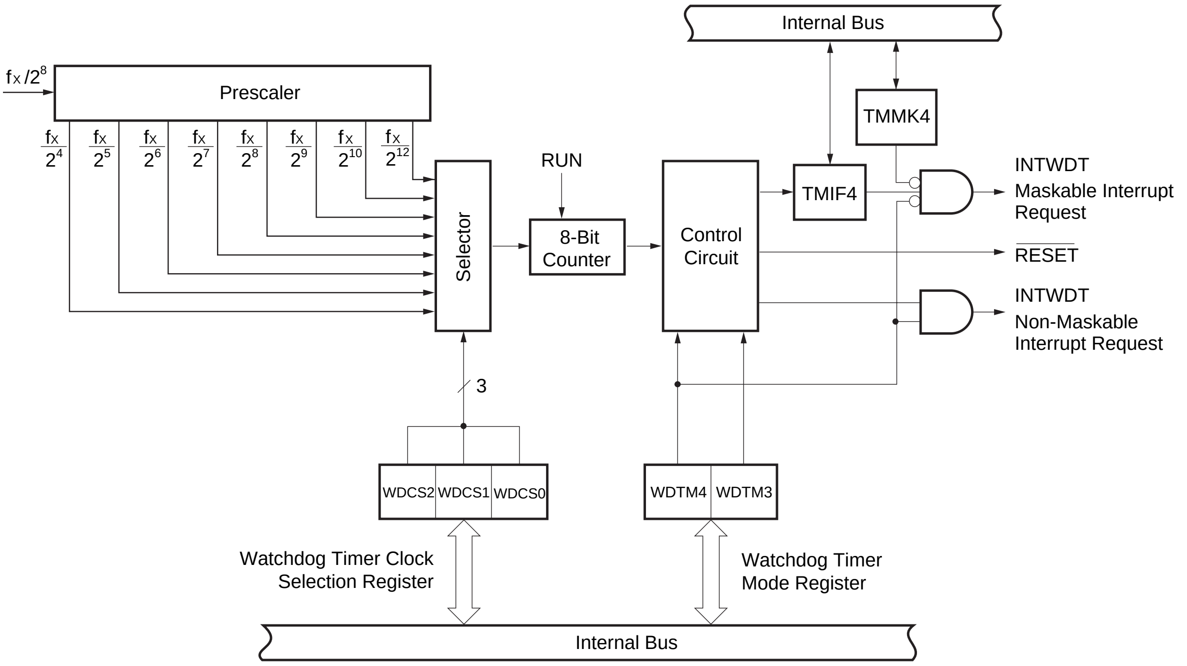
11.2 Watchdog Timer Configuration

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 an 8-bit memory manipulation instruction.

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

Figure 11-2: Timer Clock Select Register 2 Format

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

WDCS2	WDCS1	WDCS0	Overflow Time of Watchdog Timer
0	0	0	$f_X/2^{12}$ (512 µs)
0	0	1	$f_X/2^{13}$ (1 ms)
0	1	0	$f_X/2^{14}$ (2 ms)
0	1	1	$f_X/2^{15}$ (4 ms)
1	0	0	$f_X/2^{16}$ (8.19 ms)
1	0	1	$f_X/2^{17}$ (16.38 ms)
1	1	0	$f_X/2^{18}$ (32.76 ms)
1	1	1	$f_X/2^{20}$ (131 ms)

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

Remarks: 1. f_X: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with f_X = 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 an 1-bit or an 8-bit memory manipulation instruction.

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

Figure 11-3: Watchdog Timer Mode Register Format

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	R/W	FFF9H	00H

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	$f_X/2^{12}$ (512 µs)
0	0	1	$f_X/2^{13}$ (1 ms)
0	1	0	$f_X/2^{14}$ (2 ms)
0	1	1	$f_X/2^{15}$ (4 ms)
1	0	0	$f_X/2^{16}$ (8.19 ms)
1	0	1	$f_X/2^{17}$ (16.38 ms)
1	1	0	$f_X/2^{18}$ (32.76 ms)
1	1	1	$f_X/2^{20}$ (131 ms)

Remarks: 1. f_X: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with f_X = 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	$f_X/2^{12}$ (512 µs)
0	0	1	$f_X/2^{13}$ (1 ms)
0	1	0	$f_X/2^{14}$ (2 ms)
0	1	1	$f_X/2^{15}$ (4 ms)
1	0	0	$f_X/2^{16}$ (8.19 ms)
1	0	1	$f_X/2^{17}$ (16.38 ms)
1	1	0	$f_X/2^{18}$ (32.76 ms)
1	1	1	$f_X/2^{20}$ (131 ms)

Remarks: 1. f_X: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with f_X = 8.0 MHz.

[MEMO]

Chapter 12 Clock Output Control Circuit

12.1 Clock Output Control Circuit Functions

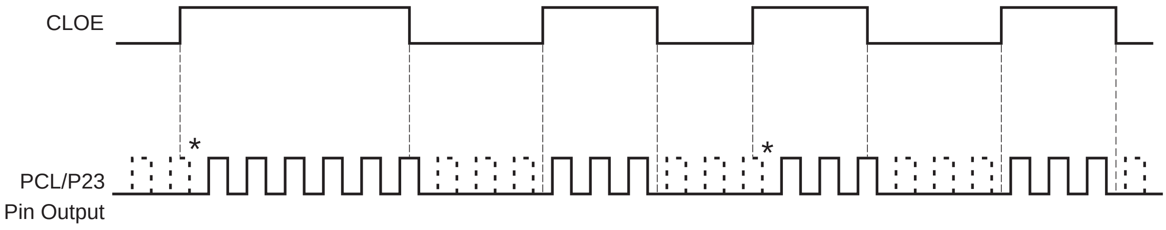
The clock output control circuit is intended for carrier output during remote controlled transmission and clock output for supply to peripheral LSI. Clocks selected with the clock output selection register (CKS) are output from the PCL/P23 pin.
Follow the procedure below to route clock pulses to the SGOA pin:

(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 P23 output latch to 0.
(3) Set bit 3 (PM23) of port mode register 2 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 P23 output latch to 1.

Remark: When clock output enable/disable is switched, the clock output control circuit does not generate pulses with smaller widths than the original signal carries. (See the portions marked with ***** in Figure 12-1).

Figure 12-1: Remote Controlled Output Application Example



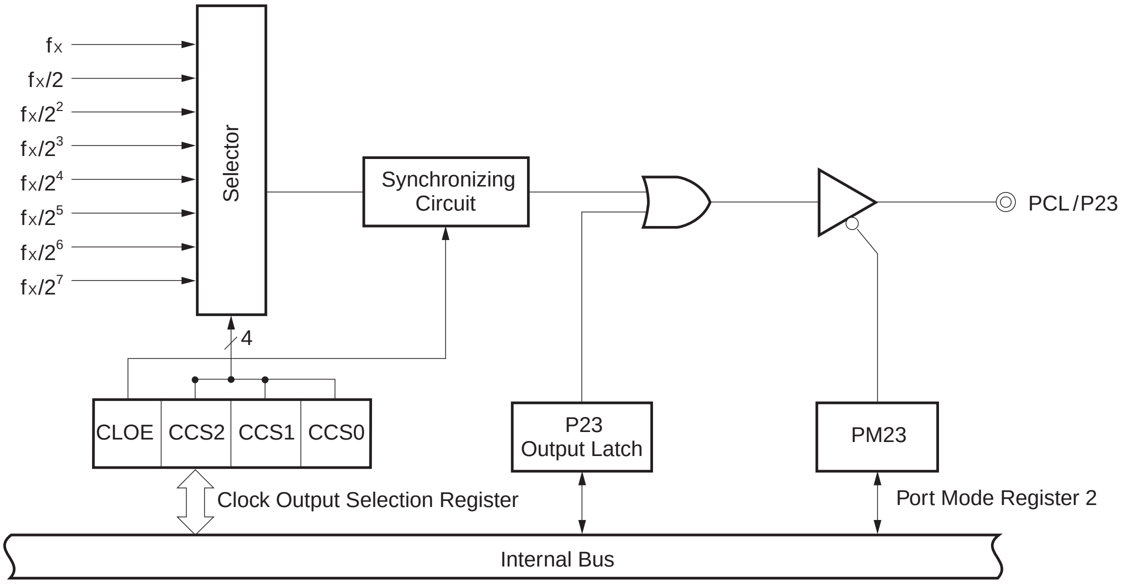
12.2 Clock Output Control Circuit Configuration

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 2 (PM2)

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 2 (PM2)

(1) Clock output selection register (CKS)

This register sets PCL output clock.
CKS is set with an 1-bit or an 8-bit memory manipulation instruction.

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

Caution: When enabling PCL output, set CCS0 to CCS2, then set 1 in CLOE with an 1-bit memory manipulation instruction.

Figure 12-3: Clock Output Selection Register (CKS) Format

	7	6	5	<4>	3	2	1	0	R/W	Address	After Reset
CKS	0	0	0	CLOE	0	CCS2	CCS1	CCS0	R/W	FF40H	00H

CCS2	CCS1	CCS0	PCL Output Clock Selection
0	0	0	f_X (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. f_X: Main system clock oscillation frequency

2. Figures in parentheses apply to operation with f_X = 8.0 MHz.

(2) Port mode register 2 (PM2)

This register sets the port mode P2 input-output in 1-bit units.
When using the P23/PCL pin for clock output function, set PM23 and output latch of P23 to 0.
PM2 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM2 to FFH.

Figure 12-4: Port Mode Register 2 Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	R/W	FF22H	FFH

PM2n	PM2n Pin Input/Output Mode Selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

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 12 analog input channels (ANI0 to ANI11).
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 ANI11, 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

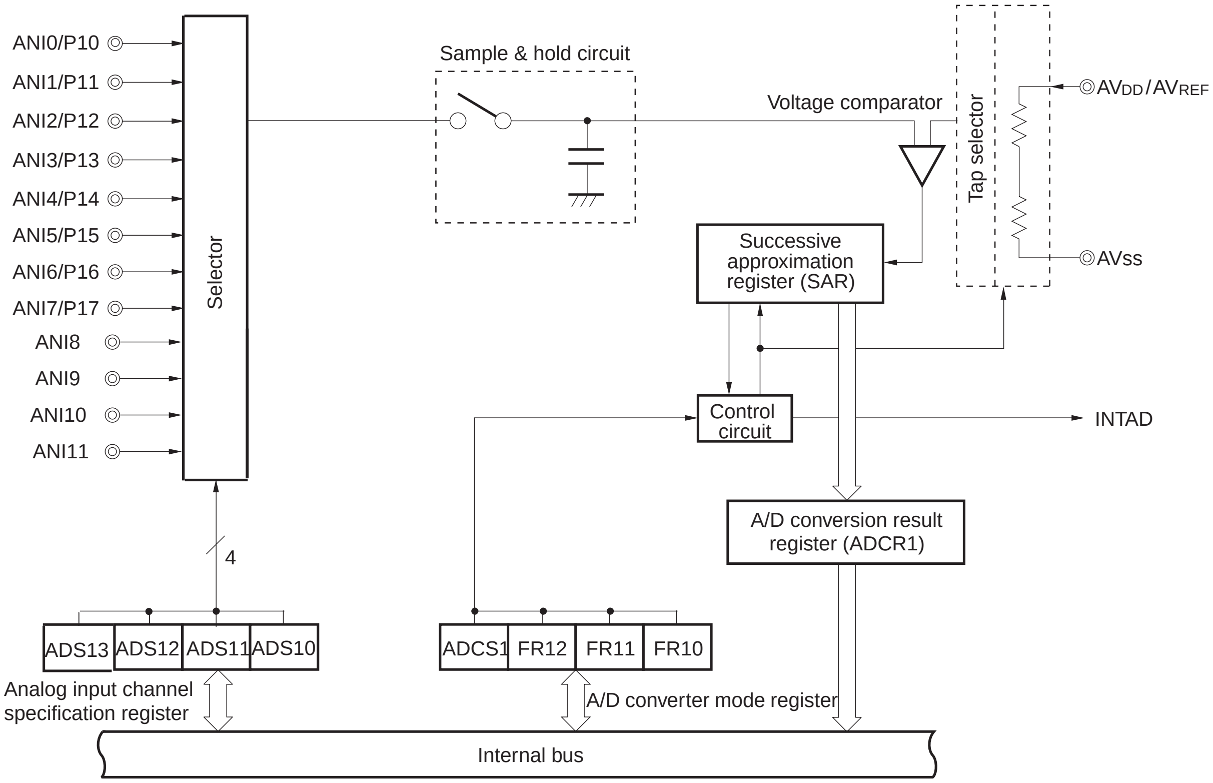
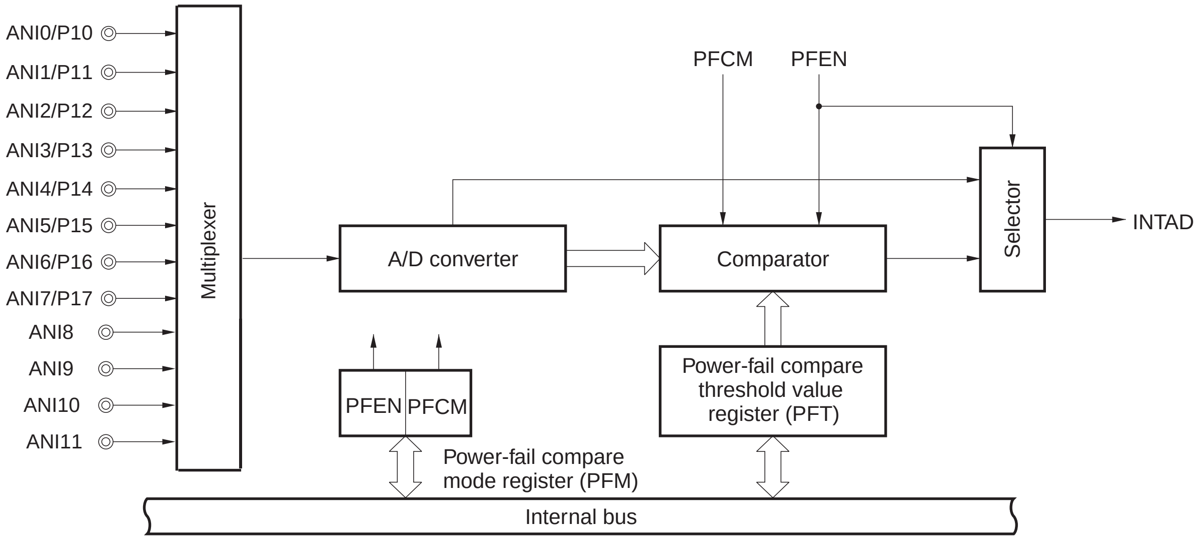


Figure 13-2: Power-Fail Detection Function Block Diagram



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	11 channels (ANI0 to ANI11)
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 sequential 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}/AV_{REF} to AV_{SS}, and generates a voltage to be compared to the analog input.

(6) ANI0 to ANI11 pins

These are twelve 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. ANI8 to ANI11 are A/D converter analog input only pins.

Caution: Use ANI0 to ANI11 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 power supply pin of the A/D converter. The supply power has to be connected when the A/D converter is used.
It converts signals input to ANI0 to ANI11 into digital signals according to the voltage applied between AV_{DD}/AV_{REF} and AV_{SS}.
Even if the AD-converter is not used AV_{DD}/AV_{REF} has to be connected to V_{DD}.

(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 1-bit or an 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input clears ADM1 to 00H.

Figure 13-3: A/D Converter Mode Register (ADM1) Format

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
ADM1	ADCS1	0	FR12	FR11	FR10	0	0	0	R/W	FF98H	00H

ADCS1	A/D Conversion Operation Control
0	Stop conversion operation
1	Enable conversion operation

FR12	FR11	FR10	Conversion Time Selection Note
0	0	0	$144/f_X$
0	0	1	$120/f_X$
0	1	0	$96/f_X$
1	0	0	$72/f_X$
1	0	1	$60/f_X$
1	1	0	$48/f_X$
Other than above			Setting prohibited

Note: Set FR12 to FR10 so that the A/D conversion time is 14 µs or more.

Caution: Bits 0 to 2 and bit 6 must be set to 0.

Remark: f_X: 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

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ADS1	0	0	0	0	ADS13	ADS12	ADS11	ADS10	R/W	FF99H	00H

ADS13	ADS12	ADS11	ADS10	Analog Input Channel Specification
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
1	0	0	0	ANI8
1	0	0	1	ANI9
1	0	1	0	ANI10
1	0	1	1	ANI11
Other than above				Setting prohibited

Caution: Bits 4 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.
PFM is set with an 8-bit manipulation instruction.
$\overline{\text{RESET}}$ input clears PFM to 00H.

Figure 13-5: Power-Fail Compare Mode Register (PFM) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PFM	PFEN	PFCM	0	0	0	0	0	0	R/W	FF9AH	00H

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)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
PFT	PFT7	PFT6	PFT5	PFT4	PFT3	PFT2	PFT1	PFT0	R/W	FF9BH	00H

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.

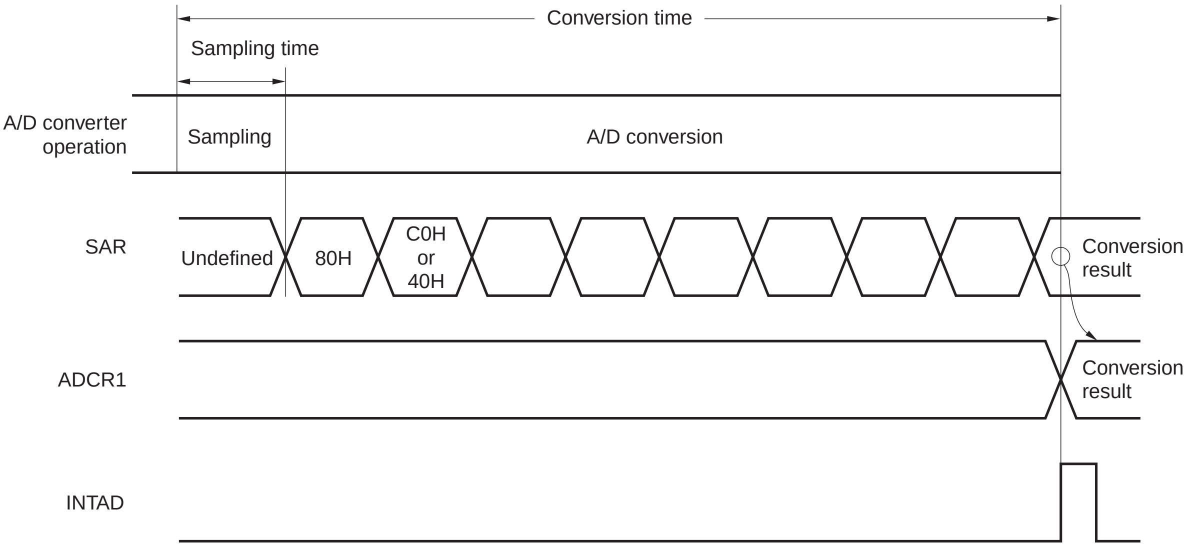
- Analog input voltage ≥ Voltage tap: Bit 6 = 1
- Analog input voltage < Voltage tap: Bit 6 = 0

<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 starting the A/D conversion (ADCS1=1) 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.

$$\mathrm{ADCR1} = \mathrm{INT}\left(\frac{V_{IN}}{AV_{DD}} \times 256 + 0.5\right)$$

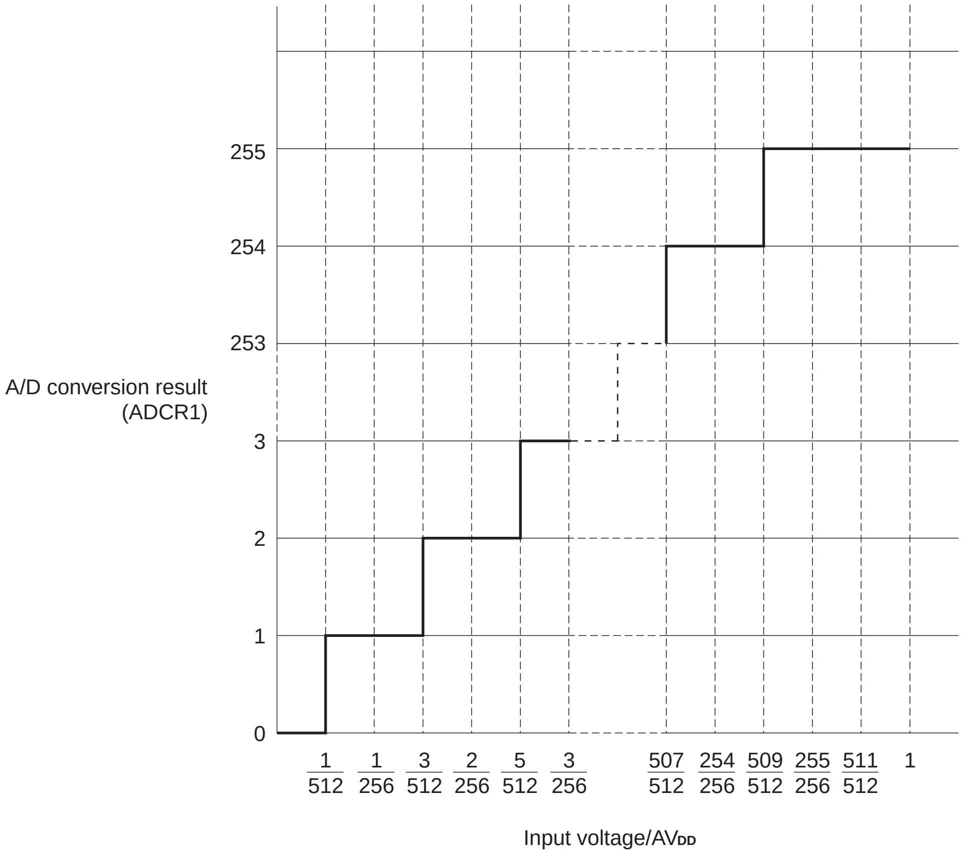
or

$$(\mathrm{ADCR1} - 0.5) \times \frac{AV_{DD}}{256} - V_{IN} < (\mathrm{ADCR1} + 0.5) \times \frac{AV_{DD}}{256}$$

where,	INT( )	:	Function which returns integer part of value in parentheses
	V_{IN}	:	Analog input voltage
	AV_{DD}/AV_{REF}	:	AV_{DD} pin voltage and A/D converter power supply
	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 ANI11 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)

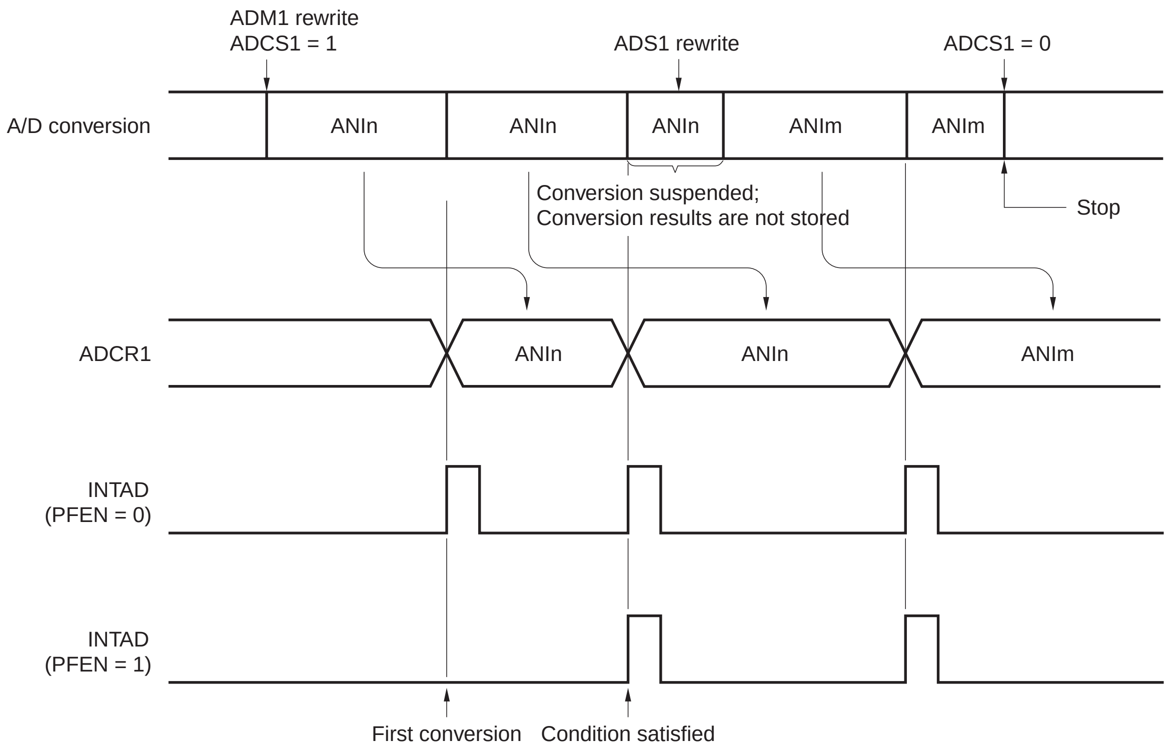
When bit 7 (ADCS1) of the A/D converter mode register (ADM1) is set to 1 and bit 7 of the power-fail compare mode register (PFM) is set to 0, A/D conversion of the voltage applied to the analog input pin specified with the analog input channel specification register (ADS1) starts.
Upon the end of the 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,..., 11

2. m = 0, 1,...,11

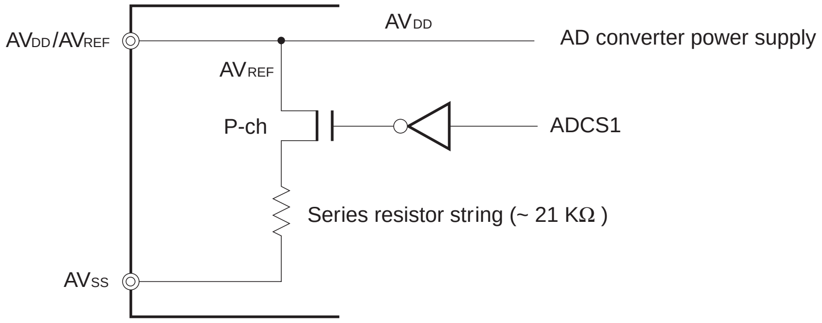
13.5 A/D Converter Precautions

(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 ANI11

The input voltages of ANI0 to ANI11 should be within the specification range. In particular, if a voltage higher than AV_{DD}/AV_{REF} 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.

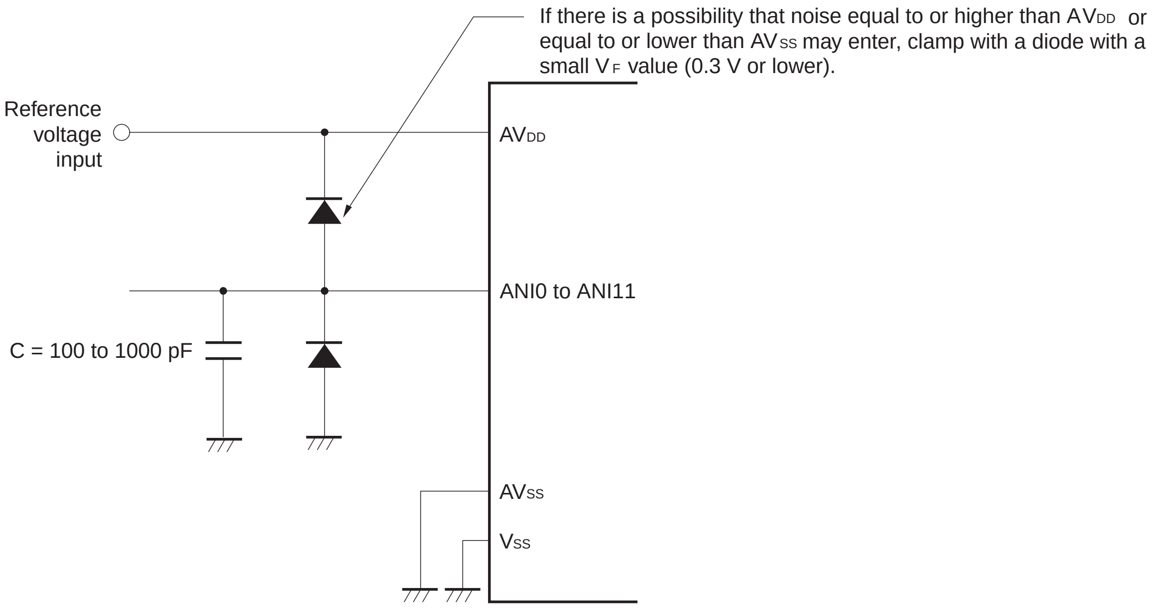
<2> Contention between ADCR1 write and A/D converter mode register (ADM1) write or analog input channel specification register (ADS1) write upon the end of conversion

ADM1 or ADS1 write is given priority. ADCR1 write is not performed, nor is the conversion end interrupt request signal (INTAD) generated.

(4) Noise counter measures

To maintain 8-bit resolution, attention must be paid to noise input to pin AV_{DD} and pins ANI0 to ANI11. 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


If there is a possibility that noise equal to or higher than AVDD or equal to or lower than AVSS may enter, clamp with a diode with a small VF value (0.3 V or lower).
Reference voltage input
AVDD
ANI0 to ANI11
C = 100 to 1000 pF
AVSS
VSS

(5) ANI0 to ANI11

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_{DD}/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.

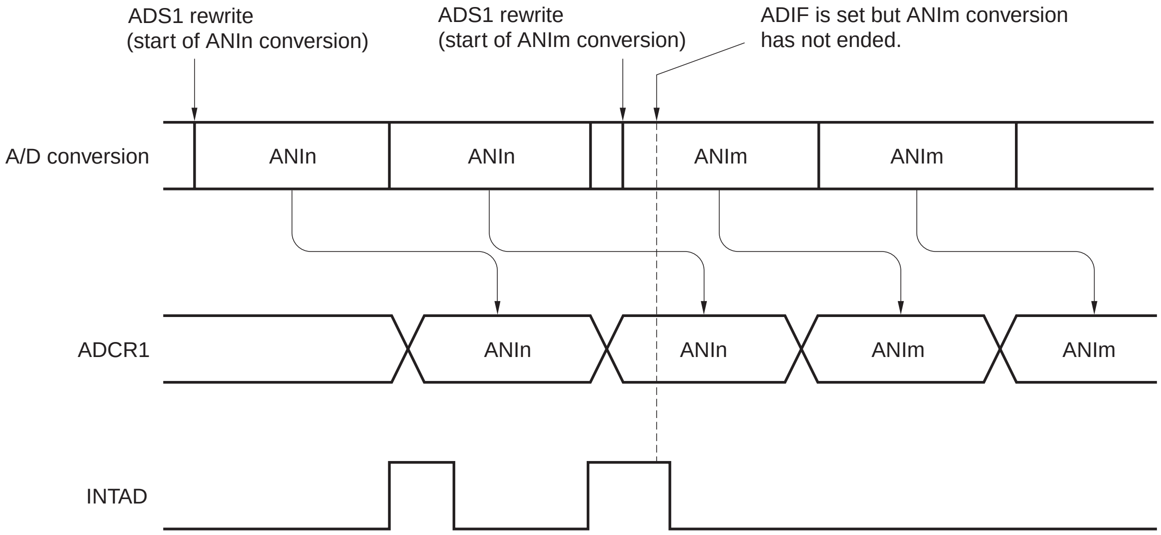
(7) Interrupt request flag (ADIF)

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


ADS1 rewrite
(start of ANIn conversion)
ADS1 rewrite
(start of ANIm conversion)
ADIF is set but ANIm conversion has not ended.
A/D conversion
ANIn
ANIn
ANIm
ANIm
ADCR1
ANIn
ANIn
ANIm
ANIm
INTAD

Remarks: 1. n = 0, 1,..., 11

2. m = 0, 1,..., 11

(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 the in-circuit emulator, the D/A converter mode register (DAM0) must be set. DAM0 is a register used to set the emulation board (IE-78K0-NS-P04).

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-78K0-NS-P04 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

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
DAM0	0	0	0	0	0	0	0	DACE	R/W	FF9CH	00H

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 device 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.

Chapter 14 Serial Interface SIO20

14.1 Serial Interface SIO20 Functions

The SIO20 has the following three modes.

- Operation stop mode
- 3-wire serial I/O mode (standard mode)
- 3-wire serial I/O mode (SPI compatible mode)

Features:

- 8-bit data length
- Simultaneous transmit and receive available
- Start bit is fixed to MSB
- Four different transmit/receive modes with selectable clock inversion and clock phase
- Master and slave modes
- Receive buffer with overflow bit to detect error condition
- Status register to monitor status of receive data buffer
- Possibility to connect to Motorola SPI^{TM} in master and slave mode

(1) Operation stop mode

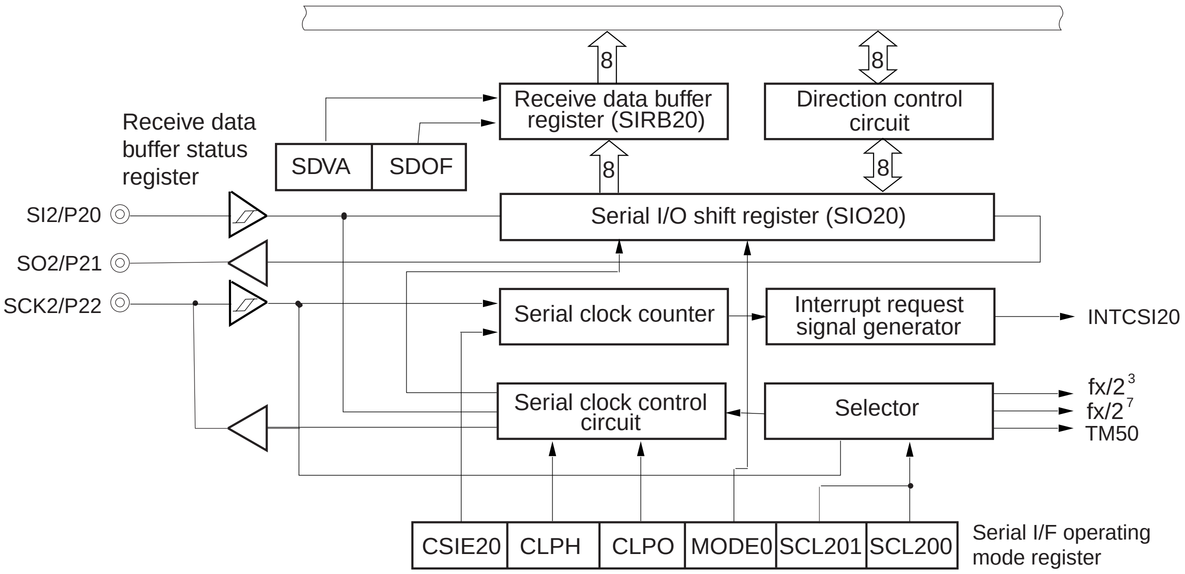
This mode is used when serial transfers are not performed.
For details, see 14.5.1 ”Operation Stop Mode” on page 225.

(2) 3-wire serial I/O mode (SPI compatible mode with fixed as MSB first)

This is an 8-bit data transfer mode using three lines: a serial clock line, serial output line (SO), and serial input line (SI).
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. The 3-wire serial interface works with additional functions to select the clock phase and the clock polarity, to see if there is new data received and error detection. After the reception of the data, the received data of the serial I/O shift register is transferred to the serial I/F receive data buffer.
3-wire serial I/O mode is useful for connection to a peripheral I/O device that includes a clock-synchronous serial interface, or a display controller, etc.
For details, see 14.5.2 ”3-wire Serial I/O Mode” on page 226 (multi-functional mode).

Figure 14-1 shows the block diagram of the SIO20.

Figure 14-1: Block diagram of SIO20


8
8
Receive data buffer register (SIRB20)
Direction control circuit
Receive data buffer status register
SDVA
SDOF
8
8
SI2/P20
Serial I/O shift register (SIO20)
SO2/P21
SCK2/P22
Serial clock counter
Interrupt request signal generator
INTCSI20
Serial clock control circuit
Selector
fx/2^3
fx/2^7
TM50
CSIE20
CLPH
CLPO
MODE0
SCL201
SCL200
Serial I/F operating mode register

14.2 Serial Interface SIO20 Configuration

The SIO20 includes the following hardware.

Table 14-1: Configuration of SIO20

Item	Configuration
Registers	Serial I/O shift register (SIO20) Serial I/F receive data buffer (SIRB20)
Control registers	Serial I/F operation mode register (CSIM20) Receive data buffer status register (SRBS20)

(1) Serial I/O shift register (SIO20)

This is an 8-bit register that performs parallel-serial conversion and serial transmit/receive (shift operations) synchronized with the serial clock.
SIO20 is set by an 8-bit memory manipulation instruction.
When “1” is set to bit 7 (CSIE20) of the serial operation mode register 20 (CSIM20), a serial operation can be started by writing data to or reading data from SIO20.
When transmitting, data written to SIO20 is output via the serial output (SO2).
When receiving, data is read from the serial input (SI2) and written to SIO20.
The $\overline{\text{RESET}}$ signal resets the register value to 00H.

Cautions: 1. Do not access to SIO20 (read/write) during transfer operation.

2. If CLPH is set to ‘1’ (clock phase shifting enabled) and the SIO20 is in slave mode, the received data should not be read from the SIO20 shift register directly. In this particular case, the result in the shift register does not match the transferred byte. The correct byte can be read from the receive data buffer. This has to be considered when writing software for the SIO20.

(2) Serial I/F operation mode register (CSIM20)

This is an 8-bit register that performs the operation of the serial interface, the selection the clock phase and polarity and the selection of the clock source.
CSIM20 is set by an 8-bit memory manipulation instruction.
When “1” is set to bit 7 (CSIE20) of the serial operation mode register 20 (CSIM20), a serial operation can be started by writing data to or reading data from SIO20.
The $\overline{\text{RESET}}$ signal resets the register value to 00H.

(3) Serial I/F receive data buffer register (SIRB20)

This is an 8-bit register read only register that contains the data that has been transferred by the SIO20. Polling of the SRBS20 register can monitor the status of this register. If an overflow occurred, the SIRB20 data will not change its contents until the status is read out and a new byte is transferred to SIRB20.
SIRB20 has to be read with an 8-bit memory manipulation instruction.
After $\overline{\text{RESET}}$ the register value is undefined.

(4) Receive data buffer status register (SRBS20)

This 8-bit read only register reflects the status of the serial I/F receive data buffer (SIRB20). It contains two flags indicating that there is unread data in the receive data buffer or that there is an overflow error.
SRBS20 can be read with an 8-bit memory manipulation instruction.
The $\overline{\text{RESET}}$ signal resets the register value to 00H.

(a) SDVA – Serial Data Valid flag

This read-only bit indicates, that the serial interface receive data buffer (SIRB20) contains unread information from a transmission. SDVA can only be cleared by reading the receive buffer SIRB. Once the SIRB20 is read, SDVA remains cleared until the next byte enters the SIRB20.

(b) SDOF – Overflow flag

This read-only register indicates an overflow error in the SIRB20.
If the SIRB contains unread data (SDVA = ‘1’), and another byte fully enters the shift register, this flag is automatically set, indicating that data will be lost. The unread data from a previous transmission can still be read out of the SIRB20, since the SIRB20 is not updated in case of an overflow condition. Reading the SRBS20 clears the SDOF flag.

SRBS20 can be read with an 1-bit or an 8-bit memory manipulation instruction.
The $\overline{\text{RESET}}$ signal resets the register value to 00H.

Caution: Even if the SIO is in overflow state, it generates an interrupt request, every time a new (lost) byte has completely entered the shift register.

14.3 Serial Interface SIO20 List of SFRS (Special Function Registers)

Table 14-2: List of SFRS (Special Function Registers)

SFR name	Symbol	R/W	Units available for bit manipulation			Value after Reset
			1-bit	8-bit	16-bit	
Serial I/F operation mode register	CSIM20	R/W	×	×	–	00H
Receive data buffer status register	SRBS20	R	–	×	–	00H
Serial I/F receive data buffer	SIRB20	R	–	×	–	undefined
Serial I/O shift register	SIO20	R/W	–	×	–	00H

14.4 Serial Interface Control Registers

The SIO20 uses the following type of register for control functions.

- Serial operation mode register 20 (CSIM20)
- Receive data buffer status register 20 (SBRS20)

(1) Serial I/F operation mode register 20 (CSIM20)

This register is used to enable or disable serial interface channel 3’s serial clock, operation modes, and specific operations.
CSIM20 can be set via an 1-bit or an 8-bit memory manipulation instruction.
The $\overline{\text{RESET}}$ input sets the value to 00H.

Figure 14-2: Serial Operation Mode Register (CSIM20) Format (1/2)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM20	CSIE20	0	0	CLPH	CLPO	MODE0	SCL201	SCL200	R/W	FFA8H	00H

CSIE20	Enable/disable specification for SIO20		
	Shift register operation	Serial counter	Port Note 1
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial function + Port function

CLPH	Clock phase selection
0	Normal mode (MSB is output on the first valid edge of SCK2)
1	Phase mode (MSB is valid before the first valid edge of SCK2)

CLPO	Clock polarity selection
0	Normal mode (send data changes on the falling edge of SCK2)
1	Inverted mode (send data changes on the rising edge of SCK2)

MODE0	Transfer operation modes and flags		
	Operation mode	Transfer start trigger	P21/SO2
0	Transmit/receive mode	Write to SIO20	SO2 output
1	Receive-only mode Note 2	Read from SIO20	Port function

Figure 14-2: Serial Operation Mode Register (CSIM20) Format (2/2)

SCL201	SCL200	Clock selection
0	0	External clock input
0	1	8-bit timer register (TM50)
1	0	$f_X/2^3$
1	1	$f_X/2^7$

Notes: 1. When CSIE20 = 0 (SIO20 operation stop status), the pins connected to SI2 and S02 can be used for port functions.

2. When MODE0 = 1 (Receive Mode), pin P21 can be used for port function.

Cautions: 1. Bits 5 and 6 must be set to 0.

2. While a serial transfer operation is enabled (CSIE20 = 1), be sure to stop the serial transfer operation once before changing the values of bits other than CSIE20 to different data.

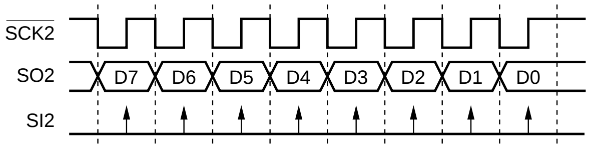
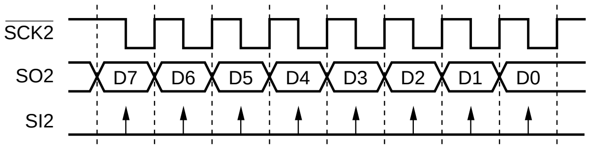
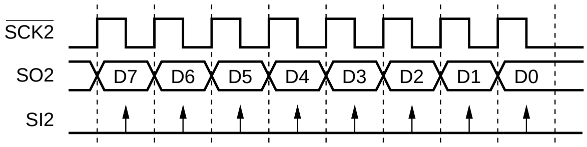
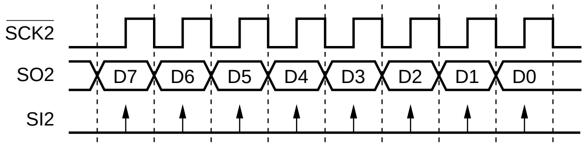
3. When operation is disabled (CSIE20 = 0) during a serial transfer operation, the operation will be stopped immediately. At this time, even if operation is enabled again (CSIE20 = 1) after it was once stopped, the operation will not start. To resume operation, set operation enable (CSIE20 = 1) and then execute an access that will be the start trigger of each transfer operation mode.

4. Changing CSIE20 and other bits at the same time is prohibited. After clearing CSIE20 to 0, change the other bits.

Remark: f_X: Main system clock oscillation frequency

The following shows the relationships between the CLPO and CLPH settings, and the serial transfer clock, data output, and input data capture timing.

Figure 14-3: Serial Transfer Operation Timing According to CLPO and CLPH Settings

CLPO	CLPH	Serial Transfer Operation Clock Selection
0	0	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0
0	1	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0
1	0	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0
1	1	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0

Remarks: **1.** $\overline{\text{SCK2}}$: Serial transfer clock

2. SO2: Data output timing

3. SI2: Input data capture timing

(2) Receive data buffer status register (SRBS20)

This register reflects that there is unread data in the serial receive data buffer register or that there is in an overflow error.
SRBS20 can be read via an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the value to 00H.

Figure 14-4: Receive Data Buffer Status Register (SRBS20) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SRBS20	0	0	0	0	0	0	SDVA	SDOF	R	FFAAH	00H

SDVA	Serial data valid
0	No valid data in serial I/F receive data buffer
1	Valid data in serial I/F receive data buffer

SDOF	Serial data transfer overflow
0	No overflow error
1	Overflow error: Receive data buffer full

(3) Serial I/F data buffer register (SIRB20)

This register contains the data that has been transferred by the SIO20.
SIRB20 can be read by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the value to undefined.

Figure 14-5: Serial I/F data buffer register (SIRB20)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SIRB20	SIRB207	SIRB206	SIRB205	SIRB204	SIRB203	SIRB202	SIRB201	SIRB200	R	FFA9H	undef.

14.5 Operations

The SIO20 has the following two operation modes.

- Operation stop mode
- 3-wire serial I/O mode

14.5.1 Operation Stop Mode

This mode does not perform serial transfers and can therefore reduce power consumption.
In operation stop mode the pins connected to SCK2, SI2 and SO2 can be used for port functions.

(1) Register settings

Operation stop mode is set via serial operation mode register 20 (CSIM20).
CSIM20 can be set via 1-bit or 8-bit memory manipulation instructions.
The $\overline{\text{RESET}}$ input sets the value to 00H.

Figure 14-6: Format of Serial Operation Mode Register (CSIM20)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM20	CSIE20	0	0	CLPH	CLPO	MODE0	SCL201	SCL200	R/W	FFA8H	00H

CSIE20	SIO20 operation enable/disable specification		
	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 CSIE20 = 0 (SIO20 operation stop status), the pins connected to SCK2, SI2 and SO2 can be used for port functions.

14.5.2 3-wire Serial I/O Mode

3-wire serial I/O mode is useful when connecting to a peripheral I/O device that includes a clock-synchronous serial interface, a display controller, etc.
This mode executes data transfers via three lines: a serial clock line (SCK2), serial output line (SO2) and serial input line (SI2).

(1) Register settings

Operation stop mode is set via serial operation mode register 20 (CSIM20).
CSIM20 can be set via 1-bit or 8-bit memory manipulation instructions.
The $\overline{\text{RESET}}$ input sets the value to 00H.

Figure 14-7: Serial Operation Mode Register (CSIM20) Format

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM20	CSIE20	0	0	CLPH	CLPO	MODE0	SCL201	SCL200	R/W	FFA8H	00H

CSIE20	Enable/disable specification for SIO20		
	Shift register operation	Serial counter	Port Note 1
0	Operation stop	Clear	Port function
1	Operation enable	Count operation enable	Serial function + Port function

CLPH	Clock phase selection
0	Normal mode (MSB is output on the first valid edge of SCK2)
1	Phase mode (MSB is valid before the first valid edge of SCK2)

CLPO	Clock polarity selection
0	Normal mode (send data changes on the falling edge of SCK2)
1	Inverted mode (send data changes on the rising edge of SCK2)

MODE0	Transfer operation modes and flags		
	Operation mode	Transfer start trigger	P21/SO2
0	Transmit/receive mode	Write to SIO20	SO2 output
1	Receive-only mode Note 2	Read from SIO20	Port function

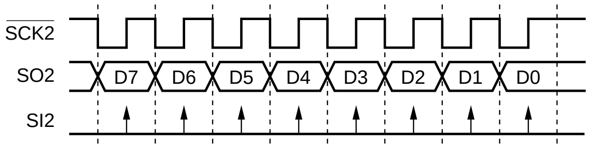
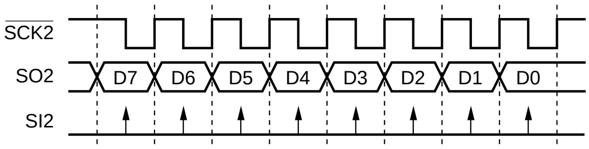
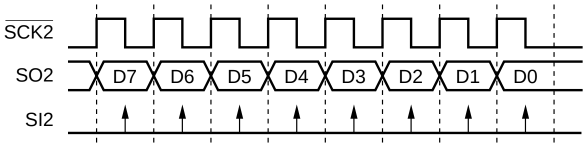
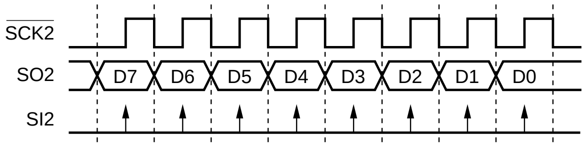
SCL201	SCL200	Clock selection
0	0	External clock input
0	1	8-bit timer register (TM50)
1	0	$f_X/2^3$
1	1	$f_X/2^7$

Notes: 1. When CSIE20 = 0 (SIO20 operation stop status), the pins connected to SI2 and S02 can be used for port functions.

2. When MODE0 = 1(Receive Mode), pin P21 can be used for port function.

The following shows the relationships between the CLPO and CLPH settings, and the serial transfer clock, data output, and input data capture timing.

