ABOV SEMICONDUCTOR Co., Ltd.
8-BIT MICROCONTROLLERS

MC96FR332A

Data Sheet (Rev.1.4)

REVISION HISTORY

REVISION 0.0 (January 28, 2010)

- Initial Version

REVISION 1.0 (April 1, 2010)

- Revision 1.0 release

REVISION 1.1 (February 11, 2011)

- Fix SFR address
- Fix 15.2 Boot Area
- Fix Table 11-9 and modify TIMER3(PWM3) registers description
- Changed names of timer2/3/carrier generator/wt registers (ex. T3LDR→T3DRL)
- Changed signal/port names of USARTx (ex. RXD→RXD(=MISO))

REVISION 1.2 (May 26, 2011)

- Fix typo errors
- Add I2C chapter
- Fix PSR0 register description for SDASWAP and SCLSWAP bits.

REVISION 1.3 (August 3, 2011)

- Remove DPH1/DPH0 registers
- Remove FSECR register for eeprom sector erase

REVISION 1.4 (January 4, 2012)

- Correct open drain control register description(removed port 0 open drain control register)
- Add Appendix B
- Changed Boot Area Size, in section 15.2 Boot Area
- Removed parallel mode description

Revision 1.4
Published by ABU

Additional information of this manual may be served by ABOV Semiconductor offices in Korea or Distributors.

ABOV Semiconductor reserves the right to make changes to any information here in at any time without notice.

The information, diagrams and other data in this manual are correct and reliable; however, ABOV Semiconductor is in no way responsible for any violations of patents or other rights of the third party generated by the use of this manual.

Table of Contents

List of Figures

MC96FR332A

CMOS 8-bit Flash Microcontroller : UR

1. OVERVIEW

1.1 Description

The MC96FR332A is an advanced 8-bit microcontroller based on CMOS process with 32K Bytes of Flash. This is a powerful device which provides a highly flexible and cost effective solution to many embedded control applications.

The MC96FR332A provides the following features : 32K Bytes of embedded FLASH ROM[NOTE1], 1024 Bytes of XRAM, 256 Bytes of IRAM, 8/16-bit Timer/Counter, WDT, WT, 10-bit PWM, USART(w/ SPI function), I2C, Carrier Generator, 8-bit Basic Interval Timer, Watch Timer and Clock control circuit. It also provides one dedicated output pin which has large current drivability specialized for remote control application. Additionally, the MC96FR332A supports power saving modes to reduce power consumption.

[NOTE1] In this document, the ROM means non-volitile memory which is read-writable.

Device Name	FLASH size	IRAM	XRAM	I/O PORT	Package
MC96FR332A	32KB	256B	1024B	23 / 27	28 TSSOP, 28/32 SOP

1.2 Features

- **CPU**
  8-bit CISC Core (8051 Compatible, 2 clocks per cycle)
- **32K Bytes On-chip FLASH**
  Endurance : 10,000 times
  Retention : 10 years
- **XRAM**
  1024 Bytes
- **IRAM**
  256 Bytes
- **General Purpose I/O**
  23/27 Ports (P0[7:0],P1[7:0],P2[2:0],P3[7:0])
- **One Basic Interval Timer**
- **Timer / Counter**
  8-bit×2ch(16-bit×1ch) + 16-bit×2ch
- **10-bit PWM(Using Timer0,1)**
- **One Watch Dog Timer**
- **One Watch Timer**
- **Two USART(with SPI feature)**
- **One I2C**
- **One Carrier Generator**
- **Key scan module**
  P0[7:0], P1[7:0]
- **Interrupt Sources**
  External : 4
  Pin Change Interrupt(P0) : 1
  USART: 4
  I2C : 1
  Key scan : 1
  Carrier Generator : 1
  WDT : 1
  WT : 1

BIT : 1
Timer0,1,2,3 : 4
FLASH : 1

- **Power On Reset**
- **Programmable Brown-Out Detector**
- **Minimum Instruction Execution Time**
  200ns (@10MHz, 1 Cycle NOP Instruction)
- **Power down mode**
  SLEEP, STOP mode
- **Operating Frequency**
  1 ~ 12MHz
- **Operating Voltage**
  1.75V ~ 5.5V (@ 1 ~ 12MHz)
- **Operating Temperature**
  -40 ~ +85℃
- **PKG Type**
  28 TSSOP, 28/32 SOP
  Available Pb free package

1.3 Ordering Information

Device name	ROM size	IRAM size	XRAM size	Package
MC96FR332AR	32KB FLASH	256B	1024B	28 TSSOP
MC96FR332AM				28 SOP
MC96FR332AD				32 SOP

Table 1-1 Ordering Information

1.4 Development Tools

1.4.1 Compiler

ABOV semiconductor does not provide any compiler for MC96FR332A. As the CPU core of MC96FR332A is Mentor 8051, you can use all kinds of third party's standard 8051 compiler.

1.4.2 OCD emulator and debugger

OCD(On Chip Debugger) program is a debugging software for ABOV semiconductor's 8051 MCU series. OCD uses only two lines to download a user code, to read and modify the internal memory or SFR(Special Function Register)s. And also OCD controls MCU's internal debugging logic, which means OCD controls emulation, step run, monitoring, etc.

OCD debugger program works on Microsoft-Windows NT, 2000, XP, Vista(32-bit) operating system.

If you want to see details more, please refer to OCD debugger manual. You can download debugger S/W and manual from out web-site.

The connecting pins between PC and MCU is as follows :

- DSCL (P2[1] of MC96FR332A)
- DSDA (P2[2] of MC96FR332A)

Figure 1-1 OCD Software and Connector

1.4.3 Programmer

To program or download user code into the ROM of MC96FR332A, ABOV semiconductor provides several tools. As a single programmer which can program only one chip at a time, there are PGMPlus for parallel programming and ISP/OCD for serial programming and debugging. On the other hand, you can program multi-chips at a time by using a gang programmer. Gang programmer can program up to 8 devices simultaneously.

1.4.3.1 Single programmer

1. PGMplus USB : This is a parallel programmer which is smaller and faster than our previous parallel programmer PGM Plus III.

Figure 1-2 PGMplus USB

2. Ez-ISP : This is one of stand alone type ISP tool. Notable thing is that it provides power to the target MCU.

Figure 1-3 Ez-ISP

3. OCD emulator : You can program or debug the MCU via OCD. Because the OCD supports ISP(In System Programming), it does not require additional H/W except for developer's target system.

1.4.3.2 Gang programmer

The gang programmer can program maximum 8 MCUs at a time. So it is mainly used in mass production line. As gang programmer is standalone type, it does not require host PC.