Figure 14-8: Serial Transfer Operation Timing According to CLPO and CLPH Settings

CLPO	CLPH	Serial Transfer Operation Clock Selection
0	0	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0
0	1	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0
1	0	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0
1	1	 $\overline{\text{SCK2}}$ SO2 SI2 D7 D6 D5 D4 D3 D2 D1 D0

Remarks: 1. SCK2: Serial transfer clock

2. SO2: Data output timing

3. SI2: Input data capture timing

(2) Receive data buffer status register (SRBS20)

This register reflects that there is unread data in the serial receive data buffer register or that there is in an overflow error.
SRBS20 can be read via an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the value to 00H.

Figure 14-9: Receive Data Buffer Status Register (SRBS20) Format

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SRBS20	0	0	0	0	0	0	SDVA	SDOF	R	FFAAH	00H

SDVA	Serial data valid
0	No valid data in serial I/F receive data buffer
1	Valid data in serial I/F receive data buffer

SDOF	Serial data transfer overflow
0	No overflow error
1	Overflow error: Receive data buffer full

(3) Serial I/F data buffer register (SIRB20)

This register contains the data that has been transferred by the SIO20.
SIRB20 can be read by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the value to undefined.

Figure 14-10: Serial I/F data buffer register (SIRB20)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SIRB20	SIRB207	SIRB206	SIRB205	SIRB204	SIRB203	SIRB202	SIRB201	SIRB200	R	FFA9H	undef.

(4) Transfer start

Serial transfer starts when the following two conditions have been satisfied.

- Transmit/receive mode
  When CSIE20 = 1 and MODE0 = 0, transfer starts when writing to SIO20.
- Receive-only mode
  When CSIE20 = 1 and MODE0 = 1, transfer starts when reading from SIO20.

Caution: The transfer of the serial interface will not start when the data is written to SIO20 if before CSIE20 bit is set to “1”.

Completion of an 8-bit transfer automatically stops the serial transfer operation and the interrupt request flag is set.

(5) Operation mode

(a) Master Mode

The SIO operates in Master mode, when one of the flags SCL201 and SCL200 is set.
Only a master SIO can initiate transmissions. Transmission starts from a master SIO by reading or writing to the SIO20 data register. The byte begins shifting out on the SO2 pin under the control of the serial clock. The SCL201 and SCL200 bits determine the speed of the transmission. Through the SCK2 pin the master also controls the shift register of the slave peripheral.
As the byte shifts out on the SO2 pin of the master, another byte shifts in from the slave on the master’s SI2 pin. The transmission ends when all eight bits are shifted out. At the end of the transmission the interrupt INTCSI20 is triggered.

(b) Slave Mode

The SIO operates in slave mode when both, the SCL201 and SCL200 flag are cleared and the external clock is selected. In slave mode the SCK2 pin is the input for the serial clock from the master serial interface. In a slave SIO, data enters the shift register under the control of the serial clock from the master serial interface. After a byte is received to the shift register of a slave SIO, it is transferred to the receive data buffer, the SDVA flag is set and an interrupt INTCSI20 is triggered. To prevent an overflow condition, the slave’s software must then read the SIO data register before another byte fully enters the shift register.
The frequency of the SCK2 for a SIO20 that has configuration as slave does not have to correspond to any particular baud rate.
A slave must complete a write to the shift register at least one bus cycle before the master starts the transmission. When the clock phase bit CLPH is cleared, the first edge of SCK2 starts the transmission. When CLPH is set, reading or writing (depending on MODE0 flag) will start the transmission.

Note: SCK2 must be in clear idle state before the slave is enabled to prevent SCK2 from appearing as a clock edge.

(6) Transmission formats

During a SIO transmission, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). A serial clock line synchronizes shifting and sampling on the two serial data lines.

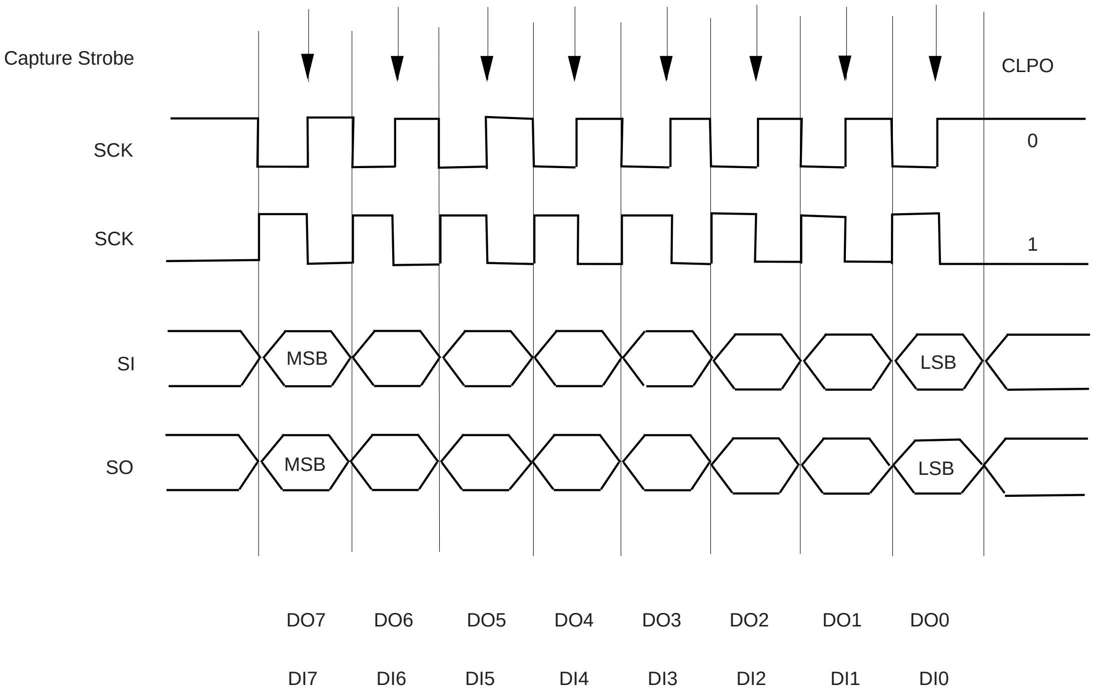
(a) Clock phase and polarity

By software any of four combination of serial clock (SCK) phase and polarity can be selected, using two bits in the SIO20 control register. The clock polarity is specified by the CLPO control bit, which selects an active high or low clock and has no significant effect on the transmission format.
The clock phase (CLPH) control bit selects on of two fundamentally different transmission formats.
The clock phase and polarity have to be identical for the master SIO device and the slave device.

(b) Transmission format when CLPH ='0'

Figure 14-11 shows a SIO transmission with the CLPH control bit set to ‘0’. Two waveforms are shown for SCK one for CLPO = ‘1’ and another one for CLPO = ‘0’. The SO signal is the signal output from the master and the SI is the signal output from the slave.
When CLPH is ‘0’, the master begins driving the MSB at its SO pin on the first active edge (either positive or negative, depending on the setting of CLPO). Therefore the slave uses the first SCK edge as a start transmission trigger.

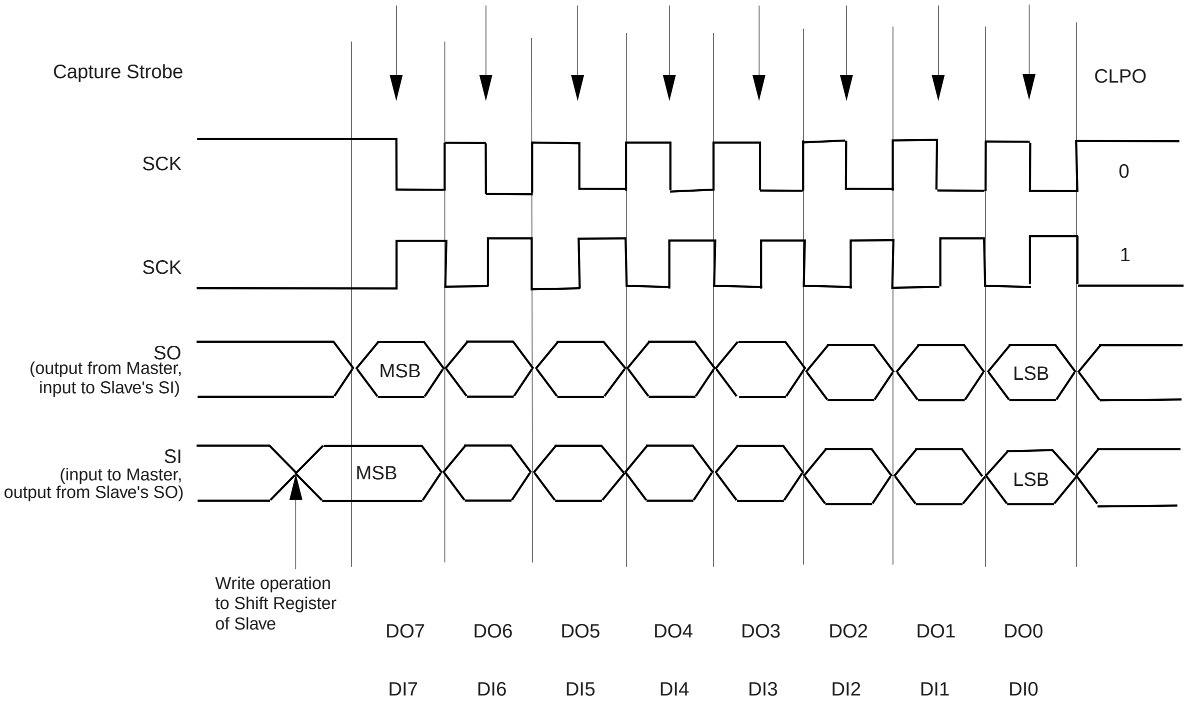
Figure 14-11: Transmission protocol for CLPH = 0



(c) Transmission format when CLPH = ‘1’

Figure 14-12 shows a SIO transmission in which CLPH is set to ‘1’. Two waveforms are shown for SCK, one for CLPO = ‘1’ and another one for CLPO = ‘0’. The SO pin is the output from the master and the SI pin is the output from the slave. When CLPH = ‘1’, the first SCK edge is the MSB capture strobe. Therefore, the slave must begin driving its data before the first SCK edge. To achieve this, the slave starts driving its MSB just after a write access to its shift register.

Figure 14-12: Transmission format for CLPH = 1

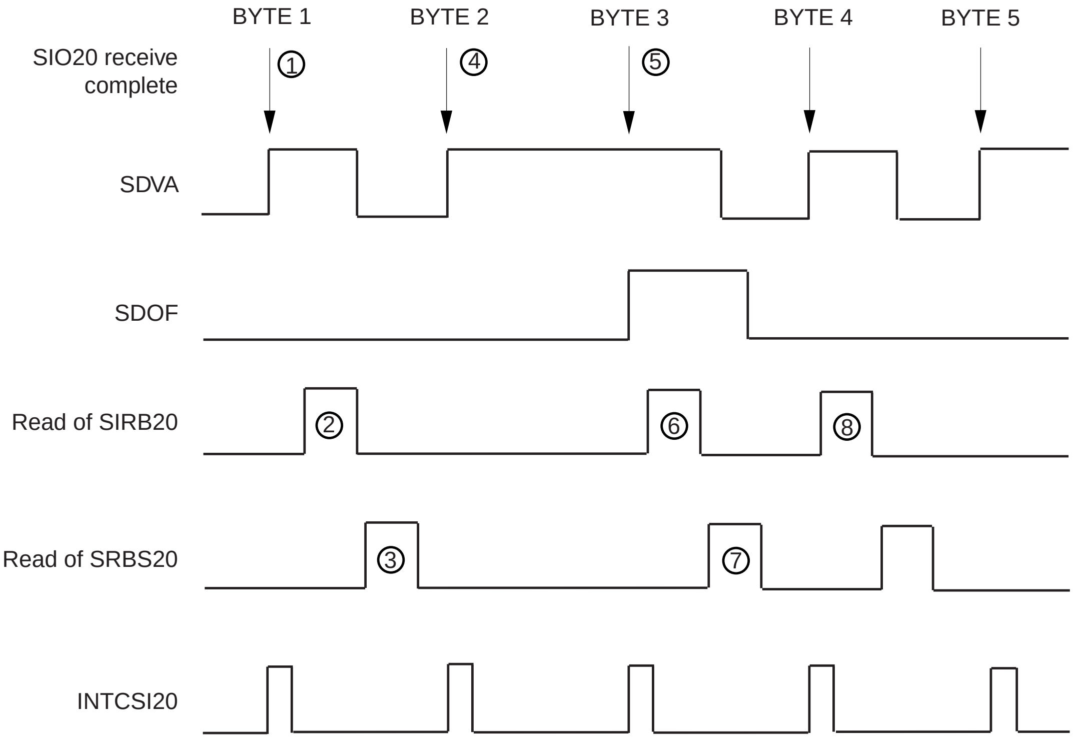


(7) Hardware detectable error condition (Overflow error)

The overflow flag (SDOF) is set if the SIO20 receive data buffer (SIRB20) still contains unread data from a previous transmission when the capture strobe of the LSB of the next transmission occurs (see figure 15-13). If an overflow occurs, the data being received is not transferred to the receive data buffer so that the unread data can still be read. Therefore, an overflow error always indicates the loss of data.

When reading the SRBS20 register the SDOF bits will be cleared. To view the status of the SIO20, the SRBS20 register should be polled every time there is an interrupt triggered by the SIO20, after reading the SIRB20.

Figure 14-13: Overflow error conditions



Remark: ① BYTE 1 transfers from shift register to data receive buffer, setting SDVA
② CPU reads BYTE 1, clearing SDVA
③ CPU reads SRBS20
④ BYTE 2 transfers from shift register to data receive buffer, setting SDVA
⑤ BYTE 3 causes overflow, BYTE 3 is lost, SDOF is
⑥ CPU reads BYTE 2, clearing SDVA
⑦ CPU reads status register SRBS20, clearing SDOF
⑧ CPU reads BYTE 4, clearing SDVA

(8) Operation during standby modes

(a) HALT mode operation

The SIO20 remains active after the execution of a HALT instruction. In Halt mode the SIO20 module registers are not accessible by the CPU. If the INTCSI20 interrupt is enabled it can bring the CPU out of HALT mode, if a transmission is completed.
If the SIO20 functions are not required during HALT mode, disabling the SIO20 module before executing the HALT instruction can reduce the power consumption.

(b) STOP mode operation

The SIO20 can operate in STOP mode, if it has configuration in slave mode. After receive of a byte has finished, the SIO20 triggers the INTCSI20 interrupt, that can, if enabled, bring the device out of STOP mode.

[MEMO]

Chapter 15 Serial Interface SIO30

15.1 SIO30 Functions

The SIO30 has the following three modes.

- Operation stop mode
- 3-wire serial I/O mode
- 2-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” on page 240**.

(2) 3-wire serial I/O mode (fixed as MSB first)

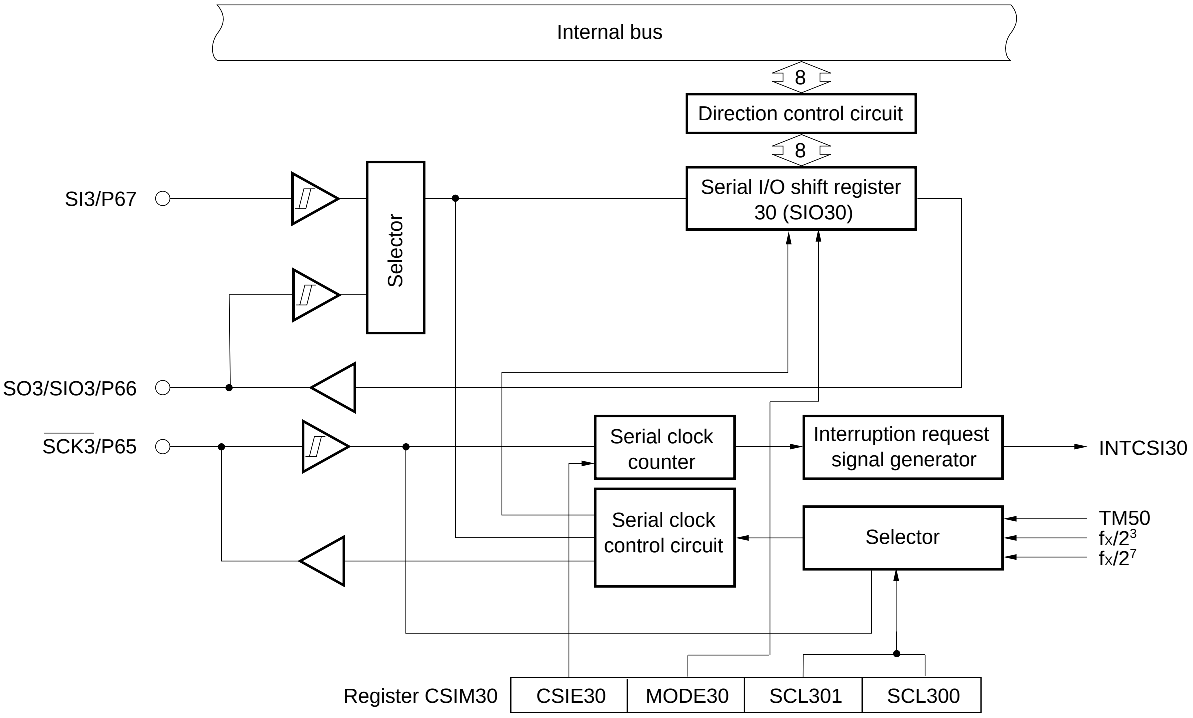
This is an 8-bit data transfer mode using three lines: a serial clock line ($\overline{\text{SCK3}}$), serial output line (SO3), and serial input line (SI3).
Since simultaneous transmit and receive operations are enabled in 3-wire serial I/O mode, the processing time for data transfers is reduced.
The 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” on page 241**.

(3) 2-wire serial I/O mode (fixed as MSB first)

This is an 8-bit data transfer mode using three lines: a serial clock line ($\overline{\text{SCK3}}$) and a serial data input/output line (SIO3).
The first bit in the 8-bit data in serial transfers is fixed as the MSB.

Figure 15-1 shows a block diagram of the SIO30.

Figure 15-1: Block Diagram of SIO30



15.2 SIO30 Configuration

The SIO30 includes the following hardware.

Table 15-1: Composition of SIO30

Item	Configuration
Registers	Serial I/O shift register (SIO30)
Control registers	Serial operation mode register (CSIM30)
	Serial mode switch register (SIOSWI)

(1) Serial I/O shift register (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 (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 (SO30).
When receiving, data is read from the serial input (SI30) 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.

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 after reset
			1-bit	8-bit	16-bit	
Serial operation mode register	CSIM30	R/W	×	×	-	00H
Serial I/O shift register	SIO30	R/W	-	×	-	00H
Serial mode switch register	SIOSWI	R/W	×	×	-	00H

15.4 Serial Interface Control Register

The SIO30 uses the following type of register for control functions.

- Serial operation mode register (CSIM30)
- Serial mode switch register (SIOSWI)

(1) Serial operation mode register (CSIM30)

This register is used to enable or disable the serial clock, selects operation modes, and defines specific operations.
CSIM30 can be set via an 1-bit or an 8-bit memory manipulation instruction.

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

Figure 15-2: Format of Serial Operation Mode Register (CSIM30)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM30	CSIE30	0	0	0	0	0	SCL301	SCL300	R/W	FFAFH	00H

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

SCL301	SCL300	Clock selection (f_X = 8.00 MHz)
0	0	External clock input to SCK3
0	1	8-bit timer TM50 input
1	0	$f_X/2^3$
1	1	$f_X/2^7$

Notes: 1. When CSIE30 = 0 (SIO30 operation stop status), the pins connected to SI3 and SO3 can be used for port functions.

2. The bits 2 to 6 have to be set to 0.

(2) Serial mode switch register (SIOSWI)

This register is used to select the SIO31's 3-wire mode or 2-wire mode data communication mode. SIOSWI is set by an 1-bit or an 8-bit memory manipulation instruction.

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

Figure 15-3: Format of Serial Mode Switch Register (SIOSWI)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SIOSWI	0	0	0	0	0	0	0	SIOSWI	R/W	FFAEH	00H

SIOSWI	SIO30 - Serial mode switch
0	3-wire mode (reset)
1	2-wire mode

The following operation modes and start trigger have to be set for the usage of the 3-wire mode or the 2-wire mode data communication mode.

Table 15-3: Operating Modes and Start Trigger

3-wire or 2-wire mode of SIO30 (SIOSWI)	Operation Mode Flag		
	Operation mode	Start trigger	Port Function
2-wire mode	Transmit/Receive mode	SIO30 write	P65: SCK3 P66: SIO3
3-wire mode	Transmit/Receive mode	SIO30 write	P65: SCK3 P66: SO3 P67: SI3

15.5 Serial Interface Operations

This section explains three modes of SIO30.

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.

Register settings
The operation stop mode can be set via the serial operation mode register (CSIM30).
CSIM30 can be set via an 1-bit or an 8-bit memory manipulation instructions.

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

Figure 15-4: Format of Serial Operation Mode Register (CSIM30)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM30	CSIE30	0	0	0	0	0	SCL301	SCL300	R/W	FFAFH	00H

CSIE30	SIO30 Operation Enable/Disable Specification		
	Shift register operation	Serial counter	Port
0	Operation stop	Clear	Port function[Note 1]
1	Operation enable	Count operation enable	Serial operation + port function[Note 2]

Note: When CSIE30 = 0 (SIO30 operation stop status), the pins SI3, SO3 and $\overline{\text{SCK3}}$ 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 ($\overline{\text{SCK3}}$), serial output line (SO3), and serial input line (SI3).

(1) Register settings

The 3-wire serial I/O mode is set via serial operation mode register (CSIM30).
CSIM30 can be set via an 1-bit or an 8-bit memory manipulation instructions.

The $\overline{\text{RESET}}$ input set the value to 00H.

Figure 15-5: Format of Serial Operation Mode Register (CSIM30)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM30	CSIE30	0	0	0	0	0	SCL301	SCL300	R/W	FFAFH	00H

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port
0	Operation stop	Clear	Port function Note 1
1	Operation enable	Count operation enable	Serial operation + port function Note 2

SCL301	SCL300	Clock selection (f_X = 8.00 MHz)
0	0	External clock input to SCK3
0	1	8-bit timer TM50 input
1	0	$f_X/2^3$
1	1	$f_X/2^7$

Notes: 1. When CSIE30 = 0 (SIO30 operation stop status), the pins SI3, SO3 and $\overline{\text{SCK3}}$ can be used for port functions.

2. The bits 2 to 6 have to be set to 0.

Caution: In the 3-wire serial I/O mode, set the port mode register as required. Set the output latch of the port to 0.

<When SIO30 is used>

Modes	Values	Settings
During serial clock output (master transmission or master reception)	PM65 = 0	Sets P65 ($\overline{\text{SCK3}}$) to output mode
	P65 = 0	Sets output latch of P65 to 0
During serial clock input (slave transmission or slave reception)	PM65 = 1	Sets P65 ($\overline{\text{SCK3}}$) to input mode
Transmit/receive mode	PM66 = 0	Sets P66 (SO3) to output mode
	P66 = 0	Sets output latch of P66 to 0
	PM67 = 1	Sets P67 (SI3) to input mode

(2) Serial mode switch register (SIOSWI)

This register is used to select the SIO31's 3-wire mode or 2-wire mode data communication mode. SIOSWI is set by an 1-bit or an 8-bit memory manipulation instruction.

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

Figure 15-6: Format of Serial Mode Switch Register (SIOSWI)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SIOSWI	0	0	0	0	0	0	0	SIOSWI	R/W	FFAEH	00H

SIOSWI	SIO30 - Serial mode switch
0	3-wire mode (reset)
1	2-wire mode

The following operation modes and start trigger have to be set for the usage of the 3-wire mode or the 2-wire mode data communication mode.

Table 15-4: Operating Modes and Start Trigger

3-wire or 2-wire mode of SIO30 (SIOSWI)	Operation Mode Flag		
	Operation mode	Start trigger	Port Function
2-wire mode	Transmit/Receive mode	SIO30 write	P65: SCK3 P66: SIO3
3-wire mode	Transmit/Receive mode	SIO30 write	P65: SCK3 P66: SO3 P67: SI3

15.5.3 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, a display controller, etc.
This mode executes the data transfer via two lines: a serial clock line ($\overline{\text{SCK3}}$) and serial input/output line (SIO3).

(1) Register settings

The 2-wire serial I/O mode is set via serial operation mode register 30 (CSIM30).
CSIM30 can be set via an 1-bit or an 8-bit memory manipulation instructions.

The $\overline{\text{RESET}}$ input set the value to 00H.

Figure 15-7: Format of Serial Operation Mode Register (CSIM30)

	<7>	6	5	4	3	2	1	0	R/W	Address	After Reset
CSIM30	CSIE30	0	0	0	0	0	SCL301	SCL300	R/W	FFAFH	00H

CSIE30	Enable/disable specification for SIO30		
	Shift register operation	Serial counter	Port
0	Operation stop	Clear	Port function[Note 1]
1	Operation enable	Count operation enable	Serial operation + port function[Note 2]

SCL301	SCL300	Clock selection (f_X = 8.00 MHz)
0	0	External clock input to SCK3
0	1	8-bit timer TM50 input
1	0	$f_X/2^3$
1	1	$f_X/2^7$

Note: When CSIE30 = 0 (SIO30 operation stop status), the pins SIO3 and $\overline{\text{SCK3}}$ can be used for port functions.

Caution: In the 2-wire serial I/O mode, set the port mode register as required. Set the output latch of the port to 0.

<When SIO30 is used>

Modes	Values	Settings
During serial clock output (master transmission or master reception)	PM65 = 0	Sets P65 ($\overline{\text{SCK3}}$) to output mode
	P65 = 0	Sets output latch of P65 to 0
During serial clock input (slave transmission or slave reception)	PM65 = 1	Sets P65 ($\overline{\text{SCK3}}$) to input mode
Transmit/receive mode	PM66 = 0	Sets P66 (SIO3) to output mode (Transmit mode)
	PM66 = 1	Sets P66 (SIO3) to input mode (Receive mode)
	P66 = 0	Sets output latch of P66 to 0

(2) Serial mode switch register (SIOSWI)

This register is used to select the SIO31's 3-wire mode or 2-wire mode data communication mode. SIOSWI is set by an 1-bit or an 8-bit memory manipulation instruction.

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

Figure 15-8: Format of Serial Mode Switch Register (SIOSWI)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
SIOSWI	0	0	0	0	0	0	0	SIOSWI	R/W	FFAEH	00H

SIOSWI	SIO30 - Serial mode switch
0	3-wire mode (reset)
1	2-wire mode

The following operation modes and start trigger have to be set for the usage of the 2-wire mode.

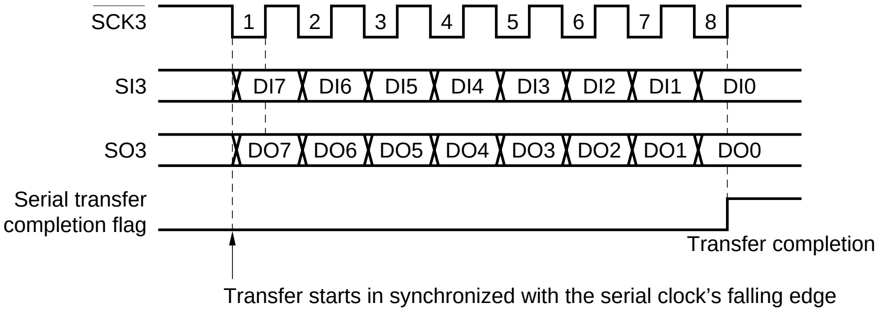
Table 15-5: Operating Modes and Start Trigger

3-wire or 2-wire mode of SIO30 (SIOSWI)	Operation Mode Flag		
	Operation mode	Start trigger	Port Function
2-wire mode	Transmit/Receive mode	SIO30 write	P65: SCK3 P66: SIO3
3-wire mode	Transmit/Receive mode	SIO30 write	P65: SCK3 P66: SO3 P67: SI3

(3) 3-wire 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 (SIO30) is shifted synchronized with the falling edge of the serial clock. The transmission data is held in the SO3 latch and is transmitted from the SO3 pin. The data is received via the SI3 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.

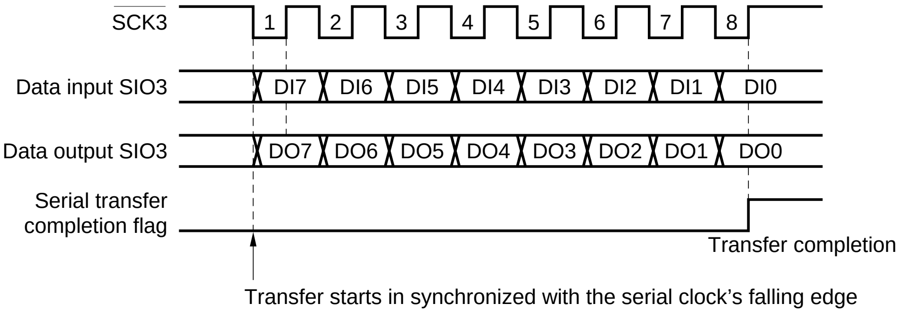
Figure 15-9: Timing of Three-wire Serial I/O Mode


SCK3
1
2
3
4
5
6
7
8
SI3
DI7
DI6
DI5
DI4
DI3
DI2
DI1
DI0
SO3
DO7
DO6
DO5
DO4
DO3
DO2
DO1
DO0
Serial transfer completion flag
Transfer completion
Transfer starts in synchronized with the serial clock's falling edge

(4) 2-wire communication operations

In the two-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 (SIO30) is shifted synchronized with the falling edge of the serial clock. The transmission data is held in the SIO3 latch and is transmitted from the SIO3 pin. The data is received via the SIO3 pin synchronized with the rising edge of the serial clock is latched to SIO3.
The completion of an 8-bit transfer automatically stops operation of SIO3 and sets interrupt request flag.

Figure 15-10: Timing of Two-wire Serial I/O Mode



(5) Transfer start

A serial transfer starts when the following conditions have been satisfied and transfer data has been set to serial I/O shift register 30 (SIO30).

- The SIO30 operation control bit must be set (CSIE = 1)
- In Transmit/receive mode
  When CSIE30 = 1, transfer starts when writing to 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.
After an 8-bit serial transfer, the internal serial clock is either stopped or is set to high level.

Chapter 16 Serial Interface Channel UART

16.1 UART Functions

The serial interface UART has the following two modes.

(1) Operation stop mode

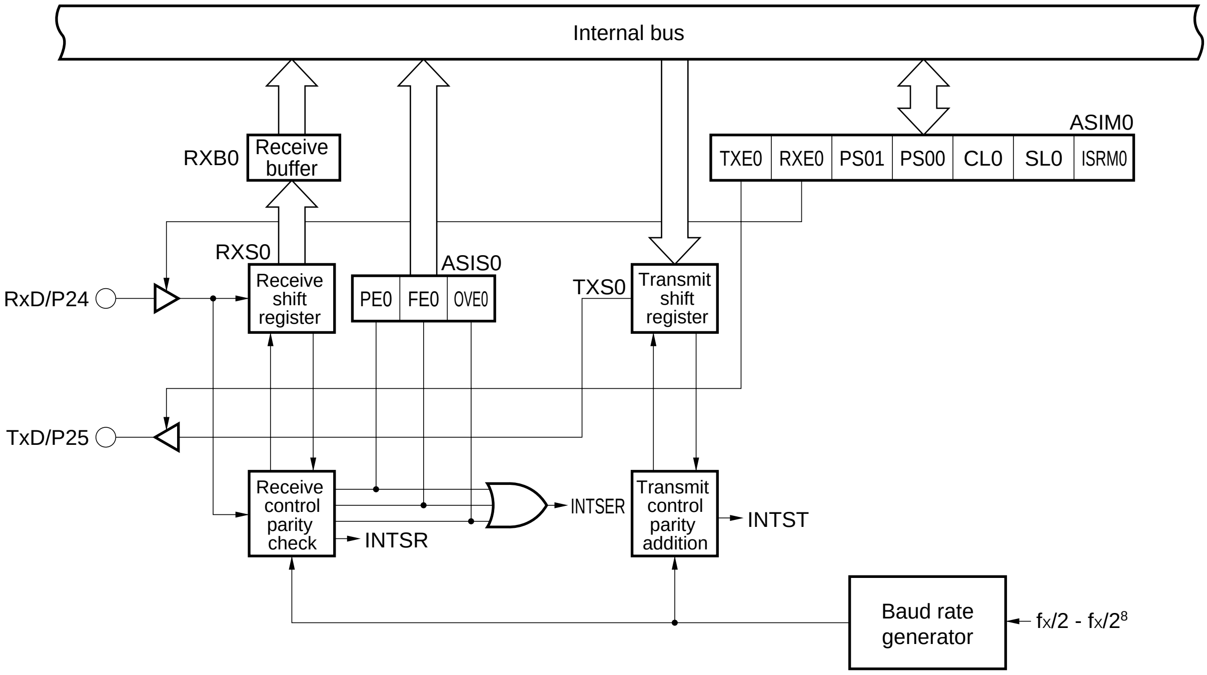
This mode is used if the serial transfer is performed to reduce power consumption.
For details, see **16.5.1 Operation Stop Mode**.

(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 **16.5.2 Asynchronous Serial Interface (UART) Mode**.

Figure 16-1 shows a block diagram of the UART macro.

Figure 16-1: Block Diagram of UART



16.2 UART Configuration

The UART includes the following hardware.

Table 16-1: 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.

Cautions: 1. Do not write to TXS0 during a transmit operation.

2. 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.

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-bit	16-bit	
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	R	-	×	-	
Baud rate generator control register	BRGC0	R/W	-	×	-	

16.4 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 an 1-bit or an 8-bit memory manipulation instructions.

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

Figure 16-2 shows the format of ASIM0.

Figure 16-2: Format of Asynchronous Serial Interface Mode Register (ASIM0) (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0	R/W	FFA0H	00H

TXE0	RXE0	Operation mode	RXD0/P62 pin function	TXD0/P63 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 transmission No parity detection during reception (parity errors do not occur)
1	0	Odd parity
1	1	Even parity

CL0	Character length specification
0	7 bits
1	8 bits

Figure 16-2: Format of Asynchronous Serial Interface Mode Register (ASIM0) (2/2)

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: Before writing different data to ASIM0, please note the following instructions:

1. **Never rewrite bits 6 or 7 (RXE0 and TXE0) during a transmit operation. Wait until transmit operation is completed.**

2. **During a receive operation you may change RXE0 only. But note that the receive operation will be stopped immediately and the contents of RXB0 and ASIS0 do not change, nor does INTSR0 or INTSER0 occur.**

3. **Never change bits 1 to 5 (ISRM0 to PS01) unless bits 6 and 7 (RXE0 and TXE0) were cleared to 0 before.**
   Bit 0 must always be 0.

(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 16-3: Format of Asynchronous Serial Interface Status Register (ASIS0)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ASIS0	0	0	0	0	0	PE0	FE0	OVE0	R	FFA1H	00H

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)

FE0	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 (BRGC0)

This register sets the serial clock for UART.
BRGC0 can be set via an 8-bit memory manipulation instruction.

When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 16-4 shows the format of BRGC0.

Figure 16-4: Format of Baud Rate Generator Control Register (BRGC0) (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00	R/W	FFA2H	00H

(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

Figure 16-4: Format of Baud Rate Generator Control Register (BRGC0) (2/2)

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 prohibited	-

Caution: Writing to BRGC0 when RXE0 and / or TXE0 are set to 1 (receive and / or transmit operation selected) may cause abnormal output from the baud rate generator and disable further communication operations. Therefore do write to BRGC0 only when RXE0 and TXE0 are set to 0.

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 ($0 \leq k \leq 14$)

16.5 Serial Interface Operations

This section explains the different modes of the UART.

16.5.1 Operation stop mode

This mode is used when serial transfer is performed to reduce power consumption.
In the operation stop mode, pins can be used as ordinary ports.

Register settings
Operation stop mode settings are made via the asynchronous serial interface mode register (ASIM0).
ASIM0 can be set via an 1-bit or an 8-bit memory manipulation instructions.
When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 16-5: Register Settings

	<7>	<6>	5	4	3	2	1	0	R/W	Address	After Reset
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0	R/W	FFA0H	00H

TXE0	RXE0	Operation mode	RXD0/P62 pin function	TXD0/P63 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: Before writing different data to ASIM0, please note the following instructions:

1. **Never rewrite bits 6 or 7 (RXE0 and TXE0) during a transmit operation. Wait until transmit operation is completed.**

2. **During a receive operation you may change RXE0 only. But note that the receive operation will be stopped immediately and the contents of RXB0 and ASIS0 do not change, nor does INTSR0 or INTSER0 occur.**

3. **Never change bits 1 to 5 (ISRM0 to PS01) unless bits 6 and 7 (RXE0 and TXE0) were cleared to 0 before.**
   Bit 0 must always be 0.

16.5.2 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 16-6: Format of Asynchronous Serial Interface Mode Register (ASIM0) (1/2)

	<7>	<6>	5	4	3	2	1	0	R/W	Address	After Reset
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0	R/W	FFA0H	00H

TXE0	RXE0	Operation mode	RXD0/P62 pin function	TXD0/P63 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 transmission No parity detection during reception (parity errors do not occur)
1	0	Odd parity
1	1	Even parity

CL0	Character length specification
0	7 bits
1	8 bits

Figure 16-6: Format of Asynchronous Serial Interface Mode Register (ASIM0) (2/2)

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: Before writing different data to ASIM0, please note the following instructions:

1. **Never rewrite bits 6 or 7 (RXE0 and TXE0) during a transmit operation. Wait until transmit operation is completed.**

2. **During a receive operation you may change RXE0 only. But note that the receive operation will be stopped immediately and the contents of RXB0 and ASIS0 do not change, nor does INTSR0 or INTSER0 occur.**

3. **Never change bits 1 to 5 (ISRM0 to PS01) unless bits 6 and 7 (RXE0 and TXE0) were cleared to 0 before.**
   Bit 0 must always be 0.

(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 16-7: Format of Asynchronous Serial Interface Status Register (ASIS0)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
ASIS0	0	0	0	0	0	PE0	FE0	OVE0	R	FFA1H	00H

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), 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.

(c) Baud rate generator control register (BRGC0)

BRGC0 can be set via an 8-bit memory manipulation instruction.
When $\overline{\text{RESET}}$ is input, its value is 00H.

Figure 16-8: Format of Baud Rate Generator Control Register (BRGC0) (1/2)

	7	6	5	4	3	2	1	0	R/W	Address	After Reset
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00	R/W	FFA2H	00H

(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

Figure 16-8: Format of Baud Rate Generator Control Register (BRGC0) (2/2)

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 prohibited	-

Caution: Writing to BRGC0 when RXE0 and / or TXE0 are set to 1 (receive and / or transmit operation selected) may cause abnormal output from the baud rate generator and disable further communication operations. Therefore do write to BRGC0 only when RXE0 and TXE0 are set to 0.

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.

- **Baud rate setting**

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.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1}(k + 16)} \ [\text{kbps}]$$

fx : Oscillation frequency of main system clock in MHz

n : Value set via TPS00 to TPS02 ($1 \leq n \leq 8$)
For details, see Table 16-3.

k : Value set via MDL00 to MDL02 ($0 \leq k \leq 14$) in register BRGC0

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 is shown in Figure 16-4, “Format of Baud Rate Generator Control Register (BRGC0) (1/2),” on page 253.

Table 16-3: Relation between 5-bit Counter’s Source Clock and “n” Value

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

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 16-4 describes the relation between the main system clock and the baud rate and
Figure 16-9 shows an example of a baud rate error tolerance range.

Table 16-4: Relation between Main System Clock and Baud Rate

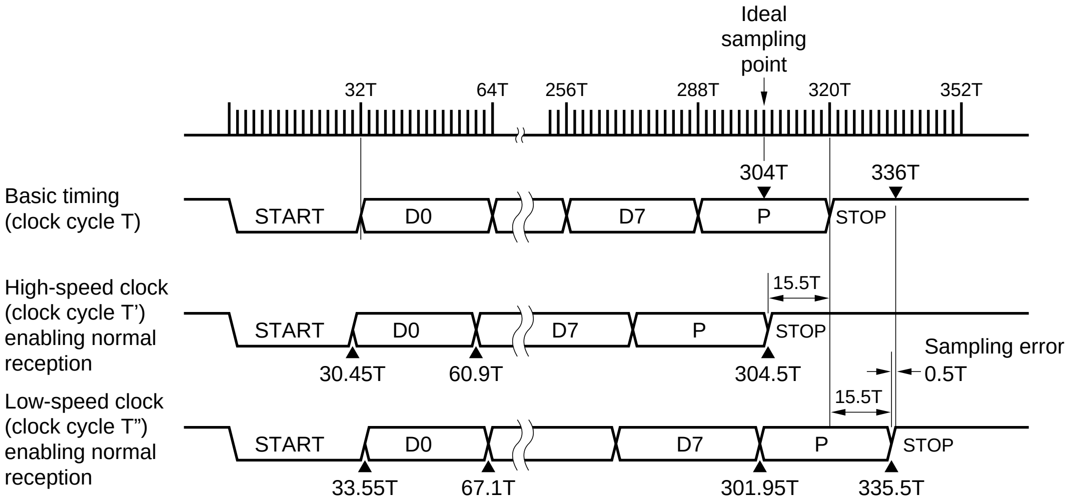
Baud rate (bps)	f_X = 8.386 MHz		f_X = 8.000 MHz		f_X = 5.000 MHz		f_X = 4.1943 MHz	
	BRGCO	ERR (%)	BRGCO	ERR (%)	BRGCO	ERR (%)	BRGCO	ERR (%)
600	7BH	1.10	7AH	0.16	70H	1.73	6BH	1.14
1200	6BH	1.10	6AH	0.16	60H	1.73	5BH	1.14
2400	5BH	1.10	5AH	0.16	50H	1.73	4BH	1.14
4800	4BH	1.10	4AH	0.16	40H	1.73	3BH	1.14
9600	3BH	1.10	3AH	0.16	30H	1.73	2BH	1.14
19200	2BH	1.10	2AH	0.16	20H	1.73	1BH	1.14
31250	21H	-1.34	20H	0	14H	0	11H	-1.31
38400	1BH	1.10	1AH	0.16	10H	1.73	0BH	1.14
76800	0BH	1.10	0AH	0.16	00H	1.73	-	-
115200	02H	1.10	01H	2.12	-	-	-	-

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 16-9: Error Tolerance (when k = 0), including Sampling Errors



Caution: The above tolerance value is the value calculated based on the ideal sample point. In the actual design, allow margins that include errors of timing for detecting a start bit.

Remark: T: 5-bit counter's source clock cycle

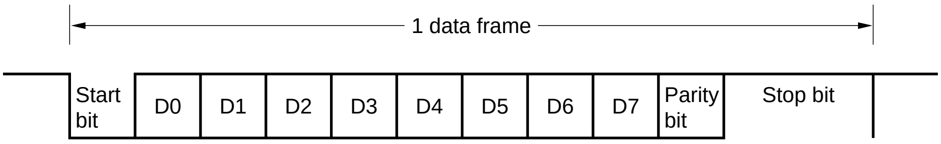
$$\text{Baud rate error tolerance (when k = 0)} = \frac{\pm 15.5 \times 100}{320} = 4.8438\ (\%)$$

(2) Communication operations

(a) Data format

As shown in Figure 16-10, 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 16-10: 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. In this case 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.

- **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.

- **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.

- **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 an “1”.

- **No parity**

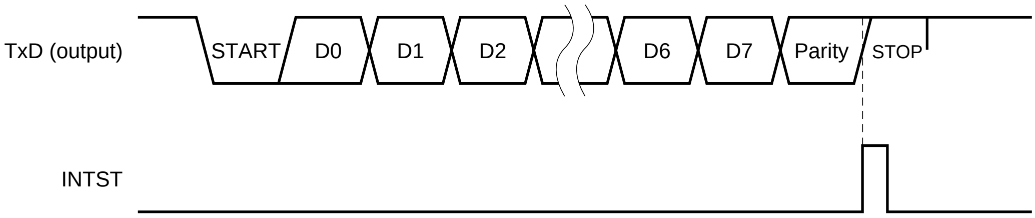
No parity bit is added to the transmit data.
During reception, receive data is regarded as having no parity bit. Since there is no parity bit, no parity errors will occur.

(c) Transmission

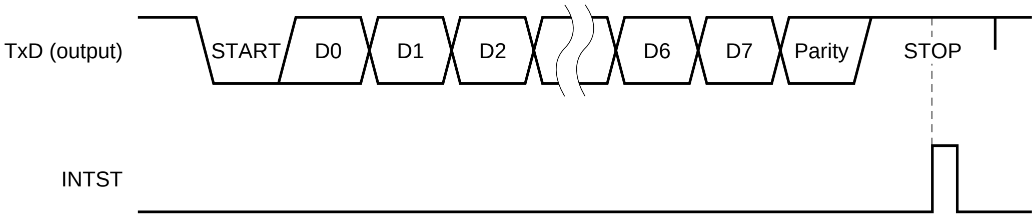
The transmit operation is started when transmit data is written to the transmit shift register (TXS0).
A start bit, parity bit, and stop bit(s) are automatically added to the data.
Starting the transmit operation shifts out the data in TXS0, thereby emptying TXS0, after which a transmit completion interrupt (INTST0) is issued.
The timing of the transmit completion interrupt is shown in Figure 16-11.

Figure 16-11: Timing of Asynchronous Serial Interface Transmit Completion Interrupt

(i) Stop bit length: 1 bit



(ii) Stop bit length: 2 bits

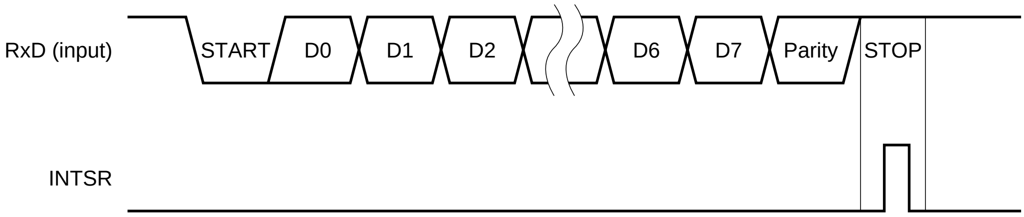


Caution: Do not write to the asynchronous serial interface mode register (ASIM0) during a transmit operation. Writing to ASIM0 during a transmit operation may disable further transmit operations (in such cases, enter a $\overline{\text{RESET}}$ to restore normal operation).
Whether or not a transmit operation is in progress can be determined via software using the transmit completion interrupt (INTST0) or the interrupt request flag (STIF) that is set by INTST0.

(d) Reception

The receive operation is enabled when bit 6 (RXE0) of the asynchronous serial interface mode register (ASIM0) is set to "1", 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, 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 (INTSR0) occurs.
Even if an error has occurred, the receive data in which the error occurred is still transferred to RXB0 and INTSR0 occurs (see Figure 14-9).
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 will change, nor does INTSR0 or INTSER0 occur.
Figure 16-12 shows the timing of the asynchronous serial interface receive completion interrupt.

Figure 16-12: Timing of Asynchronous Serial Interface Receive Completion Interrupt



Cautions: 1. 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.

2. If the receive operation is enabled with the RXD0 pin at the low level, the receive operation is immediately aborted. Make sure that the RXD0 pin input is at the high level before enabling the receive operation.

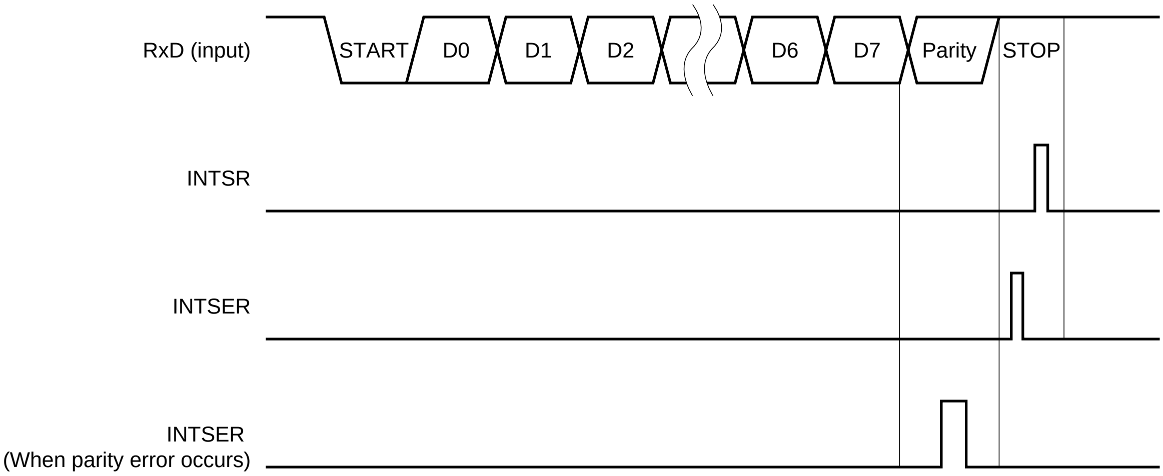
(e) Receive errors

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 (INTSER0) will occur. Receive error interrupts are generated before receive interrupts (INTSR0).
Table 16-5 lists the causes of receive errors.
As part of the receive error interrupt (INTSER0) servicing, the contents of ASIS0 can be read to determine which type of error occurred during the receive operation (see Table 16-5 and Figure 16-13).
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 16-5: 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 16-13: 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.