Figure 1-4 Gang programmer

2. BLOCK DIAGRAM

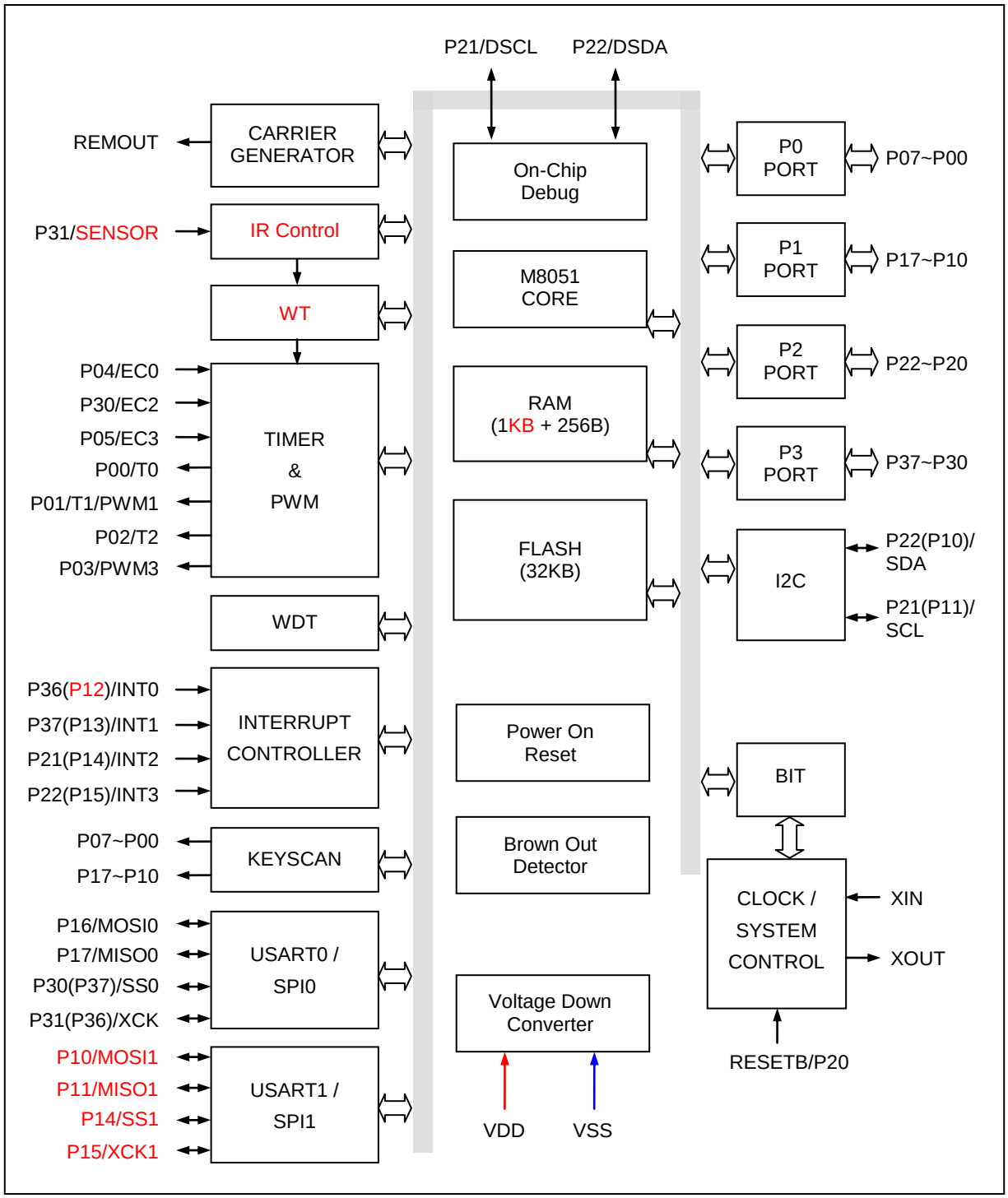


Figure 2-1 Block Diagram of MC96FR332A

PIN	Type	Option	Remarks
P20	I/O	RESETB	FUSE Control

3. PIN CONFIGURATIONS

28 TSSOP (MC96FR332AR)

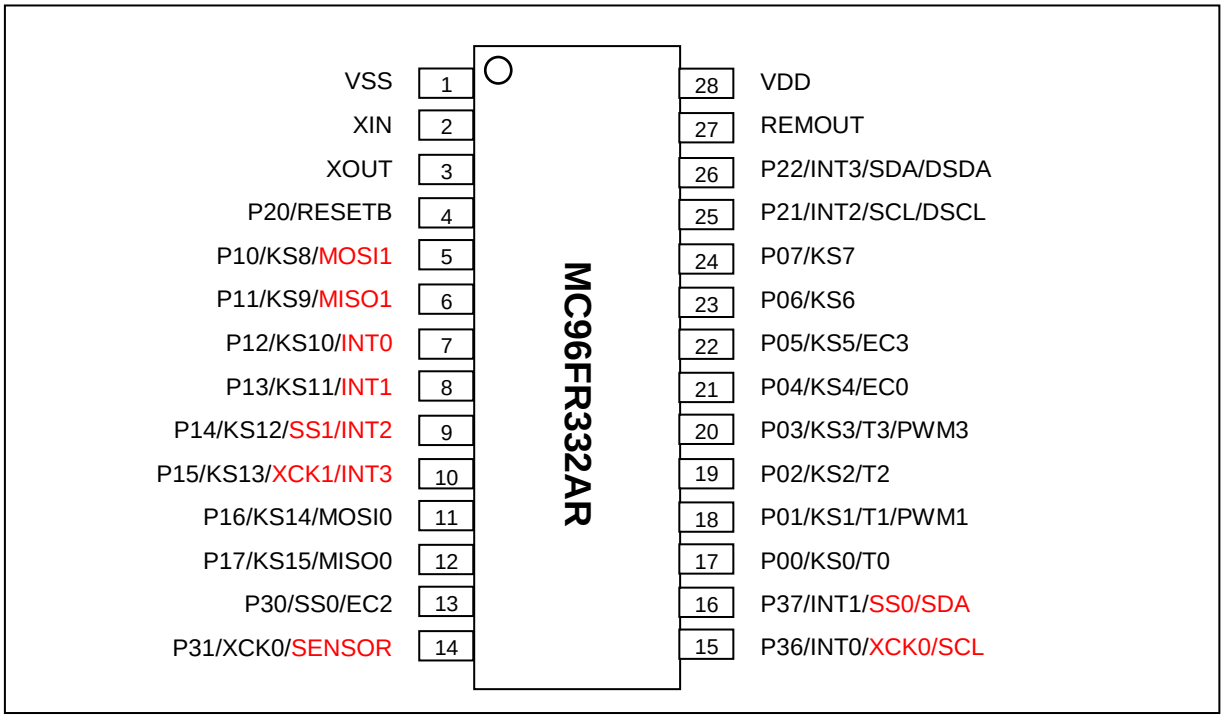


Figure 3-1 28 TSSOP Pinout MC96FR332AR

28 SOP (MC96FR332AD)

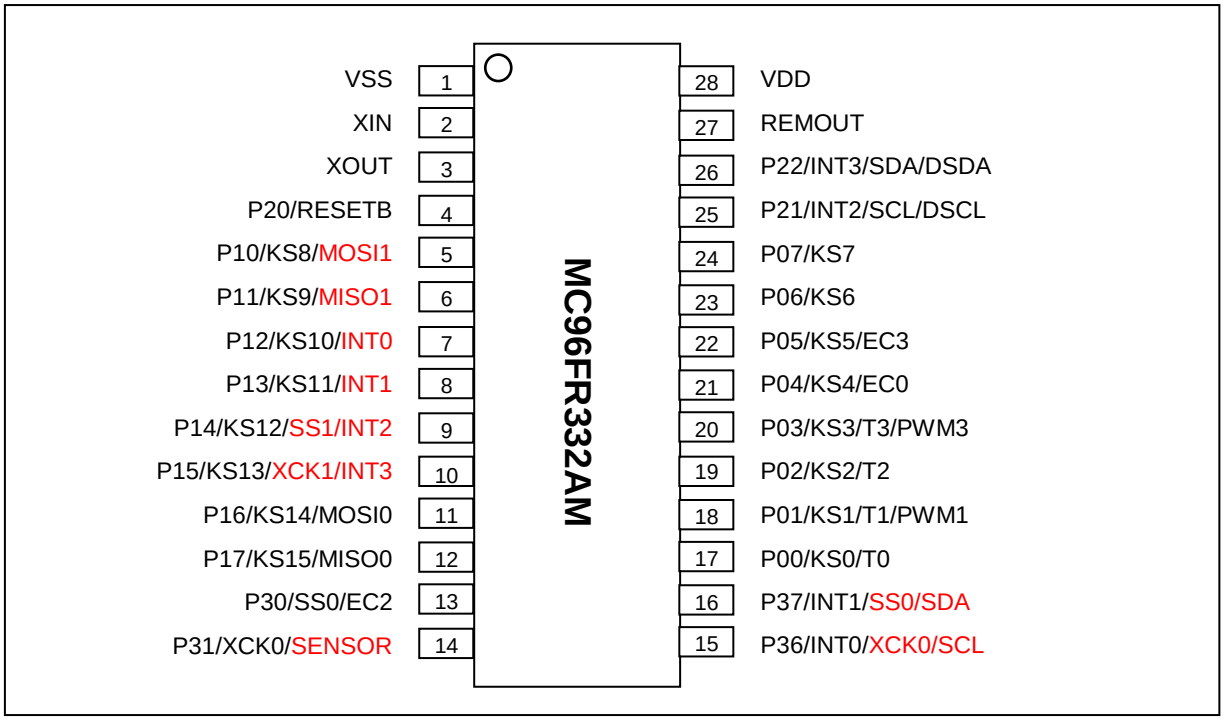


Figure 3-2 28 SOP Pinout MC96FR332AM

32 SOP (MC96FR332AD)

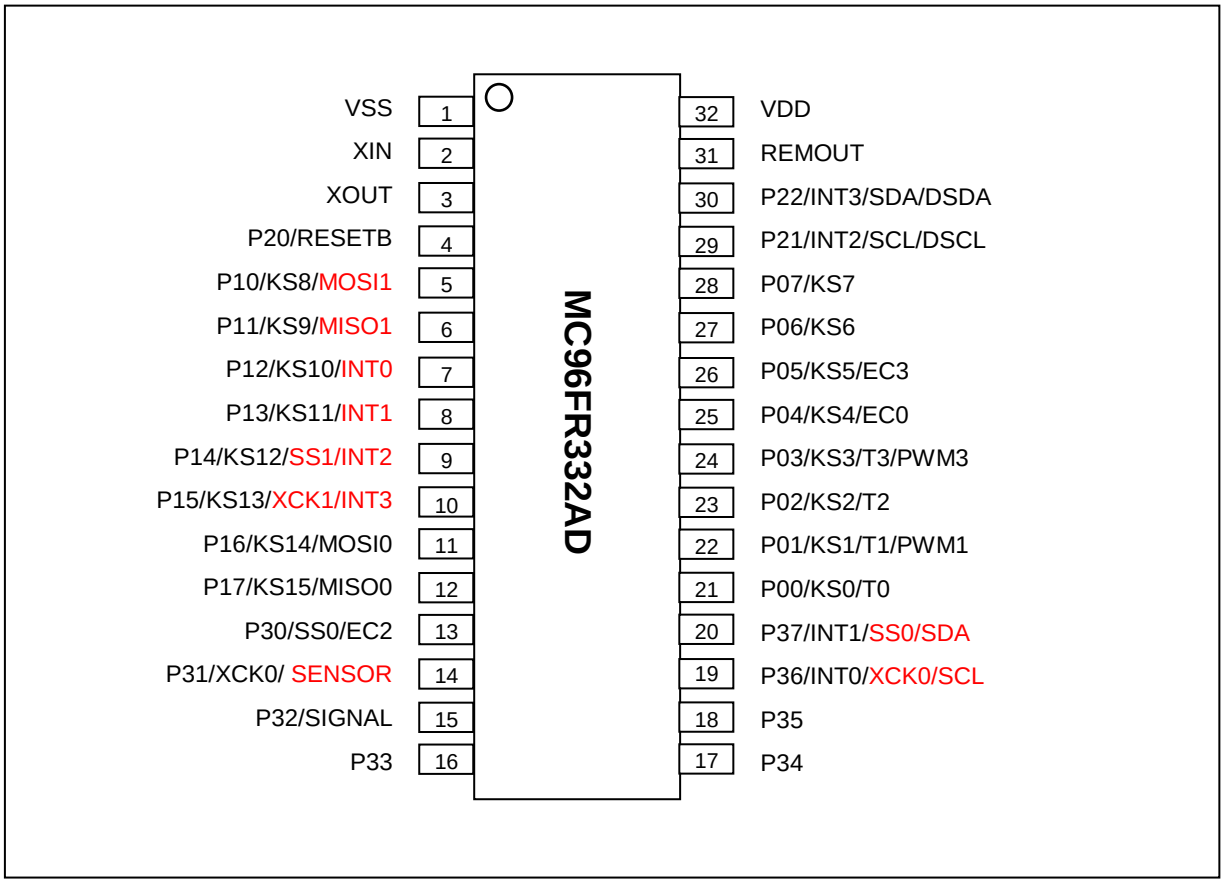


Figure 3-3 32 SOP Pinout MC96FR332AD

4. PACKAGE DIMENSION

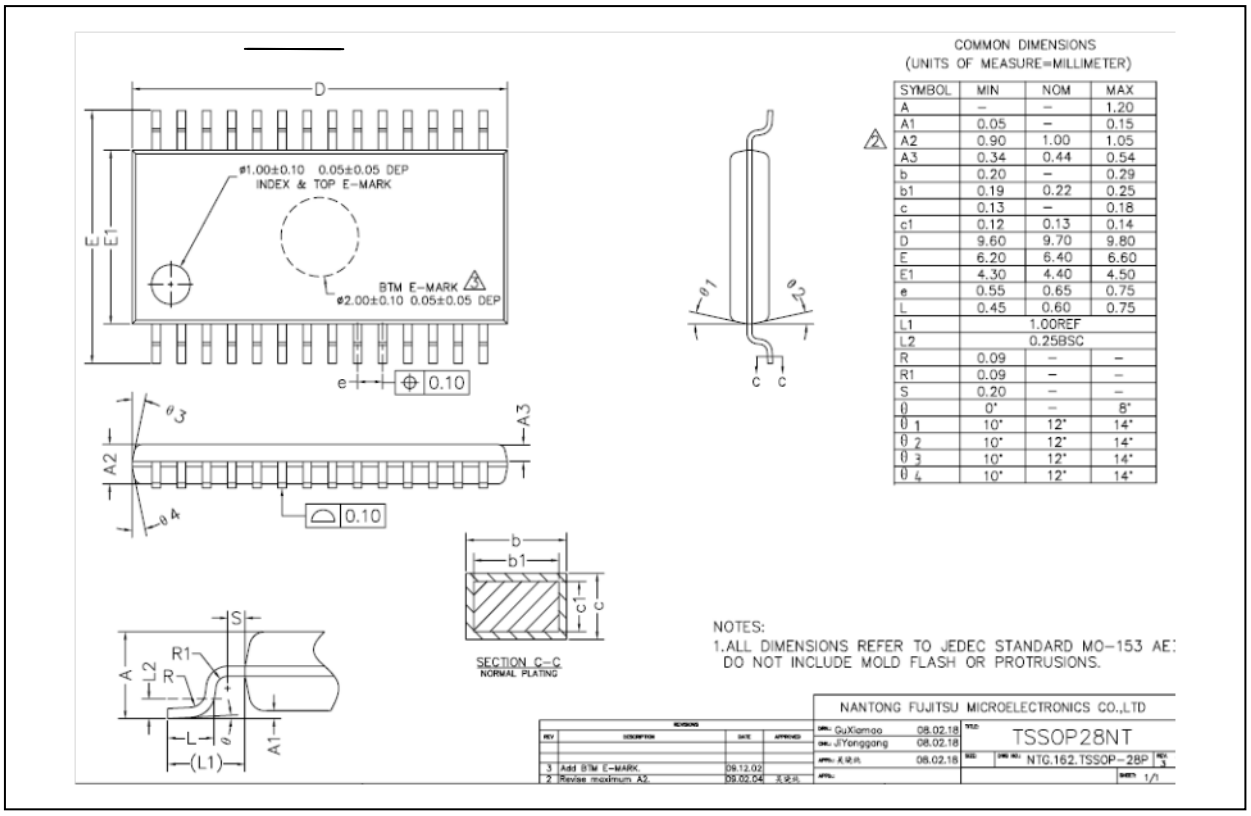


Figure 4-1 PKG DIMENSION (28 TSSOP)