16.6 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), the transmit shift register (TXS0), the receive shift register (RXS0), and the 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.

Table 17-1: Outline of the Function

Feature	Details
Protocol	CAN2.0 with active extended frame capability (Bosch specification 2.0 part B)
Baudrate	Max. 500 Kbps 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	One input receive shadow buffer (not readable by user) Up to 16 receive message objects including 2 masks Two transmit channels
Message sorting	Unique identifier on all 16 receive message objects Up to 2 message objects with mask Global mask for all messages
DCAN protocol	SFR access for general control
Interrupt	Transmit interrupt for each channel One receive interrupt with enable control for each message One error interrupt
Time functions	Support of time stamp and global time system Programmable single shot mode
Diagnostic	Readable error counters “Valid protocol activity flag” for verification of bus connection “Receive only” mode for automatic baudrate detection
Power down modes	Sleep mode: Wake up from CAN bus Stop mode: No wake-up from CAN bus

17.1 CAN Protocol

CAN is an abbreviation of "Controller Area Network", and is a class C high speed multiplexed communication protocol. CAN is specified by Bosch in the CAN specification 2.0 from September 1991 and is standardized in ISO-11898 (International Organization for Standardization) and SAE (Society of Automotive Engineers).

17.1.1 Protocol Mode Function

(1) Standard format mode

- This mode supports an 11-bit message identifier thus making it possible to differentiate between 2048 types of messages.

(2) Extended format mode

- In the extended format mode, the identifier has 29 bits. It is built by the standard identifier (11 bits) and an extended identifier (18 bits).
- When the IDE bits of the arbitration field is "recessive", the frame is sent in the extended format mode.
- When a message in extended format mode and a remote frame in standard format mode are simultaneously transmitted, the node transmitting the message with the standard mode wins the arbitration.

(3) Bus values

- The bus can have one of two complementary logical values: "dominant" or "recessive". During simultaneous transmission of "dominant" and "recessive" bits, the resulting bus value will be "dominant" (non destructive arbitration).
- For example, in case of a wired-AND implementation of the bus, the “dominant” level would be represented by a logical “0” and the “recessive” level by a logical “1”. This specific representation is used in this manual.
- Physical states (e.g. electrical voltage, light) that represent the logical levels are not given in this document.

17.1.2 Message Format

The CAN protocol message supports different types of frames. The types of frames are listed below:

- Data frame: Carries the data from a transmitter to the receiver.
- Remote frame: Transmission demand frame from the requesting node.
- Error frame: Frame sent on error detection.
- Overload frame: Frame sent when a data or remote frame would be overwritten by the next one before the receiving node could process it. The reception side did not finish its operations on the reception of the previously received frame yet.

17.1.3 Data Frame / Remote Frame

Figure 17-1: Data Frame

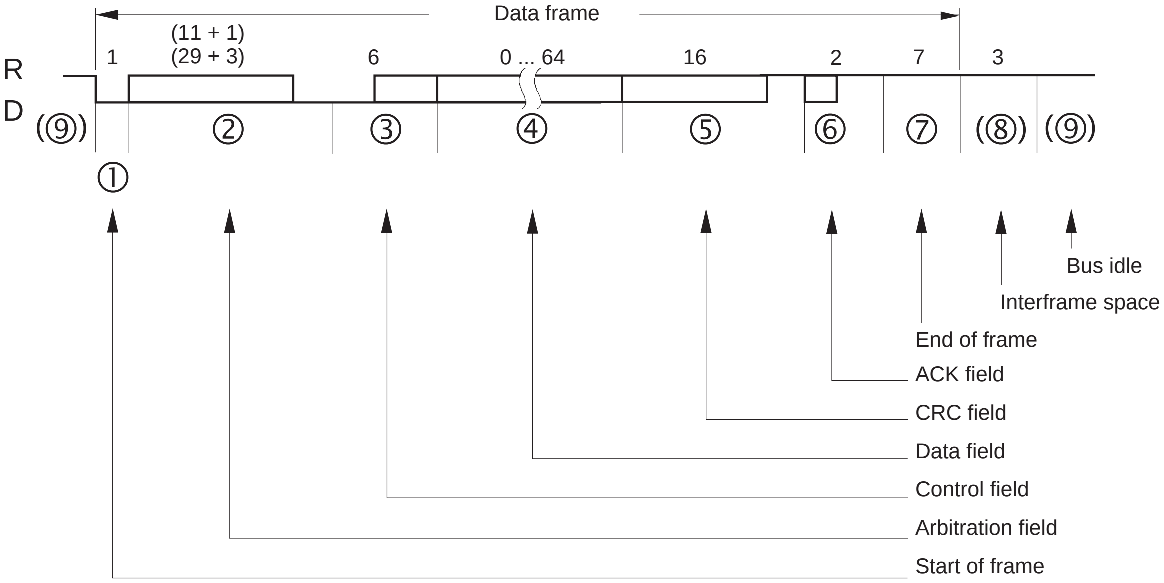
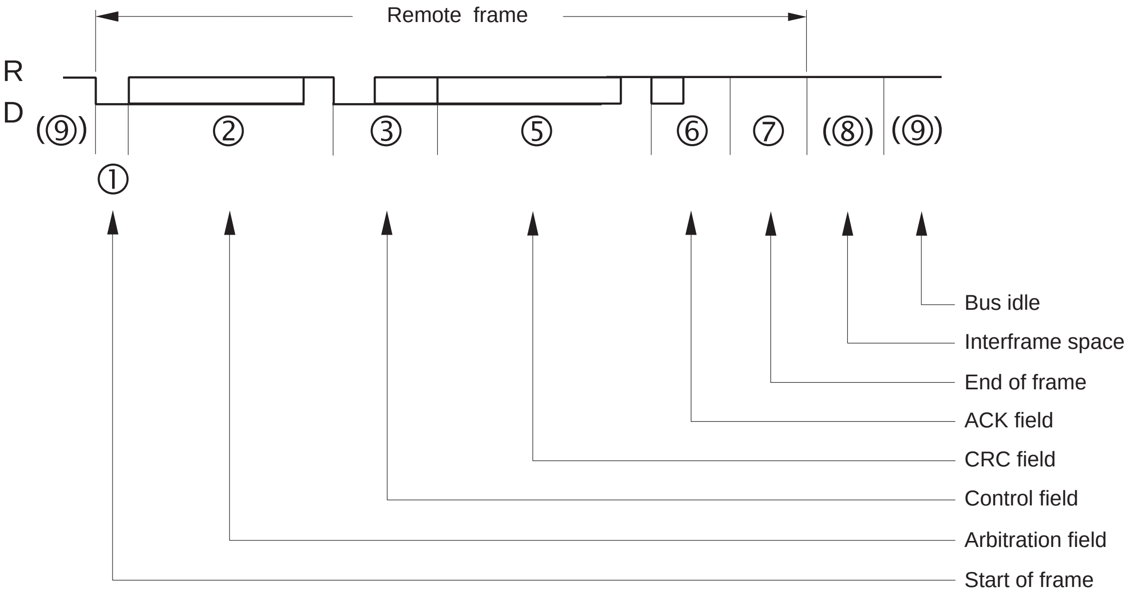


Figure 17-2: Remote Frame

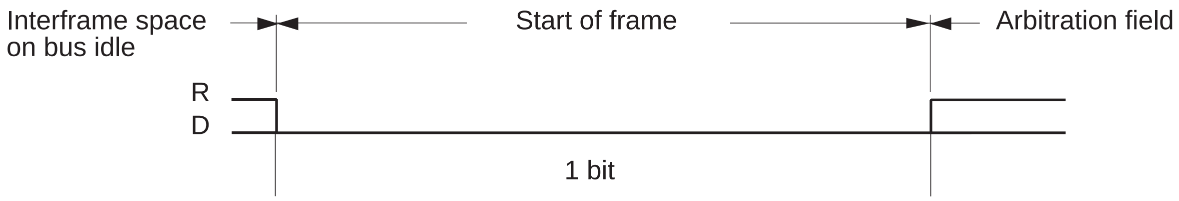


Remark: This frame is transmitted when the reception node requests transmission. Data field is not transmitted even if the data length code ≠ '0' in the control field.

17.1.4 Description of each field

(1) "R" indicates recessive level. "D" indicates dominant level.
Start of frame: The start of data frame and remote frame are indicated.

Figure 17-3: Data Frame



- The start of frame (SOF) is denoted by the falling edge of the bus signal.
- Reception continues when 'Dominant level' is detected at the sample point.
- The bus becomes idle state when 'Recessive level' is detected at a sample point.

(2) Arbitration field: Sets priority, specifies data frame or remote frame, and defines the protocol mode.

Figure 17-4: Arbitration Field/Standard Format Mode

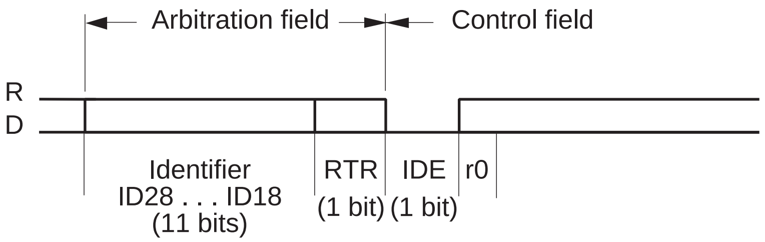
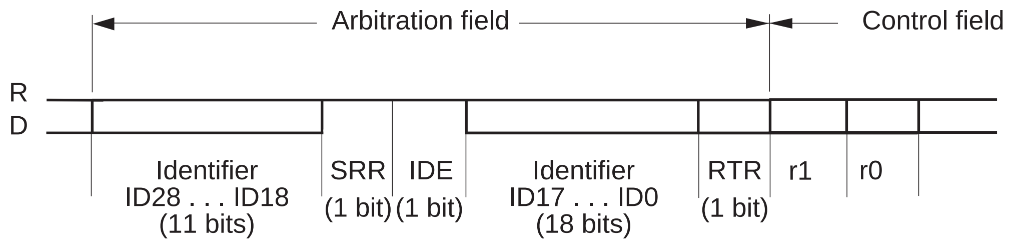


Figure 17-5: Arbitration Field/Extended Format Mode



- ID28 - ID0 is the identifier.
- The identifier is transmitted with MSB at first position.
- Substitute Remote Request (SRR) is only used in extended format mode and is always recessive.

Table 17-2: Bit Number of the Identifier

Protocol Mode Identifier	Number
Standard format mode	11 bits
Extended format mode	29 bits

Table 17-3: RTR Setting

Frame Type	RTR Bit
Data frame	0
Remote frame	1

Table 17-4: Mode Setting

Protocol Mode	IDE Bit
Standard format mode	0
Extended format mode	1

(3) Control field: The data byte number DLC in the data field specifies the number of data bytes in the current frame (DLC=0 to 8).

Figure 17-6: Control Field (Standard Format Mode)

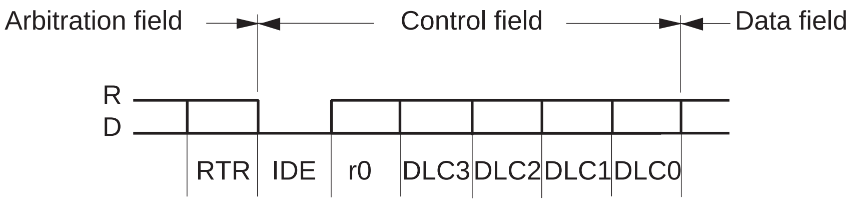
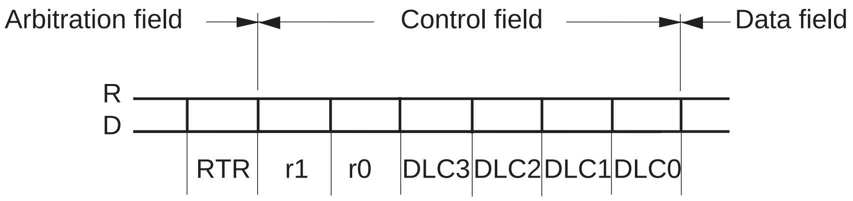


Figure 17-7: Control Field (Extended Format Mode)



- The bits r0 and r1 are reserved bits for future use and are recommended to be recessive.

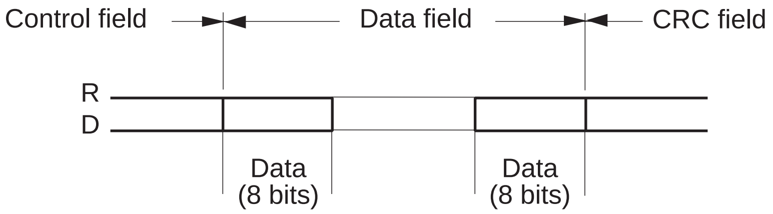
Table 17-5: Data Length Code Setting

Data Length Code				
DLC3	DLC2	DLC1	DLC0	Number of Data Bytes
0	0	0	0	0
0	0	0	1	1
.	.	.	.	
.	.	.	.	
0	1	1	1	7
1	X	X	X	8

Remark: In case of a remote frame, the data field is not generated even if data length code ≠ '0'.

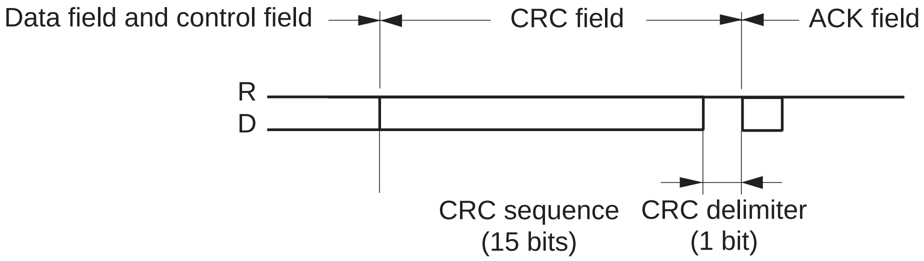
(4) Data field: This field carries the data bytes to be sent. The number of data bytes is defined by the DLC value.

Figure 17-8: Data Field



(5) CRC field: This field consists of a 15-bit CRC sequence to check the transmission error and a CRC delimiter.

Figure 17-9: CRC Field



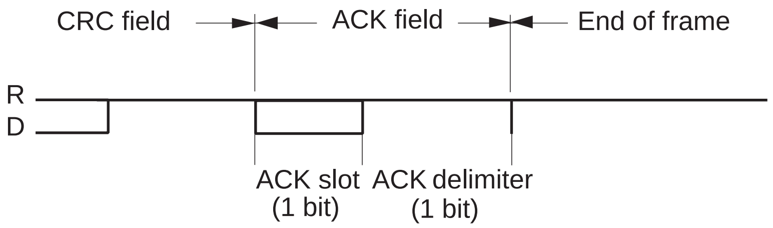
- 15 bits CRC generation polynomial is expressed by

$$P(X) = X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1.$$

- Transmission node: Transmits the CRC sequence calculated from the start of frame, arbitration field, control field and data field eliminating stuff bits.
- Reception node: The CRC received will be compared with the CRC calculated in the receiving node. For this calculation the stuff bits of the received CRC are eliminated. In case these do not match, the node issues an error frame.

(6) ACK field: For check of normal reception.

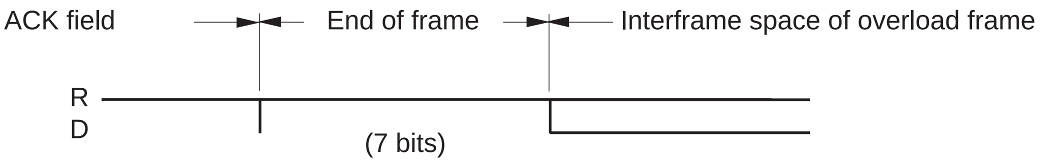
Figure 17-10: ACK Field



- Receive node sets the ACK slot to dominant level if no error was detected.

(7) End of frame: Indicates the end of the transmission/reception.

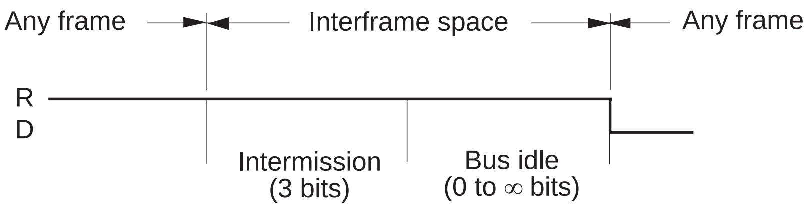
Figure 17-11: End of Frame



(8) Interframe space: This sequence is inserted after data frames, remote frames, error frames, and overload frames in the serial bitstream on the bus to indicate start or end of a frame. The length of the interframe space depends on the error state (active or passive) of the node.

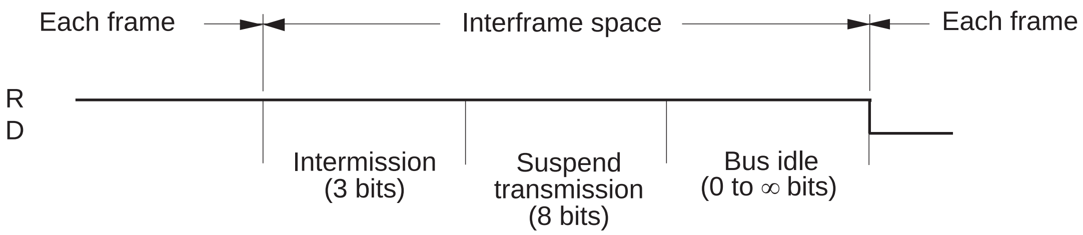
(a) Error active: Consists of 3 bits intermission and bus idle.

Figure 17-12: Interframe Space/Error Active



(b) Error passive: Consists of 3 bits intermission, suspend transmission and bus idle.

Figure 17-13: Interframe Space/Error Passive



Remark: The nominal value of the intermission field is 3 bits. However, transmission nodes may start immediately a transmission already in the 3rd bit of this field when a dominant level is detected.

Table 17-6: Operation in the Error State

Error State	Operation
Error active	Any node in this state is able to start a transmission whenever the bus is idle.
Error passive	Any node in this state has to wait for 11 consecutive recessive bits before initiating a transmission.

17.1.5 Error Frame

- This frame is sent from a node if an error is detected.
- The type of an Error Frame is defined by its error flag: ACTIVE ERROR FLAG or PASSIVE ERROR FLAG. Which kind of flag a node transmits after detecting an error condition depends on the internal count of the error counters of each node.

Figure 17-14: Error Frame

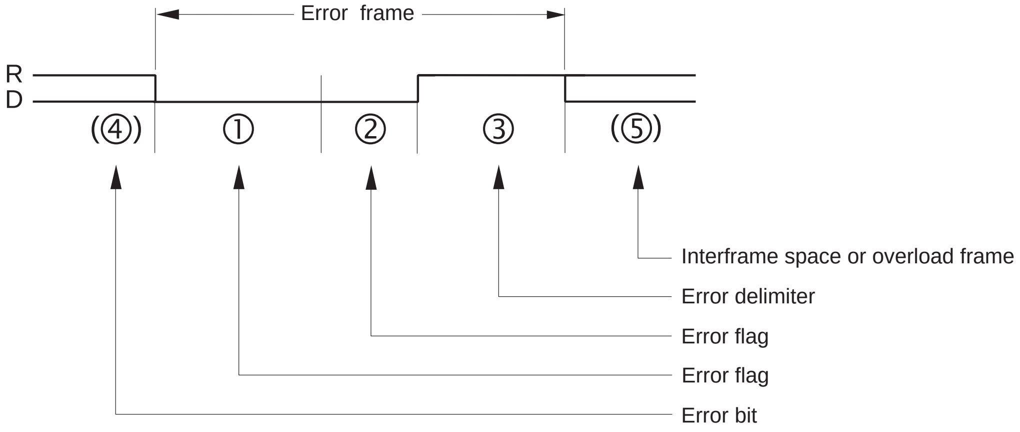


Table 17-7: Definition of each Field

No.	Name	Bit Number	Definition
1	Error flag	6	Error active node: sends 6 bits dominant level continuously. Error passive node: sends 6 bits recessive level continuously.
2	Error flag superpositioning	0 to 6	Nodes receiving an “error flag” detect bit stuff errors and issue error flags’ themselves.
3	Error delimiter	8	Sends 8 bits recessive level continuously. In case of monitoring dominant level at 8th bit, an overload frame is transmitted after the next bit.
4	Erroneous bit	-	An error frame is transmitted continuously after the bit where the error has occurred (in case of a CRC error, transmission continues after the ACK delimiter).
5	Interframe space/ overload frame	3/14 20 MAX	Interframe space or overload frame continues.

17.1.6 Overload Frame

- This frame is started at the first bit of the intermission when the reception node is busy with exploiting the receive operation and is not ready for further reception.
- When a bit error is detected in the intermission, also an overload frame is sent following the next bit after the bit error detection.
- Detecting a dominant bit during the 3rd bit of intermission will be interpreted as START OF FRAME.
- At most two OVERLOAD FRAMEs may be generated to delay the next DATA FRAME or REMOTE FRAME.

Figure 17-15: Overload Frame

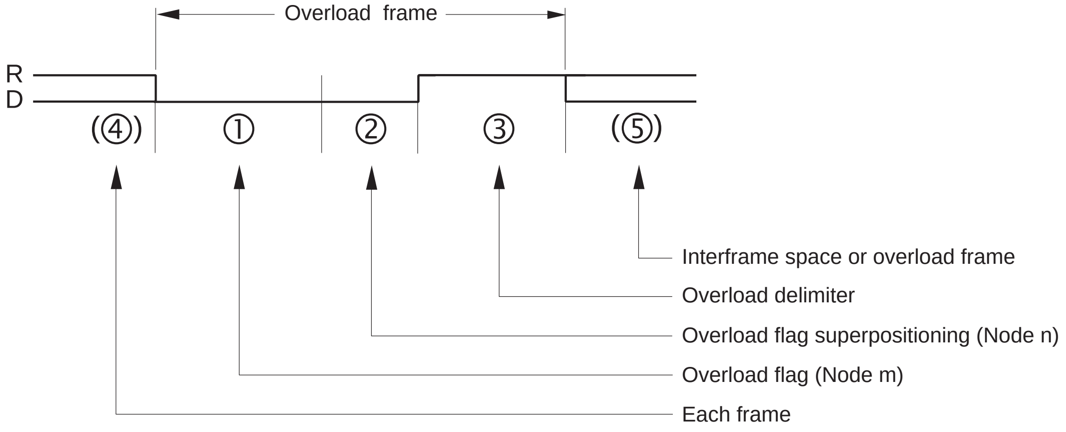


Table 17-8: Definition of each Frame

No.	Name	Bit Number	Definition
1	Overload flag	6	Sent 6 bits dominant level continuously.
2	Overload flag from any node	0 to 6	A node that receives an overload flag in the interframe space. Issues an overload flag.
3	Overload delimiter	8	Sends 8 bits recessive level continuously. In case of monitoring dominant level at 8th bit, an overload frame is transmitted after the next bit.
4	Any frame	-	Output following the end of frame, error delimiter and overload delimiter.
5	Interframe space/ overload frame	3/14 20 MAX	Interframe space or overload frame continues.

17.2 Function

17.2.1 Arbitration

If two or more nodes happen to start transmission in coincidence, the access conflict is solved by a bit-wise arbitration mechanism during transmission of the ARBITRATION FIELD.

(1) When a node starts transmission:

- During bus idle, the node having the output data can transmit.

(2) When more than one node starts transmission:

- The node with the lower identifier wins the arbitration.
- Any transmitting node compares its output arbitration field and the data level on the bus.
- It looses arbitration, when it sends recessive level and reads dominant from bus.

Table 17-9: Arbitration

Level Detection	Status of Arbitrating Node
Conformity of Level	Continuous Transmission
Non-conformity of Level	The data output is stopped from the next bit and reception operation starts.

(3) Priority of data frame and remote frame:

- When a data frame and remote frame with the same message identifier are on the bus, the data frame has priority because its RTR bit carries 'Dominant level'. The data frame wins the arbitration.

17.2.2 Bit Stuffing

When the same level continues for more than 5 bits, bit stuffing (insert 1 bit with inverse level) takes place.

- Due to this a resynchronization of the bit timing can be done at least every 10 bits.
- Nodes detecting an error condition send an error frame, violating the bit stuff rule and indicating this message to be erroneous for all nodes.

Table 17-10: Bit Stuffing

Transmission	During the transmission of a data frame and a remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, 1 bit level with reverse level of data is inserted before the following bit.
Reception	During the reception of a data frame and a remote frame, when the same level continues for 5 bits in the data between the start of frame and the ACK field, the reception is continued by deleting the next bit.

17.2.3 Multi Master

As the bus priority is determined by the identifier, any node can be the bus master.

17.2.4 Multi Cast

Any message can be received by any node (broadcast).

17.2.5 Sleep Mode/Stop Function

This is a function to put the CAN controller in waiting mode to achieve low power consumption. The SLEEP mode of the DCAN complies to the method described in ISO 11898.
Additional to this SLEEP mode, which can be woken up by bus activities, the STOP mode is fully controlled by the CPU device.

17.2.6 Error Control Function

(1) Error types

Table 17-11: Error Types

Type	Description of Error		Detection State	
	Detection Method	Detection Condition	Transmission/ Reception	Field/Frame
Bit error	Comparison of output level and level on the bus (except stuff bit)	Disagreement of both levels	Transmission/ reception node	Bit that output data on the bus at the start of frame to the end of frame, error frame and overload frame.
Stuff error	Check of the reception data at the stuff bit	6 consecutive bits of the same output level	Transmission/ reception node	Start of frame to CRC sequence
CRC error	Comparison of the CRC generated from the reception data and the received CRC sequence	Disagreement of CRC	Reception node	Start of frame to data field
Form error	Field/frame check of the fixed format	Detection of the fixed format error	Reception node	CRC delimiter ACK field End of frame Error frame Overload frame
ACK error	Check of the ACK slot by the transmission node	Detection of recessive level in ACK slot	Transmission node	ACK slot

(2) Output timing of the error frame

Table 17-12: Output Timing of the Error Frame

Type	Output timing
Bit error, stuff error, form error, ACK error	Error frame is started at the next bit timing following the detected error
Error passive	CRC error Error frame is started at the next bit timing following the ACK delimiter

(3) Measures when error occurs

- Transmission node re-transmits the data frame or the remote frame after the error frame.
- The CAN standard (ISO-11898) allows a programmable suppression of this re-transmission. It is called single shot mode.

(4) Error state

(a) Types of error state

- Three types of error state: These are error active, error passive and bus off.
- The transmission error counter (TEC) and the reception error counter (REC) control the error state.
- The error counters are incremented on each error occurrence (refer to Table 3-6).
- If the value of error counter exceeds 96, warning level for error passive state is reached.
- When only one node is active at start-up, it may not receive an acknowledgment on a transmitted message. This will increment TEC until error passive state is reached. The bus off state will not be reached because for this specific condition TEC will not increment any more if values greater than 127 are reached.
- A node in bus off state will not issue any dominant level on the CAN transmit pin. The reception of messages is not affected by the bus off state.

Table 17-13: Types of Error

Type	Operation	Value of Error Counter	Output Error Flag Type
Error active	Transmission/ reception	0 to 127	Active error flag (6 bits of dominant level continue)
Error passive	Transmission	128 to 255	Passive error flag (6 bits of recessive level continue)
	Reception	128 or more	
Bus off	Transmission	more than 255	Communication cannot be made
	Reception	-	Does not exist

(b) Error counter

- Error counter counts up when an error has occurred, and counts down upon successful transmission and reception. The error counters are updated during the first bit of an error flag.

Table 17-14: Error Counter

State	Transmission Error Counter (TEC)	Reception Error Counter (REC)
Reception node detects an error (except bit error in the active error flag or overload flag).	No change	+1
Reception node detects dominant level following the error flag of the own error frame.	No change	+8
Transmission node transmits an error flag. Exception: 1. ACK error is detected in the error passive state and dominant level is not detected in the passive error flag sent. 2. Stuff error generation in arbitration field.	+8	No change
Bit error detection during active error flag and overload flag when transmitting node is in error active state.	+8	No change
Bit error detection during active error flag and overload flag when receiving node is in error active state.	No change	+8
When the node detects fourteen continuous dominant bits counted from the beginning of the active error flag or the overload flag, and every time, eight subsequent dominant bits after that are detected. Every time when the node detects eight continuous dominant bits after the passive error flag.	+8	+8
When the transmitting node has completed to sent without error.	-1 (-0 when error counter = 0)	No change
When the reception node has completed to receive without error.	No change	-1 (1 ≤REC ≤127) –0 (REC = 0) 119-127 (REC > 127)

(c) Overload frame

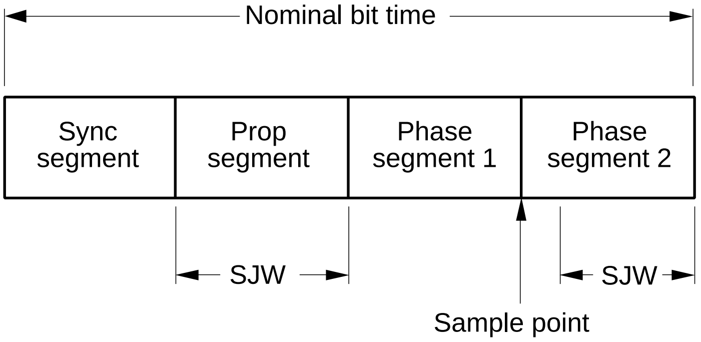
- In case the recessive level of first intermission bit is driven to dominant level, an overload frame occurs on the bus. Upon detection of an overload frame any transmit request will be postponed until the bus becomes idle.

17.2.7 Baud Rate Control Function

(1) Nominal bit time (8 to 25 time quanta)

- Definition of 1 data bit time is as follows.

Figure 17-16: Nominal Bit Time (8 to 25 Time Quanta)



[1 Minimum time for one time/quantum (TQ) = 1/fx]

- Sync segment: In this segment the bit synchronization is performed.
- Prop segment: This segment absorbs delays of the output buffer, the CAN bus and the input buffer. Prop segment time =(output buffer delay) + (CAN bus delay) + (input buffer delay).
- Phase segment 1/2: These segments compensate the data bit time error. The larger the size measured in TQ is, the larger is the tolerable error.
- The synchronization jump width (SJW) specifies the synchronization range. The SJW is programmable. SJW can have less or equal number of TQ as phase segment 2.

Table 17-15: Segment Name and Segment Length

Segment Name	Segment Length (allowed Number of TQs)
Sync segment (Synchronization segment)	1
Prop segment (Propagation segment)	Programmable 1 to 8
Phase segment 1 (Phase buffer segment 1)	Programmable 1 to 8
Phase segment 2 (Phase buffer segment 2)	Maximum of phase segment 1 and the IPT **Note**
SJW	Programmable 1 to 4

Note: IPT = Information Processing Time. It needs to be less than or equal to 2 TQ.

(2) Adjusting synchronization of the data bit

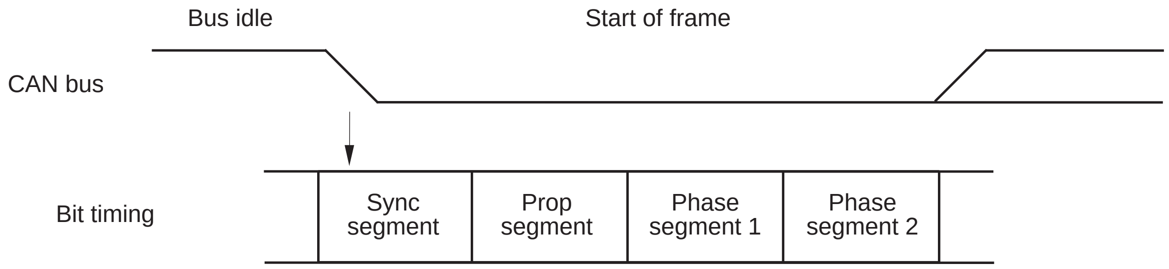
- The transmission node transmits data synchronized to the transmission node bit timing.
- The reception node adjusts synchronization at recessive to dominant edges on the bus. Depending on the protocol this synchronization can be a hard or soft synchronization.

(a) Hard synchronization

This type of synchronization is performed when the reception node detects a start of frame in the bus idle state.

- When the node detects a falling edge of a SOF, the current time quanta becomes the synchronization segment. The length of the following segments are defined by the values programmed into the SYNC0 and SYNC1 registers.

Figure 17-17: Adjusting Synchronization of the Data Bit

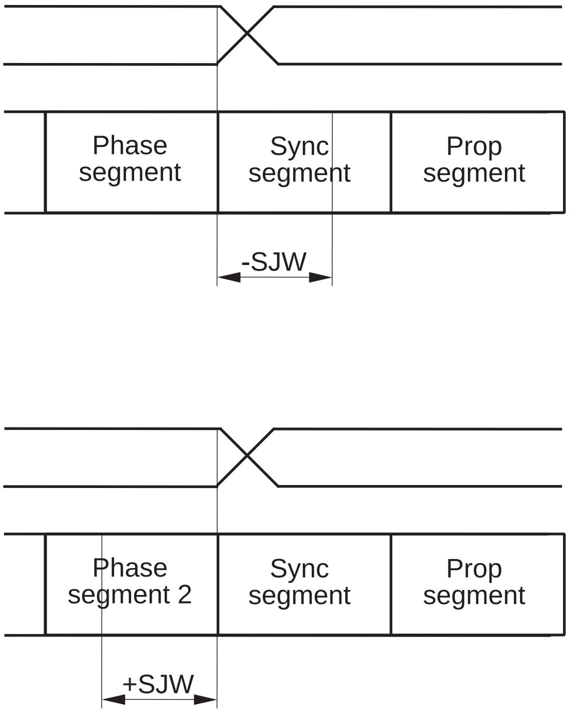


(b) Soft synchronization

When a recessive to dominant level change on the bus is detected, a soft synchronization is performed.

- If the phase error is larger than the programmed SJW value, the node will adjust the timing by applying this SJW-value. Full synchronization is achieved by subsequent adjustments on the next recessive to dominant edge(s).
- These errors that are equal or less of the programmed SJW are corrected instantly and full synchronization is achieved already for the next bit.
- The TQ at which the edge occurs becomes sync segment forcibly, if the phase error is less than or equal to SJW.

Figure 17-18: Bit Synchronization



17.2.8 State Shift Chart

Figure 17-19: Transmission State Shift Chart

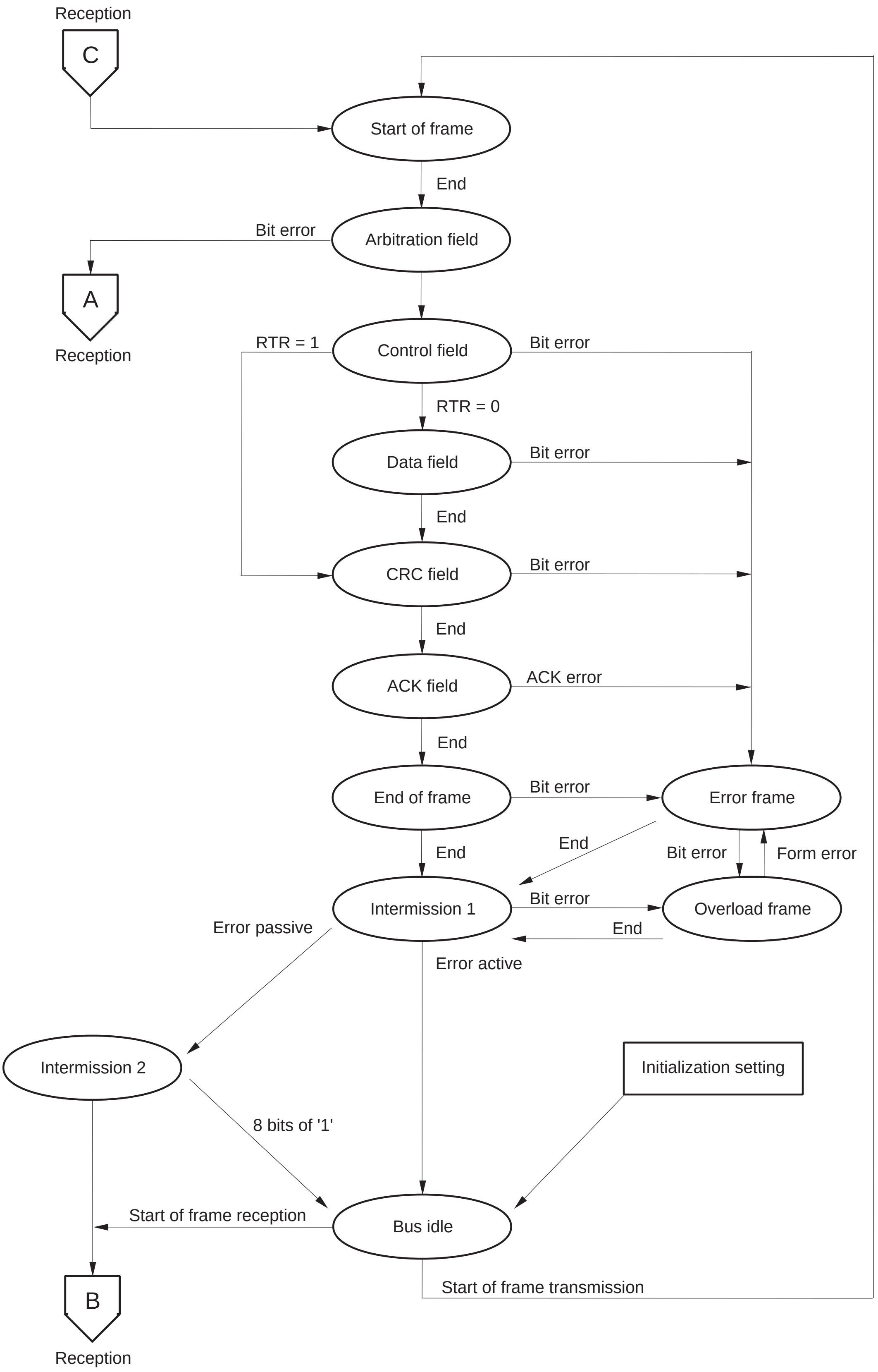


Figure 17-20: Reception State Shift Chart

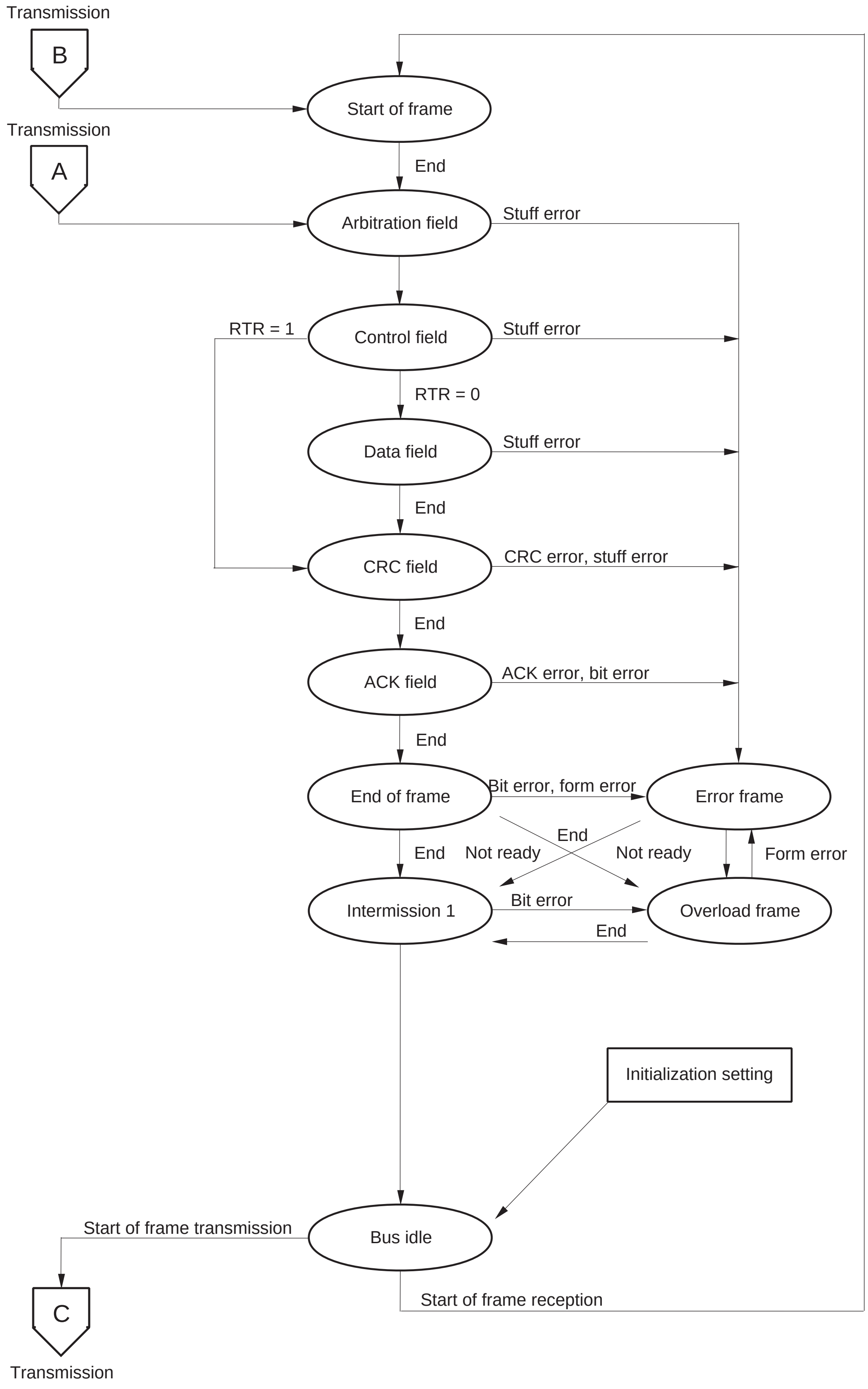
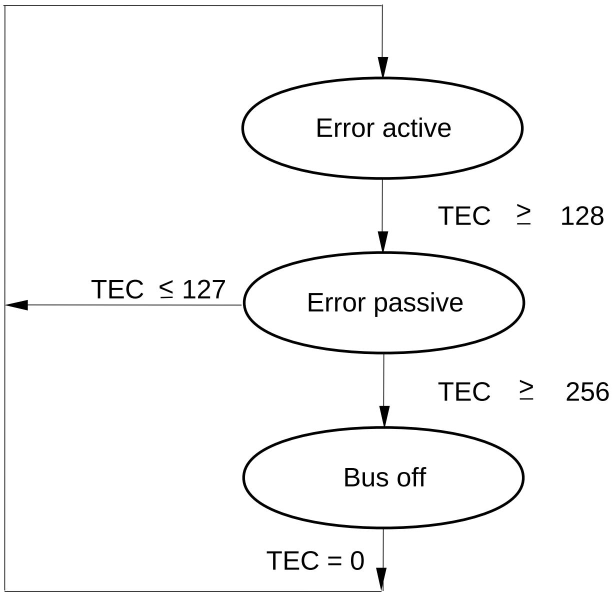


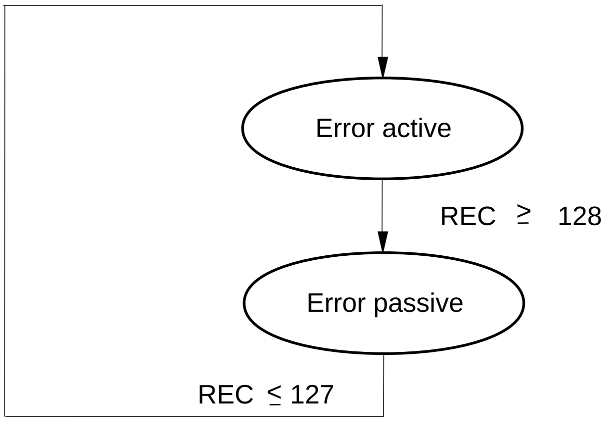
Figure 17-21: Error State Shift Chart

(a) Transmission



TEC = Transmission error counter

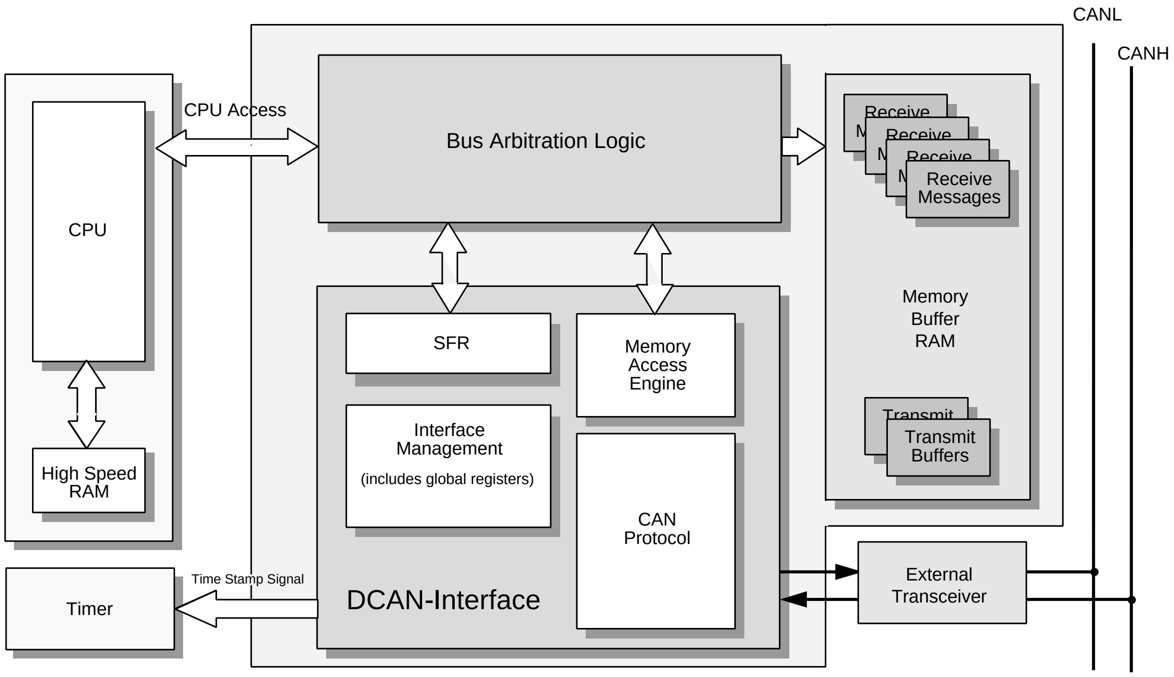
(b) Reception



REC = Reception error counter

17.3 Outline Description

Figure 17-22: Structural Block Diagram



This interface part handles all protocol activities by hardware in the CAN protocol part. The memory access engine fetches information for the CAN protocol transmission from the dedicated RAM area to the CAN protocol part or compares and sorts incoming information and stores it into predefined RAM areas.

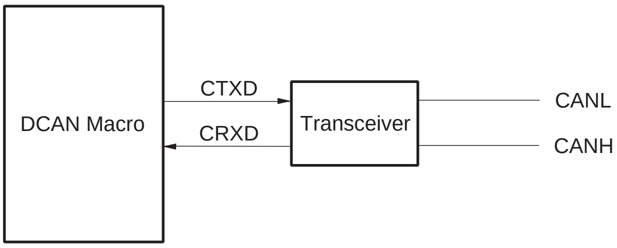
The DCAN interfaces directly to the RAM area that is accessible by the DCAN and by the CPU.

The DCAN part works with an external bus transceiver which converts the transmit data and receive data lines to the electrical characteristics of the CAN bus itself.

17.4 Connection with Target System

The DCAN Macro has to be connected to the CAN bus with an external transceiver.

Figure 17-23: Connection to the CAN Bus



17.5 CAN Controller Configuration

The CAN-module consists of the following hardware

.

Table 17-16: CAN Configuration

Item	Configuration
Message definition	In RAM area
CAN input/output	1 (CTXD) 1 (CRXD)
Control registers	CAN control register (CANC) Transmit control register (TCR) Receive message register (RMES) Redefinition control register (REDEF) CAN error status register (CANES) Transmit error counter (TEC) Receive error counter (REC) Message count register (MCNT) Bit rate prescaler (BRPRS) Synchronous control register 0 (SNYC0) Synchronous control register 1 (SYNC1) Mask control register (MASKC)

17.6 Special Function Register for CAN-module

Table 17-17: SFR Definitions

Register Name	Symbol	R/W	Bit Manipulation Units			After Reset
			1-bit	8-bit	16-bit	
CAN control register	CANC	R/W	×	×	-	01H
Transmit control register	TCR	R/W	-	×	-	00H
Receive message register	RMES	R	-	×	-	00H
Redefinition control register	REDEF	R/W	×	×	-	00H
CAN error status register	CANES	R/W	-	×	-	00H
Transmit error counter	TEC	R	-	×	-	00H
Receive error counter	REC	R	-	×	-	00H
Message count register	MCNT	R	-	×	-	C0H
Bit rate prescaler	BRPRS	R/W	-	×	-	00H
Synchronous control register 0	SYNC0	R/W	-	×	-	18H
Synchronous control register 1	SYNC1	R/W	-	×	-	0EH
Mask control register	MASKC	R/W	-	×	-	00H

The following SFR bits can be accessed with 1-bit instructions. The other SFR registers have to be accessed with 8-bit instructions.