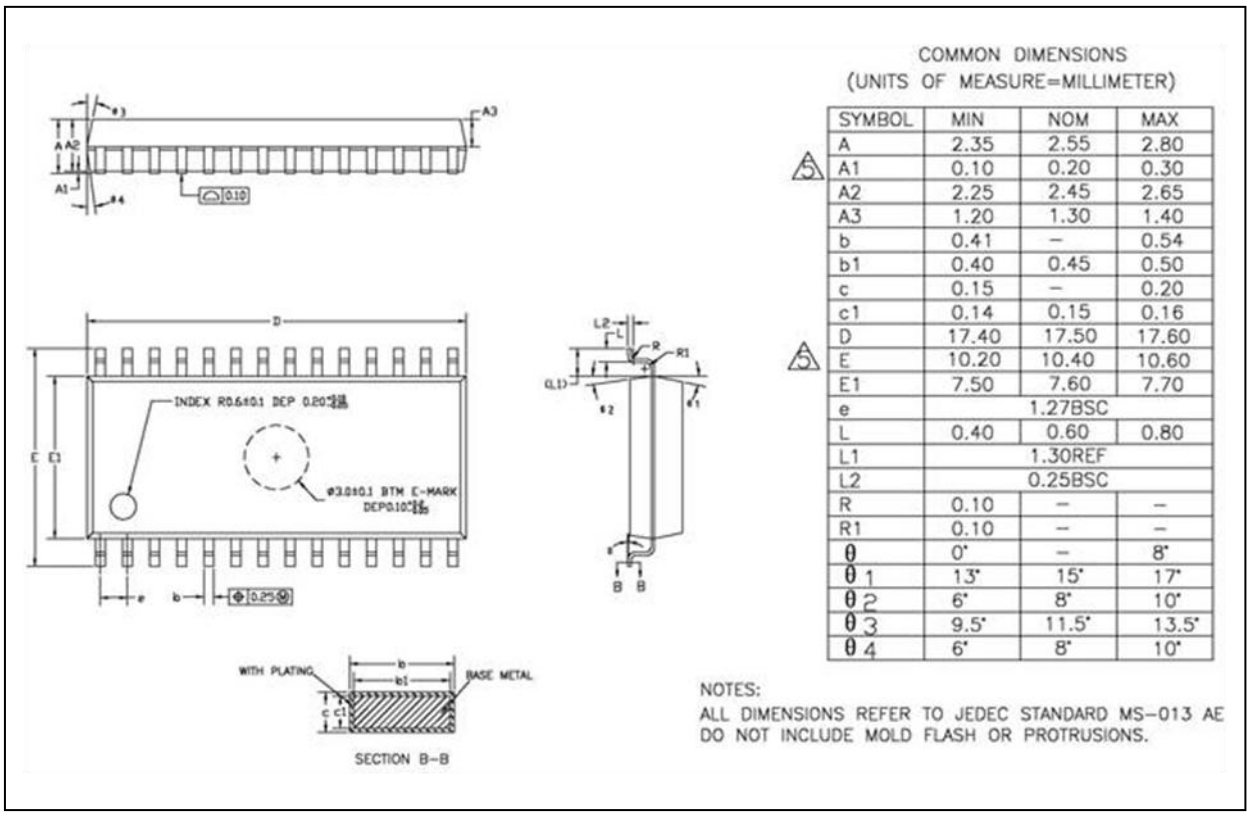


Figure 4-2 PKG DIMENSION (28 SOP)

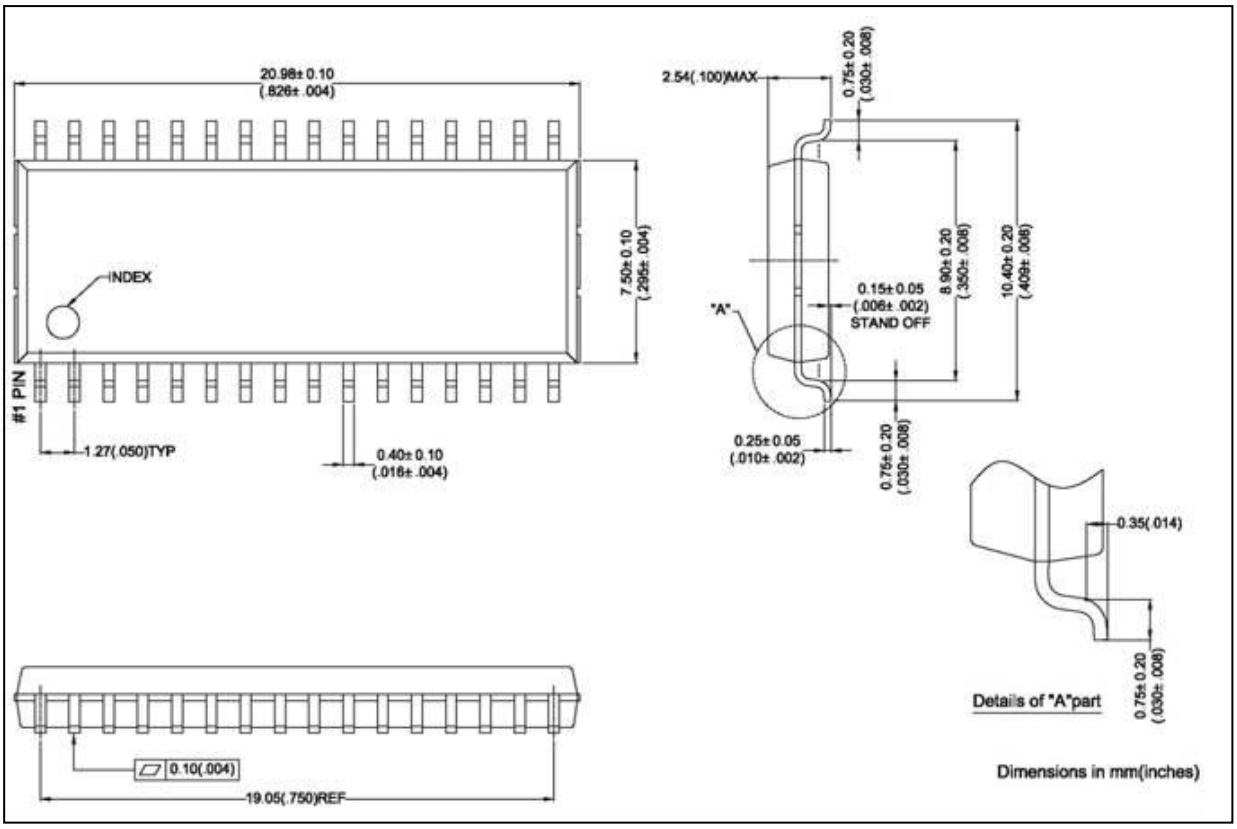


Figure 4-3 PKG DIMENSION (32 SOP)

5. PIN DESCRIPTION

PIN Name	I/O	Function	@RESET	Shared with
P00	I/O	- 8-bit I/O port, P0. - Can be set in input or output mode bitwise. - Internal pull-up resistor can be activated by setting PxnPU bit in PxPU register when this port is used as input port. - Can be configured as an open drain output mode by setting PxnOD bit in PxOD register.	Input	KS0/T0
P01				KS1/T1/PWM1
P02				KS2/T2
P03				KS3/T3/PWM3
P04				KS4/EC0
P05				KS5
P06				KS6
P07				KS7
P10	I/O	8-bit I/O port, P1. - Can be set in input or output mode bitwise. - Internal pull-up resistor can be activated by setting PxnPU bit in PxPU register when this port is used as input port. - Can be configured as an open drain output mode by setting PxnOD bit in PxOD register.	Input	KS8/MOSI1
P11				KS9/MISO1
P12				KS10/INT0 NOTE0
P13				KS11/INT1 NOTE0
P14				KS12/SS1/ INT2 NOTE0
P15				KS13/XCK1/ INT3 NOTE0
P16				KS14/MOSI0
P17				KS15/MISO0
P20	I/O	- 3-bit I/O port, P2. - Can be set in input or output mode bitwise. - Internal pull-up resistor can be activated by setting PxnPU bit in PxPU register when this port is used as input port. - Can be configured as an open drain output mode by setting PxnOD bit in PxOD register.	Input	RESETB NOTE1
P21				INT2/DSCL/ SCL NOTE2
P22				INT3/DSDA/ SDA NOTE2
-	-			-

NOTE0 INT3,2,1,0 can be triggered on P2[5:2] ports when appropriate bits in PSR0 register are set.

NOTE1 When P20 is used as a external reset pin(=RESETB) by the FUSE configuration, this pin is configured as an input port with internal pull-up resistor on.

NOTE2 SDA and SCL ports can be switched to P3[7:6] ports when appropriate bits in PSR0 register are set.

PIN Name	I/O	Function	@RESET	Shared with
P30	I/O	- 8-bit I/O port, P3. - Can be set in input or output mode bitwise. - Internal pull-up resistor can be activated by setting PxnPU bit in PxPU register when this port is used as input port. - Can be configured as an open drain output mode by setting PxnOD bit in PxOD register.	Input	SS0
P31				XCK0/ SENSOR
P32				-
P33				-
P34				-
P35				-
P36				INT0/XCK0
P37				INT1/SS0
XIN	I	Oscillator input		-
XOUT	O	Oscillator output		-
REMOUT	O	Push-pull high current output		-

6. PORT STRUCTURES

6.1 General Purpose I/O Port

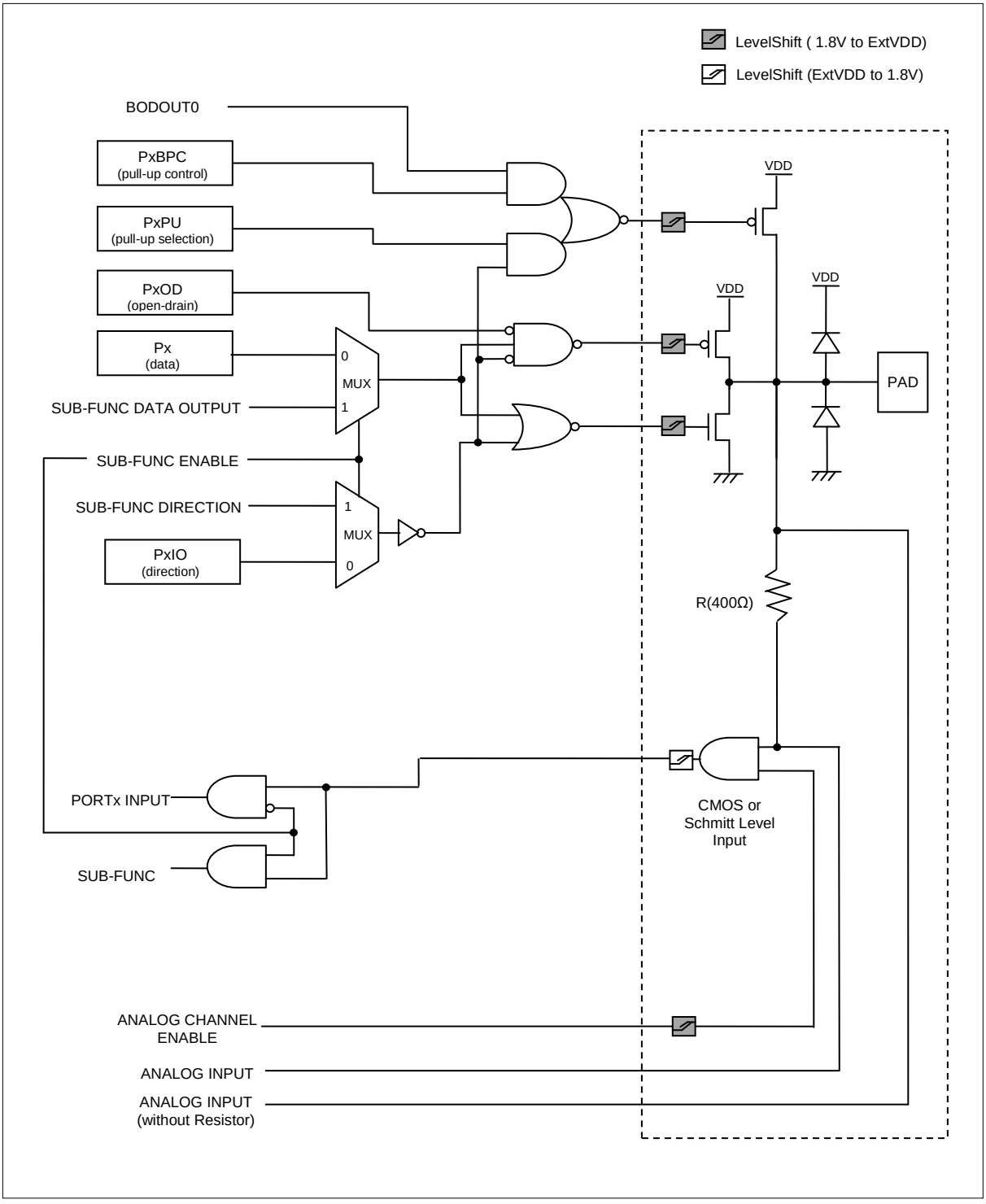


Figure 6-1 General I/O

6.2 External Interrupt I/O Port

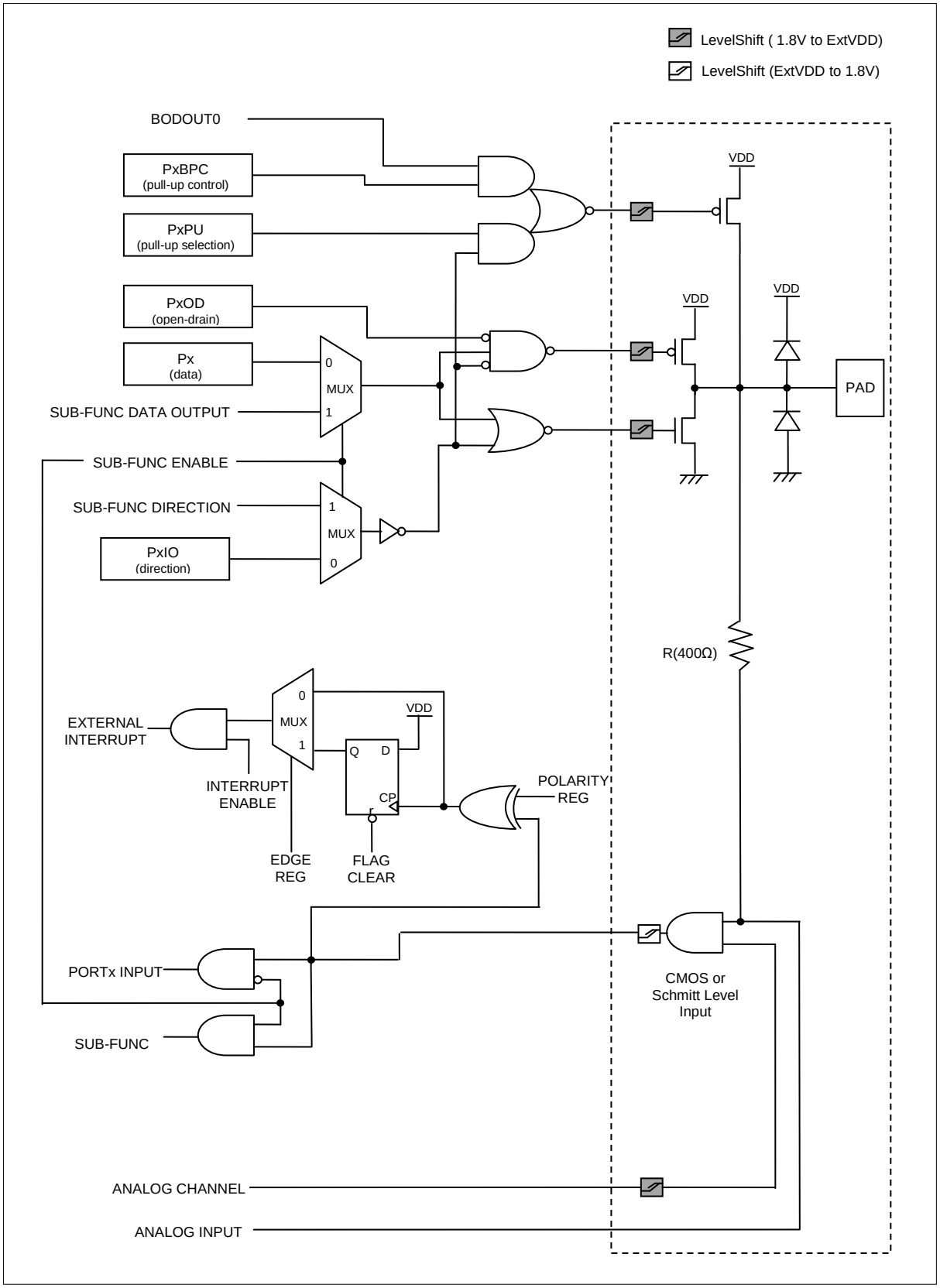


Figure 6-2 I/O with external interrupt function

7. ELECTRICAL CHARACTERISTICS

7.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Supply Voltage	VDD	-0.3~+6.5	V
	VSS	-0.3~+0.3	V
Normal Voltage Pin	VI	-0.3~VDD+0.3	V
	VO	-0.3~VDD+0.3	V
	IOH	10	mA
	∑IOH	80	mA
	IOL	20	mA
	∑IOL	160	mA
Total Power Dissipation	PT	600	mW
Storage Temperature	TSTG	-45~+125	℃

Table 7-1 Absolute Maximum Ratings

NOTE Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

7.2 RECOMMENDED OPERATING CONDITION

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Supply Voltage	VDD	f_{XIN}=1.0~12MHz	1.8	-	5.5	V
Operating Temperature	TOPR	VDD=1.65~5.5V	-40	-	85	℃
Operating Frequency	FOPR	f_{XIN}	1	-	10	MHz

Table 7-2 Recommended Operating Condition

7.3 VOLTAGE DROPOUT CONVERTER(VDC) CHARACTERISTICS

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Operating Voltage		-	1.62	-	6.05	V
Operating Temperature		-	-40	-	+85	℃
Regulation Voltage		-	1.62	1.8	1.98	V
Drop-out Voltage		-	-	-	0.02	V
Current Drivability		RUN	-	10	-	mA
		STOP	-	10	-	uA
Operating Current	IDD	RUN	-	-	1	mA
	SIDD	STOP	-	-	1	uA
Mode Transition Time	TRAN	STOP to RUN	-	-	200	uS

Table 7-3 Voltage Dropout Converter Characteristics

NOTE The operating modes of VDC itself are as follows.

RUN When the MC96FR332A is in normal operating mode, the VDC should provide enough current to the entire chip. So, in this mode of operating condition, the VDC is set to “RUN” mode to accommodate MCU’s normal RUN mode.

STOP When the MC96FR332A enters STOP mode to save current consumption, all internal logics stop operation including x-tal oscillator . In this mode, the MC96FR332A makes the VDC to enter “STOP” mode , leading to least current consumption mode.

7.4 BROWN OUT DETECTOR(BOD) CHARACTERISTICS

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Operating Voltage		-	1.5	-	6.05	V
Operating Temperature		-	-40	-	+85	℃
Detection Level	$V_{BODOUT0}$	NOTE	1.55	1.65	1.75	V
	$V_{BODOUT1}$	NOTE	1.70	1.80	1.90	V
	$V_{BODOUT2}$	NOTE	1.90	2.00	2.10	V
	$V_{BODOUT3}$	NOTE	2.10	2.20	2.30	V
	$V_{BODOUT4}$	NOTE	2.30	2.40	2.50	V
Operating Current	IDD	-	-	-	50	uA
	SIDD	-	-	-	1	uA

Table 7-4 Brown Out Detector Characteristics

NOTE $V_{BODOUT0}$ is a voltage level and BODOUT0 flag indicating it can generate internal reset due to voltage drop. When the external power drops below the $V_{BODOUT0}$ voltage level, the BOD detects the power condition and makes the device enter STOP-like mode called BOD mode. When the external power is restored, a BOD reset is generated according to pre-defined sequence and the device is initialized. $V_{BODOUT1/2/3/4}$ also indicate voltage levels and BODOUT1/2/3/4 are these flags. When the external power drops below the level indicated in abov table, the associated flag is set to ‘1’ and these values can be read through the BODSR register. These flags may be used to monitor the status of battery charging.

7.5 POWER-ON RESET CHARACTERISTICS

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Operating Voltage		-	-	-	5.5	V
Operating Temperature		-	-40	-	+85	℃
RESET Release Level		-	1.3	1.4	1.5	V
Operating Current	IDD	-	-	-	10	uA
	SIDD	-	-	-	1	uA

Table 7-5 Power-On Reset Characteristics

7.6 DC CHARACTERISTICS

(VDD =1.8~5.5V, VSS =0V, f_{XIN}=10.0MHz, TA=-40~+85℃)

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Input Low Voltage	VIL1	P0,P1,P20 (Schmitt Trigger Input)	-0.5	-	0.2VDD	V
	VIL2	P2[2:1],P3 (Normal Input)	-0.5	-	0.2VDD	V
Input High Voltage	VIH1	P0,P1,P20 (Schmitt Trigger Input)	0.8VDD	-	VDD+0.5	V
	VIH2	P2[2:1],P3 (Normal Input)	0.7VDD	-	VDD+0.5	V
Output Low Voltage	VOL1	P0,P1,P2,P3 (IOL=10mA, VDD=4.5V)	-	-	1	V
Output High Voltage	VOH1	P0,P1,P2,P3 (IOH=-4.0mA, VDD=4.5V)	3.5	-	-	V
Input High Leakage Current	IIH	P0,P1,P2,P3			1	uA
Input Low Leakage Current	IIL	P0,P1,P2,P3	-1			uA
Pull-Up Resistors	RPU	P0,P1,P2,P3 (VDD=5.5V, TA=+25℃)	22	-	55	kΩ
Power Supply Current	IDD1	RUN Mode, f_{XIN}=10MHz@5V	-	-	15	mA
	IDD2	SLEEP Mode, f_{XIN}=10MHz@5V	-	-	12	mA
	IDD3	STOP Mode @5V	-	-	10	uA

Table 7-6 DC Characteristics

7.7 AC CHARACTERISTICS

(VDD=5.0V±10%, VSS=0V, TA=-40~+85℃)