Table 17-18: SFR Bit Definitions

Name	Description	Bit
SOFE	Start of frame enable	CANC.4
SLEEP	Sleep mode	CANC.2
INIT	Initialize	CANC.0
DEF	Redefinition enable	REDEF.7

17.7 Message and Buffer Configuration

Table 17-19: Message and Buffer Configuration

Address [Note 2]	Register Name	R/W	After Reset
00xH	Transmit buffer 0	R/W	Note 1
01xH	Transmit buffer 1		
02xH	Receive message 0 / Mask 0		
03xH	Receive message 1		
04xH	Receive message 2 / Mask 1		
05xH	Receive message 3		
06xH	Receive message 4		
07xH	Receive message 5		
08xH	Receive message 6		
09xH	Receive message 7		
0AxH	Receive message 8		
0BxH	Receive message 9		
0CxH	Receive message 10		
0DxH	Receive message 11		
0ExH	Receive message 12		
0FxH	Receive message 13		
10xH	Receive message 14		
11xH	Receive message 15		

Notes: 1. Contents is undefined, because data resides in normal RAM area.

2. This address is an offset to the RAM area starting address defined with CADD0/1 in the message count register (MCNT).

17.8 Transmit Buffer Structure

The DCAN has two independent transmit buffers. The two buffers have a 16-byte data structure for standard and extended frames with the ability to send up to 8 data bytes per message. The structure of the transmit buffer is similar to the structure of the receive buffers. The CPU can use addresses that are specified as “unused” in the transmit buffer layout. As well the CPU may use unused ID addresses, unused data addresses[Note], and an unused transmit buffer of the DCAN for its own purposes. The control bits, the identification and the message data have to be stored in the message RAM area.

The transmission control is done by the TCR register. A transmission priority selection allows the customer to realize an application specific priority selection. After the priority selection the transmission can be started by setting the TXRQn bit (n = 0, 1).

In the case that both transmit buffers are used, the transmit priorities can be set. For this purpose the DCAN has the TXP bit in the TCR register. The application software has to set this priority before the transmission is started.

The two transmit buffers supply two independent interrupt lines for an interrupt controller.

Note: Message objects that need less than 8 data byte (DLC < 8) may use the remaining bytes (8 - DLC) for application purposes.

17.9 Transmit Message Format

Table 17-20: Transmit Message Format

Name	Address[Note]	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TCON	n0H	IDE	RTR	0	0	DLC3	DLC2	DLC1	DLC0
	n1H	Unused							
IDTX0	n2H	ID standard part							
IDTX1	n3H	ID standard part			0	0	0	0	0
IDTX2	n4H	ID extended part							
IDTX3	n5H	ID extended part							
IDTX4	n6H	ID extended part		0	0	0	0	0	0
	n7H	Unused							
DATA0	n8H	Message data byte 0							
DATA1	n9H	Message data byte 1							
DATA2	nAH	Message data byte 2							
DATA3	nBH	Message data byte 3							
DATA4	nCH	Message data byte 4							
DATA5	nDH	Message data byte 5							
DATA6	nEH	Message data byte 6							
DATA7	nFH	Message data byte 7							

Note: This address is a relative offset to the starting address of the transmit buffer.

(1) Transmit Message Definition

The memory location labelled TCON includes the information of the RTR bit and the bits of the control field of a data or remote frame.

TCON is set with an 1-bit or an 8-bit memory manipulation instruction.
RESET input sets TCON to an undefined value.

Figure 17-24: Transmit Message Definition Bits

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TCON	IDE	RTR	0	0	DLC3	DLC2	DLC1	DLC0	xxx0H	undefined	R/W

IDE	Identifier Extension Select
0	Transmit standard frame message; 11 bit identifier
1	Transmit extended frame message; 29 bit identifier

RTR	Remote Transmission Select
0	Transmit data frames
1	Transmit remote frames

DLC3	DLC2	DLC1	DLC0	Data Length Code Selection of Transmit Message
0	0	0	0	0 data bytes
0	0	0	1	1 data bytes
0	0	1	0	2 data bytes
0	0	1	1	3 data bytes
0	1	0	0	4 data bytes
0	1	0	1	5 data bytes
0	1	1	0	6 data bytes
0	1	1	1	7 data bytes
1	0	0	0	8 data bytes
Others than above				**Note**

Remark: The control field describes the format of frame that is generated and its length. The reserved bits of the CAN protocol are always sent in dominant state (0).

Note: The data length code selects the number of bytes which have to be transmitted. Valid entries for the data length code (DLC) are 0 to 8. If a value greater than 8 is selected, 8 bytes are transmitted in the data frame. The Data Length Code is specified in DLC3 through DLC0.

(2) Transmit Identifier Definition

These memory locations set the message identifier in the arbitration field of the CAN protocol.

IDTX0 to IDTX4 register can be set with an 1-bit or an 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets IDTX0 to IDTX4 to an undefined value.

Figure 17-25: 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
IDTX1	ID20	ID19	ID18	0	0	0	0	0	xxx3H	undefined	R/W
IDTX2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	undefined	R/W
IDTX3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	undefined	R/W
IDTX4	ID1	ID0	0	0	0	0	0	0	xxx6H	undefined	R/W

Remark: If a standard frame is defined by the IDE bit in the TCON byte then IDTX0 and IDTX1 are used only. IDTX2 to IDTX4 are free for use by the CPU for application needs.

(3) Transmit Data Definition

These memory locations set the transmit message data of the data field in the CAN frame.

DATA0 to DATA7 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to an undefined value.

Figure 17-26: Transmit Data

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DATA0									xxx8H	undefined	R/W
DATA1									xxx9H	undefined	R/W
DATA2									xxxAH	undefined	R/W
DATA3									xxxBH	undefined	R/W
DATA4									xxxCH	undefined	R/W
DATA5									xxxDH	undefined	R/W
DATA6									xxxEH	undefined	R/W
DATA7									xxxFH	undefined	R/W

Remark: Unused data bytes that are not used by the definition in the DLC bits in the TCON byte are free for use by the CPU for application needs.

17.10 Receive Buffer Structure

The DCAN has up to 16 receive buffers. The number of used buffers is defined by the MCNT register. Unused receive buffers can be used as application RAM for the CPU. The received data is stored directly in this RAM area.

The 16 buffers have a 16-byte data structure for standard and extended frames with a capacity of up to 8 data bytes per message. The structure of the receive buffer is similar to the structure of the transmit buffers. The semaphore bits DN and MUC enable a secure reception detection and data handling. For the first 8 receive message buffers the successful reception is mirrored by the DN-flags in the RMES register.

The receive interrupt request can be enabled or disabled for each used buffer separately.

17.11 Receive Message Format

Table 17-21: Receive Message Format

Name	Address$^{\text{Note 1}}$	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IDCON	n0H	0	0	0	0	0	ENI	RTR	IDE
DSTAT	n1H	DN	MUC	R1	R0	DLC			
IDREC0	n2H	ID standard part							
IDREC1	n3H	ID standard part			0	0	0	0	RTR$_{\text{REC}}$$^{\text{Note 2}}$
IDREC2	n4H	ID extended part							
IDREC3	n5H	ID extended part							
IDREC4	n6H	ID extended part		0	0	0	0	0	0
	n7H	unused							
DATA0	n8H	Message data byte 0							
DATA1	n9H	Message data byte 1							
DATA2	nAH	Message data byte 2							
DATA3	nBH	Message data byte 3							
DATA4	nCH	Message data byte 4							
DATA5	nDH	Message data byte 5							
DATA6	nEH	Message data byte 6							
DATA7	nFH	Message data byte 7							

Notes: 1. This address is a relative offset to the start address of the receive buffer.

2. RTR$_{\text{REC}}$ is the received value of the RTR message bit when this buffer is used together with a mask function.
By using the mask function a successfully received identifier overwrites the bytes IDREC0 and IDREC1 for standard frame format and IDREC0 to IDREC4 for extended frame format.

For the RTR$_{\text{REC}}$ bit exist two modes:

- RTR bit in the MCON byte of the dedicated mask is set to 0. In this case RTR$_{\text{REC}}$ will always be written to 0 together with the update of the IDn bits in IDREC1. The received frame type (data or remote) is defined by the RTR bit in IDCON of the buffer.
- RTR bit in the MCON byte of the dedicated mask is set to 1 (data and remote frames are accepted). In this case the RTR bit in IDCON has no meaning. The received message type passed the mask is shown in RTR$_{\text{REC}}$.

If a buffer is not assigned to a mask function (mask 1, mask 2 or global mask) the bytes IDREC0 to IDREC4 are only read for comparing. During initialization the RTR$_{\text{REC}}$ should be defined to 0.

(1) Receive control bits definition

The memory location labelled IDCON defines the kind of frame (data or remote frame with standard or extended format) that is monitored for the associated buffer. Notification by the receive interrupt upon successful reception can be selected for each receive buffer separately.

IDCON can be set with an 1-bit or an 8-bit memory manipulation instruction.
RESET input sets IDCON to an undefined value.

Figure 17-27: Control bits for Receive Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IDCON	0	0	0	0	0	ENI	RTR	IDE	xxx0H	undefined	R/W

IDE	Identifier Extension Select
0	Receive standard frame message; 11-bit identifier
1	Receive extended frame message; 29-bit identifier

RTR	Remote Transmission Select
0	Receive data frames
1	Receive remote frames

ENI	Enable Interrupt on Receive[Note]
0	No interrupt generated
1	Generate receive interrupt after reception of valid message

The control bits define the type of message that is transferred in the associated buffer if this type of message appears on the bus.
This byte will never be written by the DCAN. Only the host CPU can change this byte.

Note: The user has to define with the ENI bit if he wants to set a receive interrupt request when new data is received in this buffer.

(2) Receive status bits definition

The memory location labelled DSTAT sets the receive status bits of the arbitration field of the CAN protocol.

DSTAT can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DSTAT to an undefined value.

Figure 17-28: Receive Status Bits (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DSTAT	DN	MUC	R1	R0	DLC3	DLC2	DLC1	DLC0	xxx1H	undefined	R/W

The receive status reflects the current status of a message. It signals whether new data is stored or if the DCAN currently transfers data into this buffer.
In addition the data length of the last transferred data and the reserved bits of the protocol are shown.

DN	Data New
0	No change in data
1	Data changed

The DCAN-module sets DN twice. At first when it starts storing a message from the shadow buffer into the receive buffer and secondly when it finished the operation.

The CPU needs to clear this bit, to signal by itself that it has read the data. During initialization of the receive buffers the DN-bit should also be cleared. Otherwise the CPU gets no information on an update of the buffer after a successful reception.

MUC	Memory Update
0	CAN does not access data part
1	CAN is transferring new data to message buffer

The DCAN-module sets MUC when it starts transferring a message into the buffer and clears the MUC bit when the transfer is finished.

R1	Reserved Bit 1
0	Reserved bit 1 of received message was “0”
1	Reserved bit 1 of received message was “1”

R0	Reserved Bit 0
0	Reserved bit 0 of received message was “0”
1	Reserved bit 0 of received message was “1”

Figure 17-28: Receive Status Bits (2/2)

DLC3	DLC2	DLC1	DLC0	Data Length Code Selection of Receive Message
0	0	0	0	0 data bytes
0	0	0	1	1 data bytes
0	0	1	0	2 data bytes
0	0	1	1	3 data bytes
0	1	0	0	4 data bytes
0	1	0	1	5 data bytes
0	1	1	0	6 data bytes
0	1	1	1	7 data bytes
1	0	0	0	8 data bytes
Others than above				**Note**

DSTAT is written by the DCAN two times during message storage:
At the first access to this buffer DN = 1, MUC = 1, reserved bits and DLC are written.
At the last access to this buffer DN = 1, MUC = 0, reserved bits and DLC are written.

Note: Valid entries for the data length code are 0 to 8. If a value higher than 8 is received, 8 bytes are stored in the message buffer frame together with the data length code received in the DLC of the message.

(3) Receive Identifier Definition

These memory locations define the receive identifier of the arbitration field of the CAN protocol.

IDREC0 to IDREC4 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IDREC0 to IDREC4 to an undefined value.

Figure 17-29: 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
IDREC1	ID20	ID19	ID18	0	0	0	0	RTR$_{REC}$	xxx3H	undefined	R/W
IDREC2	ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	xxx4H	undefined	R/W
IDREC3	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	xxx5H	undefined	R/W
IDREC4	ID1	ID0	0	0	0	0	0	0	xxx6H	undefined	R/W

The identifier of the receive message has to be defined during the initialization of the DCAN.
The DCAN uses this data for the comparison with the identifiers received on the CAN bus. For normal message buffers without mask function this data is only read by the DCAN for comparison. In combination with a mask function this data is overwritten by the received ID that has passed the mask.

The identifier of the receive messages should not be changed without being in the initialization phase or setting the receive buffer to redefinition in the RDEF register, because the change of the contents can happen at the same time when the DCAN uses the data for comparison. This can cause inconsistent data stored in this buffer and also the ID-part can be falsified in case of using mask function.

Remarks: 1. The unused parts of the identifier (IDREC1 bit 4 - 0 always and IDREC4 bit 5 - 0 in case of extended frame reception) may be written by the DCAN to “0”. They are not released for other use by the CPU.

2. RTR$_{REC}$ is the received value of the RTR message bit when this buffer is used together with a mask function.
By using the mask function a successfully received identifier overwrites the IDREC0 and IDREC1 registers for standard frame format and the IDREC0 to IDREC4 registers for extended frame format.
For the RTR$_{REC}$ bit exists two modes:

- RTR bit in the MCON register of the dedicated mask is set to “0”. In this case RTR$_{REC}$ bit will always be written to “0” together with the update of the IDn bits (n = 18 to 20) in IDREC1. The received frame type (data or remote) is defined by the RTR bit in IDCON of the buffer.
- RTR bit in the MCON register of the dedicated mask is set to “1” (data and remote frames are accepted). In this case the RTR bit in IDCON register has no meaning. The received message type passed the mask is shown in RTR$_{REC}$ bit.

If a buffer is not dedicated to a mask function (mask 1, mask 2 or global mask) the IDREC0 to IDREC4 registers are only read for comparing. All receive identifiers should be defined to “0” before the application sets up its specific values.

(4) Receive Message Data Part

These memory locations set the receive message data part of the CAN protocol.

DATA0 to DATA7 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DATA0 to DATA7 to an undefined value.

Figure 17-30: Receive Data

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
DATA0									xxx8H	undefined	R/W
DATA1									xxx9H	undefined	R/W
DATA2									xxxAH	undefined	R/W
DATA3									xxxBH	undefined	R/W
DATA4									xxxCH	undefined	R/W
DATA5									xxxDH	undefined	R/W
DATA6									xxxEH	undefined	R/W
DATA7									xxxFH	undefined	R/W

The DCAN stores received data bytes in this memory area. Only those data bytes which are actually received and match with the identifier are stored in the receive buffer memory area.

If the DLC is less than eight, the DCAN will not write additional bytes exceeding the DLC value up to eight. The DCAN stores a maximum of 8 bytes (according to the CAN protocol rules) even when the received DLC is greater than eight.

17.12 Mask Function

Table 17-22: Mask Function

Name	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MCON	n0H							RTR	
	n1H	Unused							
MREC0	n2H	ID standard part							
MREC1	n3H	ID standard part			0	0	0	0	0
MREC2	n4H	ID extended part							
MREC3	n5H	ID extended part							
MREC4	n6H	ID extended part		0	0	0	0	0	0
	n7H	Unused							
	n8H	Unused							
	n9H	Unused							
	nAH	Unused							
	nBH	Unused							
	nCH	Unused							
	nDH	Unused							
	nEH	Unused							
	nFH	Unused							

Receive message buffer 0 and buffer 2 can be switched for masked operation with the mask control register (MASKC). In this case the message does not hold message identifier and data of the frame. Instead, it holds identifier and RTR mask information for masked compare operations for the next higher message buffer number. In case the global mask is selected, it keeps mask information for all higher message buffer numbers.

A mask does not store any information about identifier length. Therefore the same mask can be used for both types of frames (standard and extended) during global mask operation.

All unused bytes can be used by the CPU for application needs.

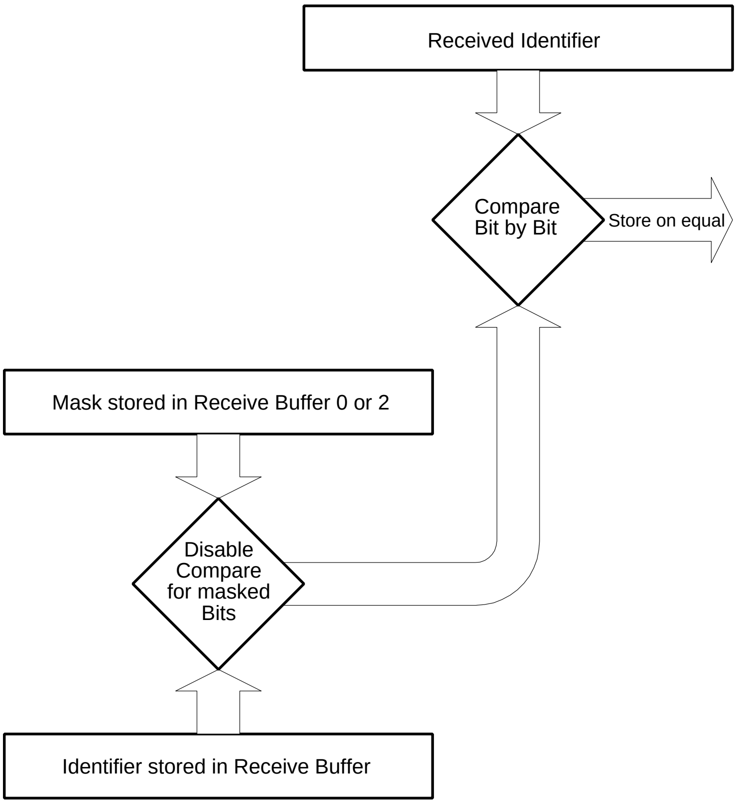
(1) Identifier Compare with Mask

The identifier compare with mask provides the possibility to exclude some bits from the comparison process. That means each bit is ignored when the corresponding bit in the mask definition is set to one.

The setup of the mask control register (MASKC) defines which receive buffer is used as a mask and which receive buffer uses which mask for comparison.

The mask does not include any information about the identifier type to be masked. This has to be defined within the dedicated receive buffer. Therefore a global mask can serve for standard receive buffers at the same time as for extended receive buffer.

Figure 17-31: Identifier Compare with Mask


Received Identifier
Compare Bit by Bit
Store on equal
Mask stored in Receive Buffer 0 or 2
Disable Compare for masked Bits
Identifier stored in Receive Buffer

This function implements the so called basic-CAN behaviour.

In this case the type of identifier is fixed to standard or extended by the setup of the IDE bit in the receive buffer. The comparison of the RTR bit can also be masked. It is possible to receive data and remote frames on the same masked receive buffer.

The following information is stored in the receive buffer:

- Identifier (11 or 29 bit as defined by IDE bit)
- Remote bit (RTR_{REC}) if both frames types (data or remote) can be received by this buffer
- Reserved bits
- Data length code (DLC)
- Data bytes as defined by DLC

Caution: All writes into the DCAN memory are byte accesses. Unused bits in the same byte will be written zero. Unused bytes will not be written and are free for application use by the CPU.

(2) Mask identifier control register (MCON)

The memory location labelled MCON sets the mask identifier control bit of the CAN protocol.

MCON can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MCON to an undefined value.

Figure 17-32: Control Bits for Mask Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MCON	0	0	0	0	0	0	RTR	0	xxx0H	undefined	R/W

RTR	Remote Transmission Select
0	Check RTR bit of received message Note 1
1	Receive message independent from RTR bit Note 2

Notes: 1. For RTR = 0 the received frame type (data or remote) is defined by the RTR bit in IDCON of the dedicated buffer. In this case RTR_{REC} will always be written to “0” together with the update of the IDn bits (n = 18 to 20) in IDREC1.

2. In case RTR in MCON is set to “1”, RTR bit in IDCON of the dedicated receive buffer has no meaning. The received message type passed the mask is shown in the RTR_{REC} bit.

(3) Mask Identifier Definition

These memory locations set the mask identifier definition of the DCAN.

MREC0 to MREC4 can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MREC0 to MREC4 to an undefined value.

Figure 17-33: Mask Identifier

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MREC0	MID28	MID27	MID26	MID25	MID24	MID23	MID22	MID21	xxx2H	undefined	R/W
MREC1	MID20	MID19	MID18	0	0	0	0	0	xxx3H	undefined	R/W
MREC2	MID17	MID16	MID15	MID14	MID13	MID12	MID11	MID10	xxx4H	undefined	R/W
MREC3	MID9	MID8	MID7	MID6	MID5	MID4	MID3	MID2	xxx5H	undefined	R/W
MREC4	MID1	MID0	0	0	0	0	0	0	xxx6H	undefined	R/W

MIDn	Mask Identifier Bit (n = 0...28)
0	Check IDn bit in IDREC0 through IDREC4 of received message
1	Receive message independent from IDn bit

17.13 Operation of the CAN Controller

17.13.1 CAN control register (CANC)

The operational modes are controlled via the CAN control register CANC.
CANC can be set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CANC to 01H.

Figure 17-34: CAN Control Register (1/2)

Symbol	7	6	5	<4>	3	<2>	1	<0>	Address	After Reset
CANC	RXF	TXF	0	SOFE	SOFSEL	SLEEP	STOP	INIT	FFB0H	01H
	R	R	R	R/W	R/W	R/W	R/W	R/W		

CANC.5 has always to be written as 0.

INIT	Request status for operational modes
0	Normal operation
1	Initialization mode

The INIT is the request bit to control the DCAN. INIT starts and stops the CAN protocol activities. Due to bus activities disabling the DCAN is not allowed any time. Therefore changing the INIT bit must not have an immediate effect to the CAN protocol activities. Setting the INIT bit is a request only.
The INITSTAT bit in the CANES register reflects if the request has been granted. The registers MCNT, SYNC0, SYNC1, and MASKC are write protected while INIT is cleared independently of INITSTAT.
Any write to these registers when INIT is set and the initialisation mode is not confirmed by the INITSTAT bit can have unexpected behaviour to the CAN bus.

STOP	Stop Mode Selection
0	Normal sleep operation / Sleep mode is released when a transition on the CAN bus is detected
1	Stop operation / Sleep mode is cancelled only by CPU access. No wake up from CAN bus

SLEEP	Sleep/Stop Request for CAN protocol
0	Normal operation
1	CAN protocol goes to sleep or stop mode depending on STOP bit

Figure 17-34: CAN Control Register (2/2)

The clock supply to the DCAN is switched off during initialization, DCAN Sleep, and DCAN Stop mode. All modes are only accepted while CAN protocol is in idle state, whereby the CRXD pin must be recessive (= high level). A sleep or stop request out of idle state is rejected and the WAKE bit in CANES is set. DCAN Sleep and DCAN Stop mode can be requested in the same manner. The only difference is that the DCAN Stop mode prevents the wake up by CAN bus activity.

Caution: The DCAN Sleep or DCAN Stop mode can not be requested as long as the WAKE bit in CANES is set.

The DCAN Sleep mode is cancelled under following conditions:

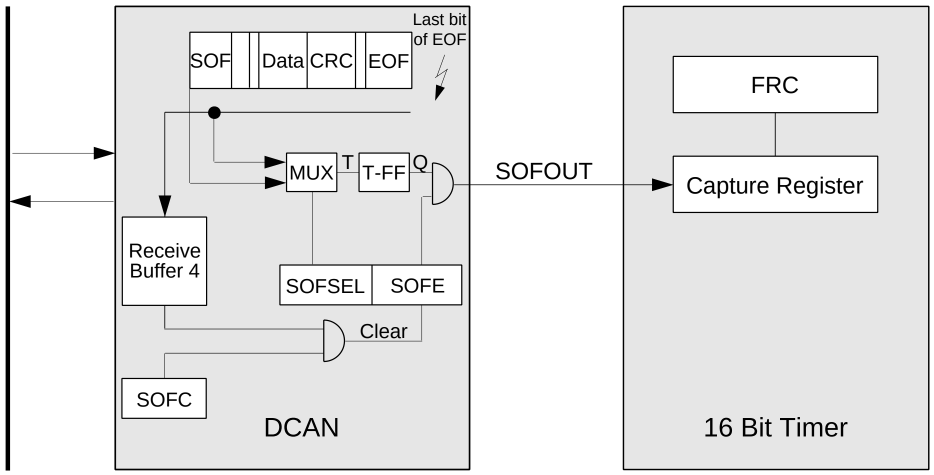
a) CPU clears the SLEEP bit.
b) Any transition while idle state on CAN bus (STOP = 0).
c) CPU sets SLEEP, but CAN protocol is active due to bus activity.

The WAKE bit in CANES is set under condition b) and c).

SOFSEL	Start of Frame Output Function Select
0	Last bit of EOF is used to generate the time stamp
1	SOF is used to generate the time stamp

SOFE	Start of Frame Enable
0	SOFOUT does not change
1	SOFOUT toggles depending on the selected mode

Figure 17-35: DCAN Support



The generation of an SOFOUT signal can be used for time measurements and for global time base synchronization of different CAN nodes as a prerequisite for time triggered communication.

Table 17-23: Possible Setup of the SOFOUT Function

SOFSEL	SOFC	SOFE	SOFOUT Function
x	x	0	Time stamp function disabled
0	x	1	Toggles with each EOF
1	0	1	Toggles with each start of frame on the CAN Bus
1	1	1	Toggles with each start of frame on the CAN bus. Clears SOFE bit when DCAN starts to store a message in receive buffer 4

SOFC is located in the synchronization register SYNC1.

$\overline{\text{RESET}}$ and setting of the INIT bit of CANC register clears the SOFOUT to 0.

Table 17-24: Transmission / Reception Flag

TXF	Transmission Flag
0	No transmission
1	Transmission active on CAN bus Note

RXF	Reception Flag
0	No data on the CAN bus
1	Reception active on the CAN bus

The TXF and RXF bits of CANC register show the present status of the DCAN to the bus.
If both bits are cleared, the bus is in idle state.
RXF and TXF bits are read-only bits. During initialization mode both bits do not reflect the bus status.

Note: Transmission is active until intermission is completed.

Figure 17-36: Time Stamp Function

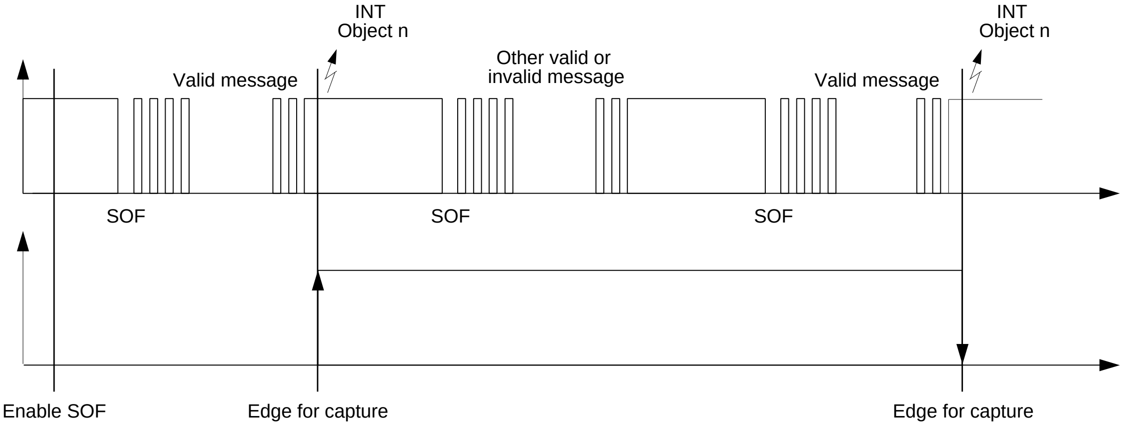


Figure 17-37: SOFOUT Toggle Function

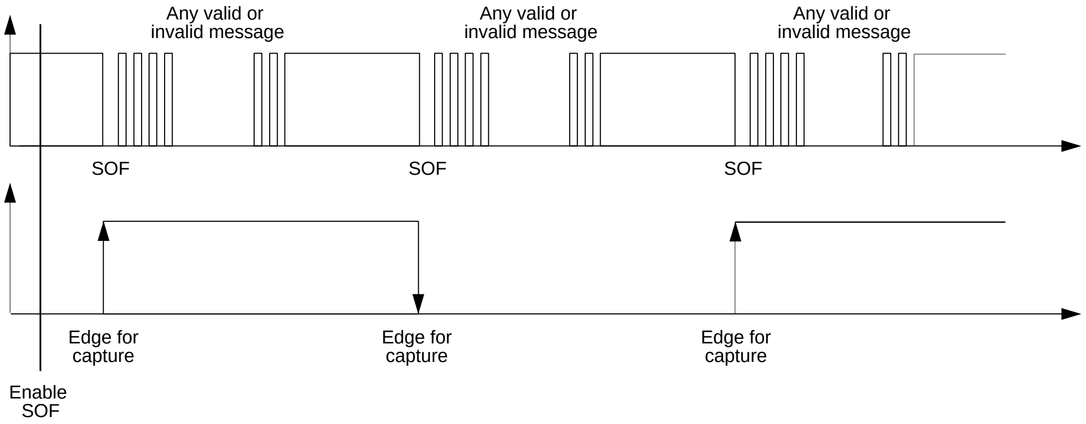
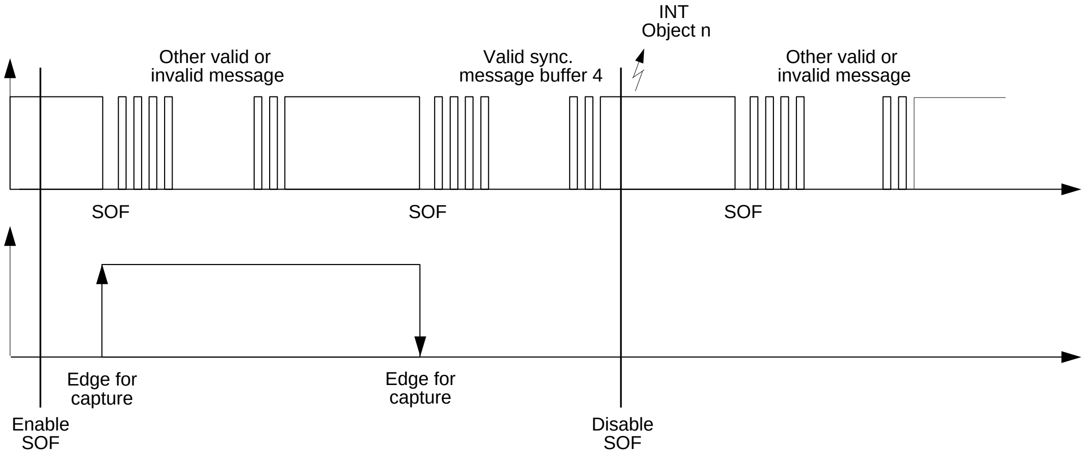


Figure 17-38: Global Time System Function



17.13.2 DCAN error status register (CANES)

This register shows the status of the DCAN.

CANES has to be set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets CANES to 00H.
The RESET sets the INIT-bit in CANC register, therefore CANES will be read as 08H after RESET release.

Figure 17-39: CAN Error Status Register (1/3)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
CANES	BOFF	RECS	TECS	0	INITSTATE	VALID	WAKE	OVER	FFB4H	00H
	R	R	R	R	R	R/W	R/W	R/W		

Remark: BOFF, RECS, TECS and INITSTATE are read only bits.

Caution: Don’t use bit operations on this SFR.
The VALID, WAKE and OVER bits have a special behavior during CPU write operations:

- **Writing a “0” to them do not change them.**
- **Writing an “1” clears the associated bit.**

This avoids any timing conflicts between CPU access and internal activities. An internal set condition of a bit overrides a CPU clear request at the same time.

BOFF	Bus Off Flag
0	Transmission error counter ≤ 255
1	Transmission error counter > 255

BOFF is cleared after receiving 128 x 11 bits recessive state (Bus idle) or by issuing a hard DCAN reset with the TLRES bit in the MCNTn register Note.
An interrupt is generated when the BOFF bit changes its value.

RECS	Reception error counter status
0	Reception error counter < 96
1	Reception error counter ≥ 96 / Warning level for error passive reached

RECS is updated after each reception.
An interrupt is generated when RECS changes its value.

Note: Issuing TLRES bit may violate the minimum recovery time as defined in ISO-11898.

Figure 17-39: CAN Error Status Register (2/3)

TECS	Transmission error counter status
0	Transmission error counter < 96
1	Transmission error counter ≥ 96 / Warning level for error passive reached

TECS is updated after each reception.
An interrupt is generated when TECS changes its value.

INITSTATE	Operational status of the DCAN
0	CAN is in normal operation
1	CAN is stopped and ready to accept new configuration data

INITSTATE changes with a delay to the INIT bit in CANC register. The delay depends on the current bus activity and the time to set all internal activities to inactive state. This time can be several bit times long. While BOFF bit is set, a request to go into the initialization mode by setting the INIT bit is ignored. In this case the INITSTATE bit will not be set until the Bus-off state is left.

VALID	Valid protocol activity detected
0	No valid message detected by the CAN protocol
1	Error free message reception from CAN bus

This bit shows valid protocol activities independent from the message definitions and the RXONLY bit setting in SYNC1n register. VALID is updated after each reception. The VALID bit will be set at the end of the frame when a complete protocol without errors has been detected.

Cautions: 1. The VALID bit is cleared if CPU writes an “1” to it, or when the INIT bit in CANC register is set.

2. Writing a “0” to the valid bit has no influence.

Figure 17-39: CAN Error Status Register (3/3)

WAKE	Wake up Condition
0	Normal operation
1	Sleep mode has been cancelled or sleep/stop mode request was not granted

This bit is set and an error interrupt is generated under the following circumstances:

a) A CAN bus activity occurs during DCAN Sleep mode.

b) Any attempt to set the SLEEP bit in the CAN control register during receive or transmit operation will immediately set the WAKE bit.

The CPU must clear this bit after recognition in order to receive further error interrupts, because the error interrupt line is kept active as long as this bit is set.

Cautions: 1. The WAKE bit is cleared to “0” if CPU writes an “1” to it, or when the INIT bit in CANC register is set.

2. Writing a “0” to the WAKE bit has no influence.

OVER	Overrun Condition
0	Normal operation
1	Overrun occurred during access to RAM

The overrun condition is set whenever the CAN can not perform all RAM accesses that are necessary for comparing and storing received data or fetching transmitted data. Typically, the overrun condition is encountered when the frequency for the macro is too low compared to the programmed baud rate. An error interrupt is generated at the same time.
The DCAN interface will work properly (i. e. no overrun condition will occur) with the following settings: The DCAN clock as defined with the PRM bits in the BRPRS register is set to a minimum of 16 times of the CAN baudrate **and** the selected CPU clock (defined in the PCC register) is set to a minimum of 16 times of the baudrate.

Possible reasons for an overrun condition are:

- Too many messages are defined.
- DMA access to RAM area is too slow compared to the CAN Baudrate.

The possible reactions of the DCAN differ depending on the situation, when the overrun occurs.

Table 17-25: Possible Reactions of the DCAN

Overrun Situation	When detected	DCAN Behavior
Cannot get transmit data.	Next data byte request from protocol. Immediate during the frame.	The frame itself conforms to the CAN specification, but its content is faulty. Corrupted data or ID in the frame. TXRQx bit (x = 0, 1) is not cleared. DCAN will retransmit the correct frame after synchronization to the bus.
Cannot store receive data.	Data storage is ongoing during the six bit of the next frame.	Data in RAM is inconsistent. No receive flags. DN and MUC bit may be set in message.
Cannot get data for ID comparison	ID compare is ongoing during six bits of next frame.	Message is not received and its data is lost.

17.13.3 CAN transmit error counter (TEC)

This register shows the transmit error counter.
TEC register can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets TEC to 00H.

Figure 17-40: Transmit Error Counter

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
TEC	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	FFB5H	00H
	R	R	R	R	R	R	R	R		

The transmit error counter reflects the status of the error counter for transmission errors as it is defined in the CAN protocol according ISO 11898.

17.13.4 CAN receive error counter (REC)

This register shows the receive error counter.
REC can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets REC to 00H.

Figure 17-41: Receive Error Counter

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
REC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	FFB6H	00H
	R	R	R	R	R	R	R	R		

The receive error counter reflects the status of the error counter for reception errors as it is defined in the CAN protocol according ISO 11898.

17.13.5 Message count register (MCNT)

This register sets the number of receive message buffers and allocates the RAM area of the receive message buffers, which are handled by the DCAN-module.
MCNT can be read with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MCNT to C0H.

Figure 17-42: Message Count Register (MCNT) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
MCNT	CADD1	CADD0	TLRES	MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	FFB7H	C0H
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		

This register is readable at any time.
Write is only permitted when the CAN is in initialization mode.

MCNT4	MCNT3	MCNT2	MCNT1	MCNT0	Receive Message Count
0	0	0	0	0	Setting prohibited
0	0	0	0	1	1 receive buffer
0	0	0	1	0	2 receive buffer
0	0	0	1	1	3 receive buffer
0	0	1	0	0	4 receive buffer
0	0	1	0	1	5 receive buffer
0	0	1	1	0	6 receive buffer
0	0	1	1	1	7 receive buffer
0	1	0	0	0	8 receive buffer
0	1	0	0	1	9 receive buffer
0	1	0	1	0	10 receive buffer
0	1	0	1	1	11 receive buffer
0	1	1	0	0	12 receive buffer
0	1	1	0	1	13 receive buffer
0	1	1	1	0	14 receive buffer
0	1	1	1	1	15 receive buffer
1	0	0	0	0	16 receive buffer
1	x	x	x	x	Setting prohibited, will be automatically changed to 16

Figure 17-42: Message Count Register (MCNT) (2/2)

TLRES	Reset function for CAN Protocol Machine
0	No Reset is issued
1	Reset of CAN protocol machine is issued if DCAN is in bus off state, DCAN will enter INIT state (CANC.0 = 1 && CANES.3 = 1)

Cautions: 1. Issuing TLRES bit may violate the minimum recovery time as defined in ISO-11898.

2. If no receive buffer is desired, define one receive buffer and disable this buffer with the REDEF function.

CADD1	CADD0	DCAN Address definition
0	0	Setting prohibited
0	1	
1	0	
1	1	F600H to F7DFH (reset value)

17.14 Baudrate Generation

(1) Bit rate prescaler register (BRPRS)

This register sets the clock for the DCAN (internal DCAN clock) and the number of clocks per time quantum (TQ).

BRPRS can be set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets BRPRS to 3FH.

Figure 17-43: Bit Rate Prescaler (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
BRPRS	PRM1	PRM0	BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	FFB8H	3FH
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		

The PRMn (n = 0, 1) bits define the clock source for the DCAN operation.
The PRM selector defines the input clock to the DCAN Macro and influences therefore all DCAN activities.

Writing to the BRPRS register is only allowed during initialization mode. Any write to this register when INIT bit is set in CANC register and the initialization mode is not confirmed by the INITSTATE bit of CANES register can cause unexpected behaviour to the CAN bus.

PRM1	PRM0	Input Clock Selector for DCAN Clock
0	0	f_X is input for DCAN
0	1	$f_X/2$ is input for DCAN
1	0	$f_X/4$ is input for DCAN
1	1	CCLK is input for DCAN

The BRPRSn bits (n = 0 to 5) define the number of DCAN clocks applied for one TQ.
For BRPRSn (n = 0 to 5) two modes are available depending on the TLMODE bit in the SYNC1 register.

Figure 17-43: Bit Rate Prescaler (2/2)

Setting of BRPRSn (n = 5 to 0) for TLMODE = 0:

BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	Bit Rate Prescaler$^{\text{Note}}$
0	0	0	0	0	0	2
0	0	0	0	0	1	4
0	0	0	0	1	0	6
0	0	0	0	1	1	8
.	.	.	.	.	.	.
.	.	.	.	.	.	2 x BRPRSn[5-0] + 2
.	.	.	.	.	.	.
1	1	1	0	1	0	118
1	1	1	0	1	1	120
1	1	1	1	0	0	122
1	1	1	1	0	1	124
1	1	1	1	1	0	126
1	1	1	1	1	1	128

Note: The bit rate prescaler value represents the DCAN clocks per TQ.

Setting of BRPRSn (n = 7 to 0) for TLMODE = 1:

BRPRS7	BRPRS6	BRPRS5	BRPRS4	BRPRS3	BRPRS2	BRPRS1	BRPRS0	Bit Rate Prescaler
0	0	0	0	0	0	0	0	1$^{\text{Note}}$
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	3
0	0	0	0	0	0	1	1	4
		.	.	.	.	.	.	.
		.	.	.	.	.	.	BRPRSn[7-0] +1
		.	.	.	.	.	.	.
1	1	1	1	1	0	1	0	123
1	1	1	1	1	0	1	1	124
1	1	1	1	1	1	0	0	125
1	1	1	1	1	1	0	1	126
1	1	1	1	1	1	1	0	127
1	1	1	1	1	1	1	1	128

Note: When using this setting the user needs to assure that phase segment 2 consists of at least 3 TQ. Phase segment 2 is given by the difference of DBT - SPT each measured in units of TQ.

BRPRS7 and BRPRS6 are located in the MASKC register.

(2) Synchronization control registers (SYNC0 and SYNC1)

These registers define the CAN bit timing. They define the length of one data bit on the CAN bus, the position of the sample point during the bit timing, and the synchronization jump width. The range of resynchronization can be adapted to different CAN bus speeds or network characteristics. Additionally, some modes related to the baud rate can be selected in SYNC1 register.

SYNC0 and SYNC1 can be read or written with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets SYNC0 to 18H.
$\overline{\text{RESET}}$ input sets SYNC1 to 0EH.

Figure 17-44: Synchronization Control Registers 0 and 1 (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SYNC0	SPT2	SPT1	SPT0	DBT4	DBT3	DBT2	DBT1	DBT0	FFB9H	18H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SYNC1	TLMODE	SOFC	SAMP	0	SJW1	SJW0	SPT4	SPT3	FFBAH	0EH	R/W

The length of a data bit time is programmable via DBT[4-0].

DBT4	DBT3	DBT2	DBT1	DBT0	Data Bit Time
Other than under					Setting prohibited
0	0	1	1	1	8 x TQ
0	1	0	0	0	9 x TQ
0	1	0	0	1	10 x TQ
0	1	0	1	0	11 x TQ
0	1	0	1	1	12 x TQ
0	1	1	0	0	13 x TQ
0	1	1	0	1	14 x TQ
0	1	1	1	0	15 x TQ
0	1	1	1	1	16 x TQ
1	0	0	0	0	17 x TQ
1	0	0	0	1	18 x TQ
1	0	0	1	0	19 x TQ
1	0	0	1	1	20 x TQ
1	0	1	0	0	21 x TQ
1	0	1	0	1	22 x TQ
1	0	1	1	0	23 x TQ
1	0	1	1	1	24 x TQ
1	1	0	0	0	25 x TQ
Other than above					Setting prohibited

Figure 17-44: Synchronization Control Registers 0 and 1 (2/2)

The position of the sample point within the bit timing is defined by SPT0n through SPT4n.

SPT4	SPT3	SPT2	SPT1	SPT0	Sample Point Position
Other than under					Setting prohibited
0	0	0	0	1	2 x TQ
0	0	0	1	0	3 x TQ
0	0	0	1	1	4 x TQ
0	0	1	0	0	5 x TQ
0	0	1	0	1	6 x TQ
0	0	1	1	0	7 x TQ
0	0	1	1	1	8 x TQ
0	1	0	0	0	9 x TQ
0	1	0	0	1	10 x TQ
0	1	0	1	0	11 x TQ
0	1	0	1	1	12 x TQ
0	1	1	0	0	13 x TQ
0	1	1	0	1	14 x TQ
0	1	1	1	0	15 x TQ
0	1	1	1	1	16 x TQ
1	0	0	0	0	17 x TQ
Other than above					Setting prohibited

TLMODE	Resolution of Bit Rate Prescaler
0	1 unit of BRPRS[5-0] in BRPRS register equals 2 DCAN clocks, BRPRS[7-6] in MASKC register are disabled (compatible to older macro versions)
1	1 unit of BRPRS[7-0] in BRPRS and MASKC register equals 2 DCAN clocks, BRPRS[7-6] in MASKC register are enabled**Note**

Note: The user needs to assure that phase segment 2 (TSEG2) consists of at least 3 TQ when using this setting. Phase segment 2 is given by the difference of DBT - SPT each measured in units of TQ.

SJW0 and SJW1 define the synchronization jump width as specified in ISO 11898.

SJW1	SJW0	Synchronisation Jump Width
0	0	1 x TQ
0	1	2 x TQ
1	0	3 x TQ
1	1	4 x TQ

Limits on defining the bit timing

The sample point position needs to be programmed between 3TQ[Note] and 17TQ, which equals a register value of **2** ≤ **SPTxn** ≤ **16** (n = 0, 1; x = 4 to 0).

The number of TQ per bit is restricted to the range from 8TQ to 25TQ, which equals a register value of **7** ≤ **DBTxn** ≤ **24** (n = 0, 1; x = 4 to 0).

The length of phase segment 2 (TSEG2) in TQ is given by the difference of TQ per bit (DBTxn) and the sample point position (SPTxn). Converted to register values the following condition applies:
2 ≤ **DBTxn - SPTxn** ≤ **8** (n = 0, 1; x = 4 to 0).

The number of TQ allocated for soft synchronization must not exceed the number of TQ for phase segment 2, but SJWyn may have as many TQ as phase segment 2:
SJWyn ≤ **DBTxn - SPTxn - 1** (n = 0, 1; x = 4 to 0; y = 0, 1).

Note: Sample point positions of 3 TQ or 4 TQ are for test purposes only. For the minimum number of TQ per bit time, 8TQ, the minimum sample point position is 5 TQ.

Example:

System clock:	fx	8 MHz
CAN parameter:	Baud rate	500 kBaud
	Sample Point	75%
	SJW	25%

At first, calculate the overall prescaler value:

$$\frac{f_X}{Baudrate} = \frac{8\ MHz}{500\ KBaud} = 16$$

16 can be split as 1 x 16 or 2 x 8. Other factors can not be mapped to the registers. Only 8 and 16 are valid values for TQ per bit. Therefore the overall prescaler value realized by BRPRSn is 2 or 1 respectively.

With TLMODE = 0 the following register settings apply:

Register value	Description	Bit fields
BRPRSn = 00h	Clock selector = fx	PRMn = 00b
		BRPRSx = 000000b
SYNC0n = A7h	CAN Bit in TQ = 8	DBTx = 00111b
	7 < (fx/Baudrate/bit rate prescaler) < 25]	
SYNC1n = 0zzz0100b	sample point 75% = 6 TQ	SPTx = 00101b
	SJW 25% = 2 TQ	SJWy = 01b
	1 TQ equals 2 clocks & BRPRS6, 7 are disabled	TLMODE = 0
	z depends on the setting of: - Number of sampling points - Receive only function - Use of time stamp or global time system	

With TLMODE = 1 the following register settings apply:

Register values	Description	Bit fields
BRPRSn = 00h	Clock selector = fx	PRMn = 00b
MASKCn = 00xx xxxxb		BRPRSn = 0000 0000b
SYNC0n = 6Fh	CAN Bit in TQ = 16	DBTn = 01111b
	7 < (fx/Baudrate/bit rate prescaler) < 25]	
SYNC1n = 1zzz 1101b	sample point 75% = 12 TQ:	SPTn = 01011b
	SJW 25% = 4 TQ	SJWn = 11b
	1 TQ equals 1 clock, BRPRS 6, 7 are enabled	TLMODE = 1
	z depends on the setting of: - Number of sampling points - Receive only function - Use of time stamp or global time system	

The VALID bit in CANES reports if the DCAN interface receives any valid message.

SAMP defines the number of sample points per bit as specified in the ISO-11898.

SAMP	Bit Sampling
0	Sample receive data one time at receive point
1	Sample receive data three times and take majority decision at sample point

SOFC works in conjunction with the SOFE and SOFSEL bits in the CAN Control Register CANC. For detailed information please refer to the bit description of that SFR register and the time function mode.

SOFC	Start of Frame Control
0	SOFE bit is independent from CAN bus activities
1	SOFE bit will be cleared when a message for receive message 4 is received and SOF mode is selected

Caution: CPU can read SYNC0/SYNC1 register at any time. Writing to the SYNC0/SYNC1 registers is only allowed during initialization mode. Any write to this register when INIT is set and the initialization mode is not confirmed by the INITSTATE bit can have unexpected behavior to the CAN bus.

17.15 Function Control

17.15.1 Transmit Control

(1) Transmit control register (TCR)

This register controls the transmission of the DCAN-module. The transmit control register (TCR) provides complete control over the two transmit buffers and their status. It is possible to request and abort transmission of both buffers independently.

TCR can be set with an 8-bit memory manipulation instruction.
RESET input sets TCR to 00H.

Figure 17-45: Transmit Control Register (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
TCR	TXP	0	TXC1	TXC0	TXA1	TXA0	TXRQ1	TXRQ0	FFB1H	00H
	R/W	R	R	R	R/W	R/W	R/W	R/W		

Caution: Don't use bit operations on this register. Also logical operations (read-modify-write) via software may lead to unexpected transmissions. Initiating a transmit request for buffer 1 while TXRQ0 is already set, is simply achieved by writing 02H or 82H. The status of the bits for buffer 0 is not affected by this write operation.

TXP	Transmission Priority
0	Buffer 0 has priority over buffer 1
1	Buffer 1 has priority over buffer 0

The user defines which buffer has to be send first in the case of both request bits are set. If only one buffer is requested by the TXRQn bits (n = 0, 1) bits, TXP bit has no influence.
TXCn (n = 0, 1) shows the status of the first transmission. It is updated when TXRQn (n = 0, 1) is cleared.