Parameter	Symbol	PIN	MIN	TYP	MAX	Unit
Operating Frequency	fMCP	XIN	1	-	10	MHz
System Clock Cycle Time	tSYS	-	100	-	1000	ns
Oscillation Stabilization Time (8MHz)	tMST1	XIN, XOUT	-	-	10	ms
External Clock “H” or “L” Pulse Width	tCPW	XIN	90	-	-	ns
External Clock Transition Time	tRCP,tFCP	XIN	-	-	10	ns
Interrupt Input Width	tIW	INT0~INT3	2	-	-	tSYS
RESETB Input Pulse “L” Width	tRST	RESETB	-	8	-	us
External Counter Input “H” or “L” Pulse Width	tECW	EC0	2	-	-	tSYS
Event Counter Transition Time	tREC,tFEC	EC0	-	-	20	ns

Table 7-7 AC Characteristics

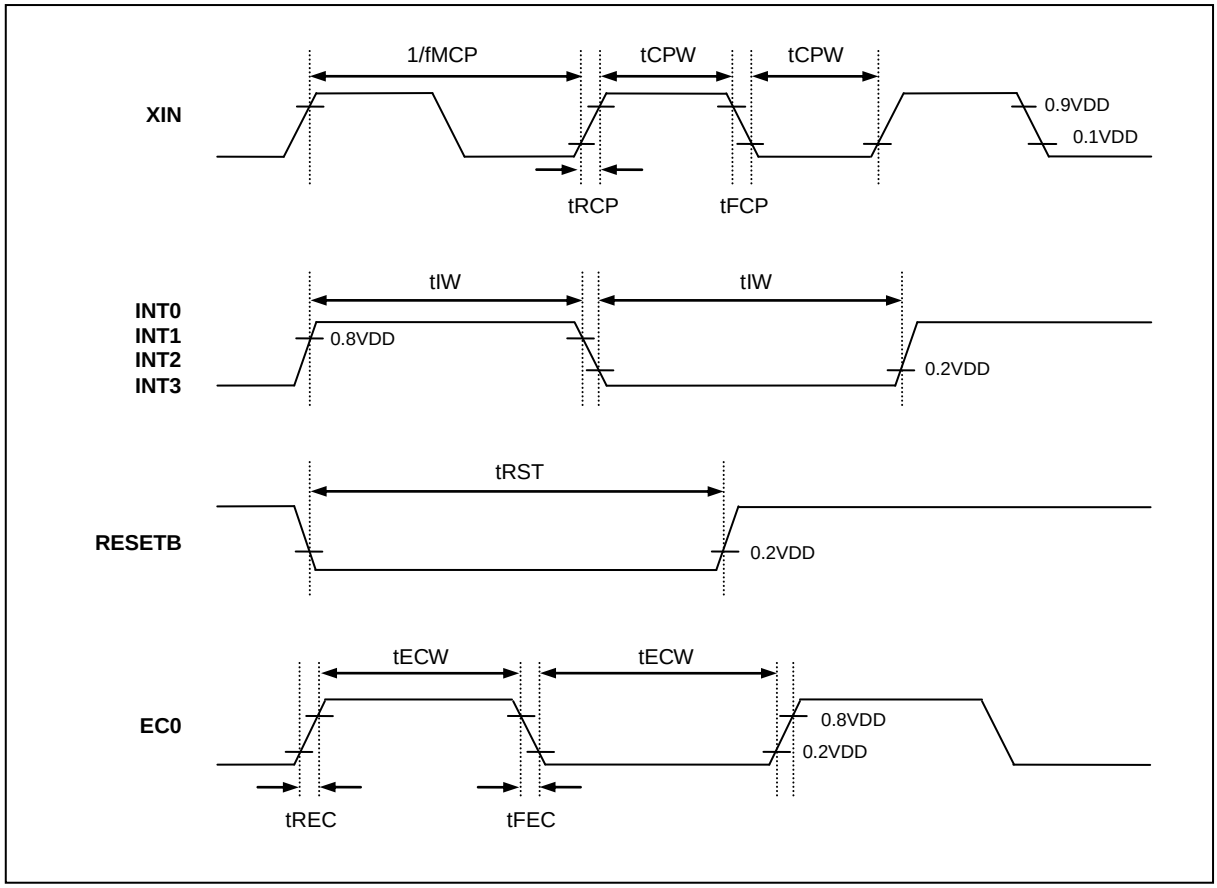


Figure 7-1 AC Timing

7.8 USART CHARACTERISTICS

The following table and figure show the timing condition of USART in SPI or Synchronous mode of operation. The USART is one of peripherals in MC96FR332A.[NOTE1].

(VDD =5.0V±10%, VSS =0V, TA=-40~+85℃)

Parameter	Symbol[NOTE2]	MIN	MAX	Unit
System clock period	t_{SCLK}	100	1000	ns
Clock (XCK) period	t_{XCK}	4	1028	t_{SCLK}
Clock (XCK) high time	t_{XCKH}	2	514	t_{SCLK}
Clock (XCK) low time	t_{XCKL}	2	514	t_{SCLK}
Lead time				
Master	t_{LEAD}	0.5 t_{XCK}	0.5 t_{XCK}	ns
Slave	t_{LEAD}	2 t_{SCLK}	-	
Lag time				
Master	t_{LAG}	0.5 t_{XCK}	0.5 t_{XCK}	ns
Slave	t_{LAG}	2 t_{SCLK}	-	
Data setup time (inputs)				
Master	t_{SIM}	2	2	t_{SCLK}
Slave	t_{SIS}	2	2	
Data hold time (inputs)				
Master	t_{HIM}	10	-	ns
Slave	t_{HIS}	10	-	
Data setup time (outputs)				
Master	t_{SOM}	2	2	t_{SCLK}
Slave	t_{SOS}	2	2	
Data hold time (outputs)				
Master	t_{HOM}	-10	-	ns
Slave	t_{HOS}	-10	-	
Disable time	t_{DIS}	1	2	t_{SCLK}

Table 7-8 Timing characteristics of USART in SYNC. or SPI mode of operations

[NOTE1] In synchronous mode, Lead and Lag time with respect to SS pin is ignored. And the case of UCPHA=0 is also applied to SPI mode only.

[NOTE2] All timing is shown with respect to 20% VDD and 80% VDD.

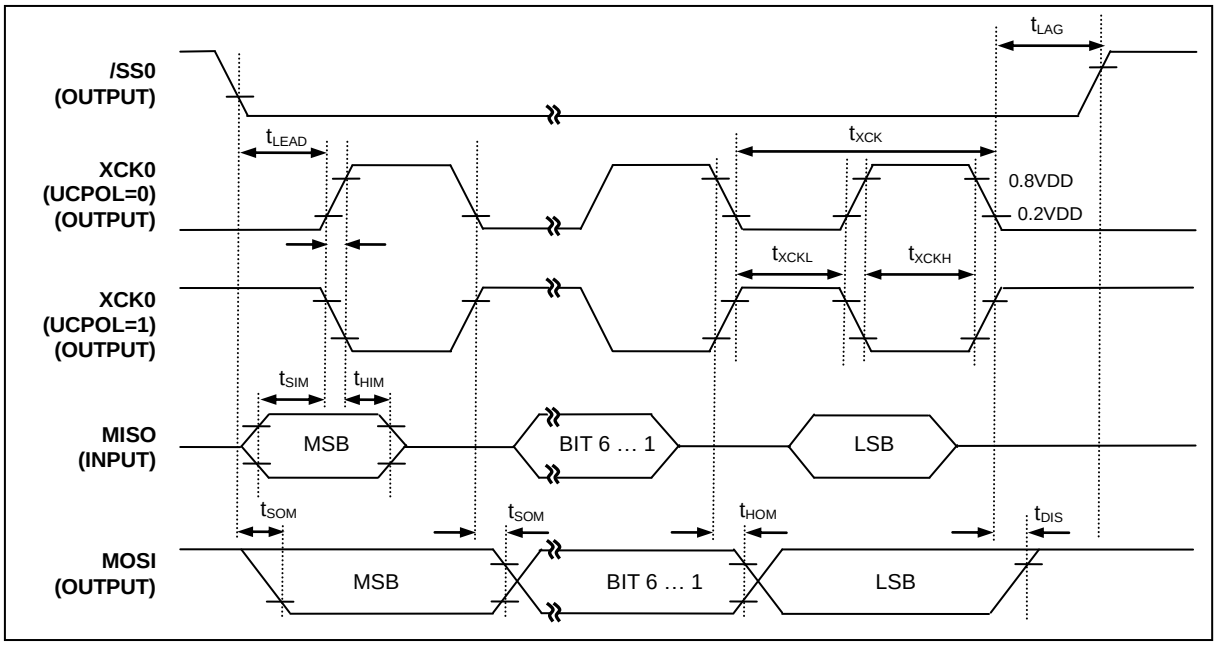


Figure 7-2 SPI master mode timing (UCPHA = 0, MSB first)

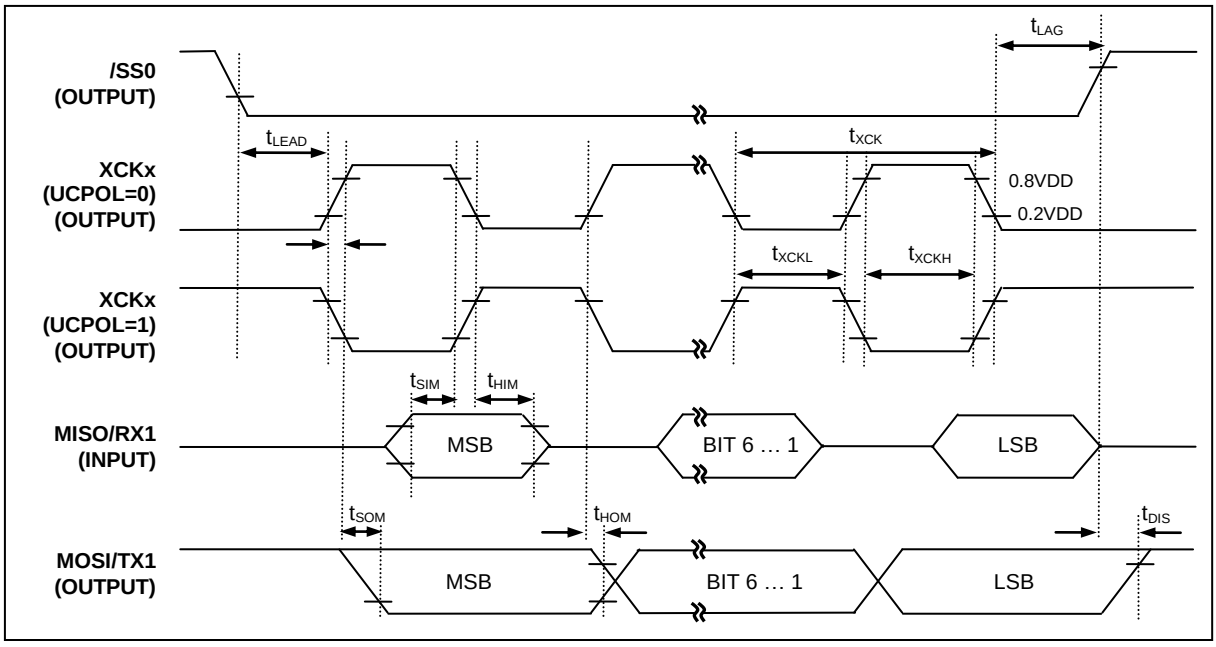


Figure 7-3 SPI / Synchronous master mode timing (UCPHA = 1, MSB first)

NOTE When in Synchronous mode, the START bit becomes MSB and the 1st or 2nd STOP bit becomes LSB.