TXAn	Transmission Abort Flag
0	Write: normal operation
	Read: no abort pending
1	Write: aborts current transmission request for this buffer n
	Read: abort is pending

TXCn	Transmission Complete Flag
0	Transmit was aborted / no data sent
1	Transmit was complete / abort had no effect

The TXAn bits (n = 0, 1) allow to free a transmit buffer with a pending transmit request. Setting the TXAn bit (n = 0, 1) by the CPU requests the DCAN to empty its buffer by clearing the respective TXRQn bit (n = 0, 1).

Figure 17-45: Transmit Control Register (2/2)

The TXAn bits (n = 0, 1) have a dual function:

1. The CPU can request an abort by writing an “1” into the bit.

2. The DCAN signals whether such an request is still pending. The bit is cleared at the same time when the TXRQn bit (n = 0, 1) is cleared.

The abort process does not affect any rules of the CAN protocol. A frame already started will continue to its end.

An abort operation can cause different results dependent on the time it is issued.

d) When an abort request is recognized by the DCAN before the start of the arbitration for transmit, the TXCn bit (n = 0, 1) is reset showing that the buffer was not send to other nodes.
e) When the abort request is recognized during the arbitration and the arbitration is lost afterwards, the TXCn bit (n = 0, 1) is reset showing that the buffer was not send to other nodes.
f) When the abort request is recognized during frame transmission and the transmission ends with an error afterwards, the TXCn bit (n = 0, 1) is reset showing that the buffer was not send to other nodes.
g) When the abort request is recognized during the frame transmission and transmission ends without error. The TXCn bit (n = 0, 1) is set showing a successful transfer of the data. I.e the abort request was not issued.

In all cases the TXRQn bit and the TXAn bit (n = 0, 1) bit will be cleared at the end of the abort operation, when the transmit buffer is available again.

Cautions: 1. The bits are cleared when the INIT bit in CANC register is set.

2. Writing a 0 to TXAn (n = 0, 1) bit has no influence

3. Do not perform read-modify-write operations on TCR.

The TXCn bit (n = 0, 1) are updated at the end of every frame transmission or abort.

TXRQn	Transmission Request Flag
0	Write: no influence
	Read: transmit buffer is free
1	Write: request transmission for buffer n
	Read: transmit buffer is occupied by former transmit request

The transmit request bits are checked by the DCAN immediately before the frame is started. The order in which the TXRQn bit (n = 0, 1) will be set does not matter as long as the first requested frame is not started on the bus.
The TXRQn bit (n = 0, 1) have dual function:

- 1. Request the transmission of a transmit buffer.
- 2. Inform the CPU whether a buffer is available or if it is still occupied by a former transmit request.

Setting the transmission request bit requests the DCAN to sent the buffer contents onto the bus. The DCAN clears the bit after completion of the transmission. Completion is either a normal transfer without error or an abort request.

An error during the transmission does not influence the transmit request status. The DCAN will automatically retry the transfer.

Cautions: 1. The bits are cleared when the INIT bit in CANC is set. A transmission already started will be finished but not retransmitted in case of an error.

2. Writing a “0” to TXRQ0 bit has no influence.

3. Do not use bit operations on this register.

4. Do not change data in transmit buffer when the corresponding TXRQ bit is set.

17.15.2 Receive Control

The receive message register mirrors the current status of the first 8 receive buffers. Each buffer has one status bit in this register. This bit is always set when a new message is completely stored out of the shadow buffer into the associated buffer. The CPU can easily find the last received message during receive interrupt handling. The bits in this register always correspond to the DN bit in the data buffers. They are cleared when the CPU clears the DN bit in the data buffer. The register itself is read only.

(1) Receive message register (RMES)

This register shows receptions of messages of the DCAN-module. More than one bit set is possible.

RMES can be read with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets RMES to 00H.

Figure 17-46: Receive Message Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset
RMES	DN7	DN6	DN5	DN4	DN3	DN2	DN1	DN0	FFB2H	00H
	R	R	R	R	R	R	R	R		

This register is read only and it is cleared when the INIT bit in CANC register is set.

DN	Data New Bit for Message n (n = 0...7)
0	No message received on message n or CPU has cleared DN bit in message n
1	Data received in message n that was not acknowledged by the CPU

DN0 bit has no meaning when receive buffer 0 is configured for mask operation in the mask control register.
DN2 bit has no meaning when receive buffer 2 is configured for mask operation in the mask control register.

17.15.3 Mask Control

The mask control register defines whether the DCAN compares all identifier bits or if some bits are not used for comparison. This functionality is provided by the use of the mask information. The mask information defines for each bit of the identifier whether it is used for comparison or not. The DCAN uses a receive buffer for this information, when it is enabled by the mask control register. In this case this buffer is not used for normal message storage. Unused bytes can be used for application needs.

(1) Mask control register (MASKC)

This register controls the mask function applied to any received message.

MASKC can be written with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets MASKC to 00H.

Figure 17-47: Mask Control Register (1/2)

Symbol	7[Note]	6[Note]	5	4	3	2	1	0	Address	After Reset
MASKC	BRPRS7	BRPRS6	SSHT	AL	0	GLOBAL	MSK1	MSK0	FFBBH	00H
	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W		

Note: BRPRS[7 - 6] are only enable if TLMODE is set to 1.

Caution: This register is readable at any time. Writing to the MASKC register is only allowed during initialization mode. Any write to this register when INIT bit is set and the initialization mode is not confirmed by the INITSTATE bit can have unexpected behavior to the CAN bus.

MSK0	Mask 0 Enable
0	Receive buffer 0 and 1 in normal operation
1	Receive buffer 0 is mask for buffer 1

MSK1	Mask 1 Enable
0	Receive buffer 2 and 3 in normal operation
1	Receive buffer 2 is mask for buffer 3

GLOBAL	Enable Global Mask
0	Normal operation
1	Highest defined mask is active for all following buffers

Figure 17-47: Mask Control Register (2/2)

SSHT	AL	Function
0	x	Single shot mode disabled
1	0	Single shot mode enabled; no re-transmission when an error occurs. Transmit message will not be queued for a second transmit request when the arbitration was lost
1	1	Single shot mode enabled; no re-transmission when an error occurs. Transmit message will be queued for a second transmit request when the arbitration was lost.

BRPRS7	BRPRS6	Prescaler values
0	0	Selects 0 - 64 DCAN clocks per time quanta
0	1	Selects 65 - 128 DCAN clocks per time quanta
1	0	Selects 129 - 192 DCAN clocks per time quanta
1	1	Selects 193 - 256 DCAN clocks per time quanta

The following table shows which compare takes place for the different receive buffers. The ID in this table always represents the ID stored in the mentioned receive buffer. The table also shows which buffers are used to provide the mask information and therefore do not receive messages. A global mask can be used for standard and extended frames at the same time. The frame type is only controlled by the IDE bit of the receiving buffer.

Table 17-26: Mask Operation Buffers

GLOBAL	MSK1	MSK0	Receive Buffer					Operation
			0	1	2	3	4-15	
X	0	0	Compare ID	Compare ID	Compare ID	Compare ID	Compare ID	Normal
0	0	1	Mask0	Compare ID & mask0	Compare ID	Compare ID	Compare ID	One mask
0	1	0	Compare ID	Compare ID	Mask1	Compare ID & mask1	Compare ID	One mask
0	1	1	Mask0	Compare ID & mask0	Mask1	Compare ID & mask1	Compare ID	Two masks
1	0	1	Mask0	Compare ID & mask0	Compare ID & mask0	Compare ID	& mask0	Global mask
1	1	0	Compare ID	Compare ID	Mask1	Compare ID	& mask1	Two normal, rest global mask
1	1	1	Mask0	Compare ID & mask0	Mask1	Compare ID	& mask1	One mask, rest global mask

Priority of receive buffers during compare

It is possible that more than one receive buffer is configured to receive a particular message. For this case an arbitrary rule for the storage of the message into one of several matching receive buffers becomes effective. The priority of a receive buffers depends on its type defined by the setup of the mask register in first place and its number in second place.

The rules for priority are:

- All non-masked receive buffers have a higher priority than the masked receive buffer.
- Lower numbered receive buffers have higher priority.

Examples:

1. All RX buffers are enabled to receive the same standard identifier 0x7FFH. Result: the message with identifier 0x7FFH is stored in RX0.
2. In difference to the previous set up, the mask option is set for RX2. Again the message 0x7FFH is stored in buffer in RX0.
3. If additionally RX0 is configured as a mask, the message will be stored in RX4.

17.15.4 Special Functions

(1) Redefinition control register (REDEF)

This register controls the redefinition of an identifier of a received buffer.

REDEF can be written with an 1-bit or an 8-bit memory manipulation instruction.
RESET input sets REDEF to 00H.

Figure 17-48: Redefinition Control Register (1/2)

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset
REDEF	DEF	0	0	0	SEL3	SEL2	SEL1	SEL0	FFB3H	00H
	R/W	R	R	R	R/W	R/W	R/W	R/W		

The redefinition register provides a way to change identifiers and other control information for one receive buffer, without disturbing the operation of the other buffers.

DEF	Redefine Permission Bit
0	Normal operation
1	Receive operation for selected message is disabled. CPU can change definition data for this message.

This bit is cleared when INIT bit in CANC is set.

Figure 17-48: Redefinition Control Register (2/2)

SEL3	SEL2	SEL1	SEL0	Buffer selection (n =0...15)
0	0	0	0	Buffer 0 is selected for redefinition
0	0	0	1	Buffer 1 is selected for redefinition
0	0	1	0	Buffer 2 is selected for redefinition
0	0	1	1	Buffer 3 is selected for redefinition
0	1	0	0	Buffer 4 is selected for redefinition
0	1	0	1	Buffer 5 is selected for redefinition
0	1	1	0	Buffer 6 is selected for redefinition
0	1	1	1	Buffer 7 is selected for redefinition
1	0	0	0	Buffer 8 is selected for redefinition
1	0	0	1	Buffer 9 is selected for redefinition
1	0	1	0	Buffer 10 is selected for redefinition
1	0	1	1	Buffer 11 is selected for redefinition
1	1	0	0	Buffer 12 is selected for redefinition
1	1	0	1	Buffer 13 is selected for redefinition
1	1	1	0	Buffer 14 is selected for redefinition
1	1	1	1	Buffer 15 is selected for redefinition
Other than above				Setting prohibited

Cautions:

1. **Keep special programming sequence. Failing to do so can cause inconsistent data or loss of receive data.**
2. **Do not change DEF bit and SEL bit at the same time. Change SEL bit only when DEF bit is cleared.**
3. **Write first SEL with DEF cleared. Write than SEL with DEF, or use bit manipulation instruction. Only clear DEF bit by keeping SEL or use bit manipulation instruction.**

Setting the redefinition bit removes the selected receive buffer from the list of possible ID hits during identifier comparisons.

Setting the DEF bit will not have immediate effect, if DCAN is preparing to store or is already in progress of storing a received message into the particular buffer. In this case the redefinition request is ignored for the currently processed message.

The application should monitor the DN flag before requesting the redefinition state for a particular buffer. A DN flag set indicates a new message that arrived or a new message that is in progress of being stored to that buffer. The application should be prepared to receive a message immediately after redefinition state was set. The user can identify this situation because the data new bit (DN) in the receive buffer will be set. This is of special importance if it is used together with a mask function because in this case the DCAN also writes the identifier part of the message to the receive buffer. Then the application needs to re-write the configuration of the message buffer.

17.16 Interrupt Information

17.16.1 Interrupt Vectors

The DCAN peripheral supports four interrupt sources as shown in the following table.

Table 17-27: Interrupt Sources

Function	Source	Interrupt Flag
Error	Error counter Overrun error Wake up	CEIF
Receive	Received frame is valid	CRIF
Transmit buffer 0	TXRQ0 is cleared	CTIF0
Transmit buffer 1	TXRQ1 is cleared	CTIF1

17.16.2 Transmit Interrupt

The transmit interrupt is generated when all following conditions are fulfilled:

- The transmit interrupt 0 is generated when TXRQ0 bit is cleared.
- The transmit interrupt 1 is generated when TXRQ1 bit is cleared.

Clearing of these bits releases the buffer for writing a new message into it. This event can occur due to a successful transmission or due to an abort of a transmission. Only the DCAN can clear this bit. The CPU can only request to clear the TXRQn bit by setting the ABORTn bit (n = 0, 1).

17.16.3 Receive Interrupt

The receive interrupt is generated when all of the following conditions are fulfilled:

- CAN protocol part marks received frame valid.
- The received frame passes the acceptance filter. In other words, a message buffer with an identifier/mask combination fits to the received frame.
- The memory access engine successfully stored data in the message buffer.
- The message buffer is marked for interrupt generation with ENI bit set.

The memory access engine can delay the interrupt up to the 7th bit of the next frame because of its compare and store operations.

17.16.4 Error Interrupt

The error interrupt is generated when any of the following conditions are fulfilled:

- Transmission error counter (BOFF) changes its state.
- Transmission error counter status (TECS) changes its state.
- Reception error counter status (RECS) changes its state.
- Overrun during RAM access (OVER) becomes active.
- The wake-up condition (WAKE) becomes active.

The wake-up condition activates an internal signal to the interrupt controller. In order to receive further error interrupts generated by other conditions, the CPU needs to clear the WAKE bit in CANES register every time a wake-up condition was recognized.
No further interrupt can be detected by the CPU as long as the WAKE bit is set.

17.17 Influence of the standby Function of the CAN Controller

17.17.1 CPU Halt Mode

The CPU halt mode is possible in conjunction with DCAN Sleep mode.

17.17.2 CPU Stop Mode

The DCAN stops any activity when its clock supply stops due to a CPU Stop mode issued. This may cause an erroneous behaviour on the CAN bus. Entering the CPU Stop Mode is not allowed when the DCAN is in normal mode, i.e. online to the CAN bus.

The DCAN will reach an overrun condition, when it receives clock supply again.

CPU Stop mode is possible when the DCAN was set to initialization state, sleep mode or stop mode beforehand. Note that the CPU will not be started again if the DCAN Stop mode was entered previously.

17.17.3 DCAN Sleep Mode

The DCAN Sleep mode is intended to lower the power consumption during phases where no communication is required.
The CPU requests the DCAN Sleep mode. The DCAN will signal with the WAKE bit, if the request was granted or if it is not possible to enter the sleep mode due to ongoing bus activities.
After a successful switch to the DCAN Sleep mode, the CPU can safely go into halt, watch or stop mode. However, the application needs to be prepared that the DCAN cancels the sleep mode any time due to bus activities. If the wake-up interrupt is serviced, the CPU Stop mode has not to be issued.
Otherwise the CPU will not be released from CPU Stop mode even when there is ongoing bus activity.
The wake-up is independent from the clock. The release time for the CPU Stop mode of the device is of no concern because the DCAN synchronizes again to the CAN bus after clock supply has started.

The following example sketches the general approach on how to enter the DCAN Sleep mode. Note that the function may not return for infinite time when the CAN bus is busy. The user may apply time out controls to avoid excessive run-times.

Code example:

```
DCAN_Sleep_Mode(void){
CANES = 0x02;                 // clear Wake bit
CANC = 0x04                   // request DCAN Sleep mode
while (CANES & 0x02)          // check if DCAN Sleep mode was accepted
        {
        CANES = 0x02;         // try again to get DCAN asleep
        CANC = 0x04;
        }
}
```

The following code example assures a safe transition into CPU Stop mode for all timing scenarios of a suddenly occurring bus activity. The code prevents that the CPU gets stuck with its oscillator stopped despite CAN bus activity.

Code example:

```
........                      //any application code

DCAN_Sleep_Mode;              //request and enter DCAN sleep mode

........                      //any application code

DI();                         //disable interrupts
NOP;Note
NOP;
if (wakeup_interrupt_occurred == FALSE)
                              // the variable wakeup_interrupt occurred
                              // needs to be initialized at system reset
                              // and it needs to be set TRUE when servicing
                              // the wake-up interrupt.
        {
        CPU_STOP;             //enter CPU Stop mode
        }
NOP:Note
NOP:
NOP;
EI();                         // enable interrupts
.........                     // resume with application code
```

Note: The interrupt acknowledge needs some clock cycles (depends on host core). In order to prevent that the variable wakeup_interrupt_occurred is already read before DI(); becomes effective some NOP-instruction have to be inserted. As well the number of NOP-instructions after the CPU Stop instruction is dependent on the host core. The given example is tailored for 78K0.

17.17.4 DCAN Stop Mode

The CPU requests this mode from DCAN. The procedure equals the request for DCAN Sleep mode. The DCAN will signal with the WAKE bit, if the request was granted or if it is not possible to enter the DCAN Stop mode due to ongoing bus activities.

After a successful switch to the DCAN Stop mode, the CPU can safely go into halt, watch or stop mode without any precautions. The DCAN can only be woken up by the CPU. Therefore the CPU needs to clear the SLEEP bit in the CANC register.

This mode reduces the power consumption of the DCAN to a minimum.

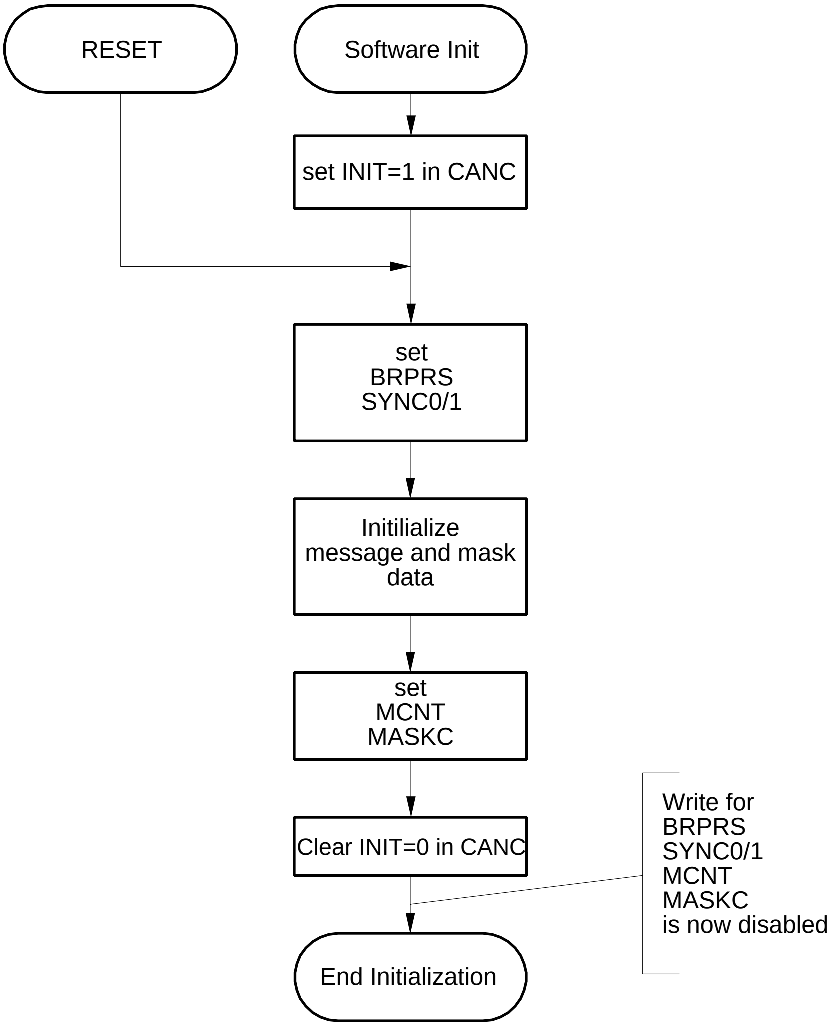
Code example:

```
DCAN_Stop_Mode(void){
CANES = 0x02;                      // clear Wake bit
CANC = 0x06                        // request DCAN Stop mode
while (CANES & 0x02)               // check if DCAN Stop mode was accepted
        {
        CANES = 0x02;              // try again to get DCAN into stop mode
        CANC = 0x06;
        }
}
```

17.18 Functional Description by Flowcharts

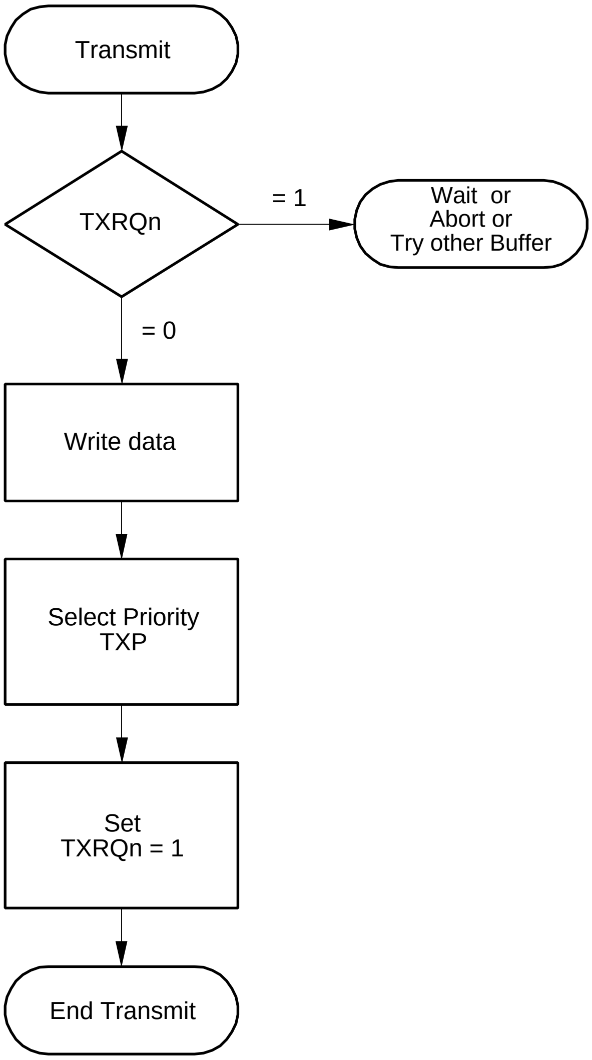
17.18.1 Initialization

Figure 17-49: Initialization Flow Chart



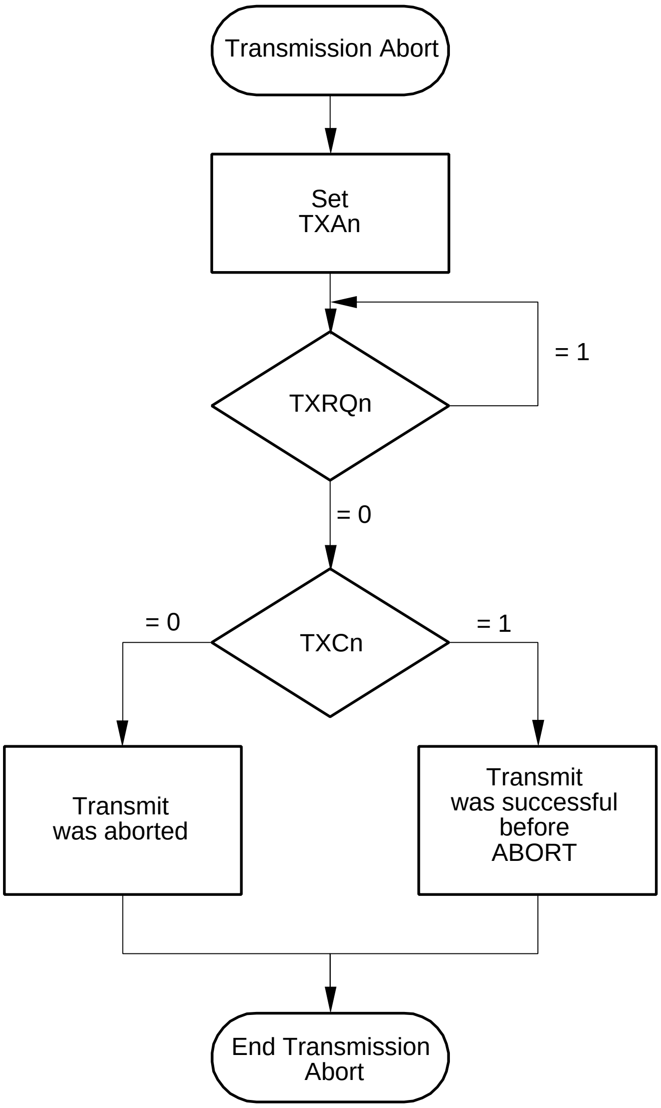
17.18.2 Transmit Preparation

Figure 17-50: Transmit Preparation



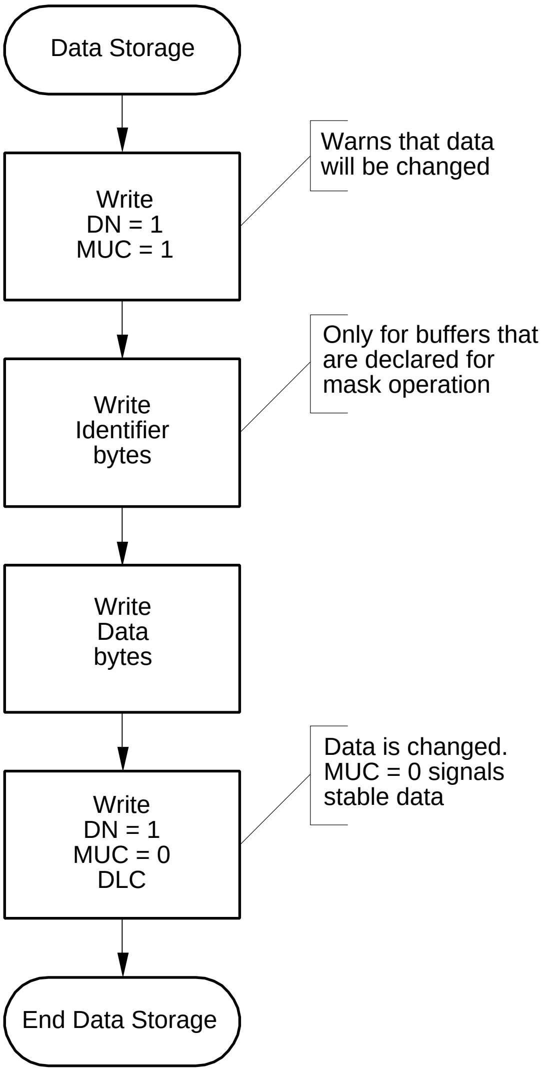
17.18.3 Abort Transmit

Figure 17-51: Transmit Abort



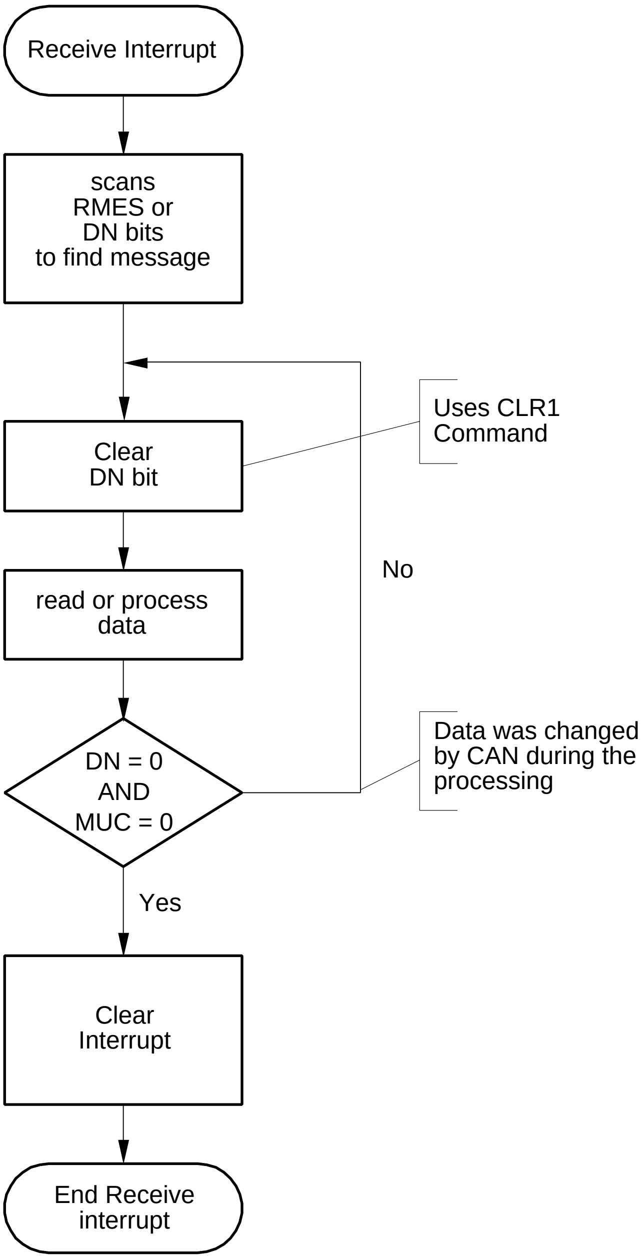
17.18.4 Handling by the DCAN

Figure 17-52: Handling of Semaphore Bits by DCAN-Module


Data Storage
Write
DN = 1
MUC = 1
Warns that data will be changed
Write
Identifier
bytes
Only for buffers that are declared for mask operation
Write
Data
bytes
Write
DN = 1
MUC = 0
DLC
Data is changed. MUC = 0 signals stable data
End Data Storage

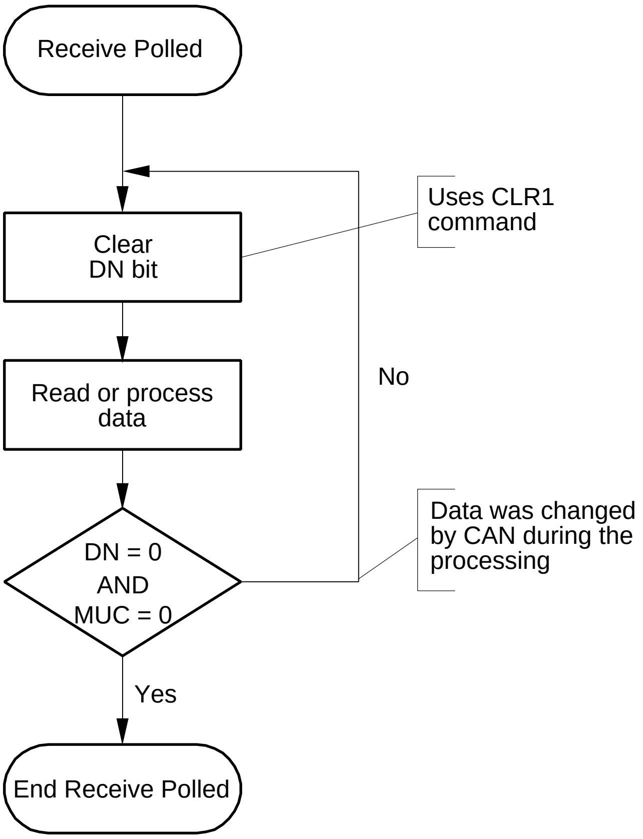
17.18.5 Receive Event Oriented

Figure 17-53: Receive with Interrupt, Software Flow



17.18.6 Receive Task Oriented

Figure 17-54: Receive, Software Polling



17.19 CAN Controller Precautions

(1) Resynchronization

According to the CAN protocol specification (BOSCH CAN specification, version 2.0, Sept. 1991, part A, chapter 8) a CAN node has to perform a soft-synchronization, when acting as a transmitter sending a dominant bit, if a recessive to dominant edge occurs after the sample point within phase segment 2. This scenario is only encountered in case of a disturbance. For this case the soft-synchronization is not performed by the implementations listed below. Due to this, the nominal length of an error frame, that follows this disturbance, can be extended by the amount of time quanta, allocated for the synchronization jump width.

[MEMO]

Chapter 18 Interrupt Functions

18.1 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.
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 18-1, “Interrupt Source List,” on page 348).
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.

18.2 Interrupt Sources and Configuration

There are total of 26 interrupt sources: non-maskable, maskable, and software interrupts.

Table 18-1: Interrupt Source List

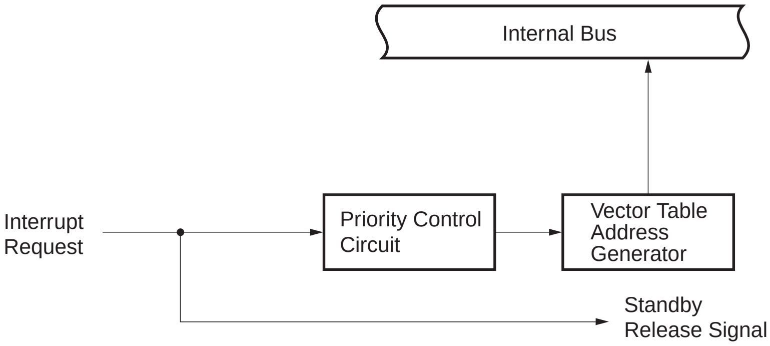
Mask-ability	Interrupt Priority Note 1	Interrupt Source		Internal/ External	Vector Address	Basic Structure Type Note 2
		Name	Trigger			
Non-maskable	–	INTWDT	Overflow of watchdog timer (When the Watchdog timer NMI is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (When the 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 capture register (CR20) match signal		000AH	
	4	INTTM21	Generation of 16-bit timer capture register (CR21) match signal		000CH	
	5	INTTM22	Generation of 16-bit timer 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	INTCE	CAN Error	Internal	0018H	(D)
	11	INTCR	CAN Receive		001AH	
	12	INTCT0	CAN Transmit buffer 0		001CH	
	13	INTCT1	CAN Transmit buffer 1		001EH	
	14	INTCSI20	End of serial interface SIO20 transfer		0020H	
	15	INTSER	Serial interface UART reception error generation		0022H	
	16	INTSR	End of Serial interface UART reception		0024H	
	17	INTST	End of Serial interface UART transfer		0026H	
	18	INTTM00	Generation of 16-bit timer 0 capture/compare register (CR00) match signal		0028H	
	19	INTTM01	Generation of 16-bit timer 0 capture/compare register (CR01) match signal		002AH	
	20	INTTM50	Generation of 8-bit timer/event counter 50 match signal		002CH	
	21	INTTM51	Generation of 8-bit timer/event counter 51 match signal		002EH	
	22	INTWTI	Reference time interval signal from watch timer		0032H	
	23	INTWT	Reference time interval signal from watch timer		0034H	
	24	INTKR	Key Return interrupt signal		0036H	(E)
	25	INTCSI30	End of serial interface SIO30 transfer		0038H	(D)
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 25 is the lowest priority.

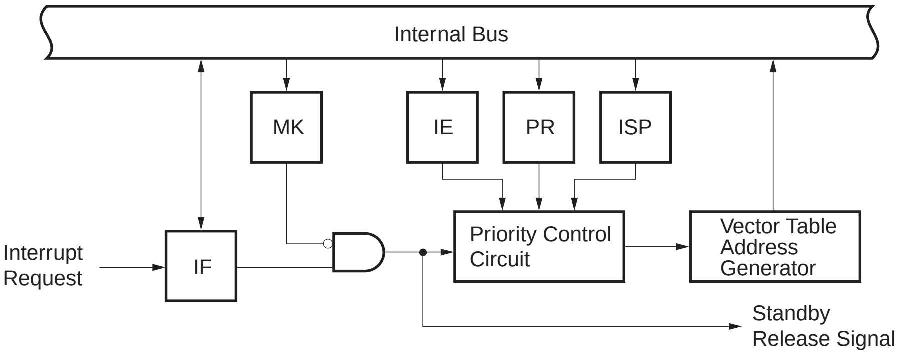
2. Basic configuration types (A) to (E) correspond to (A) to (E) of Figure 18-1on page 349.

Figure 18-1: Basic Configuration of Interrupt Function (1/2)

(a) Internal non-maskable interrupt



(b) Internal maskable interrupt

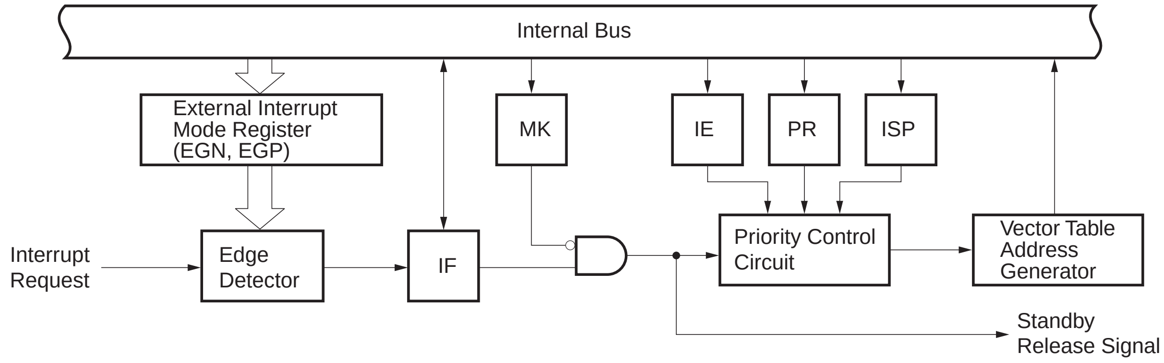


Remark:

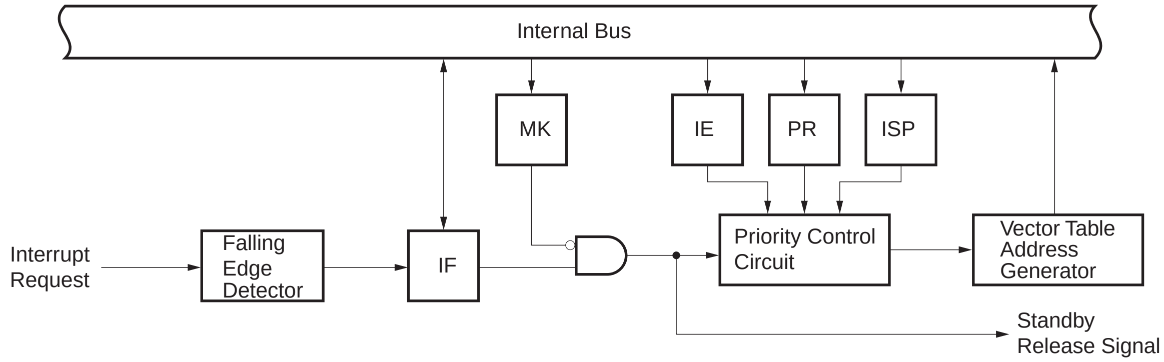
IF	:	Interrupt request flag
IE	:	Interrupt enable flag
ISP	:	In-service priority flag
MK	:	Interrupt mask flag
PR	:	Priority specify flag

Figure 18-1: Basic Configuration of Interrupt Function (2/2)

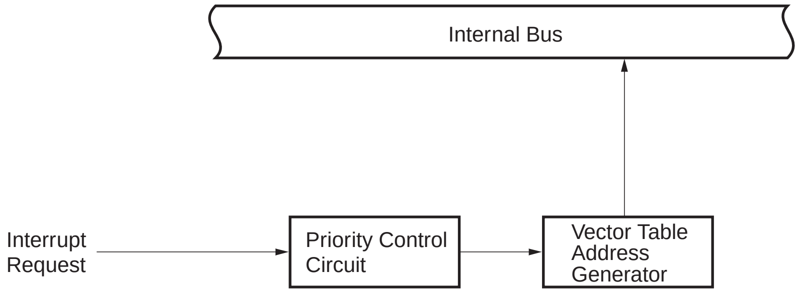
(c) External maskable interrupt



(d) External maskable interrupt (INTKR)



(e) Software interrupt



Remark:			
	IF	:	Interrupt request flag
	IE	:	Interrupt enable flag
	ISP	:	In-service priority flag
	MK	:	Interrupt mask flag
	PR	:	Priority specify flag

18.3 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 18-2 gives a listing of interrupt request flags, interrupt mask flags, and priority specify flags corresponding to interrupt request sources.

Table 18-2: 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
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
INTM52	TMIF52	TMMK52	TMPR52
INTWTI	WTIIF	WTIMK	WTIPR
INTWT	WTIF	WTMK	WTPR
INTWDT	WDTIF	WDTMK	WDTPR
INTAD	ADIF	ADMK	ADPR
INTCSI20	CSIIF20	CSIMK20	CSIPR20
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
INTKR	KRIF	KRMK	KRPR
INTCSI30	CSIIF30	CSIMK30	CSIPR30

(1) Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H)

The interrupt request flag is set to 1 when the corresponding interrupt request is generated. It is cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon application of $\overline{\text{RESET}}$ input.

IF0L, IF0H, IF1L and IF1H are set with an 1-bit or an 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register IF0, use a 16-bit memory manipulation instruction for the setting.

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

Figure 18-2: 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	SERIF	CSIIF30	CTIF1	CTIF0	CRIF	CEIF	PIF3	PIF2	FFE1H	00H	R/W
IF1L	WTIF	0	TMIF51	TMIF50	TMIF01	TMIF00	STIF	SRIF	FFE2H	00H	R/W
IF1H	0	0	0	0	0	CSIIF30	KRIF	WTIF	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 the 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. MK0L, MK0H, MK1L and MK1H are set with an 1-bit or an 8-bit memory manipulation instruction. If MK0L and MK0H are used as a 16-bit register MK0, use a 16-bit memory manipulation instruction for the setting.
RESET input sets these registers to FFH.

Figure 18-3: 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	FFH	R/W
MK0H	SERMK	CSIMK20	CTMK1	CTMK0	CRMK	CEMK	PMK3	PMK2	FFE5H	FFH	R/W
MK1L	WTMK	1	TMMK51	TMMK50	TMMK01	TMMK00	STMK	SRMK	FFE6H	FFH	R/W
MK1H	1	1	1	1	1	CSIMK30	KRMK	WTMK	FFE7H	FFH	R/W

xxMKx	Interrupt Servicing Control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions: 1. If TMMK4 flag is read when the 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 an 1-bit or an 8-bit memory manipulation instruction. If PR0L and PR0H 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 18-4: 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	FFH	R/W
PR0H	SERPR	CSIPR20	CTPR1	CTPR0	CRPR	CEPR	PPR3	PPR2	FFE9H	FFH	R/W
PR1L	WTPR	1	TMPR51	TMPR50	TMPR01	TMPR00	STPR	SRPR	FFEAH	FFH	R/W
PR1H	1	1	1	1	1	CSIPR30	KRPR	WTPR	FFEBH	FFH	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 to 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 P03.
EGP and EGN can be read or written to with an 1-bit or an 8-bit memory manipulation instruction.
These registers are set to 00H when the $\overline{\text{RESET}}$ signal is output.

Figure 18-5: Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
EGP	0	0	0	0	EGP3	EGP2	EGP1	EGP0	FF48H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
EGN	0	0	0	0	EGN3	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	Valid edge of INTPn pin (n = 0 - 3)
0	0	Interrupt disable
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

(5) Program status word (PSW)

The program status word is a register to hold the instruction execution result and the current status for interrupt request. The IE flag to set maskable interrupts (enable/disable) and the ISP flag to control multiple interrupt servicing are mapped.
Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, and when the BRK instruction is executed, the contents of PSW automatically is saved onto the stack and the IE flag is reset to 0. If a maskable interrupt request is acknowledged contents of the priority specify flag of the acknowledged interrupt are transferred to the ISP flag. The acknowledged contents of PSW is also saved onto the stack with the PUSH PSW instruction. It is retrieved from the stack with the RETI, RETB, and POP PSW instructions.
$\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 18-6: Program Status Word Format

Symbol	7	6	5	4	3	2	1	0	After Reset	R/W
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY	02H	R/W

ISP	Priority of Interrupt Currently Being Received
0	High-priority interrupt servicing (low-priority interrupt disable)
1	Interrupt request not acknowledged or low-priority interrupt servicing (all-maskable interrupts enable)

IE	Interrupt Request Acknowledge Enable/Disable
0	Disable
1	Enable

18.4 Interrupt Servicing Operations

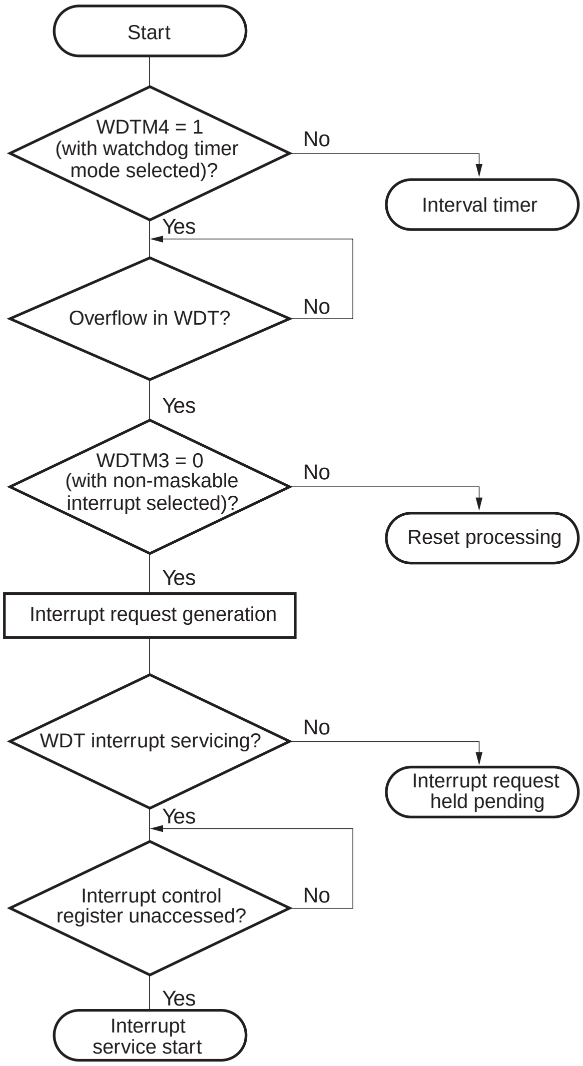
18.4.1 Non-maskable interrupt request acknowledge operation

A non-maskable interrupt request is unconditionally acknowledged even if in an interrupt request acknowledge disable state. It does not undergo interrupt priority control and has highest priority over all other interrupts.

If a non-maskable interrupt request is acknowledged, PSW and PC are pushed on the stack. The IE and ISP flags are reset to 0, and the vector table contents are loaded into PC.

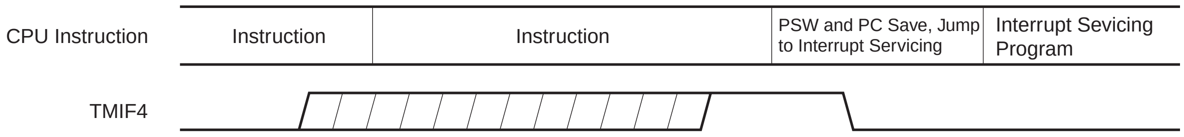
A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after the current execution of the non-maskable interrupt servicing program is terminated (following RETI instruction execution) and one main routine instruction is executed. If a new non-maskable interrupt request is generated twice or more during a non-maskable interrupt service program execution, only one non-maskable interrupt request is acknowledged after termination of the non-maskable interrupt service program execution.

Figure 18-7: Flowchart from Non-Maskable Interrupt Generation to Acknowledge


Start
WDTM4 = 1 (with watchdog timer mode selected)?
No
Interval timer
Yes
Overflow in WDT?
No
Yes
WDTM3 = 0 (with non-maskable interrupt selected)?
No
Reset processing
Yes
Interrupt request generation
WDT interrupt servicing?
No
Interrupt request held pending
Yes
Interrupt control register unaccessed?
No
Yes
Interrupt service start

Remark: WDTM : Watchdog timer mode register
WDT : Watchdog timer

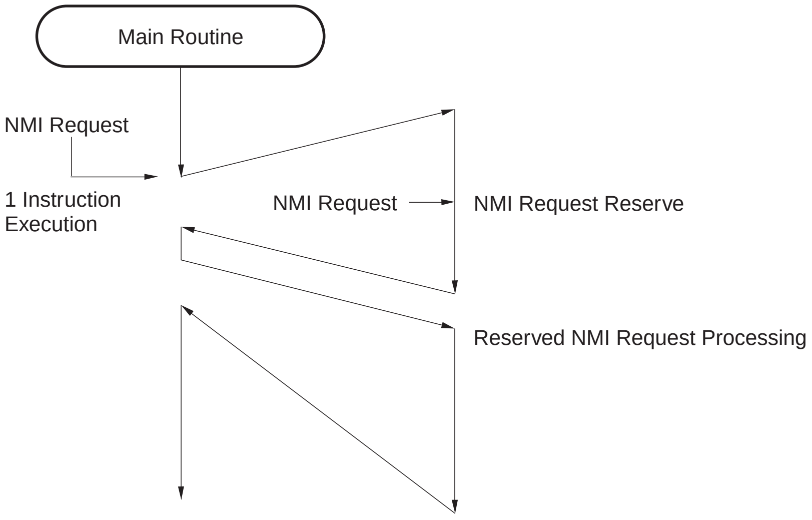
Figure 18-8: Non-Maskable Interrupt Request Acknowledge Timing


CPU Instruction
Instruction
Instruction
PSW and PC Save, Jump to Interrupt Servicing
Interrupt Sevicing Program
TMIF4

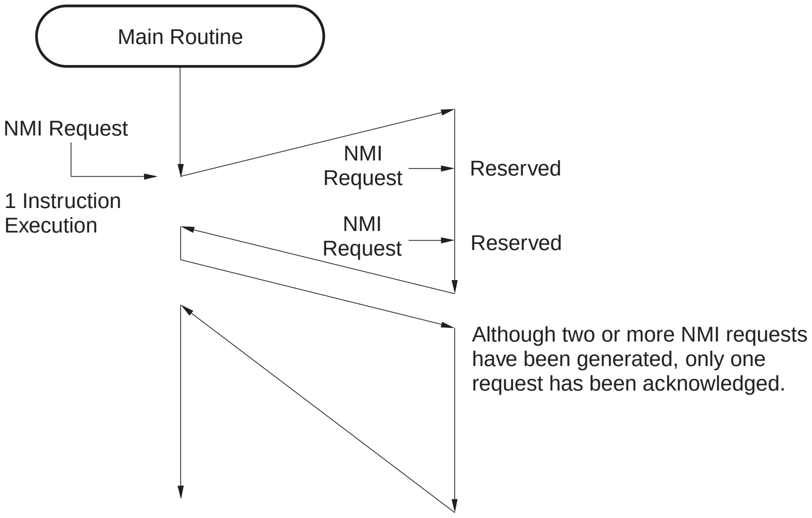
Remark: WDTIF : Watchdog timer interrupt request flag

Figure 18-9: 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


Main Routine
NMI Request
1 Instruction Execution
NMI Request
Reserved
NMI Request
Reserved
Although two or more NMI requests have been generated, only one request has been acknowledged.

18.4.2 Maskable interrupt request acknowledge operation

A maskable interrupt request becomes acknowledgeable when an interrupt request flag is set to 1 and the interrupt mask (MK) flag is cleared to 0. A vectored interrupt request is acknowledged in an interrupt enable state (with IE flag set to 1). However, a low-priority interrupt request is not acknowledged during high-priority interrupt service (with ISP flag reset to 0).
Wait times from maskable interrupt request generation to interrupt servicing are as follows.

Table 18-3: Times from Maskable Interrupt Request Generation to Interrupt Service

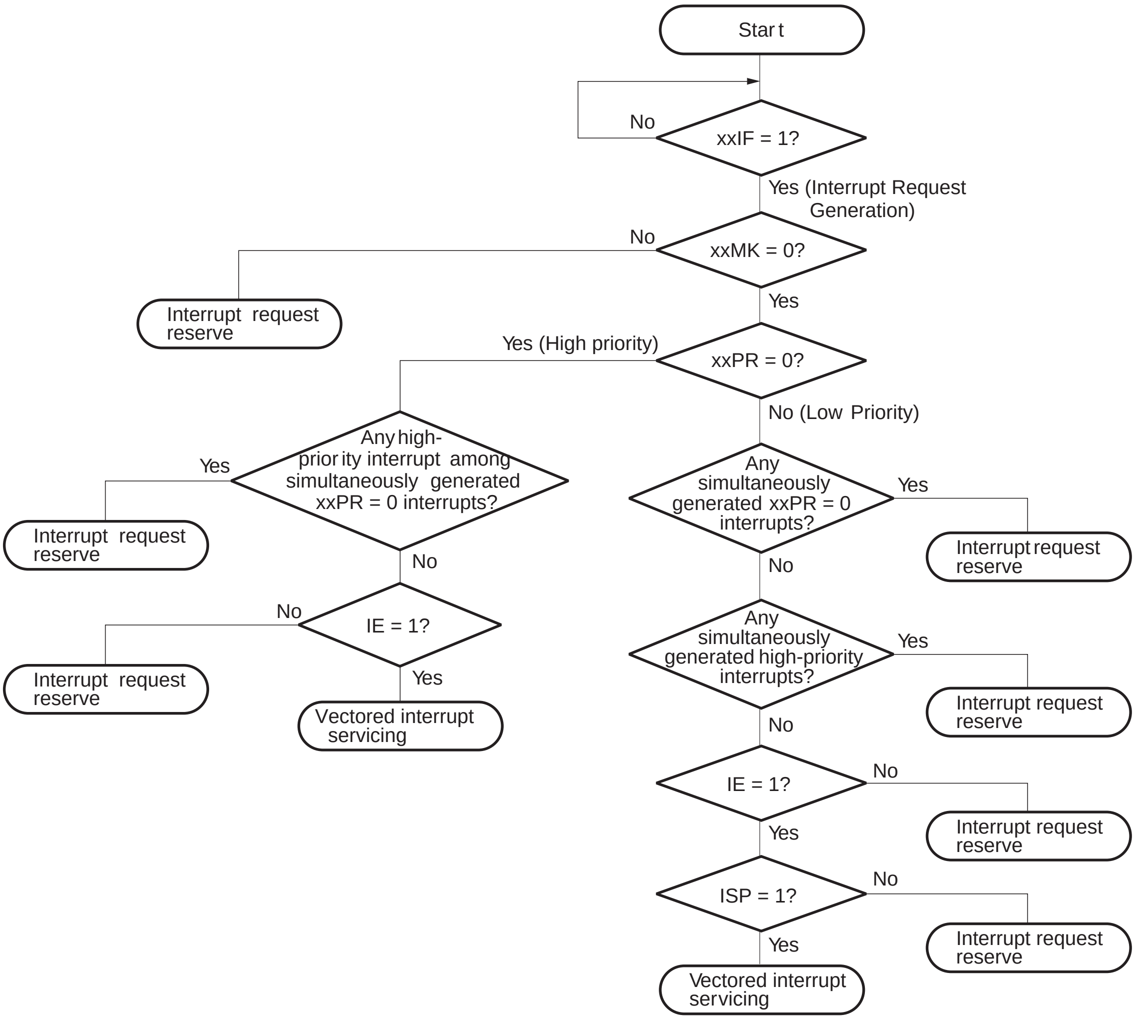
	Minimum Time	Maximum Time[Note]
When xxPRx = 0	7 clocks	32 clocks
When xxPRx = 1	8 clocks	33 clocks

Note: If an interrupt request is generated just before a divide instruction, the wait time is maximized.

Remark: 1 clock: 1/ f_{CPU}
(f_{CPU}: CPU clock)

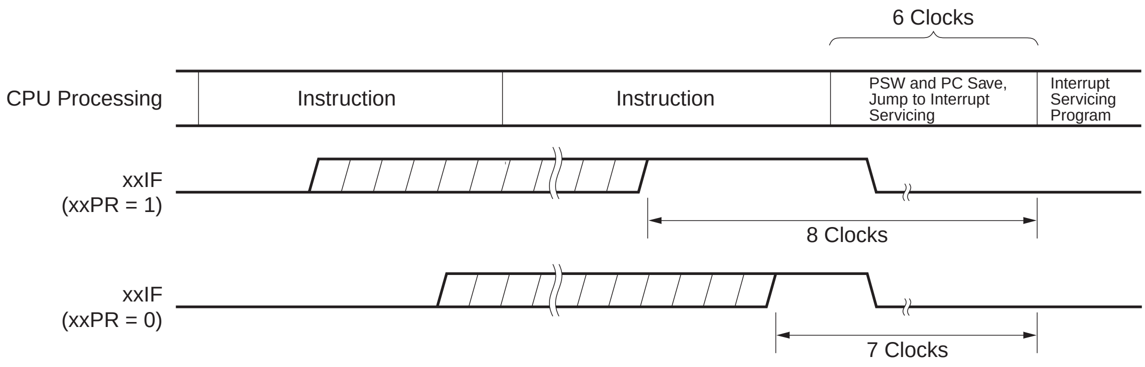
If two or more maskable interrupt requests are generated simultaneously, the request specified for higher priority with the priority specify flag is acknowledged first. If two or more requests are specified for the same priority with the priority specify flag, the interrupt request with the higher default priority is acknowledged first.
Any reserved interrupt requests are acknowledged when they become acknowledgeable.
Figure 18-10 on page 360 shows interrupt request acknowledge algorithms.
When a maskable interrupt request is acknowledged, the contents of program status word (PSW) and program counter (PC) are saved in this order onto the stack. Then, the IE flag is reset (to 0), and the value of the acknowledged interrupt priority specify flag is transferred to the ISP flag. Further, the vector table data determined for each interrupt request is loaded into PC and the program will branch accordingly.
Return from the interrupt is possible with the RETI instruction.

Figure 18-10: Interrupt Request Acknowledge Processing Algorithm


Start
xxIF = 1?
No
Yes (Interrupt Request Generation)
xxMK = 0?
No
Interrupt request reserve
Yes
xxPR = 0?
Yes (High priority)
No (Low Priority)
Any high-priority interrupt among simultaneously generated xxPR = 0 interrupts?
Yes
Interrupt request reserve
No
IE = 1?
No
Interrupt request reserve
Yes
Vectored interrupt servicing
Any simultaneously generated xxPR = 0 interrupts?
Yes
Interrupt request reserve
No
Any simultaneously generated high-priority interrupts?
Yes
Interrupt request reserve
No
IE = 1?
No
Interrupt request reserve
Yes
ISP = 1?
No
Interrupt request reserve
Yes
Vectored interrupt servicing

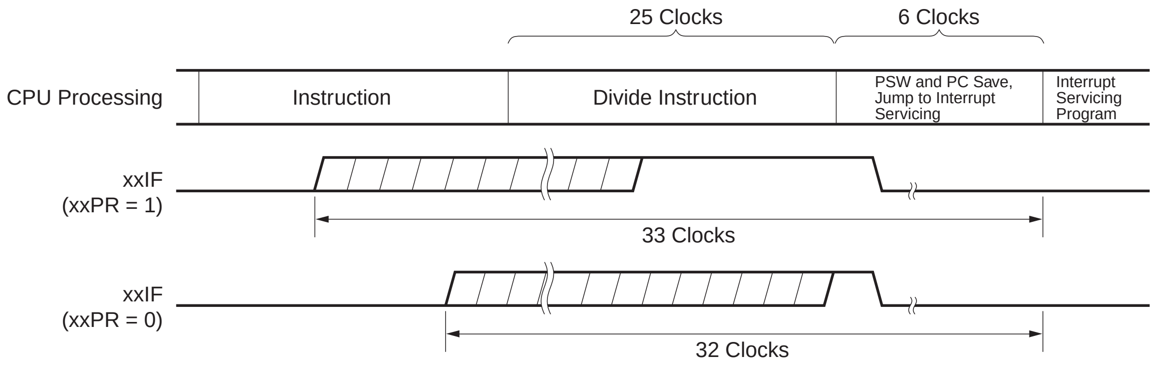
Remark:			
	xxIF	:	Interrupt request flag
	xxMK	:	Interrupt mask flag
	xxPR	:	Priority specify flag
	IE	:	Flag to control maskable interrupt request acknowledge
	ISP	:	Flag to indicate the priority of interrupt being serviced (0 = an interrupt with higher priority is being serviced, 1 = interrupt request is not acknowledged or an interrupt with lower priority is being serviced).

Figure 18-11: Interrupt Request Acknowledge Timing (Minimum Time)



Remark: 1 clock: 1/ f_{CPU}
(f_{CPU}: CPU clock)

Figure 18-12: Interrupt Request Acknowledge Timing (Maximum Time)



Remark: 1 clock: 1/ f_{CPU}
(f_{CPU}: CPU clock)

18.4.3 Software interrupt request acknowledge operation

A software interrupt request is acknowledged by BRK instruction execution. Software interrupt cannot be disabled.
If a software interrupt is acknowledged, the contents of program status word (PSW) and program counter (PC) are saved to stacks, in this order. Then the IE flag is reset (to 0), and the contents of the vector tables (003EH and 003FH) are loaded into PC and the program branches accordingly.
Return from the software interrupt is possible with the RETB instruction.

Caution: Do not use the RETI instruction for returning from the software interrupt.

18.4.4 Multiple interrupt servicing

A multiple interrupt service consists in acknowledging another interrupt during the execution of another interrupt routine.
A multiple interrupt service is generated only in the interrupt request acknowledge enable state (IE = 1) (except non-maskable interrupt). As soon as an interrupt request is acknowledged, it enters the acknowledge disable state (IE = 0). Therefore, in order to enable multiple interrupts, it is necessary to set the interrupt enable state by setting the IE flag (1) with the EI instruction during interrupt servicing.
Even in an interrupt enabled state, a multiple interrupt may not be enabled. However, it is controlled according to the interrupt priority. There are two priorities, the default priority and the programmable priority. The multiple interrupt is controlled by the programmable priority control.
If an interrupt request with the same or higher priority than that of the interrupt being serviced is generated, it is acknowledged as a multiple interrupt. In the case of an interrupt with a priority lower than that of the interrupt being processed, it is not acknowledged as a multiple interrupt.
An interrupt request not acknowledged as a multiple interrupt due to interrupt disable or a low priority is reserved and acknowledged following one instruction execution of the main processing after the completion of the interrupt being serviced.
During non-maskable interrupt servicing, multiple interrupts are not enabled.
Table 18-4 on page 363 shows an interrupt request enabled for multiple interrupt during interrupt servicing, and Figure 18-13 on page 364 shows multiple interrupt examples.

Table 18-4: Interrupt Request Enabled for Multiple Interrupt during Interrupt Servicing

Maskable Interrupt Request Interrupt being serviced		Non-maskable Interrupt Request	Maskable Interrupt Request			
			xxPR = 0		xxPR = 1	
			IE = 1	IE = 0	IE = 1	IE = 0
Non-maskable interrupt		D	D	D	D	D
Maskable Interrupt	ISP = 0	E	E	D	D	D
	ISP = 1	E	E	D	E	D
Software interrupt		E	E	D	E	D

Remarks: 1. E : Multiple interrupt enable

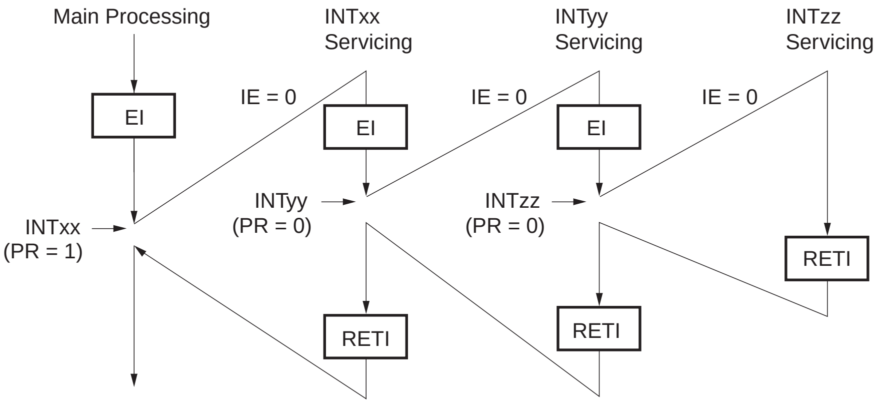
2. D : Multiple interrupt disable

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

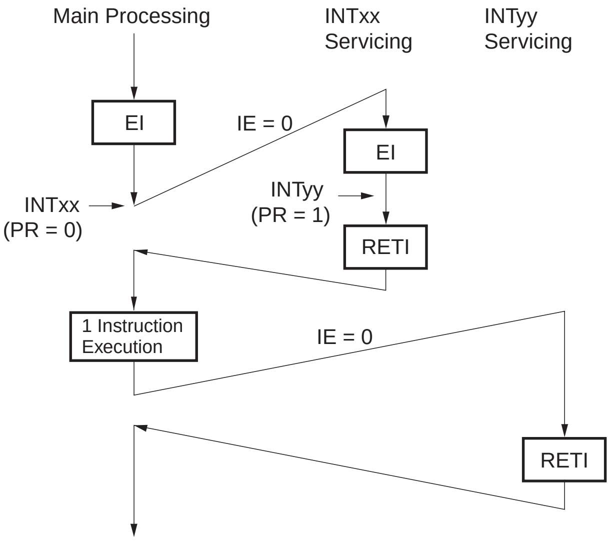
Figure 18-13: Multiple Interrupt Example (1/2)

(a) Example 1. Two multiple interrupts generated



During interrupt INTxx servicing, two interrupt requests, INTyy and INTzz are acknowledged, and a multiple interrupt is generated. An EI instruction is issued before each interrupt request acknowledge, and the interrupt request acknowledge enable state is set.

(b) Example 2. Multiple interrupt is not generated by priority control

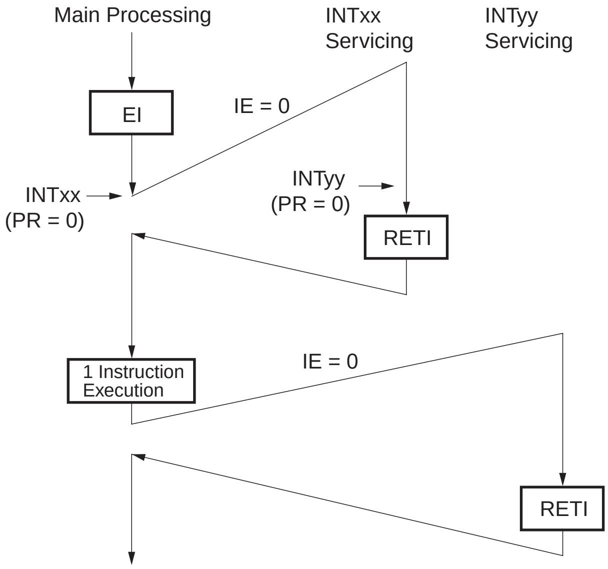


The interrupt request INTyy generated during interrupt INTxx servicing is not acknowledged because the interrupt priority is lower than that of INTxx, and a multiple interrupt is not generated. INTyy request is retained and acknowledged after execution of 1 instruction execution of the main processing.

Remark: PR = 0 : Higher priority level
PR = 1 : Lower priority level
IE = 0 : Interrupt request acknowledge disable

Figure 18-13: Multiple Interrupt Example (2/2)

(c) Example 3. A multiple interrupt is not generated because interrupts are not enabled



Because interrupts are not enabled in interrupt INTxx servicing (an EI instruction is not issued), interrupt request INTyy is not acknowledged, and a multiple interrupt is not generated. The INTyy request is reserved and acknowledged after 1 instruction execution of the main processing.

Remark: PR = 0 : Higher priority level
IE = 0 : Interrupt request acknowledge disable

18.4.5 Interrupt request reserve

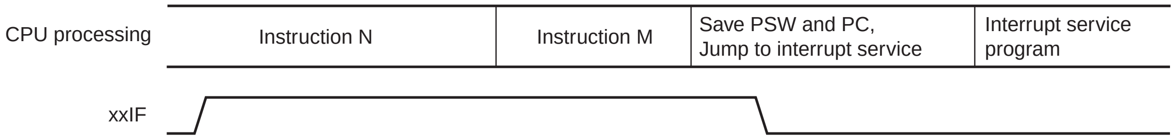
Some instructions may reserve the acknowledge of an instruction request until the completion of the execution of the next instruction even if the interrupt request is generated during the execution.
The following list shows such instructions (interrupt request reserve instruction).

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW.bit, CY
- MOV1 CY, PSW.bit
- AND1 CY, PSW.bit
- OR1 CY, PSW.bit
- XOR CY, PSW.bit
- SET1/CLR1 PSW.bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW.bit, $addr16
- BF PSW.bit, $addr16
- BTCLR PSW.bit, $addr16
- EI
- DI
- Manipulate instructions:
  for IF0L, IF0H, IF1L, MK0L, MK0H, MK1L, PR0L, PR0H, PR1L, EGP, EGN

Caution: BRK instruction is not an interrupt request reserve instruction described above. However, in a software interrupt started by the execution of BRK instruction, the IE flag is cleared to 0. Therefore, interrupt requests are not acknowledged even when a maskable interrupt request is issued during the execution of the BRK instruction. However, non-maskable interrupt requests are acknowledged.

Figure 18-14 shows the interrupt request hold timing.

Figure 18-14: 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 19 Key Return Mode

19.1 Key Return Mode Functions

The Key Return Mode allows it to build up a keyboard by using a detection of a low level at any bit of port 4.

Caution: When the Key Return Mode is enabled, a low level at any bit of port 4 generates a Key Return Interrupt. Port pins that should not generate a Key Return Interrupt can be disabled by switching the respective port pin to output mode.

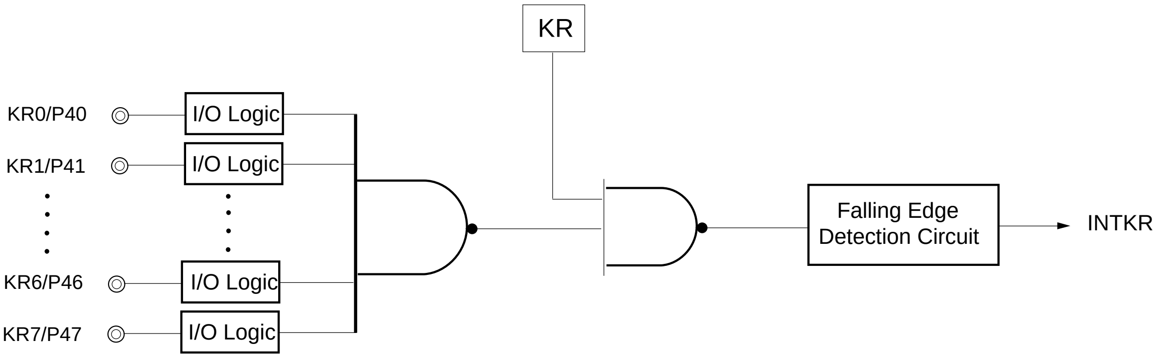
19.2 Key Return Mode Circuit Configuration

The Key Return Mode consists of the following hardware.

Table 19-1: Key Return Mode Configuration

Item	Configuration
Control Register	Key Return Mode register (KRM) Port Mode register 4 (PM4)

Figure 19-1: Key Return Mode Circuit Block Diagram



19.3 Key Return Mode Control Registers

The following two types of registers are used to control the key return mode:

- Key Return Mode Register (KRM)
- Port Mode Register (PM4)

(1) Key return mode register (KRM)

The register enables the key return mode. KRM is set with an 1-bit or an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets KRM to 00H.

Figure 19-2: Key Return Mode Register (KRM) Format

Symbol	7	6	5	4	3	2	1	<0>	Address	After Reset	R/W
KRM	0	0	0	0	0	0	0	KR	FF47H	00H	R/W

KR	Key Return Mode Selection
0	Key return mode disabled
1	Key return mode enabled

Caution: When the Key Return Mode is enabled, a low level at any bit of port 4 generates a Key Return Interrupt. Port pins that should not generate a Key Return Interrupt can be disabled by switching the respective port pin to output mode.

(2) Port mode register 4 (PM4)

This register sets port 4 in input/output mode in 1-bit units. When using port 4 in key return mode, set PM4 to input.

PM4 is set with an 1-bit or an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM4 to FFH.

Figure 19-3: Port Mode Register 4 (PM4) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W

PM4n	P4n Pin Input/Output Mode Selection (n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

[MEMO]

Chapter 20 Standby Function

20.1 Standby Function and Configuration

20.1.1 Standby function

The standby function is designed to decrease the 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 much as in the STOP mode. The HALT mode is capable of 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 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 entering the standby mode are held. The input/output port output latch and output buffer status 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 afterwards.

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.

20.1.2 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 $2^{17}/f_X$ until the STOP mode is cleared by $\overline{\text{RESET}}$ input.

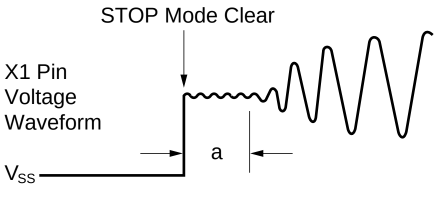
Figure 20-1: Oscillation Stabilization Time Select Register (OSTS) Format

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)
1	1	1	$2^{16}/f_X$ (8.9 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 Figure 20-2 below) from STOP mode clear to clock oscillation start, regardless of clearance by $\overline{\text{RESET}}$ input or by interrupt generation.

Figure 20-2: Standby Timing



Remarks: 1. f_X: Main system clock oscillation frequency

2. Values in parentheses apply to operating at f_X = 8.00 MHz

20.2 Standby Function Operations

20.2.1 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 20-1: HALT Mode Operation 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	Operation stops
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 f_{XT} 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	
External interrupt (INTP0 to INTP3 and INTKR)	Operable	

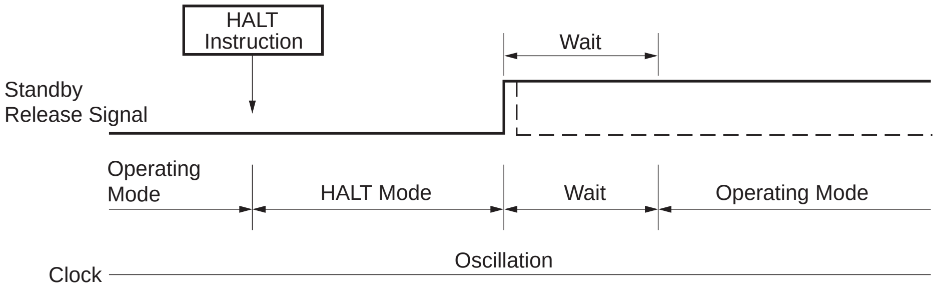
(2) HALT mode clear

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 20-3: 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

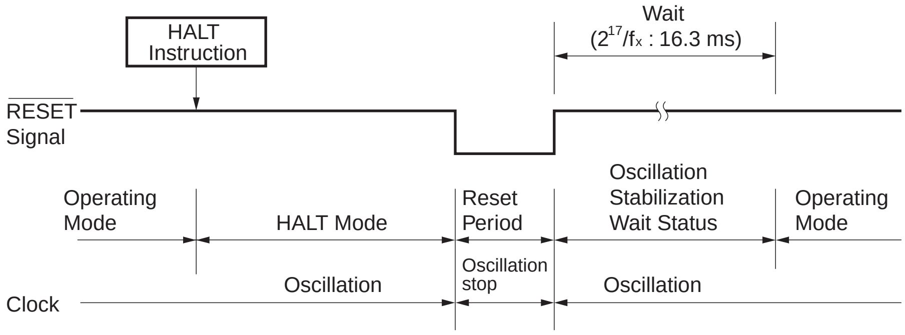
(b) Clear upon non-maskable interrupt request

The HALT mode is cleared and vectored interrupt service is carried out whether interrupt acknowledge is enabled or disabled.

(c) Clear upon $\overline{\text{RESET}}$ input

As is the case with normal reset operation, a program is executed after branch to the reset vector address.

Figure 20-4: HALT Mode Release by $\overline{\text{RESET}}$ Input



Remarks: 1. f_X: Main system clock oscillation frequency

2. Values in parentheses apply to operation at f_X = 8.0 MHz

Table 20-2: 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

Remark: x: Don’t care

20.2.2 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 time 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 20-3: STOP Mode Operating Status

SMOS 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 (TM00)	Operation stops	
16-bit timer (TM2)	Operation stops	
8-bit timer event counter 5 and 6	Operable when TI is selected as count clock	
Watch timer	Operable when f_{XT} 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	
External interrupt (INTP0 to INTP4 and INTKR)	Operable	

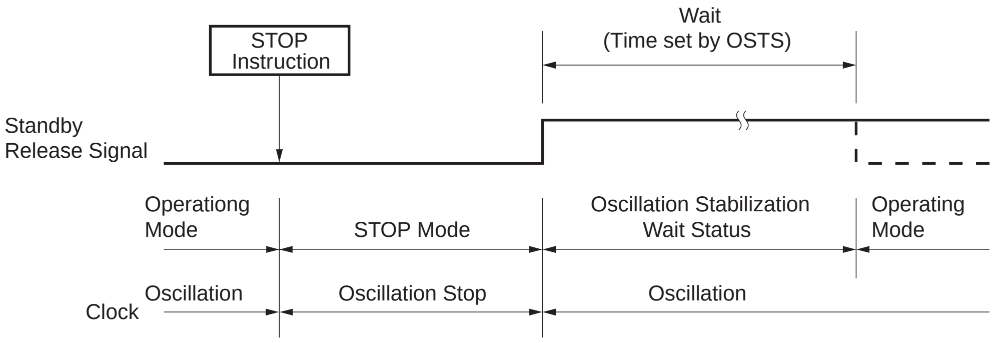
(2) STOP mode release

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 20-5: STOP Mode Release by Interrupt Generation

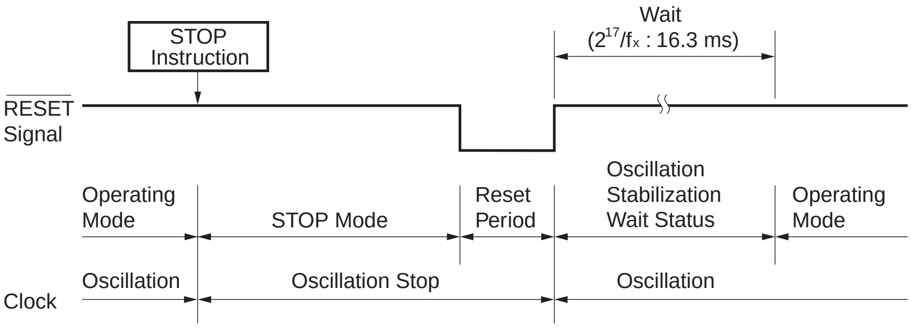


Remark: The broken line indicates the case when the interrupt request which has cleared the standby status is acknowledged.

(b) Release by $\overline{\text{RESET}}$ input

The STOP mode is cleared and after the lapse of oscillation stabilization time, reset operation is carried out.

Figure 20-6: Release by STOP Mode $\overline{\text{RESET}}$ Input



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

2. Values in parentheses apply to operation at f_X = 8.0 MHz

Table 20-4: 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
$\overline{\text{RESET}}$ input	-	-	x	x	Reset processing

Remark: x: Don’t care

Chapter 21 Reset Function

21.1 Reset Function

The following three operations are available to generate the reset signal.

- External reset input with $\overline{\text{RESET}}$ pin
- Internal reset by watchdog timer overrun time detection
- Internal reset by main clock failure 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 21-1. 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 ($2^{17}/f_X$). The reset applied by watchdog timer overflow is automatically cleared after a reset and program execution starts after the lapse of oscillation stabilization time ($2^{17}/f_X$) (see Figure 21-2, “Timing of Reset Input by RESET Input,” on page 382, Figure 21-3, “Timing of Reset due to Watchdog Timer Overflow,” on page 382, and Figure 21-4, “Timing of Reset Input in STOP Mode by RESET Input,” on page 382).

Cautions: 1. For an external reset, apply a low level for 10 µs or more to the $\overline{\text{RESET}}$ pin.

2. During reset the main system clock oscillation remains stopped but the subsystem clock oscillation continues.

3. When the STOP mode is cleared by reset, the STOP mode contents are held during reset. However, the port pin becomes high-impedance.

Figure 21-1: Block Diagram of Reset Function

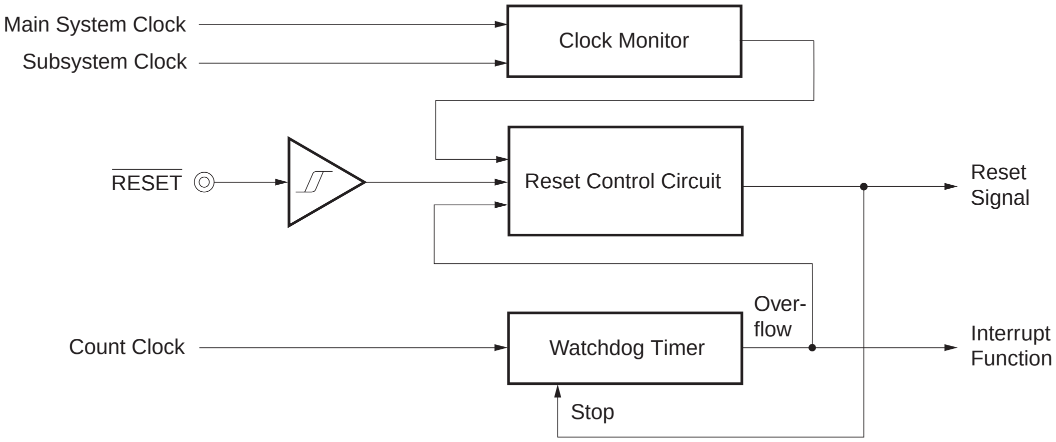


Figure 21-2: Timing of Reset Input by $\overline{\text{RESET}}$ Input

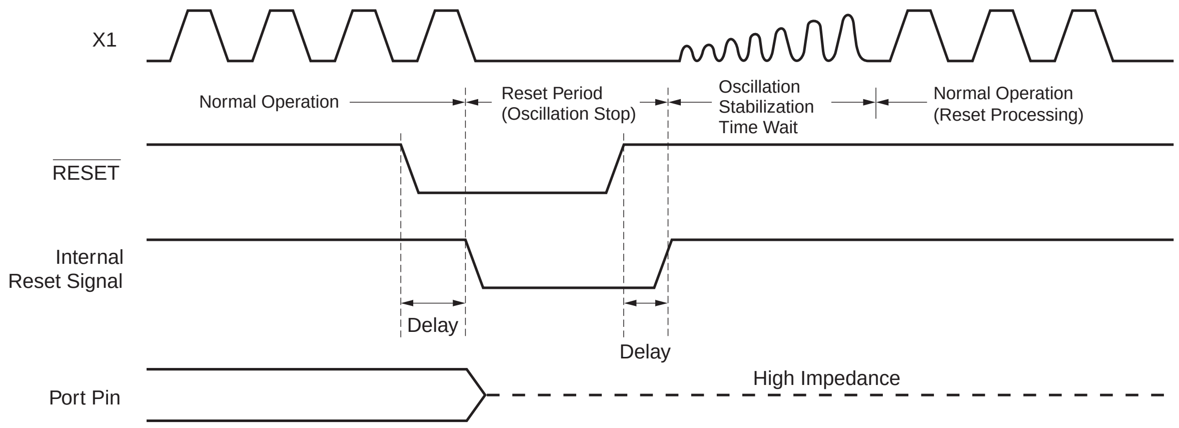


Figure 21-3: Timing of Reset due to Watchdog Timer Overflow

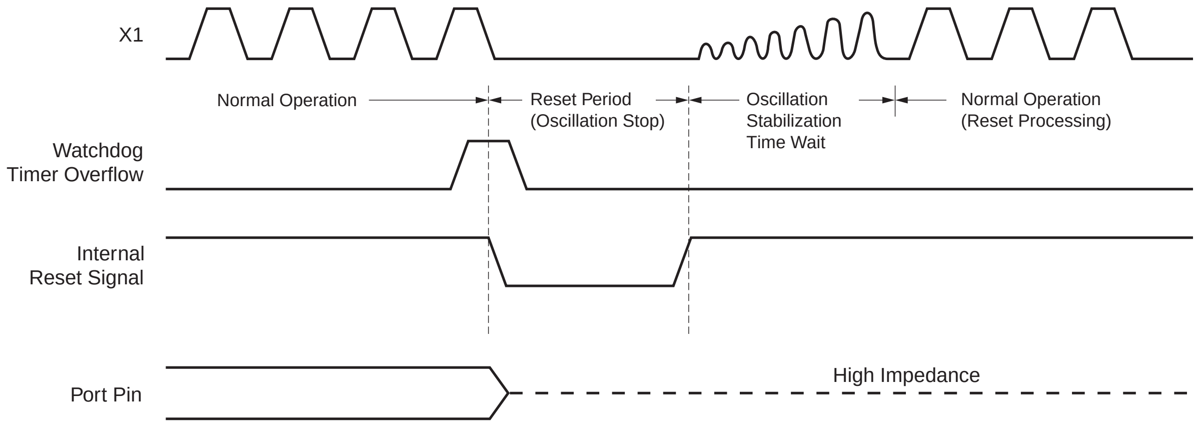


Figure 21-4: Timing of Reset Input in STOP Mode by $\overline{\text{RESET}}$ Input

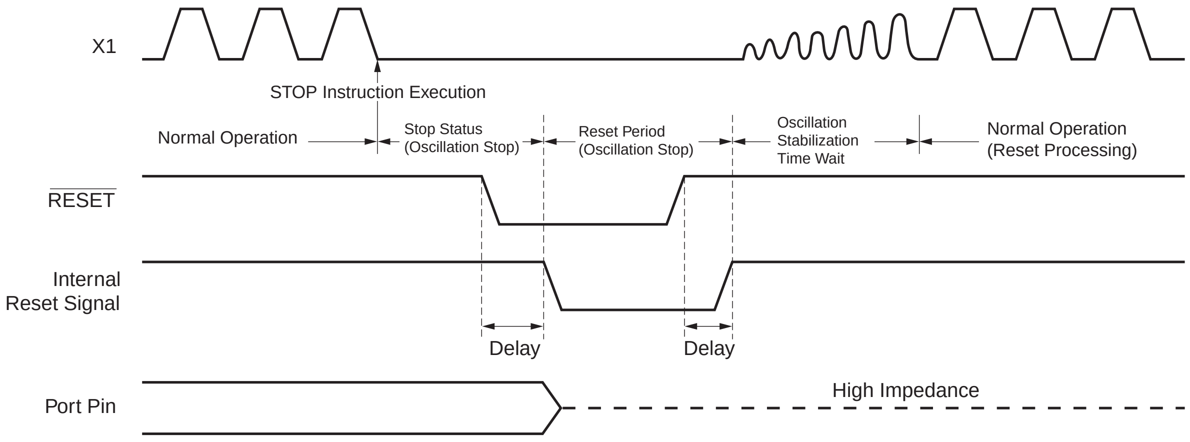


Table 21-1: Hardware Status after Reset (1/2)

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]
	LCD Display Data Memory	Note 4
Port (Output latch)	Ports 0, 2, 4, 5, 6, 7 (P0, P2, P4, P5,P6, P7)	00H
Port mode register (PM0, PM2, PM4, PM5, PM6, PM7)		FFH
Pull-up resistor option register (PU0, PU2, PU4, PU5, PU6, PU7)		00H
Port function selection (PF2)		00H
Processor clock control register (PCC)		04H
Memory size switching register (IMS)		CFH
Internal expansion RAM size switching register (IXS)		Note 3
Oscillation stabilization time select register (OSTS)		04H
16-bit timer/event counter 0	Timer register (TM0)	00H
	Capture/compare register (CR00, CR01)	00H
	Prescaler mode register (PRM0)	00H
	Mode control register (TMC0)	00H
	Capture/compare control register 0 (CRC0)	00H
	Output control register (TOC0)	00H
16-bit timer/event counter 2	Timer register (TM2)	00H
	Capture control register (CR20, CR21, CR22)	00H
	Prescaler mode register (PRM2)	00H
	Mode control register (TMC2)	00H
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)	00H
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

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.

3. The value after RESET depends on the product (see Table 22-4, “Values when the Internal Expansion RAM Size Switching Register is Reset,” on page 387)

Table 21-1: Hardware Status after Reset (2/2)

Hardware		Status after Reset
Serial interface	Operating mode register (CSIM20, CSIM30)	00H
	Shift register (SIO20, SIO30)	00H
	Serial interface switch register (SIOSWI)	00H
	Serial receive data buffer (SIRB20)	00H
	Receive data buffer status (SRBRS20)	00H
	Asynchronous mode register (ASIM0)	00H
	Asynchronous status register (ASIS0)	00H
	Baudrate generator control register (BRGR0)	00H
	Transmit shift register (TXS0)	FFH
	Receive buffer register (RXB0)	
A/D converter	Mode register (ADM1)	00H
	Conversion result register (ADCR1)	00H
	Input select register (ADS1)	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
Flash self-programming	Flash self-programming mode control register (FLPMC)	08H
	Self-programming and oscillation control register (SPOC)	08H
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)	00H
	Bit rate prescaler register (BRPRS)	3FH
	Synchronous control register (SYNC0)	18H
	Synchronous control register (SYNC1)	0EH
	Mark control register (MASKC)	00H
	Counter Register (SMCNT)	00H
	PWM timer control register (MCNTC)	00H
Main clock monitor	Clock monitor mode register (CLM)	00H
Key return input	Key return mode register (KRM)	00H

Chapter 22 µPD78F0818 and Memory Definition

The flash memory version of the µPD780816 Subseries includes the µPD78F0818.

The µPD78F0818 replaces the internal mask ROM of the µPD780816 Subseries with flash memory to which a program can be written, deleted and overwritten while mounted on the substrata.
Table 22-1 lists the differences among the µPD78F0818 and the mask ROM versions.

Table 22-1: Differences among µPD78F0818 and Mask ROM Versions

Item	µPD78F0818	Mask ROM Versions
IC pin	None	Available
V_{PP} pin	Available	None
Electrical characteristics	Please refer to Chapter 24 ”Electrical Specifications” on page 409 of this document.	

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.

22.1 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 CFH.

Caution: When a device of the µPD780816 Subseries is selected, be sure to set the value specified in Table 22-2 to IMS. Other settings are prohibited.

Figure 22-1: 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	CFH	R/W

ROM3	ROM2	ROM1	ROM0	Internal ROM size selection
1	0	0	0	32 K bytes
1	1	0	0	48 K bytes
1	1	1	1	60 K bytes
Other than above				Setting prohibited

RAM2	RAM1	RAM0	Internal high-speed RAM size selection
1	1	0	1024 bytes
Other than above			Setting prohibited

Notes: 1. The values to be set after reset depend on the product (See Table 22-2).

2. Even if the flash version has a memory size of 59.5 K flash memory, the register has to be set to a flash memory size of 60 K.

Table 22-2: Values to be set after Reset of the Memory Size Switching Register

Part Number	Reset Value
µPD780814	C8H
µPD780816	CCH
µPD78F0818	CFH

22.2 Internal Expansion RAM Size Switching Register

The µPD78F0818 allows users to define its internal extension RAM size by using the internal expansion RAM size switching register (IXS), so that the same memory mapping as that of a mask ROM version with a different internal expansion RAM is possible.
The IXS is set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ signal input sets IXS to the value indicated in Table 22-4.

Caution: When a device of the µPD780816 Subseries is selected, be sure to set the value specified in Table 22-3 to IXS. Other settings are prohibited.

Figure 22-2: Internal Expansion RAM Size Switching Register (IXS) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IXS	0	0	0	0	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	**Note 1**	W

IXRAM3	IXRAM2	IXRAM1	IXRAM0	Internal Expansion RAM capacity selection
1	0	1	1	480 bytes
1	0	0	0	2016 bytes
Other than above				Setting prohibited

Notes: 1. The values after Reset depend on the product (see Table 22-4).

2. The value which is set in the IXS that has the identical memory map to the mask ROM versions is given in Table 22-3.

Table 22-3: Examples of internal Expansion RAM Size Switching Register Settings

Relevant Mask ROM Version	IXS Setting
µPD780814	0BH
µPD780816	0BH
µPD78F0818	08H

Table 22-4: Values when the Internal Expansion RAM Size Switching Register is Reset

Part Number	Reset Value
µPD780814	0CH
µPD780816	0CH
µPD78F0818	08H

22.3 Self-Programming and Oscillation Control Register

The µPD78F0818 allows users to reduce the power consumption in HALT mode by a selection of the clock supply of the flash memory.
The SPOC register is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ signal input sets SPOC to 08H.

Figure 22-3: Self-Programming and Oscillation Control Register (SPOC) Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SPOC	0	0	0	0	0	0	HCSEL1	HCSEL0	FF51H	08H	R/W

HCSEL1	HCSEL0	HALT Mode Clock Select
0	0	$f_X/2^4$ (500 kHz)
0	1	$f_X/2^5$ (250 kHz)
1	0	$f_X/2^6$ (125 kHz)
1	1	$f_X/2^7$ (62.5 kHz)

Caution: Be sure to keep bits 2 to 7 = “0”.
After Reset the read value of the SPOC register will be 00H.

22.4 Flash memory programming with flash programmer

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 to the host machine and the target system.
Moreover, writing to flash memory can also be performed using a flash memory writing adapter connected to flash programmer.

22.4.1 Selection of transmission method

Writing to flash memory is performed using flash programmer and serial communication. Select the transmission method for writing from Table 22-5. For the selection of the transmission method, a format like the one shown in Figure 22-4 is used. The transmission methods are selected with the V_{PP} pulse numbers shown in Table 22-5.

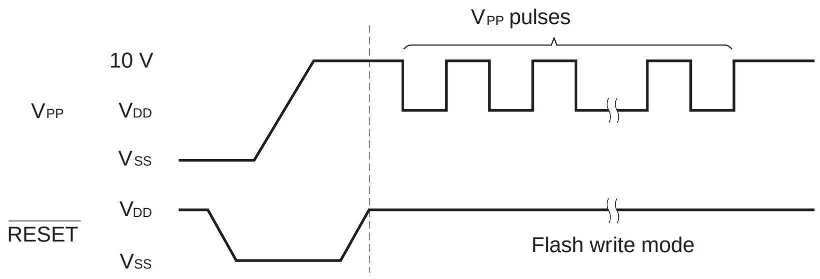
Table 22-5: Transmission Method List

Transmission Method	Number of Channels	Pin Used	Number of V_{PP} Pulses
3-wire serial I/O (SIO30)	1	SI2/P20, SO2/P21, SCK2/P22	0
UART	1	RXD/P24, TXD/P27	8

Cautions: 1. Be sure to select the number of V_{PP} pulses shown in Table 22-5 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 22-4: Transmission Method Selection Format



22.4.2 Initialization of the programming mode

When V_{PP} reaches up to 10 V with $\overline{\text{RESET}}$ terminal activated, on-board programming mode becomes available.
After release of $\overline{\text{RESET}}$, the programming mode is selected by the number of V_{PP} pulses.

22.4.3 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 22-6.

Table 22-6: Main Functions of Flash Memory Programming

Function	Description
Reset	Detects write stop and transmission synchronization
Chip verify	Compares the entire memory contents and input data
Chip internal verify	Compares the entire memory contents internally
Chip blank check	Checks the deletion status of the entire flash memory
High-speed write	Performs writing to the flash memory according to the write start address and the number of write data (bytes)
Continuous write	Performs successive write operations using the data input with high-speed write operation
Chip pre-write	Performs the write operation with 00H to the entire flash memory
Area verify	Compares the entire flash area contents and input data
Area internal verify	Compares the entire flash area contents internally
Area erase	Erases the entire flash area
Area write back	Performs the write back function after the erase of the flash area
Area blank check	Checks the deletion status of the entire flash area
Area pre-write	Performs the write operation with 00H to the entire flash area
Oscillation frequency setting	Inputs the resonator oscillation frequency information
Erase time setting	Defines the flash memory erase time
Baudrate setting	Sets the transmission rate when the UART method is used
Write back time setting	Defines the flash memory write back time
Silicon signature read	Outputs the device name, memory capacity, and device block information

22.4.4 Flash programmer connection

Connection of flash programmer and µPD78F0818 differs depending on communication method (3-wire serial I/O, UART). Each case of connection shows in Figures 22-5 and 22-6.

Figure 22-5: Connection of using the 3-Wire SIO30 Method

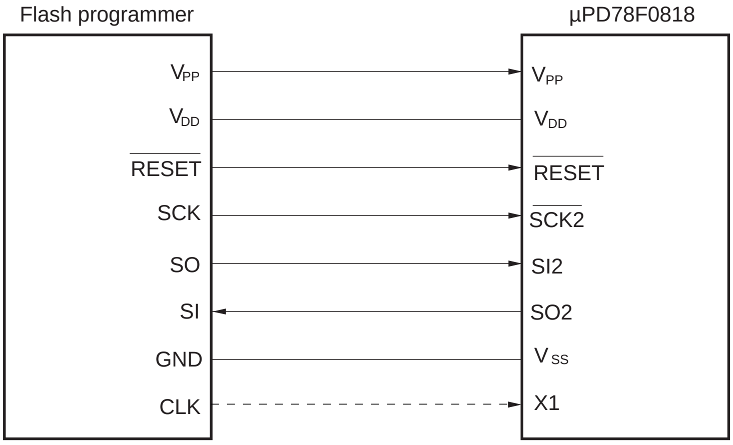
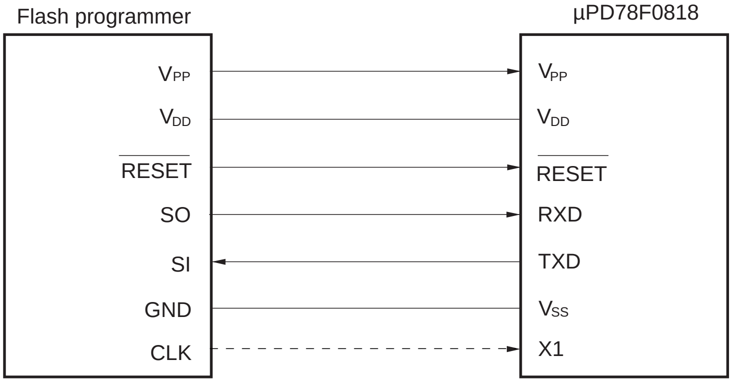


Figure 22-6: Connection of using the UART Method



V_{PP} : Programming voltage applied from the o-board programming tool.

$\overline{\text{RESET}}$: A $\overline{\text{RESET}}$ is generated and the device is set to the on-board programming mode.

System clock : The CPU clock for the device CLK may be supplied by the on-board program tool.
CLK, X1 : Alternatively 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.

V_{DD} : 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 V_{SS}.

$\overline{\text{SCK2}}$: Serial clock generated by the on-board programming tool.

SI2 : Serial data sent by the on-board programming tool.

SO2 : Serial data sent by the device.

RXD : Serial data sent by the on-board programming tool.

TXD : Serial data sent by the device.

22.4.5 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 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.

22.5 Flash Self-Programming Control

The µPD78F0818 provides the secure self-programming with real-time support. further details are provided in an application note (U14995E).

22.5.1 Flash self-programming mode control register (FLPMC)

The flash programming mode control register allows to enable/disable the self-programming mode of the µPD78F0818.
The FLPMC register is set with an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets FLPMC to 08H.