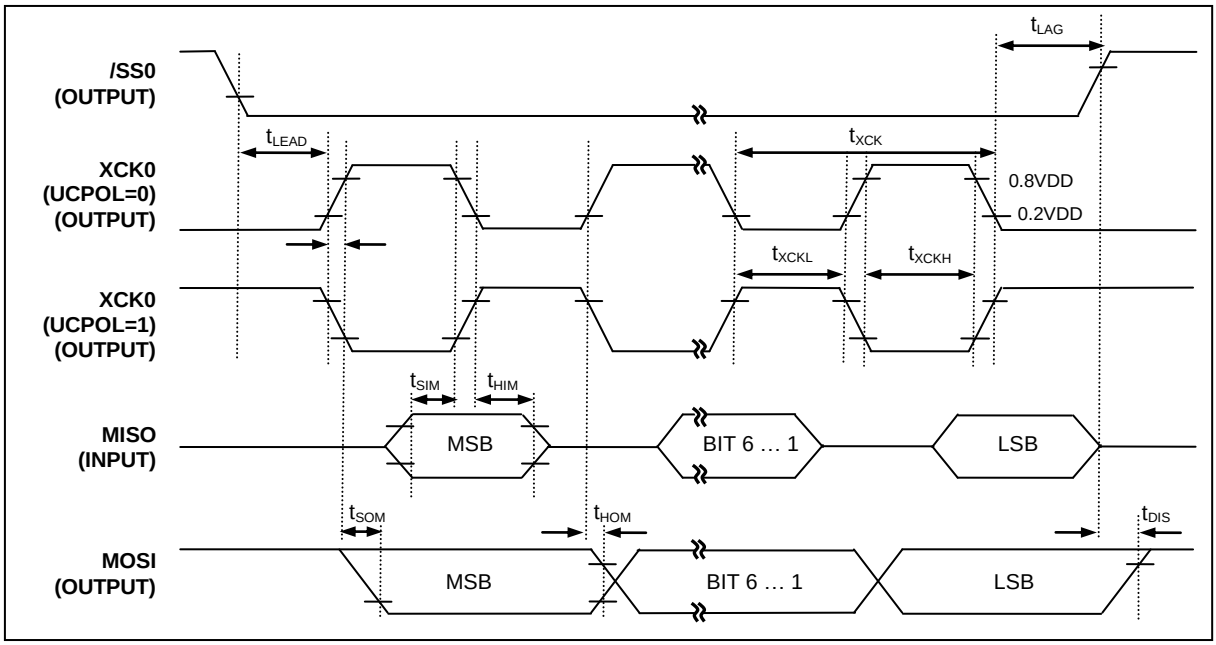


Figure 7-4 SPI slave mode timing (UCPHA = 0, MSB first)

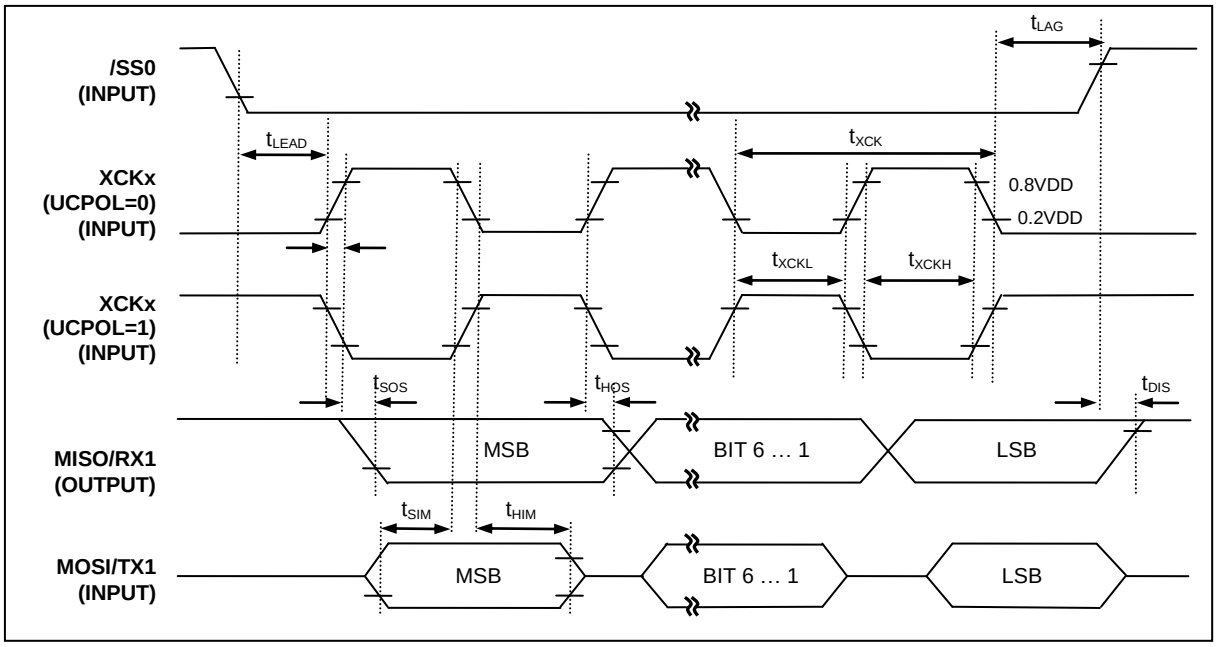


Figure 7-5 SPI / Synchronous slave mode timing (UCPHA = 1, MSB first)

NOTE1 When in Synchronous mode, the START bit becomes MSB and the 1st or 2nd STOP bit becomes LSB.

7.9 REMOUT PORT CHARACTERISTICS (To be modified..)

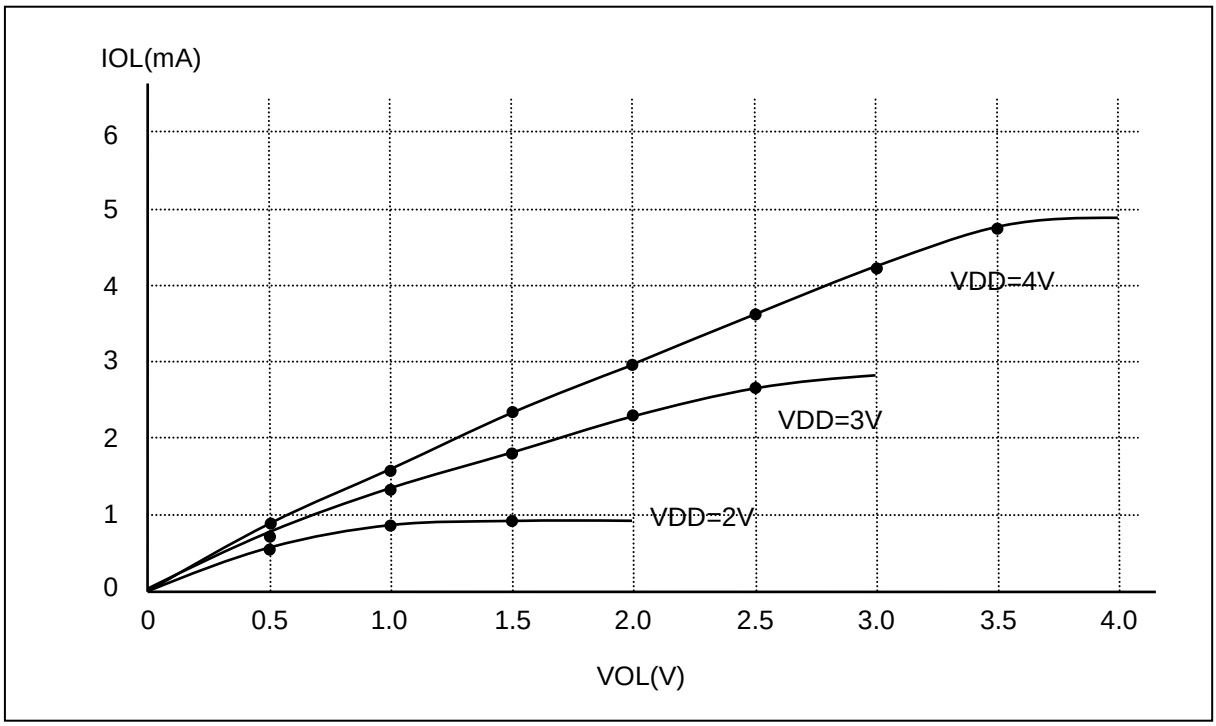


Figure 7-6 IOL vs VOL

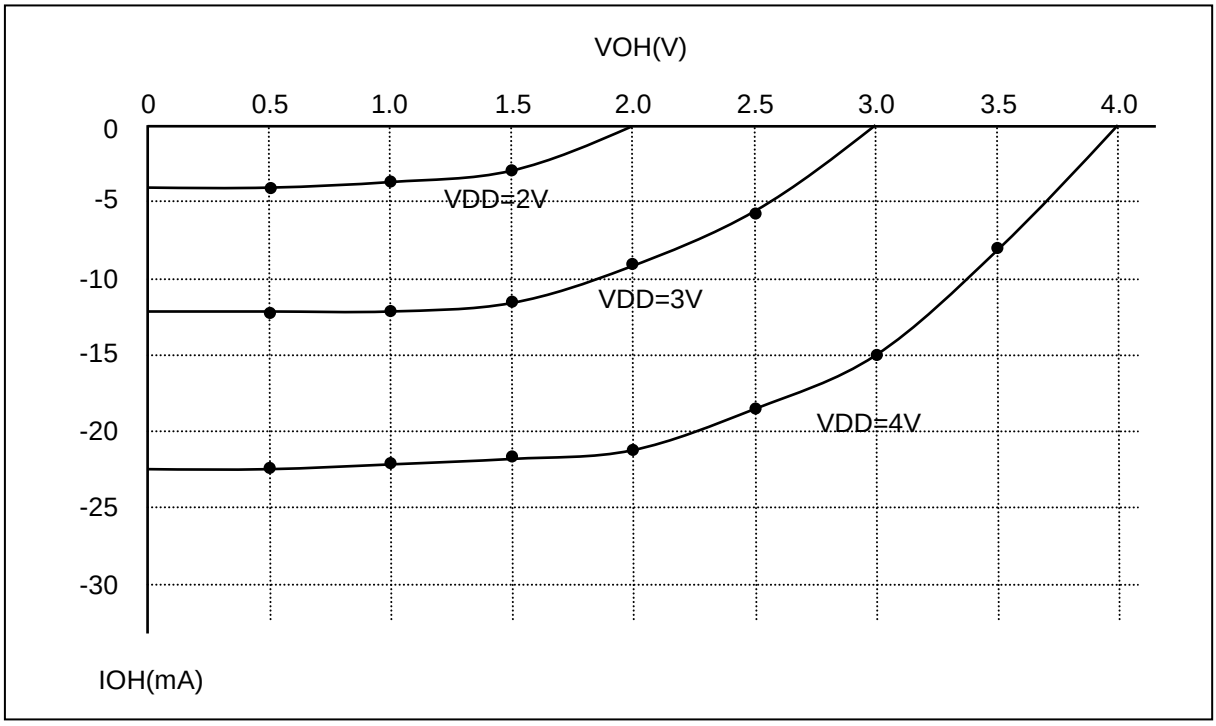


Figure 7-7 IOH vs VOH

7.10 TYPICAL CHARACTERISTICS

These graphs and tables provided in this section are for design guidance only and are not tested or guaranteed. In some graphs or tables the data presented are outside specified operating range (e.g. outside specified VDD range). This is for information only and devices are guaranteed to operate properly only within the specified range.