Figure 22-7: Flash Self-Programming Mode Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
FLPMC	0	0	0	0	1	V_{PP}	0	FLSPM0	FF50H	08H	R/W

V_{PP}	Programming Voltage Detected
0	No
1	Yes

FLSPM0	Self-Programming Mode Selection
0	Normal operation mode
1	Self-programming mode

Remark: The bit V_{PP} is a read-only flag.

[MEMO]

Chapter 23 Instruction Set

This chapter describes each instruction set of the µPD780816 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)**.”

23.1 Legends Used in Operation List

23.1.1 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 23-1: 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 3-5, “Special Function Register List,” on page 62.

23.1.2 Description of “operation” column

A	:	A register; 8-bit accumulator
X	:	X register
B	:	B register
C	:	C register
D	:	D register
E	:	E register
H	:	H register
L	:	L register
AX	:	AX register pair; 16-bit accumulator
BC	:	BC register pair
DE	:	DE register pair
HL	:	HL register pair
PC	:	Program counter
SP	:	Stack pointer
PSW	:	Program status word
CY	:	Carry flag
AC	:	Auxiliary carry flag
Z	:	Zero flag
RBS	:	Register bank select flag
IE	:	Interrupt request enable flag
NMIS	:	Non-maskable interrupt servicing flag
( )	:	Memory contents indicated by address or register contents in parentheses
X_H, X_L	:	Higher 8 bits and lower 8 bits of 16-bit register
	:	Logical product (AND)
	:	Logical sum (OR)
	:	Exclusive logical sum (exclusive OR)
——	:	Inverted data
addr16	:	16-bit immediate data or label
jdisp8	:	Signed 8-bit data (displacement value)

25.1.3 Description of “flag operation” column

(Blank)	:	Not affected
0	:	Cleared to 0
1	:	Set to 1
X	:	Set/cleared according to the result
R	:	Previously saved value is restored

23.2 Operation List

Table 23-2: Operation List (1/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	r, #byte	2	4	-	r← byte			
		saddr, #byte	3	6	7	(saddr) ← byte			
		sfr, #byte	3	-	7	str ← byte			
		A, r Note 3	1	2	-	A ← r			
		r, A Note 3	1	2	-	r ← A			
		A, saddr	2	4	5	A ←(saddr)			
		saddr, A	2	4	5	(saddr) ← A			
		A, sfr	2	-	5	A ← sfr			
		sfr, A	2	-	5	sfr ← A			
		A, !addr16	3	8	9 + n	A ← (addr16)			
		!addr16, A	3	8	9 + m	(addr16) ← A			
		PSW, #byte	3	-	7	PSW ← byte	×	×	×
		A, PSW	2	-	5	A ← PSW			
		PSW, A	2	-	5	PSW ← A	×	×	×
		A, [DE]	1	4	5 + n	A ← (DE)			
		[DE], A	1	4	5 + m	(DE) ← A			
		A, [HL]	1	4	5 + n	A ← (HL)			
		[HL], A	1	4	5 + m	(HL) ← A			
		A, [HL + byte]	2	8	9 + n	A ← (HL + byte)			
		[HL + byte], A	2	8	9 + m	(HL + byte) ← A			
		A, [HL + B]	1	6	7 + n	A ← (HL + B)			
		[HL + B], A	1	6	7 + m	(HL + B) ← A			
		A, [HL + C]	1	6	7 + n	A ← HL + C)			
		[HL + C], A	1	6	7 + m	(HL + C) ← A			
	XCH	A, r Note 3	1	2	-	A ↔ r			
		A, saddr	2	4	6	A ↔ (saddr)			
		A, sfr	2	-	6	A ↔ (sfr)			
		A, !addr16	3	8	10+n+m	A ↔ (addr16)			
		A, [DE]	1	4	6+n+m	A ↔ (DE)			
		A, [HL]	1	4	6+n+m	A ↔ (HL)			
		A, [HL + byte]	2	8	10+n+m	A ↔ (HL + byte)			
		A, [HL + B]	2	8	10+n+m	A ↔ (HL + B)			
		A, [HL + C]	2	8	10+n+m	A ↔ (HL + C)			

Notes: **1.** When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: **1.** One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (2/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	MOVW	rp, #word	3	6	-	rp ← word			
		saddrp, #word	4	8	10	(saddrp) ← word			
		sfrp, #word	4	-	10	sfrp ← word			
		AX, saddrp	2	6	8	AX ← (saddrp)			
		saddrp, AX	2	6	8	(saddrp) ← AX			
		AX, sfrp	2	-	8	AX ← sfrp			
		sfrp, AX	2	-	8	sfrp ← AX			
		AX, rp **Note 4**	1	4	-	AX ← rp			
		rp, AX **Note 4**	1	4	-	rp ← AX			
		AX, !addr16	3	10	12 + 2n	AX ← (addr16)			
		!addr16, AX	3	10	12 + 2m	(addr16) ← AX			
	XCHW	AX, rp **Note 4**	1	4	-	AX × rp			
8-bit operation	ADD	A, #byte	2	4	-	A, CY ← A + byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	×	×	×
		A, r **Note 3**	2	4	-	A, CY ← A + r	×	×	×
		r, A	2	4	-	r, CY ← r + A	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A + (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C)	×	×	×
	ADDC	A, #byte	2	4	-	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r **Note 3**	2	4	-	A, CY ← A + r + CY	×	×	×
		r, A	2	4	-	r, CY ← r + A + CY	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16) + CY	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A + (HL) + CY	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte) + CY	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B) + CY	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C) + CY	×	×	×

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (3/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	**Note 2**		Z	AC	CY
8-bit operation	SUB	A, #byte	2	4	-	A, CY ← A - byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) - byte	×	×	×
		A, r **Note 3**	2	4	-	A, CY ← A - r	×	×	×
		r, A	2	4	-	r, CY ← r - A	×	×	×
		A, saddr	2	4	5	A, CY ← A - (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A - (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A - (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A - (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A - (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A - (HL + C)	×	×	×
	SUBC	A, #byte	2	4	-	A, CY ← A - byte - CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) - byte - CY	×	×	×
		A, r **Note 3**	2	4	-	A, CY ← A - r - CY	×	×	×
		r, A	2	4	-	r, CY ← r - A - CY	×	×	×
		A, saddr	2	4	5	A, CY ← A - (saddr) - CY	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A - (addr16) - CY	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A - (HL) - CY	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A - (HL + byte) - CY	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A - (HL + B) - CY	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A - (HL + C) - CY	×	×	×
	AND	A, #byte	2	4	-	A ← A ∧ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	×		
		A, r **Note 3**	2	4	-	A ← A ∧ r	×		
		r, A	2	4	-	r ← r ∧ A	×		
		A, saddr	2	4	5	A ← A ∧ (saddr)	×		
		A, !addr16	3	8	9 + n	A ← A ∧ (addr16)	×		
		A, [HL]	1	4	5 + n	A ← A ∧ (HL)	×		
		A, [HL + byte]	2	8	9 + n	A ← A ∧ (HL + byte)	×		
		A, [HL + B]	2	8	9 + n	A ← A ∧ (HL + B)	×		
		A, [HL + C]	2	8	9 + n	A ← A ∧ (HL + C)	×		

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (4/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	OR	A, #byte	2	4	-	A ← A ∨ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∨ byte	×		
		A, r Note 3	2	4	-	A ← A ∨ r	×		
		r, A	2	4	-	r ← r ∨ A	×		
		A, saddr	2	4	5	A ← A ∨ (saddr)	×		
		A, !addr16	3	8	9 + n	A ← A ∨ (addr16)	×		
		A, [HL]	1	4	5 + n	A ← A ∨ (HL)	×		
		A, [HL + byte]	2	8	9 + n	A ← A ∨ (HL + byte)	×		
		A, [HL + B]	2	8	9 + n	A ← A ∨ (HL + B)	×		
		A, [HL + C]	2	8	9 + n	A ← A ∨ (HL + C)	×		
	XOR	A, #byte	2	4	-	A ← A ⊻ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ⊻ byte	×		
		A, r Note 3	2	4	-	A ← A ⊻ r	×		
		r, A	2	4	-	r ← r ⊻ A	×		
		A, saddr	2	4	5	A ← A ⊻ (saddr)	×		
		A, !addr16	3	8	9 + n	A ← A ⊻ (addr16)	×		
		A, [HL]	1	4	5 + n	A ← A ⊻ (HL)	×		
		A, [HL + byte]	2	8	9 + n	A ← A ⊻ (HL + byte)	×		
		A, [HL + B]	2	8	9 + n	A ← A ⊻ (HL + B)	×		
		A, [HL + C]	2	8	9 + n	A ← A ⊻ (HL + C)	×		
	CMP	A, #byte	2	4	-	A - byte	×	×	×
		saddr, #byte	3	6	8	(saddr) - byte	×	×	×
		A, r Note 3	2	4	-	AA - r	×	×	×
		r, A	2	4	-	r - A	×	×	×
		A, saddr	2	4	5	A - (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A - (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A - (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A - (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A - (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A - (HL + C)	×	×	×
16-bit operation	ADDW	AX, #word	3	6	-	AX, CY ← AX + word	×	×	×
	SUBW	AX, #word	3	6	-	AX, CY ← AX - word	×	×	×
	CMPW	AX, #word	3	6	-	AX – word	×	×	×
Multiply/ divide	MULU	X	2	16	-	AX ← A x X			
	DIVUW	C	2	25	-	AX (Quotient), C (Remainder) ← AX ÷ C			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (5/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Increment/ decrement	INC	r	1	2	-	r $\leftarrow$ r + 1	×	×	
		saddr	2	4	6	(saddr) $\leftarrow$ (saddr) + 1	×	×	
	DEC	r	1	2	-	r $\leftarrow$ r – 1	×	×	
		saddr	2	4	6	(saddr) $\leftarrow$ (saddr) – 1	×	×	
	INCW	rp	1	4	-	rp $\leftarrow$ rp + 1			
	DECW	rp	1	4	-	rp $\leftarrow$ rp - 1			
Rotate	ROR	A, 1	1	2	-	(CY, $A_7 \leftarrow A_0$, $A_m - 1 \leftarrow A_m$) x 1 time			×
	ROL	A, 1	1	2	-	(CY, $A_0 \leftarrow A_7$, $A_m + 1 \leftarrow A_m$) x 1 time			×
	RORC	A, 1	1	2	-	(CY $\leftarrow A_0$, $A_7 \leftarrow$ CY, $A_m - 1 \leftarrow A_m$) x 1 time			×
	ROLC	A, 1	1	2	-	(CY $\leftarrow A_7$, $A_0 \leftarrow$ CY, $A_m + 1 \leftarrow A_m$) x 1 time			×
	ROR4	[HL]	2	10	12+n+m	$A_{3-0} \leftarrow (HL)_{3-0}$, $(HL)_{7-4} \leftarrow A_{3-0}$, $(HL)_{3-0} \leftarrow (HL)_{7-4}$			
	ROL4	[HL]				$A_{3-0} \leftarrow (HL)_{7-4}$, $(HL)_{3-0} \leftarrow A_{3-0}$, $(HL)_{7-4} \leftarrow (HL)_{3-0}$			
BCD adjust	ADJBA		2	4	-	Decimal Adjust Accumulator after Addition	×	×	×
	ADJBS		2	4	-	Decimal Adjust Accumulator after Subtract	×	×	×
Bit manipulate	MOV1	CY, saddr.bit	3	6	7	CY $\leftarrow$ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY $\leftarrow$ sfr.bit			×
		CY, A.bit	2	4	-	CY $\leftarrow$ A.bit			×
		CY, PSW.bit	3	-	7	CY $\leftarrow$ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY $\leftarrow$ (HL).bit			×
		saddr.bit, CY	3	6	8	(saddr.bit) $\leftarrow$ CY			
		sfr.bit, CY	3	-	8	sfr.bit $\leftarrow$ CY			
		A.bit, CY	2	4	-	A.bit $\leftarrow$ CY			
		PSW.bit, CY	3	-	8	PSW.bit $\leftarrow$ CY	×	×	
		[HL].bit, CY	2	6	8+n+m	(HL).bit $\leftarrow$ CY			
	AND1	CY, saddr.bit	3	6	7	CY $\leftarrow$ CY $\wedge$ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY $\leftarrow$ CY $\wedge$ sfr.bit			×
		CY, A.bit	2	4	-	CY $\leftarrow$ CY $\wedge$ A.bit			×
		CY, PSW.bit	3	-	7	CY $\leftarrow$ CY $\wedge$ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY $\leftarrow$ CY $\wedge$ (HL).bit			×

Notes:

1. When the internal high-speed RAM area is accessed or instruction with no data access
2. When an area except the internal high-speed RAM area is accessed.
3. Except “r = A”
4. Only when rp = BC, DE or HL

Remarks:

1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.
2. This clock cycle applies to internal ROM program.
3. n is the number of waits when external memory expansion area is read from.
4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (6/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	**Note 2**		Z	AC	CY
Bit manipulate	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY ← CY ∨ sfr.bit			×
		CY, A.bit	2	4	-	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	-	7	CY ← CY ∨ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∨ (HL).bit			×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ⊻ saddr.bit)			×
		CY, sfr.bit	3	-	7	CY ← CY ⊻ sfr.bit			×
		CY, A.bit	2	4	-	CY ← CY ⊻ A.bit			×
		CY, PSW.bit	3	-	7	CY ← CY ⊻ PSW.bit			×
		CY, [HL].bit	2	6	7 + n	CY ← CY ⊻ (HL).bit			×
	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1			
		sfr.bit	3	-	8	sfr.bit ← 1			
		A.bit	2	4	-	A.bit ← 1			
		PSW.bit	2	-	6	PSW.bit ← 1	×	×	×
		[HL].bit	2	6	8+n+m	(HL).bit ← 1			
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0			
		sfr.bit	3	-	8	sfr.bit ← 0			
		A.bit	2	4	-	A.bit ← 0			
		PSW.bit	2	-	6	PSW.bit ← 0	×	×	×
		[HL].bit	2	6	8+n+m	(HL).bit ← 0			
	SET1	CY	1	2	-	CY ← 1			1
	CLR1	CY	1	2	-	CY ← 0			0
	NOT1	CY	1	2	-	CY ← $\overline{\text{CY}}$			×

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (7/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Call/return	CALL	!addr16	3	7	-	$(SP - 1) \leftarrow (PC + 3)_H$, $(SP - 2) \leftarrow (PC + 3)_L$, PC ←addr16, SP ←SP – 2			
	CALLF	!addr11	2	5	-	$(SP - 1) \leftarrow (PC + 2)_H$, $(SP - 2) \leftarrow (PC + 2)_L$, $PC_{15-11} \leftarrow 00001$, $PC_{10-0} \leftarrow$ addr11, SP ← SP – 2			
	CALLT	[addr5]	1	6	-	$(SP - 1) \leftarrow (PC + 1)_H$, $(SP - 2) \leftarrow (PC + 1)_L$, $PC_H \leftarrow$ (00000000, addr5 + 1), $PC_L \leftarrow$ (00000000, addr5), SP ← SP – 2			
	BRK		1	6	-	(SP – 1) ← PSW, $(SP - 2) \leftarrow (PC + 1)_H$, $(SP - 3) \leftarrow (PC + 1)_L$, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0			
	RET		1	6	-	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$,SP ← SP + 2			
	RETI		1	6	-	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$, PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R	R	R
	RETB		1	6	-	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R	R	R
Stack manipulate	PUSH	PSW	1	2	-	(SP – 1) ← PSW, SP ← SP – 1			
		rp	1	4	-	$(SP - 1) \leftarrow rp_H$, $(SP - 2) \leftarrow rp_L$, SP ← SP – 2			
	POP	PSW	1	2	-	PSW ← (SP), SP ← SP + 1	R	R	R
		rp	1	4	-	$rp_H \leftarrow (SP + 1)$, $rp_L \leftarrow (SP)$, SP ← SP + 2			
	MOVW	SP, #word	4	-	10	SP ← word			
		SP, AX	2	-	8	SP ← AX			
		AX, SP	2	-	8	AX ← SP			
Uncondi-tional branch	BR	!addr16	3	6	-	PC ← addr16			
		$addr16	2	6	-	PC ← PC + 2 + jdisp8			
		AX	2	8	-	$PC_H \leftarrow$ A, PCL ← X			
Condi-tional branch	BC	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if CY = 1			
	BNC	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if CY = 0			
	BZ	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if Z = 1			
	BNZ	$addr16	2	6	-	PC ← PC + 2 + jdisp8 if Z = 0			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

Table 23-2: Operation List (8/8)

Instruction Group	Mnemonic	Operands	Byte	Clock		Operation	Flag		
				Note 1	**Note 2**		Z	AC	CY
Conditional branch	BT	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if(saddr.bit) = 1			
		sfr.bit, $addr16	4	-	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1			
		A.bit, $addr16	3	8	-	PC ← PC + 3 + jdisp8 if A.bit = 1			
		PSW.bit, $addr16	3	-	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1			
		[HL].bit, $addr16	3	10	11 + n	PC ← PC + 3 + jdisp8 if (HL).bit = 1			
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if(saddr.bit) = 0			
		sfr.bit, $addr16	4	-	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0			
		A.bit, $addr16	3	8	-	PC ← PC + 3 + jdisp8 if A.bit = 0			
		PSW.bit, $addr16	4	-	11	PC ← PC + 4 + jdisp8 if PSW. bit = 0			
		[HL].bit, $addr16	3	10	11 + n	PC ← PC + 3 + jdisp8 if (HL).bit = 0			
	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if(saddr.bit) = 1 then reset(saddr.bit)			
		sfr.bit, $addr16	4	-	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit			
		A.bit, $addr16	3	8	-	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit			
		PSW.bit, $addr16	4	-	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	×	×	×
		[HL].bit, $addr16	3	10	12+n+m	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit			
	DBNZ	B, $addr16	2	6	-	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0			
		C, $addr16	2	6	-	C ← C –1, then PC ← PC + 2 + jdisp8 if C ≠ 0			
		saddr. $addr16	3	8	10	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if(saddr) ≠ 0			
CPU control	SEL	RBn	2	4	-	RBS1, 0 ← n			
	NOP		1	2	-	No Operation			
	EI		2	-	6	IE ← 1(Enable Interrupt)			
	DI		2	-	6	IE ← 0(Disable Interrupt)			
	HALT		2	6	-	Set HALT Mode			
	STOP		2	6	-	Set STOP Mode			

Notes: 1. When the internal high-speed RAM area is accessed or instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except “r = A”

4. Only when rp = BC, DE or HL

Remarks: 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to internal ROM program.

3. n is the number of waits when external memory expansion area is read from.

4. m is the number of waits when external memory expansion area is written to.

23.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

Table 23-3: 8-bit instructions

2nd Operand 1st Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte] [HL + B] [HL + C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP										DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL + byte] [HL + B] [HL + C]		MOV											
X													MULU
C													DIVUW

Note: Except r = A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

Table 23-4: 16-bit instructions

2nd Operand 1st Operand	#word	AX	rp[Note]	sfrp	saddrp	!addr16	sp	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
sp	MOVW	MOVW						

Note: Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

Table 23-5: Bit manipulation instructions

2nd Operand 1st Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call/instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

Table 23-6: Call/instructions/branch instructions

2nd Operand 1st Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

[MEMO]

Chapter 24 Electrical Specifications

24.1 Absolute Maximum Ratings

(1) µPD780814(A), µPD780816(A), µPD78F0818(A)

(T_A = 25°C)

Parameter	Symbol	Conditions		Rating	Unit
Supply voltage	V_{DD}			-0.3 to + 6.0	V
	V_{PP}	µPD78F0818(A)		-0.3 to + 11.0	
	AV_{DD}/ AV_{REF}	AV_{DD}/ AV_{REF} = V_{DD}		-0.3 to V_{DD} + 0.3	
	AV_{SS}			-0.3 to + 0.3	
Input voltage	V_{I1}	P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, X1, X2, CL1, $\overline{\text{RESET}}$, CRXD		-0.3 to V_{DD} +0.3	
Output voltage	V_O			-0.3 to V_{DD} +0.3	
Analog input voltage	V_{AN}	P10 to P17, ANI8 to ANI11	Analog input pin	AV_{SS} -0.3 to AV_{DD}+0.3	
High level output current	I_{OH}	1 pin		-10	mA
		P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD total		-20	
Low level output current	I_{OL} Note	1 pin	Peak	20	
			Effective	10	
		P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD total	Peak	50	
			Effective	25	
Operating ambient temperature	T_{OPT}			-40 to +85	°C
Storage temperature	TSTG	µPD780814(A) and µPD780816(A)		-65 to +150	
		µPD78F0818(A)		-40 to +125	

Note: Effective value should be calculated as follows: [Effective value] = [Peak value] × $\sqrt{\text{duty}}$

Caution: Product quality may suffer if the absolute maximum ratings are exceeded for even a single parameter or even momentarily. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions which ensure that the absolute maximum ratings are not exceeded.

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

(2) μPD780814(A1), μPD780816(A1)
(T_A = 25°C)

These specifications are only target values and may not be satisfied by mass-produced products.

Parameter	Symbol	Conditions		Rating	Unit
Supply voltage	V_{DD}			-0.3 to + 6.0	V
	AV_{DD}/ AV_{REF}	AV_{DD}/ AV_{REF} = V_{DD}		-0.3 to V_{DD} + 0.3	
	AV_{SS}			-0.3 to + 0.3	
Input voltage	V_{I1}	P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, X1, X2, CL1, $\overline{\text{RESET}}$, CRXD		-0.3 to V_{DD} +0.3	
Output voltage	V_O			-0.3 to V_{DD} +0.3	
Analog input voltage	V_{AN}	P10 to P17, ANI8 to ANI11	Analog input pin	AV_{SS} -0.3 to AV_{DD}+0.3	
High level output current	I_{OH}	1 pin		-10	mA
		P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD total		-20	
Low level output current	I_{OL} Note	1 pin	Peak	20	
			Effective	10	
		P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD total	Peak	50	
			Effective	25	
Operating ambient temperature	T_{OPT}			-40 to +110	°C
Storage temperature	TSTG			-65 to +150	

Note: Effective value should be calculated as follows: [Effective value] = [Peak value] × $\sqrt{\text{duty}}$

Caution: Product quality may suffer if the absolute maximum ratings are exceeded for even a single parameter or even momentarily. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions which ensure that the absolute maximum ratings are not exceeded.

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

(3) µPD780814(A2), µPD780816(A2)
(T_A = 25°C)

Parameter	Symbol	Conditions		Rating	Unit
Supply voltage	V_{DD}			-0.3 to + 6.0	V
	AV_{DD}/ AV_{REF}	AV_{DD}/ AV_{REF} = V_{DD}		-0.3 to V_{DD} + 0.3	
	AV_{SS}			-0.3 to + 0.3	
Input voltage	V_{I1}	P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, X1, X2, CL1, $\overline{\text{RESET}}$, CRXD		-0.3 to V_{DD} +0.3	
Output voltage	V_O			-0.3 to V_{DD} +0.3	
Analog input voltage	V_{AN}	P10 to P17, ANI8 to ANI11	Analog input pin	AV_{SS} -0.3 to AV_{DD}+0.3	
High level output current	I_{OH}	1 pin		-10	mA
		P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD total		-20	
Low level output current	I_{OL}Note	1 pin	Peak	20	
			Effective	10	
		P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD total	Peak	50	
			Effective	25	
Operating ambient temperature	T_{OPT}			-40 to +125	°C
Storage temperature	TSTG			-65 to +150	

Note: Effective value should be calculated as follows: [Effective value] = [Peak value] × $\sqrt{\text{duty}}$

Caution: **Product quality may suffer if the absolute maximum ratings are exceeded for even a single parameter or even momentarily. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions which ensure that the absolute maximum ratings are not exceeded.**

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

24.2 Capacitance

(1) µPD780814(A), µPD780816(A), µPD78F0818(A)
(T_A = 25°C, V_{DD} = V_{SS} = 0 V)

Parameter	Symbol	Function		Min.	Typ.	Max.	Unit
Input capacitance	C_{IN}	f = 1 MHz Other than measured pins: 0 V				15	pF
Input/output capacitance	C_{IO}	f = 1 MHz Other than measured pins: 0 V	P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71			15	pF

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

(2) µPD780814(A1), µPD780816(A1)
(T_A = 25°C, V_{DD} = V_{SS} = 0 V)

These specifications are only target values and may not be satisfied by mass-produced products.

Parameter	Symbol	Function		Min.	Typ.	Max.	Unit
Input capacitance	C_{IN}	f = 1 MHz Other than measured pins: 0 V				15	pF
Input/output capacitance	C_{IO}	f = 1 MHz Other than measured pins: 0 V	P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71			15	pF

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

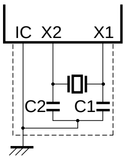
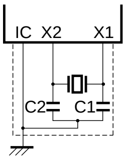
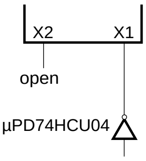
(3) µPD780814(A2), µPD780816(A2)
(T_A = 25°C, V_{DD} = V_{SS} = 0 V)

Parameter	Symbol	Function		Min.	Typ.	Max.	Unit
Input capacitance	C_{IN}	f = 1 MHz Other than measured pins: 0 V				15	pF
Input/output capacitance	C_{IO}	f = 1 MHz Other than measured pins: 0 V	P00 - P03, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71			15	pF

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

24.3 Main System Clock Oscillation Circuit Characteristics

(1) μPD780814(A), μPD780816(A), μPD78F0818(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		Oscillator frequency (f_X) **Note 1**	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		Oscillation stabilization time **Note 2**	After V_{DD} reaches oscillator voltage range MIN. 4.0 V			10	ms
Crystal resonator		Oscillator frequency (f_X) **Note 1**	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		Oscillation stabilization time **Note 2**	After V_{DD} reaches oscillator voltage range MIN. 4.0 V			10	ms
External clock		X1 input frequency (f_X) **Note 1**	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		X1 input high/low-level width (t_{XH}, t_{XL})	V_{DD} = 4.0 to 5.5 V	55		125	ns

Notes: 1. Indicates only oscillation circuit characteristics. Refer to “AC Characteristics” for instruction execution time.

2. Time required to stabilize oscillation after reset or STOP mode release.

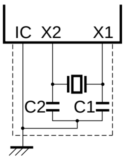
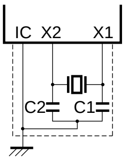
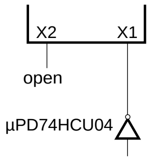
Cautions: 1. When using the main system clock oscillation circuit, wiring in the area enclosed with the broken line should be carried out as follows to avoid an adverse effect from wiring capacitance.

- **Wiring should be as short as possible.**
- **Wiring should not cross other signal lines.**
- **Wiring should not be placed close to a varying high current.**
- **The potential of the oscillation circuit capacitor ground should always be the same as that of V_{SS}.**
- **Do not ground wiring to a ground pattern in which a high current flows.**
- **Do not fetch a signal from the oscillation circuit.**

2. When the main system clock is stopped and the system is operated by the subsystem clock, the subsystem clock should be switched again to the main system clock after the oscillation stabilization time is secured by the program.

(2) μPD780814(A1), μPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V)

These specifications are only target values and may not be satisfied by mass-produced products.

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		Oscillator frequency (f_X) Note 1	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		Oscillation stabilization time Note 2	After V_{DD} reaches oscillator voltage range MIN. 4.0 V			10	ms
Crystal resonator		Oscillator frequency (f_X) Note 1	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		Oscillation stabilization time Note 2	After V_{DD} reaches oscillator voltage range MIN. 4.0 V			10	ms
External clock		X1 input frequency (f_X) Note 1	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		X1 input high/low-level width (t_{XH}, t_{XL})	V_{DD} = 4.0 to 5.5 V	55		125	ns

Notes: 1. Indicates only oscillation circuit characteristics. Refer to “AC Characteristics” for instruction execution time.

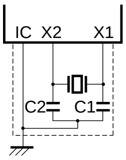
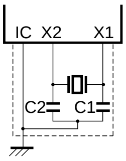
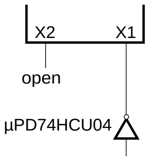
2. Time required to stabilize oscillation after reset or STOP mode release.

Cautions: 1. When using the main system clock oscillation circuit, wiring in the area enclosed with the broken line should be carried out as follows to avoid an adverse effect from wiring capacitance.

- **Wiring should be as short as possible.**
- **Wiring should not cross other signal lines.**
- **Wiring should not be placed close to a varying high current.**
- **The potential of the oscillation circuit capacitor ground should always be the same as that of V_{SS}.**
- **Do not ground wiring to a ground pattern in which a high current flows.**
- **Do not fetch a signal from the oscillation circuit.**

2. When the main system clock is stopped and the system is operated by the subsystem clock, the subsystem clock should be switched again to the main system clock after the oscillation stabilization time is secured by the program.

(3) μPD780814(A2), μPD780816(A2)
(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		Oscillator frequency (f_X) Note 1	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		Oscillation stabilization time Note 2	After V_{DD} reaches oscillator voltage range MIN. 4.0 V			10	ms
Crystal resonator		Oscillator frequency (f_X) Note 1	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		Oscillation stabilization time Note 2	After V_{DD} reaches oscillator voltage range MIN. 4.0 V			10	ms
External clock		X1 input frequency (f_X) Note 1	V_{DD} = 4.0 to 5.5 V	4.0	8.0	8.38	MHz
		X1 input high/low-level width (t_{XH}, t_{XL})	V_{DD} = 4.0 to 5.5 V	55		125	ns

Notes: 1. Indicates only oscillation circuit characteristics. Refer to “AC Characteristics” for instruction execution time.

2. Time required to stabilize oscillation after reset or STOP mode release.

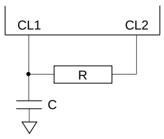
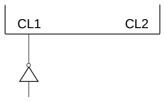
Cautions: 1. When using the main system clock oscillation circuit, wiring in the area enclosed with the broken line should be carried out as follows to avoid an adverse effect from wiring capacitance.

- **Wiring should be as short as possible.**
- **Wiring should not cross other signal lines.**
- **Wiring should not be placed close to a varying high current.**
- **The potential of the oscillation circuit capacitor ground should always be the same as that of V_{SS}.**
- **Do not ground wiring to a ground pattern in which a high current flows.**
- **Do not fetch a signal from the oscillation circuit.**

2. When the main system clock is stopped and the system is operated by the subsystem clock, the subsystem clock should be switched again to the main system clock after the oscillation stabilization time is secured by the program.

24.4 Subsystem Clock Oscillation Circuit Characteristics

(1) µPD780814(A), µPD780816(A), µPD78F0818(A) (T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended circuit	Parameter	Test Conditions	MIN.	TYP.	MAX.	Unit
RC osc.Note 1		Oscillator frequency (f_{XT})	4.0 V ≤ V_{DD} ≤ 5.5 V R = 300 KΩ C = 33 pF	32	40	100	kHz
External clock Note 1		CL1 Input Note 2 frequency (f_{XT})	4.0 V ≤ V_{DD} ≤ 5.5 V	0.032	8.0	8.38	MHz
		CL1 Input high/low level width (t_{XTH}, t_{XTL})	4.0 V ≤ V_{DD} ≤ 5.5 V	0.055		15.6	µs

Notes: 1. Only oscillator circuit characteristics are shown. Regarding instruction execute time, please refer to AC characteristics.

2. The input frequency of 8.00 MHz to CL1 is only valid as frequency input to the DCAN.

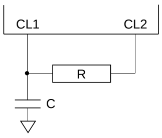
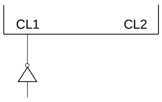
Cautions: 1. When using the subsystem clock oscillation circuit, wiring in the area enclosed with the broken line should be carried out as follows to avoid an adverse effect from wiring capacitance.

- **Wiring should be as short as possible.**
- **Wiring should not cross other signal lines.**
- **Wiring should not be placed close to a varying high current.**
- **The potential of the oscillation circuit capacitor ground should always be the same as that of V_{SS}.**
- **Do not ground wiring to a ground pattern in which a high current flows.**
- **Do not fetch a signal from the oscillation circuit.**

2. The subsystem clock oscillation circuit is designed to be a circuit with a low amplification level, for low power consumption more prone to mis-operation due to noise than that of the main system clock. Therefore, when using the subsystem clock, take special cautions for wiring methods.

(2) μPD780814(A1), μPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V)

These specifications are only target values and may not be satisfied by mass-produced products.

Resonator	Recommended circuit	Parameter	Test Conditions	MIN.	TYP.	MAX.	Unit
RC osc.[Note 1]		Oscillator frequency (f_{XT})	4.0 V ≤ V_{DD} ≤ 5.5 V R = 300 KΩ C = 33 pF	32	40	100	kHz
External clock [Note 1]		CL1 Input [Note 2] frequency (f_{XT})	4.0 V ≤ V_{DD} ≤ 5.5 V	0.032	8.0	8.38	MHz
		CL1 Input high/low level width (t_{XTH}, t_{XTL})	4.0 V ≤ V_{DD} ≤ 5.5 V	0.055		15.6	μs

Notes: 1. Only oscillator circuit characteristics are shown. Regarding instruction execute time, please refer to AC characteristics.

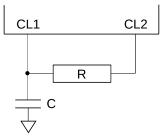
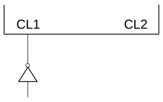
2. The input frequency of 8.00 MHz to CL1 is only valid as frequency input to the DCAN.

Cautions: 1. When using the subsystem clock oscillation circuit, wiring in the area enclosed with the broken line should be carried out as follows to avoid an adverse effect from wiring capacitance.

- **Wiring should be as short as possible.**
- **Wiring should not cross other signal lines.**
- **Wiring should not be placed close to a varying high current.**
- **The potential of the oscillation circuit capacitor ground should always be the same as that of V_{SS}.**
- **Do not ground wiring to a ground pattern in which a high current flows.**
- **Do not fetch a signal from the oscillation circuit.**

2. The subsystem clock oscillation circuit is designed to be a circuit with a low amplification level, for low power consumption more prone to mis-operation due to noise than that of the main system clock. Therefore, when using the subsystem clock, take special cautions for wiring methods.

(3) µPD780814(A2), µPD780816(A2)
(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended circuit	Parameter	Test Conditions	MIN.	TYP.	MAX.	Unit
RC osc.Note 1		Oscillator frequency (f_{XT})	4.0 V ≤ V_{DD} ≤ 5.5 V R = 300 KΩ C = 33 pF	32	40	100	kHz
External clock Note 1		CL1 Input Note 2 frequency (f_{XT})	4.0 V ≤ V_{DD} ≤ 5.5 V	0.032	8.0	8.38	MHz
		CL1 Input high/low level width (t_{XTH}, t_{XTL})	4.0 V ≤ V_{DD} ≤ 5.5 V	0.055		15.6	µs

Notes: 1. Only oscillator circuit characteristics are shown. Regarding instruction execute time, please refer to AC characteristics.

2. The input frequency of 8.00 MHz to CL1 is only valid as frequency input to the DCAN.

Cautions: 1. When using the subsystem clock oscillation circuit, wiring in the area enclosed with the broken line should be carried out as follows to avoid an adverse effect from wiring capacitance.

- **Wiring should be as short as possible.**
- **Wiring should not cross other signal lines.**
- **Wiring should not be placed close to a varying high current.**
- **The potential of the oscillation circuit capacitor ground should always be the same as that of V_{SS}.**
- **Do not ground wiring to a ground pattern in which a high current flows.**
- **Do not fetch a signal from the oscillation circuit.**

2. The subsystem clock oscillation circuit is designed to be a circuit with a low amplification level, for low power consumption more prone to mis-operation due to noise than that of the main system clock. Therefore, when using the subsystem clock, take special cautions for wiring methods.

24.5 DC Characteristics

(1) µPD780814(A), µPD780816(A), µPD78F0818(A)

(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
High-level input voltage	V_{IH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CRXD, ANI0 - ANI11		0.7 V_{DD}		V_{DD}	V
	V_{IH2}	$\overline{\text{RESET}}$		0.8 V_{DD}		V_{DD}	
	V_{IH4}	X1, X2, CL1		V_{DD} - 0.5		V_{DD}	
Low-level input voltage	V_{IL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CRXD, ANI0 - ANI11		0		0.3 V_{DD}	
	V_{IL2}	$\overline{\text{RESET}}$		0		0.2 V_{DD}	
	V_{IL4}	X1, X2, CL1		0		0.4	
High-level output voltage	V_{OH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD	V_{DD} = 4.0 - 5.5 V I_{OH} = -1 mA	V_{DD} - 1.0		V_{DD}	
Low-level output voltage	V_{OL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD	V_{DD} = 4.0 - 5.5 V I_{OL} = 1.6 mA	0		0.4	
High-level input leakage current	I_{LIH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, $\overline{\text{RESET}}$, CRXD, ANI0 - ANI11	V_{IN} = V_{DD}	0		3	µA
	I_{LIH2}	X1, X2, CL1, CL2		0		20	
Low-level input leakage current	I_{LIL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, $\overline{\text{RESET}}$, CRXD, ANI0 - ANI11	V_{IN} = 0 V	0		-3	
	II_{LIL2}	X1, X2, CL1, CL2		0		-20	
High-level output leakage current	I_{LOH}	V_{OUT} = V_{DD}		0		3	
Low-level output leakage current	I_{LOL}	V_{OUT} = 0 V		0		-3	
Software pull-up resistor	R2	V_{IN} = 0 V	4.5 V ≤ V_{DD} ≤ 5.5 V	10	30	100	KΩ

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

(2) µPD780814(A), µPD780816(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Mask ROM Version

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Power supply current Note 1	I_{DD1}	f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 2		5.5	11	mA
		f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 3		9.5	19	
	I_{DD3}	RC oscillation operating mode (f_{XT} = 40 kHz)		150	500	µA
	I_{DD4}	RC oscillation HALT mode (f_{XT} = 40 kHz)		60	180	
	I_{DD5}	CL1 = V_{DD} STOP mode		1	30	

Notes: 1. Current through V_{DD0}, V_{DD1} respectively through V_{SS0}, V_{SS1}.
Excluded is the current through the inside pull-up resistors, through AV_{DD}/AV_{REF}, the port current.

2. CPU is operable.
The other peripherals like: CAN controller, Timer 0, Timer 2, serial interfaces, A/D converter etc. are stopped.

3. CPU and all peripherals (except for the A/D converter) are in operating mode and PCL output is f_X.

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

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

3. The typical values are with respect to T_A = 25°C.

(3) µPD78F0818(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

Flash EEPROM Version

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Power supply current Note 1	I_{DD1}	f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 2		9.5	19	mA
		f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 3		15	28.5	
	I_{DD3}	RC oscillation operating mode (f_{XT} = 40 kHz)		180	560	µA
	I_{DD4}	RC oscillation HALT mode (f_{XT} = 40 kHz)		60	180	
	I_{DD5}	CL1 = V_{DD} STOP mode		1	30	

Notes: 1. Current through V_{DD0}, V_{DD1} respectively through V_{SS0}, V_{SS1}.
Excluded is the current through the inside pull-up resistors, through AV_{DD}/AV_{REF}, the port current.

2. CPU is operable.
The other peripherals like: CAN controller, Timer 0, Timer 2, serial interfaces, A/D converter etc. are stopped.

3. CPU and all peripherals (except for the A/D converter) are in operating mode and PCL output is f_X.

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

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

3. The typical values are with respect to T_A = 25°C.

(4) µPD780814(A1), µPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V)

Mask ROM Version

These specifications are only target values and may not be satisfied by mass-produced products.

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
High-level input voltage	V_{IH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CRXD, ANI0 - ANI11		0.7 V_{DD}		V_{DD}	V
	V_{IH2}	$\overline{\text{RESET}}$		0.8 V_{DD}		V_{DD}	
	V_{IH4}	X1, X2, CL1		V_{DD} - 0.5		V_{DD}	
Low-level input voltage	V_{IL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CRXD, ANI0 - ANI11		0		0.3 V_{DD}	
	V_{IL2}	$\overline{\text{RESET}}$		0		0.2 V_{DD}	
	V_{IL4}	X1, X2, CL1		0		0.4	
High-level output voltage	V_{OH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD	V_{DD} = 4.0 - 5.5 V I_{OH} = -1 mA	V_{DD} - 1.0		V_{DD}	
Low-level output voltage	V_{OL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD	V_{DD} = 4.0 - 5.5 V I_{OL} = 1.6 mA	0		0.4	
High-level input leakage current	I_{LIH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, $\overline{\text{RESET}}$, CRXD, ANI0 - ANI11	V_{IN} = V_{DD}	0		10	µA
	I_{LIH2}	X1, X2, CL1, CL2		0		20	
Low-level input leakage current	I_{LIL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, $\overline{\text{RESET}}$, CRXD, ANI0 - ANI11	V_{IN} = 0 V	0		-10	
	II_{LIL2}	X1, X2, CL1, CL2		0		-20	
High-level output leakage current	I_{LOH}	V_{OUT} = V_{DD}		0		10	
Low-level output leakage current	I_{LOL}	V_{OUT} = 0 V		0		-10	
Software pull-up resistor	R2	V_{IN} = 0 V	4.5 V ≤ V_{DD} ≤ 5.5 V	10	30	100	KΩ

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

(5) µPD780814(A1), µPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V)

These specifications are only target values and may not be satisfied by mass-produced products.

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Power supply current Note 1	I_{DD1}	f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 2		5.5	12	mA
		f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 3		9.5	20	
	I_{DD3}	RC oscillation operating mode (f_{XT} = 40 kHz)		150	1560	µA
	I_{DD4}	RC oscillation HALT mode (f_{XT} = 40 kHz)		60	1180	
	I_{DD5}	CL1 = V_{DD} STOP mode		1	1000	

Notes: 1. Current through V_{DD0}, V_{DD1} respectively through V_{SS0}, V_{SS1}.
Excluded is the current through the inside pull-up resistors, through AV_{DD}/AV_{REF}, the port current.

2. CPU is operable.
The other peripherals like: CAN controller, Timer 0, Timer 2, serial interfaces, A/D converter etc. are stopped.

3. CPU and all peripherals (except for the A/D converter) are in operating mode and PCL output is f_X.

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

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

3. The typical values are with respect to T_A = 25°C.

(6) µPD780814(A2), µPD780816(A2)
(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V)

Mask ROM Version

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
High-level input voltage	V_{IH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CRXD, ANI0 - ANI11		0.7 V_{DD}		V_{DD}	V
	V_{IH2}	$\overline{\text{RESET}}$		0.8 V_{DD}		V_{DD}	
	V_{IH4}	X1, X2, CL1		V_{DD} - 0.5		V_{DD}	
Low-level input voltage	V_{IL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CRXD, ANI0 - ANI11		0		0.3 V_{DD}	
	V_{IL2}	$\overline{\text{RESET}}$		0		0.2 V_{DD}	
	V_{IL4}	X1, X2, CL1		0		0.4	
High-level output voltage	V_{OH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD	V_{DD} = 4.0 - 5.5 V I_{OH} = -1 mA	V_{DD} - 1.0		V_{DD}	
Low-level output voltage	V_{OL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, CTXD	V_{DD} = 4.0 - 5.5 V I_{OL} = 1.6 mA	0		0.4	
High-level input leakage current	I_{LIH1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, $\overline{\text{RESET}}$, CRXD, ANI0 - ANI11	V_{IN} = V_{DD}	0		10	µA
	I_{LIH2}	X1, X2, CL1, CL2		0		20	
Low-level input leakage current	I_{LIL1}	P00 - P03, P10 - P17, P20 - P27, P40 - P47, P50 - P57, P60 - P67, P70, P71, $\overline{\text{RESET}}$, CRXD, ANI0 - ANI11	V_{IN} = 0 V	0		-10	
	II_{LIL2}	X1, X2, CL1, CL2		0		-20	
High-level output leakage current	I_{LOH}	V_{OUT} = V_{DD}		0		10	
Low-level output leakage current	I_{LOL}	V_{OUT} = 0 V		0		-10	
Software pull-up resistor	R2	V_{IN} = 0 V	4.5 V ≤ V_{DD} ≤ 5.5 V	10	30	100	KΩ

Remark: The characteristics of the dual-function pins are the same as those of the port pins unless otherwise specified.

(7) µPD780814(A2), µPD780816(A2)
(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Power supply current Note 1	I_{DD1}	f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 2		5.5	12	mA
		f_X = 8 MHz, crystal/ceramic oscillation operating mode (PCC = 00H)Note 3		9.5	20	
	I_{DD3}	RC oscillation operating mode (f_{XT} = 40 kHz)		150	1560	µA
	I_{DD4}	RC oscillation HALT mode (f_{XT} = 40 kHz)		60	1180	
	I_{DD5}	CL1 = V_{DD} STOP mode		1	1000	

Notes: 1. Current through V_{DD0}, V_{DD1} respectively through V_{SS0}, V_{SS1}.
Excluded is the current through the inside pull-up resistors, through AV_{DD}/AV_{REF}, the port current.

2. CPU is operable.
The other peripherals like: CAN controller, Timer 0, Timer 2, serial interfaces, A/D converter etc. are stopped.

3. CPU and all peripherals (except for the A/D converter) are in operating mode and PCL output is f_X.

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

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

3. The typical values are with respect to T_A = 25°C.

24.6 AC Characteristics

24.6.1 Basic Operation

(1) μPD780814(A), μPD780816(A), μPD78F0818(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

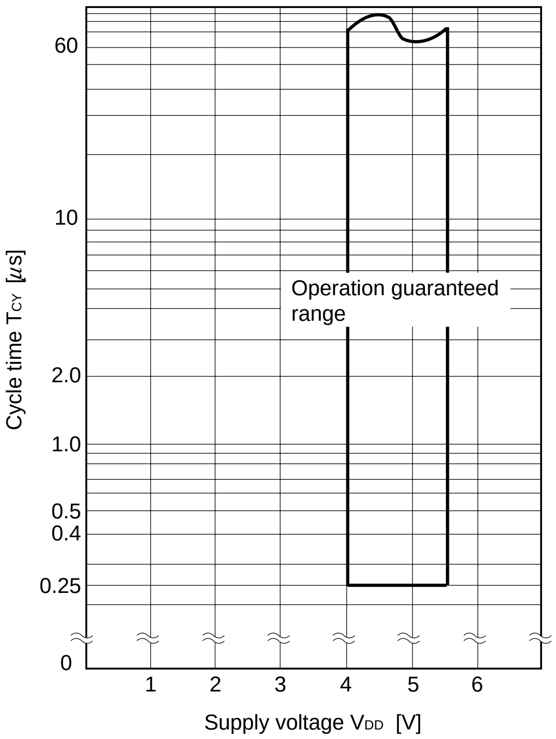
Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Cycle time Note 1	T_{CY}	4.0 V ≤ V_{DD} ≤ 5.5 V	0.25		125	μs
TI50, TI51 input frequency	f_{TI5}		0		4	MHz
TI50, TI51 input high/low level width	t_{TIH5} t_{TIL5}		100			ns
TI20, TI21, TI22 input high/low level width	t_{TIH2} t_{TIL2}		$3/f_{SMP2}$ **Note 2**			μs
TI00, TI01 input high/low level width	T_{CAPH} T_{CAPL}		$3/f_{SMP0}$ **Note 3**			
Interrupt input high/low level width	T_{INTH} T_{INTL}	INTP0 - INTP3, P40 - P47	1			
$\overline{\text{RESET}}$ low level width	t_{RSL}		10			

Notes: 1. The cycle time equals to the minimum instruction execution time.
For example:
1 NOP instruction corresponds to 2 CPU clock cycles (f_{CPU}) selected by the processor clock control register (PCC).

2. f_{SMP2} (sampling clock) = $f_X/8$, $f_X/16$, $f_X/32$, $f_X/64$

3. f_{SMP0} (sampling clock) = $f_X/2$, $f_X/16$, $f_X/128$.
Selection of f_{SMP0} = $f_X/2$, $f_X/16$, $f_X/128$ is possible using bits 0 and 1 (PRM00, PRM01) of prescaler mode register PRM0. However, if the TI00 valid edge is selected as the count clock, the value becomes f_{SMP0} = $f_X/2$.

T_{CY} vs. V_{DD}


60
10
2.0
1.0
0.5
0.4
0.25
0
1
2
3
4
5
6
Cycle time TCY [μs]
Supply voltage VDD [V]
Operation guaranteed range

(2) μPD780814(A1), μPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V)

These specifications are only target values and may not be satisfied by mass-produced products.

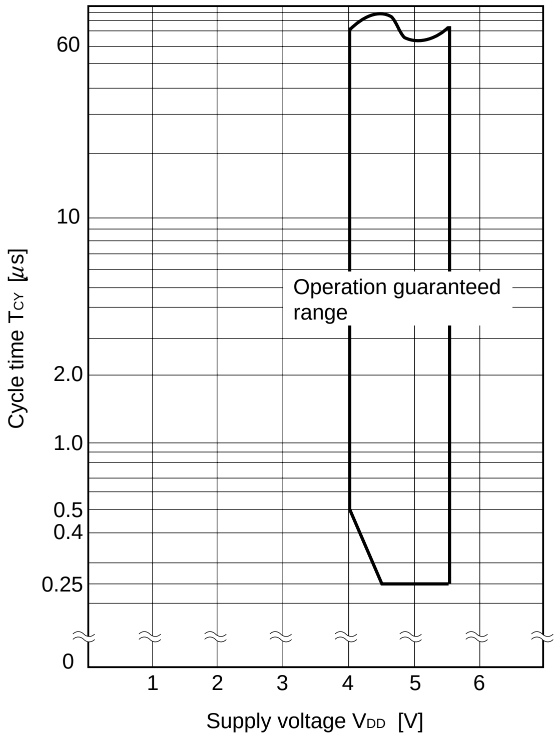
Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Cycle time **Note 1**	T_{CY}	4.5 V ≤ V_{DD} ≤ 5.5 V	0.25		125	μs
		4.0 V ≤ V_{DD} ≤ 5.5 V	0.5		125	
TI50, TI51 input frequency	f_{TI5}		0		4	MHz
TI50, TI51 input high/low level width	t_{TIH5} t_{TIL5}		100			ns
TI20, TI21, TI22 input high/low level width	t_{TIH2} t_{TIL2}		$3/f_{SMP2}$ **Note 2**			μs
TI00, TI01 input high/low level width	T_{CAPH} T_{CAPL}		$3/f_{SMP0}$ **Note 3**			
Interrupt input high/low level width	T_{INTH} T_{INTL}	INTP0 - INTP3, P40 - P47	1			
$\overline{\text{RESET}}$ low level width	t_{RSL}		10			

Notes: 1. The cycle time equals to the minimum instruction execution time.
For example:
1 NOP instruction corresponds to 2 CPU clock cycles (f_{CPU}) selected by the processor clock control register (PCC).

2. f_{SMP2} (sampling clock) = $f_X/8$, $f_X/16$, $f_X/32$, $f_X/64$

3. f_{SMP0} (sampling clock) = $f_X/2$, $f_X/16$, $f_X/128$.
Selection of f_{SMP0} = $f_X/2$, $f_X/16$, $f_X/128$ is possible using bits 0 and 1 (PRM00, PRM01) of prescaler mode register PRM0. However, if the TI00 valid edge is selected as the count clock, the value becomes f_{SMP0} = $f_X/2$.

T_{CY} vs. V_{DD}


60
10
2.0
1.0
0.5
0.4
0.25
0
1
2
3
4
5
6
Cycle time TCY [μs]
Supply voltage VDD [V]
Operation guaranteed range

(3) µPD780814(A2), µPD780816(A2)

(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V)

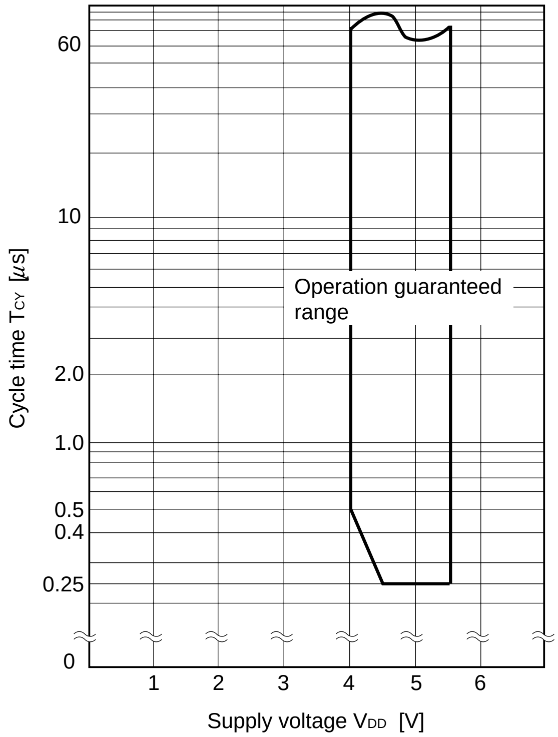
Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Cycle time [Note 1]	T_{CY}	4.5 V ≤ V_{DD} ≤ 5.5 V	0.25		125	µs
		4.0 V ≤ V_{DD} ≤ 5.5 V	0.5		125	
TI50, TI51 input frequency	f_{TI5}		0		4	MHz
TI50, TI51 input high/low level width	t_{TIH5} t_{TIL5}		100			ns
TI20, TI21, TI22 input high/low level width	t_{TIH2} t_{TIL2}		$3/f_{SMP2}$ **Note 2**			µs
TI00, TI01 input high/low level width	T_{CAPH} T_{CAPL}		$3/f_{SMP0}$ **Note 3**			
Interrupt input high/low level width	T_{INTH} T_{INTL}	INTP0 - INTP3, P40 - P47	1			
$\overline{\text{RESET}}$ low level width	t_{RSL}		10			

Notes: 1. The cycle time equals to the minimum instruction execution time.
For example:
1 NOP instruction corresponds to 2 CPU clock cycles (f_{CPU}) selected by the processor clock control register (PCC).

2. f_{SMP2} (sampling clock) = $f_X/8$, $f_X/16$, $f_X/32$, $f_X/64$

3. f_{SMP0} (sampling clock) = $f_X/2$, $f_X/16$, $f_X/128$.
Selection of f_{SMP0} = $f_X/2$, $f_X/16$, $f_X/128$ is possible using bits 0 and 1 (PRM00, PRM01) of prescaler mode register PRM0. However, if the TI00 valid edge is selected as the count clock, the value becomes f_{SMP0} = $f_X/2$.

T_{CY} vs. V_{DD}


60
10
2.0
1.0
0.5
0.4
0.25
0
1
2
3
4
5
6
Cycle time TCY [μs]
Supply voltage VDD [V]
Operation guaranteed range

24.6.2 Serial Interface

(1) µPD780814(A), µPD780816(A), µPD78F0818(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V)

(a) Serial interface Channel CSI (SIO2)

3-wire serial I/O mode ($\overline{\text{SCK2}}$ Internal clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK2}}$ cycle time	t_{KCY1}		1000		ns
$\overline{\text{SCK2}}$ high/low-level width	t_{KH1}, t_{KL1}		t_{KCY1}/2 - 50		
SI2 setup time (to $\overline{\text{SCK2}}$) ↑	t_{SIK1}		100		
SI2 hold time (from $\overline{\text{SCK2}}$) ↑	t_{KSI1}		400		
SO2 output delay time (from $\overline{\text{SCK2}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO2, $\overline{\text{SCK2}}$ output line

3-wire serial I/O mode ($\overline{\text{SCK2}}$ External clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK2}}$ cycle time	t_{KCY1}		800		ns
$\overline{\text{SCK2}}$ high/low-level width	t_{KH1}, t_{KL1}		400		
SI2 setup time (to $\overline{\text{SCK2}}$) ↑	t_{SIK1}		100		
SI2 hold time (from $\overline{\text{SCK2}}$) ↑	t_{KSI1}		400		
SO2 output delay time (from $\overline{\text{SCK2}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO2, $\overline{\text{SCK2}}$ output line

(b) Serial interface Channel CSI (SIO3)

3-wire serial I/O mode ($\overline{SCK3}$ Internal clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{SCK3}$ cycle time	t_{KCY1}		1000		ns
$\overline{SCK3}$ high/low-level width	t_{KH1}, t_{KL1}		t_{KCY1}/2 - 50		
SI3 setup time (to $\overline{SCK3}$) ↑	t_{SIK1}		100		
SI3 hold time (from $\overline{SCK3}$) ↑	t_{KSI1}		400		
SO3 output delay time (from $\overline{SCK3}$) ↓	t_{KSO1}	C = 100 pF [Note]		300	

Note: C is the load capacitance of SO3, $\overline{SCK3}$ output line

3-wire serial I/O mode ($\overline{SCK3}$ External clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{SCK3}$ cycle time	t_{KCY1}		800		ns
$\overline{SCK3}$ high/low-level width	t_{KH1}, t_{KL1}		400		
SI3 setup time (to $\overline{SCK3}$) ↑	t_{SIK1}		100		
SI3 hold time (from $\overline{SCK3}$) ↑	t_{KSI1}		400		
SO3 output delay time (from $\overline{SCK3}$) ↓	t_{KSO1}	C = 100 pF [Note]		300	

Note: C is the load capacitance of SO3, $\overline{SCK3}$ output line

(c) Serial interface Channel UART

UART mode (Dedicated baud rate generator output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					125	Kbps

(2) μPD780814(A1), μPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V)

Mask Version

(a) Serial interface Channel CSI (SIO2)

These specifications are only target values and may not be satisfied by mass-produced products.

3-wire serial I/O mode ($\overline{\text{SCK2}}$ Internal clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK2}}$ cycle time	t_{KCY1}		1000		ns
$\overline{\text{SCK2}}$ high/low-level width	t_{KH1}, t_{KL1}		t_{KCY1}/2 - 50		
SI2 setup time (to $\overline{\text{SCK2}}$) ↑	t_{SIK1}		100		
SI2 hold time (from $\overline{\text{SCK2}}$) ↑	t_{KSI1}		400		
SO2 output delay time (from $\overline{\text{SCK2}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO2, $\overline{\text{SCK2}}$ output line

These specifications are only target values and may not be satisfied by mass-produced products.

3-wire serial I/O mode ($\overline{\text{SCK2}}$ External clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK2}}$ cycle time	t_{KCY1}		800		ns
$\overline{\text{SCK2}}$ high/low-level width	t_{KH1}, t_{KL1}		400		
SI2 setup time (to $\overline{\text{SCK2}}$) ↑	t_{SIK1}		100		
SI2 hold time (from $\overline{\text{SCK2}}$) ↑	t_{KSI1}		400		
SO2 output delay time (from $\overline{\text{SCK2}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO2, $\overline{\text{SCK2}}$ output line

(b) Serial interface Channel CSI (SIO3)

These specifications are only target values and may not be satisfied by mass-produced products.

3-wire serial I/O mode ($\overline{\text{SCK3}}$ Internal clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK3}}$ cycle time	t_{KCY1}		1000		ns
$\overline{\text{SCK3}}$ high/low-level width	t_{KH1}, t_{KL1}		$t_{KCY1}/2$ - 50		
SI3 setup time (to $\overline{\text{SCK3}}$) ↑	t_{SIK1}		100		
SI3 hold time (from $\overline{\text{SCK3}}$) ↑	t_{KSI1}		400		
SO3 output delay time (from $\overline{\text{SCK3}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO3, $\overline{\text{SCK3}}$ output line

These specifications are only target values and may not be satisfied by mass-produced products.

3-wire serial I/O mode ($\overline{\text{SCK3}}$ External clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK3}}$ cycle time	t_{KCY1}		800		ns
$\overline{\text{SCK3}}$ high/low-level width	t_{KH1}, t_{KL1}		400		
SI3 setup time (to $\overline{\text{SCK3}}$) ↑	t_{SIK1}		100		
SI3 hold time (from $\overline{\text{SCK3}}$) ↑	t_{KSI1}		400		
SO3 output delay time (from $\overline{\text{SCK3}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO3, $\overline{\text{SCK3}}$ output line

(c) Serial interface Channel UART

These specifications are only target values and may not be satisfied by mass-produced products.

UART mode (Dedicated baud rate generator output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					125	Kbps

(3) μPD780814(A2), μPD780816(A2)

(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V)

(a) Serial interface Channel CSI (SIO2)

3-wire serial I/O mode ($\overline{\text{SCK2}}$ Internal clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK2}}$ cycle time	t_{KCY1}		1000		ns
$\overline{\text{SCK2}}$ high/low-level width	t_{KH1}, t_{KL1}		t_{KCY1}/2 - 50		
SI2 setup time (to $\overline{\text{SCK2}}$) ↑	t_{SIK1}		100		
SI2 hold time (from $\overline{\text{SCK2}}$) ↑	t_{KSI1}		400		
SO2 output delay time (from $\overline{\text{SCK2}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO2, $\overline{\text{SCK2}}$ output line

3-wire serial I/O mode ($\overline{\text{SCK2}}$ External clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK2}}$ cycle time	t_{KCY1}		800		ns
$\overline{\text{SCK2}}$ high/low-level width	t_{KH1}, t_{KL1}		400		
SI2 setup time (to $\overline{\text{SCK2}}$) ↑	t_{SIK1}		100		
SI2 hold time (from $\overline{\text{SCK2}}$) ↑	t_{KSI1}		400		
SO2 output delay time (from $\overline{\text{SCK2}}$) ↓	t_{KSO1}	C = 100 pF **Note**		300	

Note: C is the load capacitance of SO2, $\overline{\text{SCK2}}$ output line

(b) Serial interface Channel CSI (SIO3)

3-wire serial I/O mode ($\overline{\text{SCK3}}$ Internal clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK3}}$ cycle time	t_{KCY1}		1000		ns
$\overline{\text{SCK3}}$ high/low-level width	t_{KH1}, t_{KL1}		$t_{KCY1}/2$ - 50		
SI3 setup time (to $\overline{\text{SCK3}}$) ↑	t_{SIK1}		100		
SI3 hold time (from $\overline{\text{SCK3}}$) ↑	t_{KSI1}		400		
SO3 output delay time (from $\overline{\text{SCK3}}$) ↓	t_{KSO1}	C = 100 pF Note		300	

Note: C is the load capacitance of SO3, $\overline{\text{SCK3}}$ output line

3-wire serial I/O mode ($\overline{\text{SCK3}}$ External clock output)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK31}}$ cycle time	t_{KCY1}		800		ns
$\overline{\text{SCK31}}$ high/low-level width	t_{KH1}, t_{KL1}		400		
SI31 setup time (to $\overline{\text{SCK31}}$) ↑	t_{SIK1}		100		
SI31 hold time (from $\overline{\text{SCK31}}$) ↑	t_{KSI1}		400		
SO31 output delay time (from $\overline{\text{SCK31}}$) ↓	t_{KSO1}	C = 100 pF Note		300	

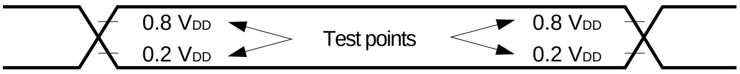
Note: C is the load capacitance of SO3, $\overline{\text{SCK3}}$ output line

(c) Serial interface Channel UART

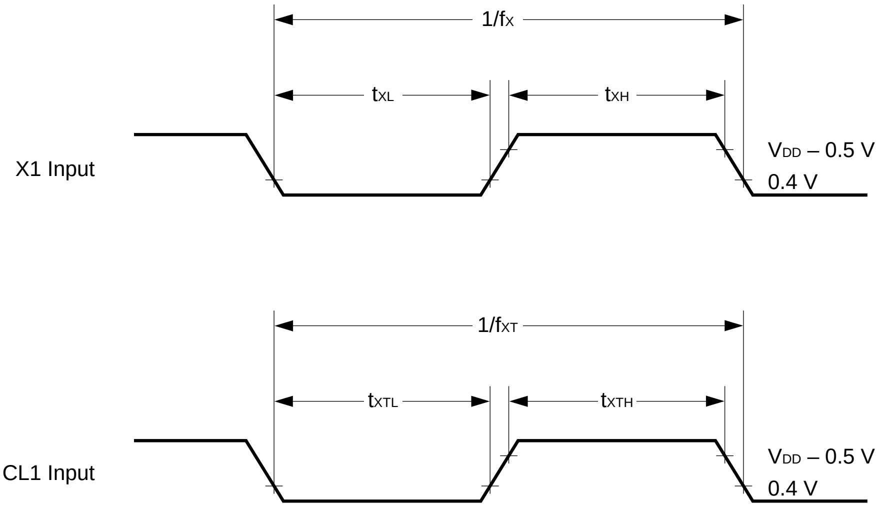
UART mode (Dedicated baud rate generator output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					125	Kbps

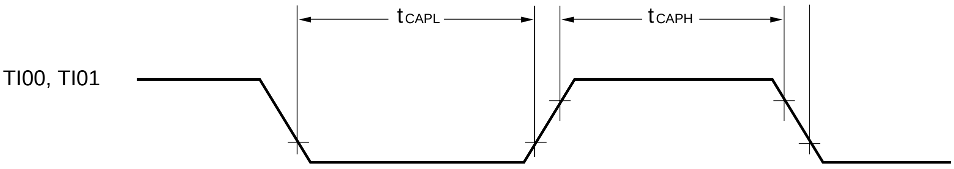
AC Timing Test Points (excluding X1, CL1 Input)

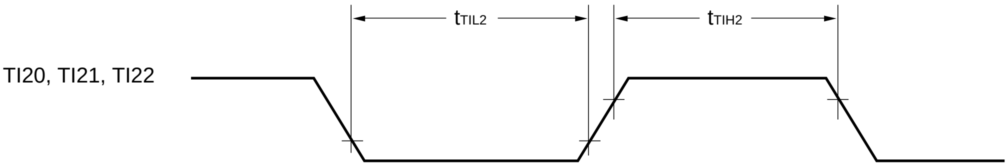


Clock Timing

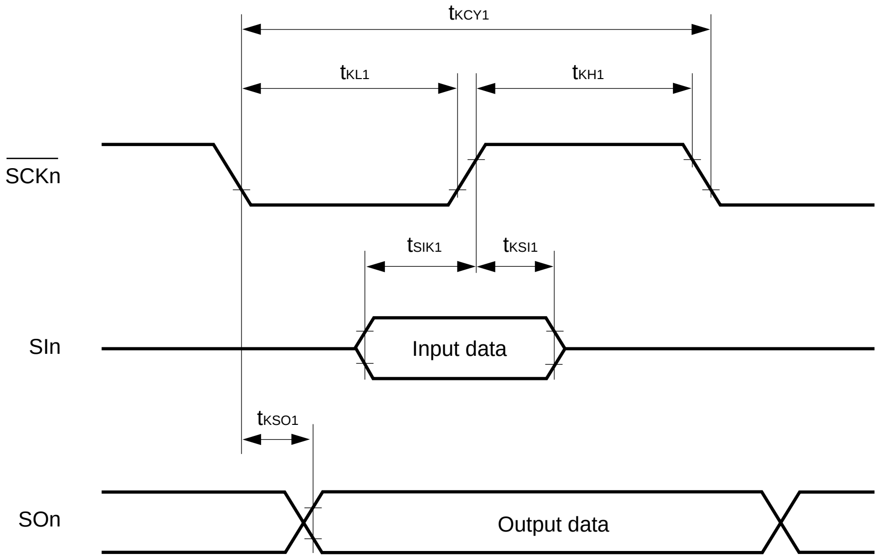


TI Timing





3-wire serial I/O mode / 2-wire serial I/O mode



Remark: n = 2, 3

24.6.3 A/D Converter Characteristics

(1) µPD780814(A), µPD780816(A), µPD78F0818(A)
(T_A = -40°C to +85°C, V_{DD} = 4.0 to 5.5 V, AV_{SS} = V_{SS} = 0 V, f_X = 8 MHz)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8	8	8	bit
Overall error Note					± 0.6	%
Conversion time	t_{CONV}		14			µs
Analog input voltage	V_{IAN}		AV_{SS}		AV_{DD}	V
Reference voltage	AV_{DD} / AV_{REF}		V_{DD}	V_{DD}	V_{DD}	
AV_{DD}/AV_{REF} current	I_{REF}	ADCS-bit = 1		750	1500	µA
		ADCS-bit = 0		0	3	

Note: Overall error excluding quantization (± 1/2 LSB). It is indicated as a ratio to the full-scale value.

Remark: f_X: Main system clock oscillation frequency.

(2) µPD780814(A1), µPD780816(A1)
(T_A = -40°C to +110°C, V_{DD} = 4.0 to 5.5 V, AV_{SS} = V_{SS} = 0 V, f_X = 8 MHz)

These specifications are only target values and may not be satisfied by mass-produced products.

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8	8	8	bit
Overall error					± 1.3	%
Conversion time	t_{CONV}		14			µs
Analog input voltage	V_{IAN}		AV_{SS}		AV_{DD}	V
Reference voltage	AV_{DD} / AV_{REF}		V_{DD}	V_{DD}	V_{DD}	
AV_{DD}/AV_{REF} current	I_{REF}	ADCS-bit = 1		750	1500	µA
		ADCS-bit = 0		0	3	

Remark: f_X: Main system clock oscillation frequency.

(3) µPD780814(A2), µPD780816(A2)

(T_A = -40°C to +125°C, V_{DD} = 4.0 to 5.5 V, AV_{SS} = V_{SS} = 0 V, f_X = 8 MHz)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8	8	8	bit
Overall error					± 1.3	%
Conversion time	t_{CONV}		14			µs
Analog input voltage	V_{IAN}		AV_{SS}		AV_{DD}	V
Reference voltage	AV_{DD} / AV_{REF}		V_{DD}	V_{DD}	V_{DD}	
AV_{DD}/AV_{REF} current	I_{REF}	ADCS-bit = 1		750	1500	µA
		ADCS-bit = 0		0	3	

Remark: f_X: Main system clock oscillation frequency.

24.6.4 Data Memory Stop Mode Low Supply Voltage Data Retention Characteristics

(1) µPD780814(A), µPD780816(A), µPD78F0818(A)
(T_A = -40°C to +85°C)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Data retention power supply voltage	V_{DDDR}		2.0		5.5	V
Data retention power supply current	I_{DDDR}	V_{DDDR} = 2.0 V		1	30	µA
Release signal set time	t_{SREL}		0			µs
Oscillation stabilization wait time	t_{WAIT}	Release by $\overline{\text{RESET}}$		$2^{17}/f_X$		ms
		Release by interrupt		**Note**		

Note: In combination with bits 0 to 2 (OSTS0 to OSTS2) of oscillation stabilization time select register, selection of $2^{12}/f_X$ and $2^{14}/f_X$ to $2^{17}/f_X$ is possible.

Remark: f_X: Main system clock oscillation frequency.

(2) µPD780814(A1), µPD780816(A1)
(T_A = -40°C to +110°C)

These specifications are only target values and may not be satisfied by mass-produced products.

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Data retention power supply voltage	V_{DDDR}		2.5		5.5	V
Data retention power supply current	I_{DDDR}	V_{DDDR} = 2.0 V		1	1000	µA
Release signal set time	t_{SREL}		0			µs
Oscillation stabilization wait time	t_{WAIT}	Release by $\overline{\text{RESET}}$		$2^{17}/f_X$		ms
		Release by interrupt		**Note**		

Note: In combination with bits 0 to 2 (OSTS0 to OSTS2) of oscillation stabilization time select register, selection of $2^{12}/f_X$ and $2^{14}/f_X$ to $2^{17}/f_X$ is possible.

Remark: f_X: Main system clock oscillation frequency.

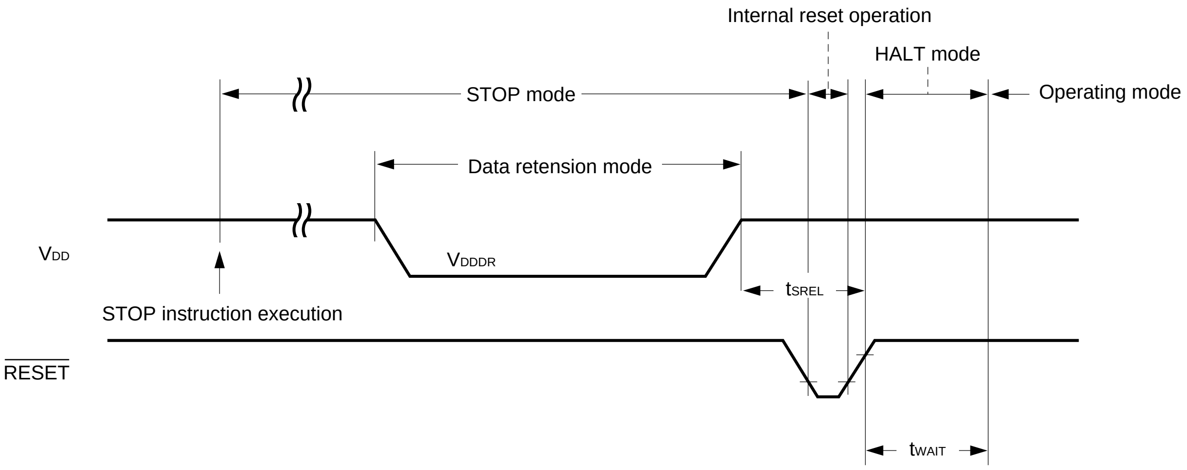
(3) µPD780814(A2), µPD780816(A2)
(T_A = -40°C to +125°C)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Data retention power supply voltage	V_{DDDR}		2.5		5.5	V
Data retention power supply current	I_{DDDR}	V_{DDDR} = 2.0 V		1	1000	µA
Release signal set time	t_{SREL}		0			µs
Oscillation stabilization wait time	t_{WAIT}	Release by $\overline{\text{RESET}}$		$2^{17}/f_X$		ms
		Release by interrupt		**Note**		

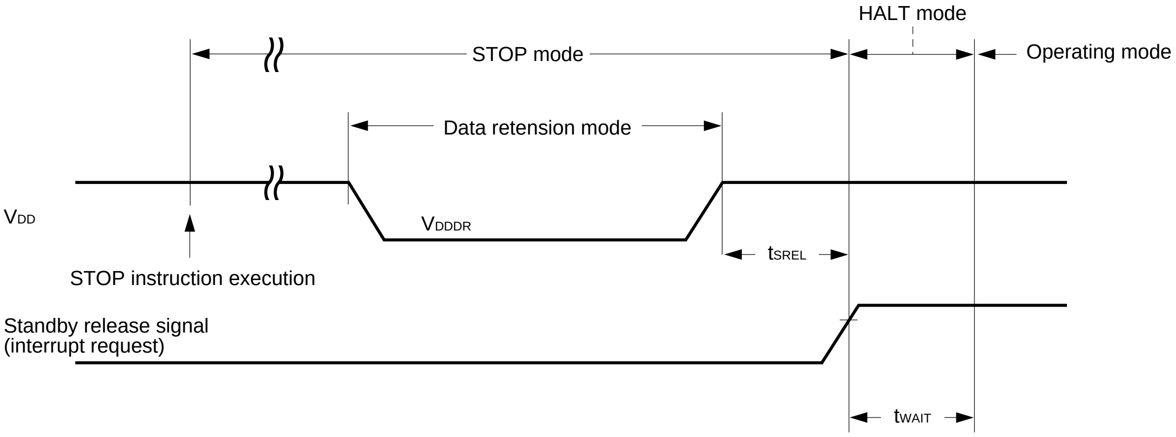
Note: In combination with bits 0 to 2 (OSTS0 to OSTS2) of oscillation stabilization time select register, selection of $2^{12}/f_X$ and $2^{14}/f_X$ to $2^{17}/f_X$ is possible.

Remark: f_X: Main system clock oscillation frequency.

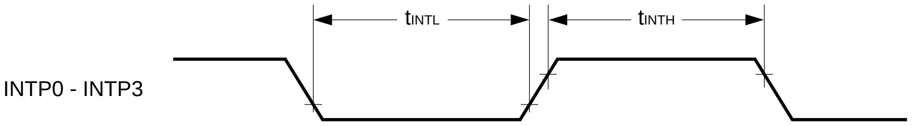
Data Retention Timing (STOP mode release by RESET)



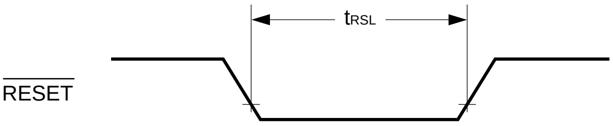
Data Retention Timing (Standby release signal: STOP mode release by Interrupt signal)



Interrupt Input Timing



RESET Input Timing



24.6.5 Flash Memory Programming Characteristics: µPD78F0818(A)

(T_A = 10°C to 40°C, V_{DD} = AV_{DD} = 4.5 to 5.5 V, V_{SS} = AV_{SS} = 0 V, V_{PP} = 9.7 to 10.3 V)

(1) Basic characteristics

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Operating frequency	f_X		4.0		8.38	MHz
Supply voltage	V_{DD}		4.0		5.5	V
	V_{PPL}	When V_{PP} low-level is detected	0		0.2 V_{DD}	V
	V_{PP}	When V_{PP} high-level is detected	0.8 V_{DD}	V_{DD}	1.2 V_{DD}	V
	V_{PPH}	When V_{PP} high-voltage is detected and for programming	9.7	10.0	10.3	V
Number of rewrites	C_{WRT}		20**Note**			Times
Programming temperature	t_{PRG}		10		+40	°C

Note: Operation is not guaranteed for over 20 rewrites.

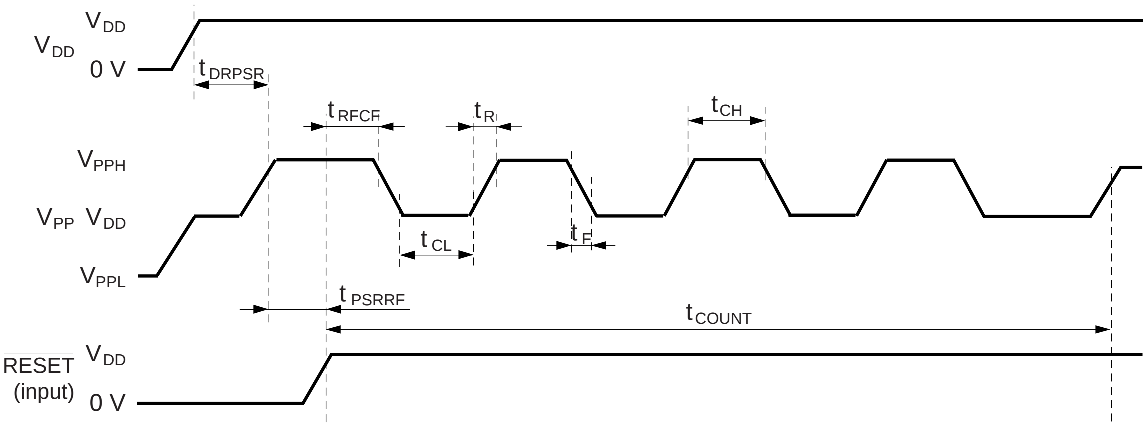
Remark: After execution of the program command, execute the verify command and check that the writing has been completed normally.

(2) Serial write operation characteristics

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Set time from V_{DD} ↑ to V_{PP} ↑	t_{DRPSR}	V_{PP} high voltage	10			µs
Set time from V_{PP} ↑ to $\overline{\text{RESET}}$ ↑	t_{PSRRF}	V_{PP} high voltage	1.0			
V_{PP} ↑ count start time from $\overline{\text{RESET}}$ ↑	t_{RFCF}	V_{PP} high voltage	1.0			
Count execution time	t_{COUNT}				2.0	ms
V_{PP} counter high-level width	t_{CH}		8.0		**Note**	µs
V_{PP} counter low-level width	t_{CL}		8.0		**Note**	
V_{PP} counter rise/fall time	t_R, t_F				1.0	

Note: For maximum t_{CH} / t_{CL}, please make sure to finish the pulses within the time t_{COUNT}.

Flash Write Mode Setting Timing



(3) Write erase characteristics

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
V_{PP} supply voltage	V_{PP2}	During flash memory programming	9.7	10.0	10.3	V
V_{DD} supply current	I_{DD}	When V_{PP} = 10 V, f_X = 8.38 MHz			50	mA
V_{PP} supply current	I_{PP}	When V_{PP} = 10 V			100	mA
Step erase time	t_{ER}	**Note 1**		0.2		s
Overall erase time per area	t_{ERA}	When step erase time = 0.2 s **Note 2**			20	s/area
Write-back time	t_{WB}	**Note 3**	49.4	50	50.6	ms
Number of write-backs per write-back command	c_{WB}	When write-back time = 50 ms **Note 4**			60	Times/ write-back command
Number of erase/ write-backs	c_{ERWB}				16	Times
Step write time	t_{WR}	**Note 5**	48	50	52	µs
Overall write time per word	t_{WRW}	When step write time = 50 µs (1 word = 1 byte) **Note 6**	48		520	µs/ word
Number of rewrites per area	c_{ERWR}	1 erase + 1 write after erase = = 1 rewrite **Note 7**	20			Times/ area

Notes: 1. The recommended setting value for the step erase time is 0.2 s.

2. The prewrite time before erasure and the erase verify time (write-back time) is not included.

3. The recommended setting value for the write-back time is 50 ms.

4. Write-back is executed once by the issuance of the write-back command. Therefore, the number of retries must be the maximum value minus the number of commands issued.

5. Recommended step write setting value is 50 µs.

6. The actual write time per word is 100 µs longer. The internal verify time during or after a write is not included.

7. When a product is first written after shipment, "erase → write" and "write only" are both taken as one rewrite.

Example: P: Write, E: Erase

Shipped product →	P → E → P → E → P	: 3 rewrites
Shipped product → E →	P → E → P → E → P	: 3 rewrites

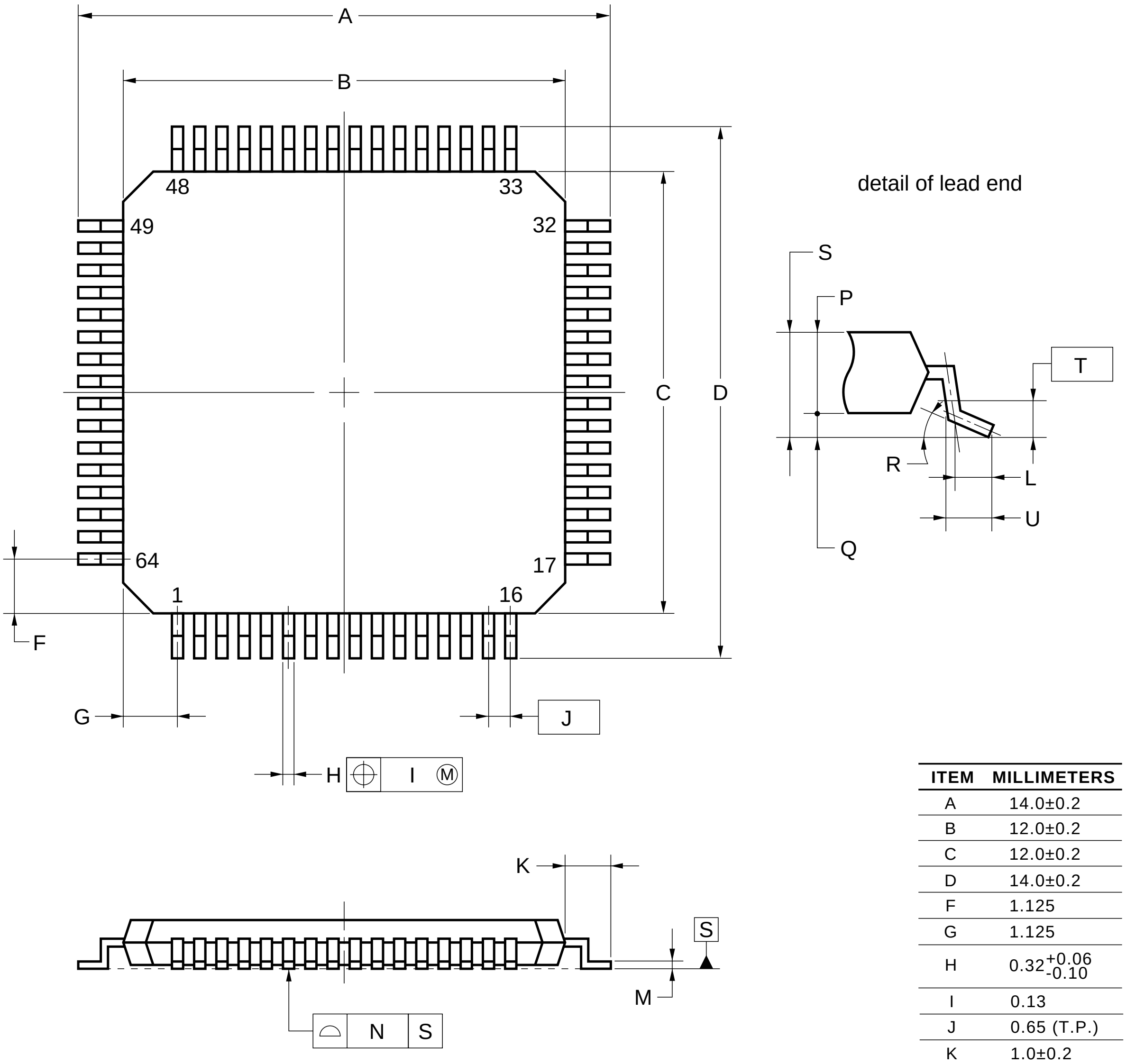
Remarks: 1. The range of the operating clock during flash memory programming is the same as the range during normal operation.

2. When using the *flash*MASTER, the time parameters that need to be downloaded from the parameter files for write/erase are automatically set. Unless otherwise directed, do not change the set values.

[MEMO]

Chapter 25 Package Drawing

64-PIN PLASTIC TQFP (12x12)



ITEM	MILLIMETERS
A	14.0±0.2
B	12.0±0.2
C	12.0±0.2
D	14.0±0.2
F	1.125
G	1.125
H	$0.32^{+0.06}_{-0.10}$
I	0.13
J	0.65 (T.P.)
K	1.0±0.2
L	0.5
M	$0.17^{+0.03}_{-0.07}$
N	0.10
P	1.0
Q	0.1±0.05
R	$3^{\circ}{}^{+4^{\circ}}_{-3^{\circ}}$
S	1.1±0.1
T	0.25
U	0.6±0.15

P64GK-65-9ET-2

NOTE

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

Remark: The shape and material of the ES product is the same as the mass produced product.

[MEMO]

Chapter 26 Recommended Soldering Conditions

The µPD780816 Subseries should be soldered and mounted under the conditions in the table below. For detail of recommended soldering conditions, refer to the information document **Semiconductor Device Mounting Technology Manual (IEI-1207)**.
For soldering methods and conditions other than those recommended below, consult our sales personnel.

• µPD780814GK(A)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)
• µPD780814GK(A1)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)
• µPD780814GK(A2)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)
• µPD780816GK(A)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)
• µPD780816GK(A1)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)
• µPD780816GK(A2)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)
• µPD78F0818GK(A)-XXX-9ET	: 64-pin plastic QFP (12 × 12 mm)

Surface Mounting Type Soldering Conditions

Soldering Method	Soldering conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C. Duration: 30 sec max. (at 210°C or above). Number of times: twice max. <Precautions> (1) The second reflow should be started after the first reflow device temperature has returned to the ordinary state. (2) Flux washing must not be performed by the use of water after the first reflow.	IR35-107-2
VPS	Package peak temperature: 215°C. Duration: 40 sec max. (at 210°C or above). Number of times: twice max. <Precautions> (1) The second reflow should be started after the first reflow device temperature has returned to the ordinary state. (2) Flux washing must not be performed by the use of water after the first reflow.	VR15-107-2
Wave soldering	Soldering bath temperature: 260°C max. Duration: 10 sec max. Number of times: once. Preheating temperature: 120°C max. (package surface temperature)	WS60-107-1
Pin part heating	Pin temperature: 350°C max. Duration: 3 sec max. (per device side)	P350

Caution: Use of more than one soldering method should be avoided (except in the case of pin part heating).

[MEMO]

Appendix A Development Tools

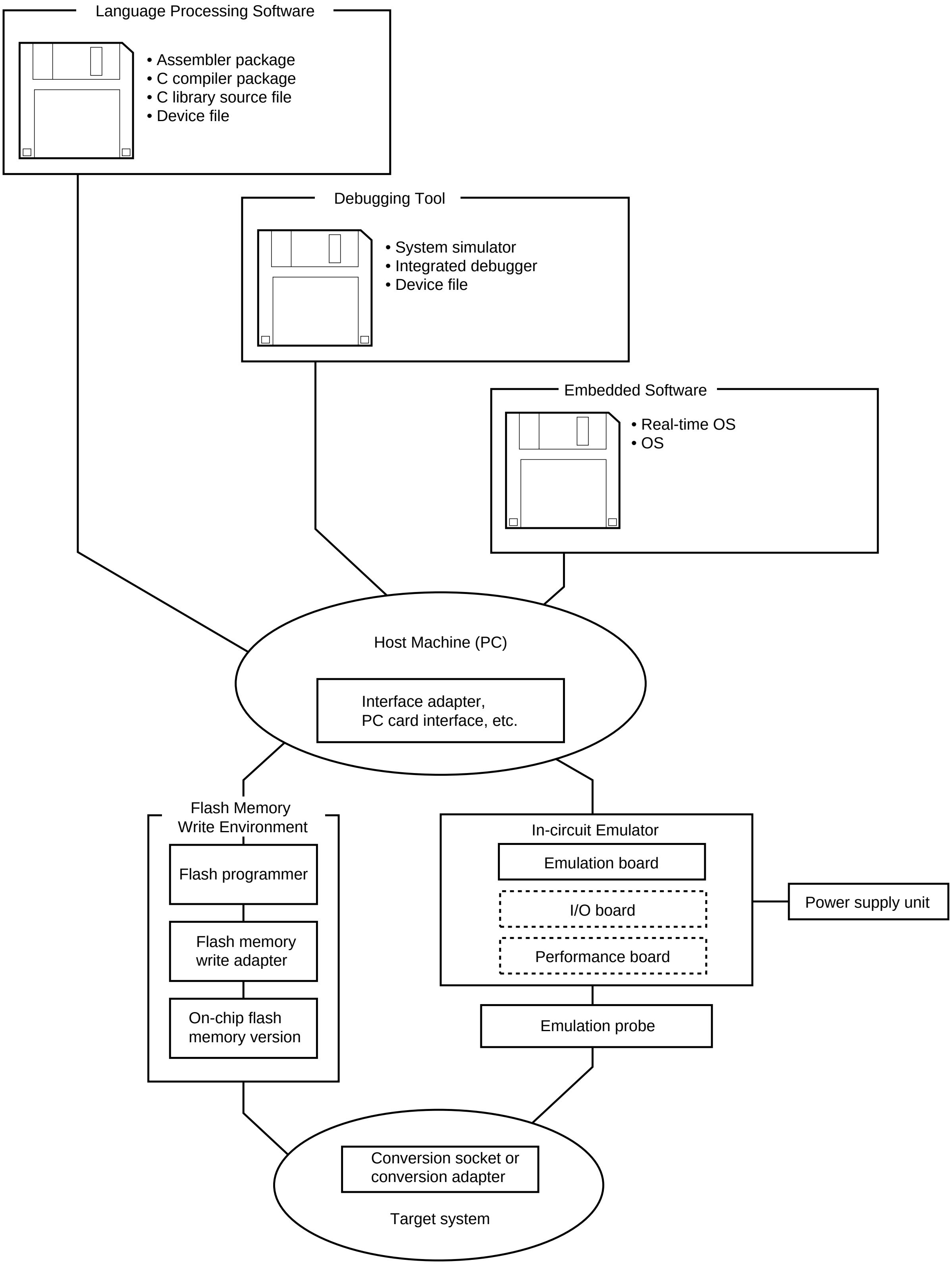
The following development tools are available for the development of systems that employ the µPD780816 Subseries.
Figure A-1 shows the development tool configuration.

- Support for PC98-NX series
  Unless otherwise specified, products compatible with IBM PC/ATTM computers are compatible with PC98-NX series computers. When using PC98-NX series computers, refer to the explanation for IBM PC/AT computers.

- Windows (Unless otherwise specified, “Windows” means the following OS).
- Windows 95/98
- Windows NT Version 4.0
- Windows 2000

Figure A-1: Development Tool Configuration

(a) When using the in-circuit emulator IE-78K0-NS-A



Remark: Items in broken line boxes differ according to the development environment. See A.3.1 Hardware.

A.1 Language Processing Software

NEC Software

RA78K/0 Assembler Package	This assembler converts programs written in mnemonics into an object codes executable with a microcontroller. Further, this assembler is provided with functions capable of automatically creating symbol tables and branch instruction optimization. This assembler should be used in combination with an optional device file. **<Precaution when using RA78K/0 in PC environment>** This assembler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) on Windows.
CC78K/0 C Compiler Package	This compiler converts programs written in C language into object codes executable with a microcontroller. This compiler should be used in combination with an optical assembler package and device file. **<Precaution when using CC78K/0 in PC environment>** This C compiler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) on Windows.
Device File	This file contains information peculiar to the device. This device file should be used in combination with an optical tool (RA78K/0, CC78K/0, SM78K0, ID78K0-NS, and ID78K0). Corresponding OS and host machine differ depending on the tool to be used with.
CC78K/0-L C Library Source File	This is a source file of functions configuring the object library included in the C compiler package (CC78K/0). This file is required to match the object library included in C compiler package to the customer's specifications.

IAR Software

A78000	Assembler package used for the 78K0 series.
ICC78000	C compiler package used for the 78K0 series.
XLINK	Linker package used for the 78K0 series.

A.2 Flash Memory Writing Tools

FlashMASTER Flashpro III (part number: FL-PR3, PG-FP3) Flashpro IV (part number: PG-FP4) Flash Programmer	Flash programmer dedicated to microcontrollers with on-chip flash memory.
FA-64GK-9ET Flash Memory Writing Adapter	Flash memory writing adapter used connected to the Flashpro II, Flashpro III and *flash*MASTER. • FA-64GK-9ET: 63-pin plastic QFP (GK-9ET type)

A.3 Debugging Tools

A.3.1 Hardware

(1) When using the In-Circuit Emulator IE-78K0-NS-A

IE-78K0-NS-A In-circuit Emulator	The in-circuit emulator serves to debug hardware and software when developing application systems using a 78K/0 Series product. It corresponds to integrated debugger (ID78K0-NS). This emulator should be used in combination with power supply unit, emulation probe, and interface adapter which is required to connect this emulator to the host machine.
IE-70000-MC-PS-B Power Supply Unit	This adapter is used for supplying power from a receptacle of 100-V to 240-V AC.
EB-Power FW 7301/05 Power Supply Unit	This adapter is used for supplying power from a receptable of 100 V to 240 V AC
IE-70000-98-IF-C Interface Adapter	This adapter is required when using the PC-9800 series computer (except notebook type) as the IE-78K0-NS-A host machine (C bus compatible).
IE-70000-CD-IF-A PC Card Interface	This is PC card and interface cable required when using notebook-type computer as the IE-78K0-NS-A host machine (PCMCIA socket compatible).
IE-70000-PC-IF-C Interface Adapter	This adapter is required when using the IBM PC compatible computers as the IE-78K0-NS-A host machine (ISA bus compatible).
IE-70000-PCI-IF-A Interface Adapter	This adapter is required when using a computer with PCI bus as the IE-78K0-NS host machine.
IE-78K0-NS-P04 Emulation Board	This board emulates the operations of the peripheral hardware peculiar to a device. It should be used in combination with an in-circuit emulator.
IE-780818-NS-EM4 Probe Board	This board provides the connection and buffers between the emulation board and the connector of the emulation probe.
NP-64GK Emulation Probe	This probe is used to connect the in-circuit emulator to a target system and is designed for use with 64-pin plastic QFP.
NQPACK064SB YQPACK064SB YQSOCKET064SBF HQPACK064SB1410 Conversion Adapter	This conversion adapter connects the NP-64GK to a target system board designed for a 64-pin plastic QFP.

(2) Socket Details

NQPACK064SB	Socket for soldering on the target.
YQPACK064SB	Adapter socket for connecting the probe to the NQPACK064SB
HQPACK064SB140	Lid socket for connecting the device to the NQPACK064SB
YQSOCKET064SBF	High adapter between the device to the YQPACK064SB and the probe

A.3.2 Software

SM78K0 System Simulator	This system simulator is used to perform debugging at C source level or assembler level while simulating the operation of the target system on a host machine. This simulator runs 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 should be used in combination with the optional device file.
ID78K0-NS Integrated Debugger (supporting In-Circuit Emulator IE-78K0-NS-A)	This debugger is a control program to debug 78K/0 Series microcontrollers. It adopts a graphical user interface, which is equivalent visually and operationally to Windows or OSF/Motif™. It also has an enhanced debugging function for C language programs, and thus trace results can be displayed on screen in C-language level by using the windows integration function which links a trace result with its source program, disassembled display, and memory display. In addition, by incorporating function modules such as task debugger and system performance analyzer, the efficiency of debugging programs, which run on real-time OSs can be improved. It should be used in combination with the optional device file.

[MEMO]

Appendix B Embedded Software

For efficient development and maintenance of the µPD780816 Subseries, the following embedded software products are available.

B.1 Real-Time OS

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
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.

Caution: When purchasing the RX78K/0, fill in the purchase application form in advance and sign the User Agreement.

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 Fuzzy inference debugger	Support software for evaluation and adjustment of fuzzy knowledge data by using in-circuit emulator and at hardware level.
	Part number: µSxxxxFD78K0 (PC-9800 Series, PC/AT and compatible machines)

[MEMO]

Appendix C Register Index

Appendix D Revision History

The following shows the revision history up to present. Application portions signifies the chapter of each edition.

(1/2)

Edition No.	Major items revised	Revised Sections
EE4	Revision of Chapter 1 (page 29)	1.6
	Revision of Figure 11-1 (page 191)	11.2
	Revision of Figure 16-2 (page 251), Caution added	16.4
	Revision of Figure 16-4 (page 254), Caution added	16.4
	Revision of Caution in Figure 16-5 (page 255)	16.5
	Revision of Caution in Figure 16-6 (page 257)	16.5
	Revision of Caution in Figure 16-8 (page 260)	16.5
	Revision of Table 16-4 (page 262)	16.5
	Addition of Caution in Figure 16-9 (page 262)	16.5
	Revision of Caution in Figure 16-12 (page 266)	16.5
	RXONLY-mode deleted (pages 322, 325)	17
	CAN Controller Precautions added (page 345)	17.19
	Revision of Figure 22-3 (page 386)	22.3
	Revision of Chapter 24.5 (pages 407, 408, 409, 421. 423)	24.5
	Revision of Chapter 24.6.1 (pages 424, 426, 428)	24.6.1
	Revision of chapter 26 (page 449)	26

(2/2)

Edition No.	Major items revised	Revised Sections

NEC

Facsimile Message

From:

Name

Company

Tel. FAX

Address

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.

Thank you for your kind support.

North America
NEC Electronics America Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Market Communication Dept.
Fax: +49(0)-211-6503-1344

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

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

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-2719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-6250-3583

Japan
NEC Semiconductor Technical Hotline
Fax: +81- 44-435-9608

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	□	□	□	□