The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. “Typical” represents the mean of the distribution while “max” or “min” represents (mean + 3σ) and (mean - 3σ) respectively where σ is standard deviation.

8. MEMORY

The MC96FR332A has separate address spaces for Program and Data Memory. The logical separation of Program and Data Memory allows the Data Memory to be accessed by 8-bit addresses, which can be more quickly stored and manipulated by an 8-bit CPU. Nevertheless, 16-bit Data Memory addresses can also be generated through the DPTR register.

Program Memory contains user software and is read-only while the device is in normal running mode. But the Program Memory can be erased or programmed by ISP(In System Programming) method. The MC96FR332A can assign maximum 32KB to Program Memory which is divided into 2 banks.

Data Memory is composed of Internal RAM (IRAM), External RAM (XRAM). IRAM is read-writable and address space is 256B including Stack Pointer. XRAM has 8KB of memory depth and also read-writable.

NOTE The terms IRAM and XRAM are used just to classify kind of memory. XRAM doesn't have to reside outside the device. In MC96FR332A, both IRAM and XRAM reside in device.

8.1 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 32K bytes of program memory space located in one bank 0. The following figure shows a map of program memory in MC96FR332A. After reset, the CPU begins program execution from address 0000_H. All interrupt vector is assigned to their fixed location in program memory. An interrupt causes the CPU to jump to it's vector location, where the CPU commences execution of the service routine. External interrupt 0, for example, is assigned to location $000B_H$. If user wants to use external interrupt 0 as an interrupt source, its service routine must begin at location $000B_H$. If the interrupt is not used, its service location is available as general purpose program memory. Because all interrupt vector can use 8 bytes from each vector address NOTE, the service routine can reside entirely within that 8 bytes if an interrupt service routine is short enough (as is often the case in control applications). Normally an interrupt service routine is longer than 8 bytes, so the service routine starts with a jump instruction.

NOTE Refer to Table 10-2 Interrupt Vector Address.

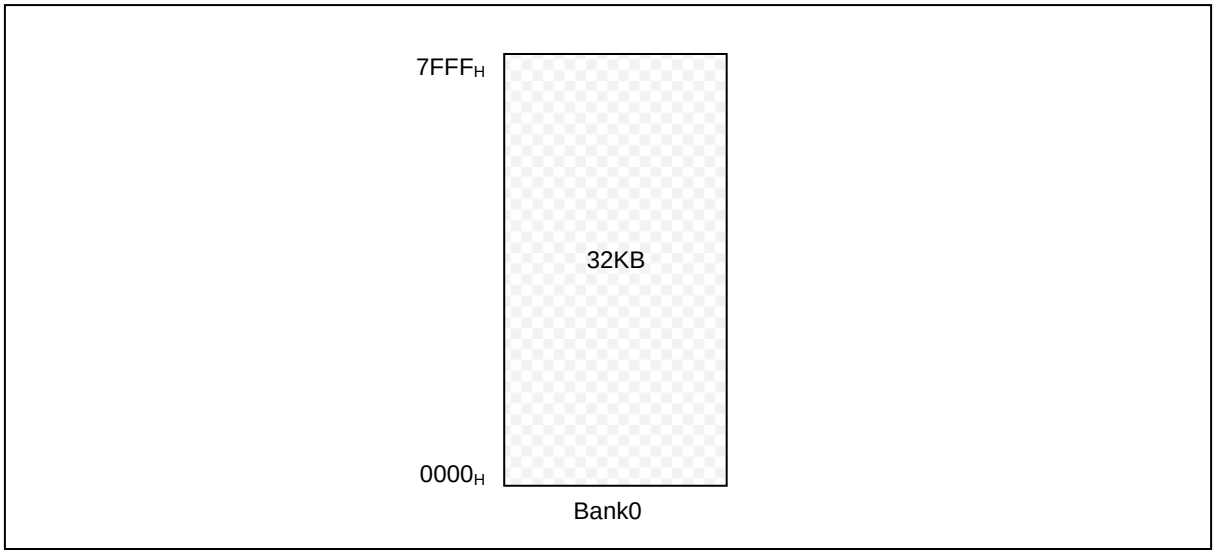


Figure 8-1 Program Memory

8.2 IRAM

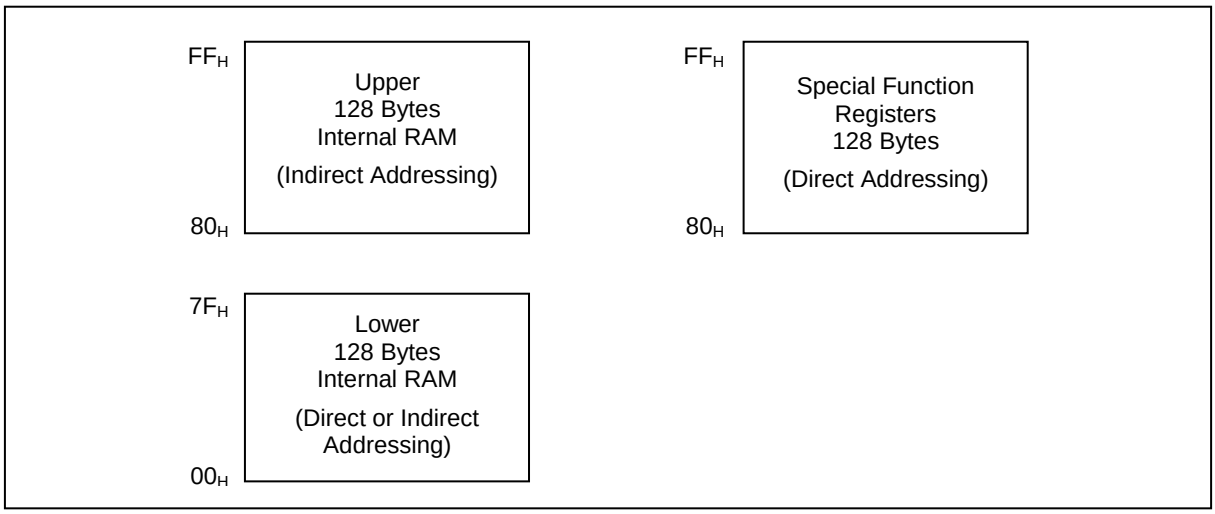


Figure 8-2 DATA MEMORY (IRAM)

Internal Data Memory is mapped in Figure 8-2. The memory space is shown divided into three blocks, which are generally referred to as the Lower 128, the Upper 128, and SFR space.

Internal Data Memory addresses are always one byte wide, which implies an address space of only 256 bytes. However, the addressing modes for internal RAM can in fact accommodate 384 bytes, using a simple trick. Direct addresses higher than $7F_H$ access one memory space, and indirect addresses higher than $7F_H$ access a different memory space. Thus Figure 8-2 shows the Upper 128 and SFR space occupying the same block of addresses, 80_H through FF_H, although they are physically separate entities.

The Lower 128 bytes of RAM are present in all devices using MCS-51 devices as mapped in Figure 8-2. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The MCS-51 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00_H through $7F_H$.

All of the bytes in the Lower 128 can be accessed by either direct or indirect addressing. The Upper 128 can only be accessed by indirect addressing. These spaces are used for user RAM and stack pointer.

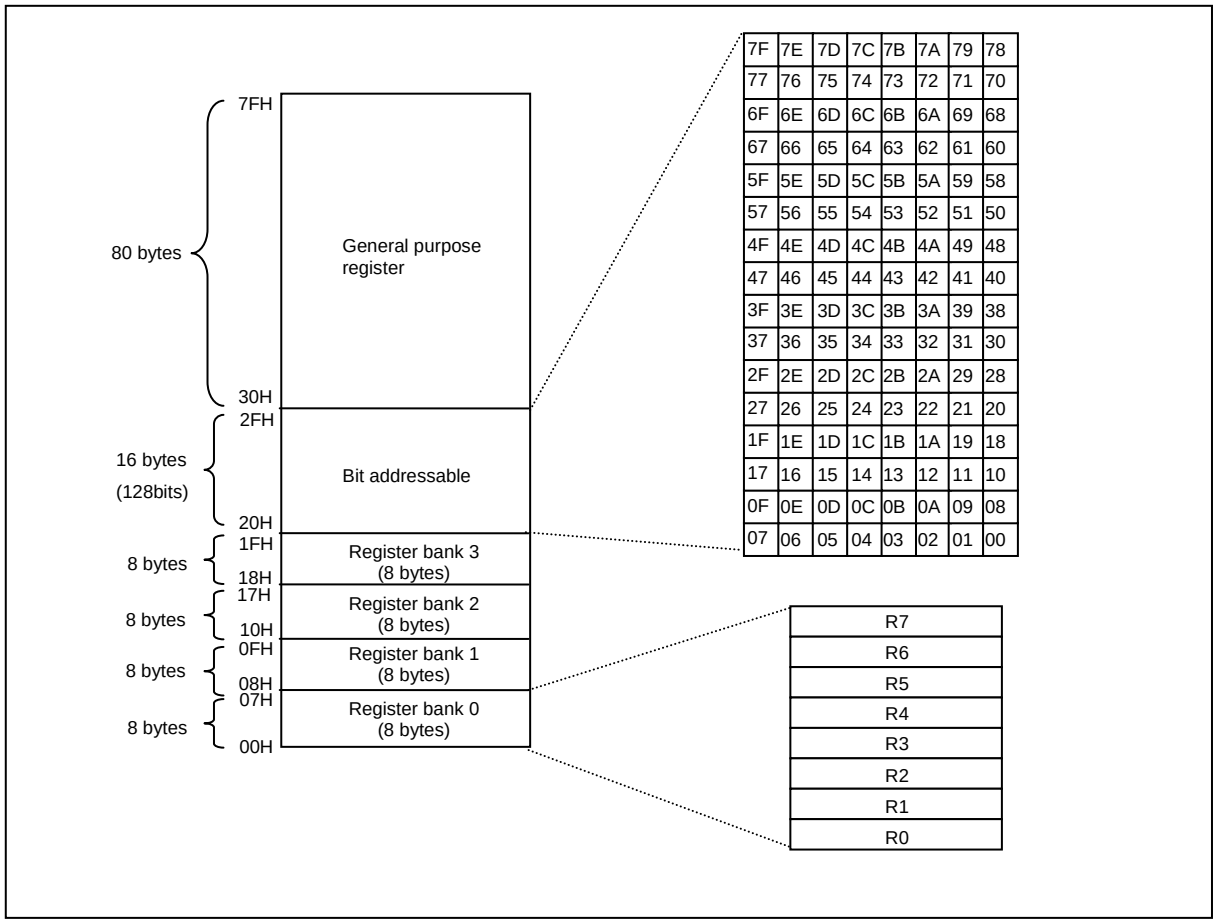


Figure 8-3 Lower 128 Byte of IRAM

8.2.1 Indirect Address Area

Note that in Figure 8.2 the SFRs and the indirect address RAM have the same addresses (80_H~FF_H). Nevertheless, they are two separate areas and accessed in two different ways.

For example the instruction

MOV 80H, #0AAH

writes $0AA_H$ to Port 0 which is one of the SFRs and the instruction

MOV R0, #80H

MOV @R0, #0BBH

writes $0BB_H$ in location 80_H of data RAM. Thus, after execution of both of the above instructions Port 0 will contain $0AA_H$ and location 80_H of the RAM will contain $0BB_H$.

Note that stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space in MC96FR332A.

8.2.2 Direct And Indirect Address Area

The 128 bytes of RAM which can be accessed by both direct and indirect addressing can be divided into 3 segments as listed below and shown in Figure 8.3.

Register Bank 0~3 Locations 00_H through $1F_H$ (32 bytes). ASM-51 and the device after reset default to register bank 0. To use the other register banks the user must select them in the software (refer to the MCS-51 Micro Assembler User's Guide). Each register bank contains 8 one-byte registers, 0 through 7.

Reset initializes the Stack Pointer to location 07_H and it is incremented once to start from location 08_H which is the first register (R0) of the second register bank. Thus, in order to use more than one register bank, the SP should be initialized to a different location of the RAM where it is not used for data storage (ie, higher part of the RAM).

Bit Addressable Area 16 bytes have been assigned for this segment, 20_H~$2F_H$. Each one of the 128 bits of this segment can be directly addressed (00_H~$7F_H$).

The bits can be referred to in two ways both of which are acceptable by the ASM-51. One way is to refer to their addresses, ie. 00_H to $7F_H$. The other way is with reference to bytes 20_H to $2F_H$. Thus, bits 0_H~7_H can also be referred to as bits 20.0~20.7, and bits 8_H ~F_H are the same as 21.0~21.7 and so on.

Scratch Pad Area Bytes 30_H through $7F_H$ are available to the user as data RAM. However, if the stack pointer has been initialized to this area, enough number of bytes should be left aside to prevent SP data destruction.

8.2.3 Special Function Registers

All I/O and peripherals operation for the MC96FR332A accessed via Special Function Registers (SFRs). These registers occupy direct Internal Data Memory space locations in the range 80_H to FF_H. Their names and addresses are given in the Table 8.9. Note these SFRs are implemented using flip-flops within the core, not as RAM.

The MC96FR332A has special registers which are provided by M8051 core. These are Program Counter(PC), Accumulator(A), B register(B), the Stack Pointer(SP), the Program Status Word(PSW), general purpose register(R0~R7) and DPTR (Data pointer register).

NOTE There's some address space in the SFRs which are not implemented. Reading these address space may return arbitrary value, and writing to these reserved SFR address may result in un-expected operation. So cautions are needed when accessing reserved address.

Accumulator (ACC) This register provides one of the operands for most ALU operations. It is denoted as 'A' in the instruction table included later in this document. On reset this register returns 00_H.

B register (B) This register provides the second operand for multiply or divide instructions. Otherwise, it may be used as a scratch pad register. On reset this register returns 00_H.

Stack Pointer (SP) The SP register contains the Stack Pointer. The Stack Pointer is used to load the program counter into Internal Data Memory during LCALL and ACALL instructions and to retrieve the program counter from memory during RET and RETI instructions. Data may also be saved on or retrieved from the stack using PUSH and POP instructions. Instructions that manipulate the stack automatically pre-increment or post-decrement the Stack Pointer so that the Stack Pointer always points to the last byte written to the stack, i.e. the top of the stack. On reset the Stack Pointer is set to 07_H.

It falls to the programmer to ensure that the location of the stack in Internal Data Memory does not interfere with other data stored therein.

Program Counter (PC) The Program Counter consists of two 8-bit registers PCH and PCL. This counter indicates the address of the next instruction to be executed. On reset, the program counter is initialized to reset routine address (PCH:00_H, PCL:00_H).

Data Pointer Register (DPTR) The Data Pointer (DPTR) is a 16-bit register which is used to form 16-bit addresses for External Data Memory accesses (MOVX A, @DPTR and MOVX @DPTR, A), for program byte moves (MOVC A, @A+DPTR) and for indirect program jumps (JMP @A+DPTR).

Two true 16-bit operations are allowed on the Data Pointer – load immediate (MOV DPTR, #data) and increment (INC DPTR).

Program Status Word (PSW) The PSW contains several status bits that reflect the current state of the CPU. The PSW, shown in Figure 8.1, resides in SFR space. It contains the Carry bit, the Auxiliary Carry (for BCD operations), the two register bank select bits, the Overflow flag, a Parity bit, and two user-definable status flags.

CY The Carry bit, other than serving the function of a Carry bit in arithmetic operations, also serves as the “Accumulator” for a number of Boolean operations.

AC The Auxiliary Carry bit, this bit is set when there is a carry from bit 3 of ALU or there is no borrow from bit 4 of ALU after operation.

RS0, RS1 The bits RS0 and RS1 are used to select one of the four register banks shown in Figure 8.3. A number of instructions refer to these RAM locations as R0 through R7. The selection of which of the four banks is being referred to is made on the basis of the bits RS0 and RS1 at execution time.

OV Overflow flag. This bit is set to “1” when an overflow occurs as the result of an arithmetic operation involving signs. An overflow occurs when the result of an addition or subtraction exceeds +127($7F_H$) or 128(80_H). The CLRV instruction clears the overflow flag. There is no set instruction. When the BIT instruction is executed, bit 6 of memory is copied to this flag.

P The Parity bit reflects the number of 1s in the Accumulator: P=1 if the Accumulator contains an odd number of 1s, and P=0 if the Accumulator contains an even number of 1s. Thus the number of 1s in the Accumulator plus P is always even.

Two bits in the PSW are uncommitted and may be used as general purpose status flags.

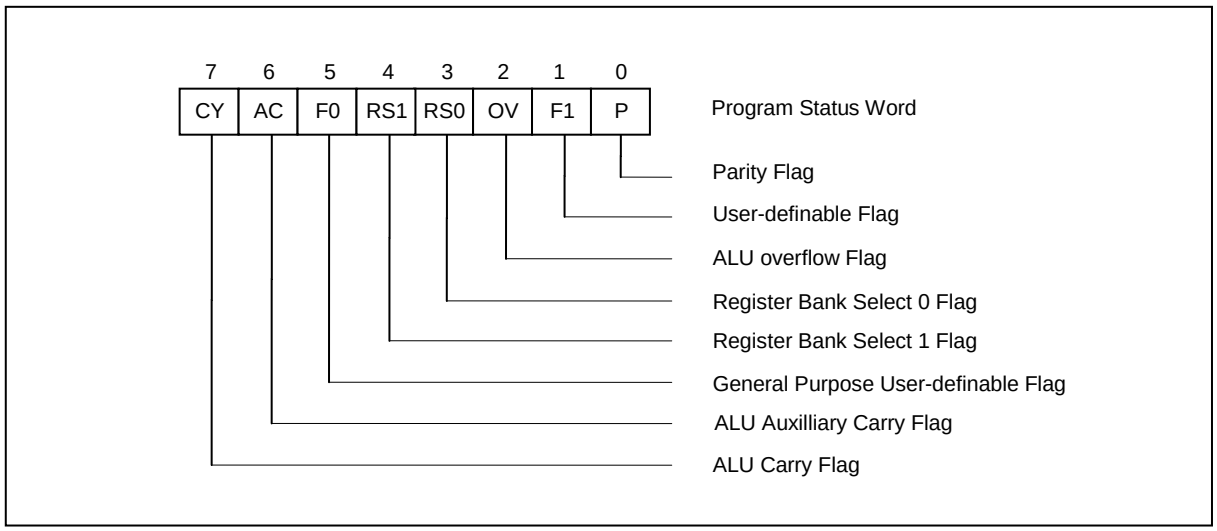


Figure 8-4 PSW Register

8.3 XRAM

There's another kind of RAM called XRAM (External RAM) in MC96FR332A and the size is 1KB, 0000_H through $03FF_H$. This address space is assigned to XDATA[NOTE] region and used for data storage.

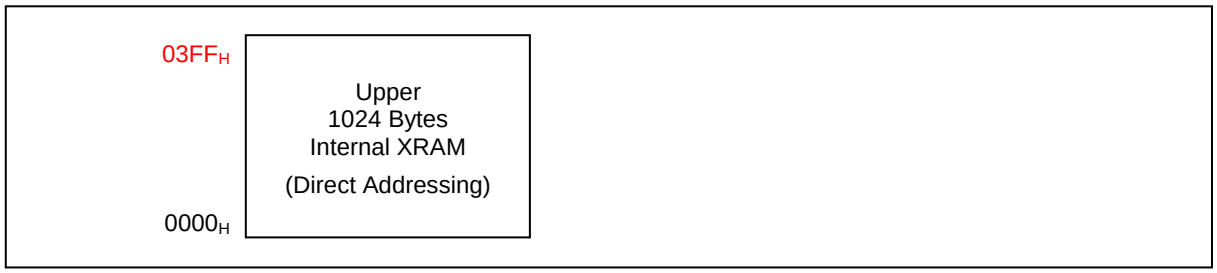


Figure 8-5 DATA MEMORY (XRAM)

NOTE XRAM, 64Bytes of page buffers and some eXtended SFR(XSFR) are assigned to XDATA area in MC96FR332A. And these address space are accessed via MOVX instruction.

8.4 Registers

8.4.1 SFR Map

-	*Reserved*
	M8051 Compatible

	0H/8H[NOTE]	1H/9H	2H/AH	3H/BH	4H/CH	5H/DH	6H/EH	7H/FH
$F8_H$	IP1 --00_0000	CFGRR	UCTRL11	UCTRL12	UCTRL13	USTAT1	UBAUD1	UDATA1
$F0_H$	B 0000_0000	WTCR0H	WTCR0L	WTCR1H	WTCR1L	WTCR2H	WTCR2L	KITSR
$E8_H$	RMR	FARH	FARM	FARL	FCR	FSR	FTCR	-
$E0_H$	ACC 0000_0000	FMR	UCTRL01	UCTRL02	UCTRL03	USTAT0	UBAUD0	UDATA0
$D8_H$	-	-	-	P5IO ----_0000	WTDRH	IRCC0	IRCC1	IRCC2
$D0_H$	PSW 0000_0000	WTMR	SMRR0	SMRR1	WTR1	WTR0	SRLC0	SRLC1
$C8_H$	-	T3CR2	T3CR	T3L/CDR3L/ PWM3DRL	T3H/CDR3H /PWM3DRH	T3DRL/PW M3PRL	T3DRH/PW M3PRH	T2L/T2DRL/ CDR2L
$C0_H$	P3	P0PC 0000_0000	RDBH	RDBL	RDRH	RDRL	T2CR	T2H/T2DRH /CDR2H
$B8_H$	IP --00_0000	-	RDCH	CFRH	CFRL	RDCL	RODR	ROB
$B0_H$	P2IO ----_-000	-	T0CR	T0/CDR0/T0 DR	T1CR	T1DR/ PWM1PR	T1/CDR1/ PWM1DR	PWM1HR
$A8_H$	IE 0000_0000	IE1	IE2	IE3	EIFLAG	EIEDGE	EIPOLA	EIENAB
$A0_H$	P1IO 0000_0000	-	EO 0000_0000	I2CSDHR	P4IO 0000_0000	I2CDR	I2CSAR	I2CSAR1
98_H	P0IO 0000_0000	-	-	P3IO 0000_0000	I2CMR	I2CSR	I2CSCLLR	I2CSCLHR
90_H	P2 ----_-000	-	PSR0	-	MEX1	MEX2	MEX3	MEXSP
88_H	P1 0000_0000	-	SCCR	BCCR	BITR	WDTMR	WDTR	BODSR
80_H	P0 0000_0000	SP 0000_0111	DPL 0000_0000	DPH 0000_0000	-	-	BODR 1000_0001	PCON 0000_0000

Table 8-1 SFR Map

NOTE These registers are bit-addressable.

Caution : Writing to reserved registers may result in un-expected function.

8.4.2 XSFR Map

Page Buffer (64Bytes)

	0H/8H	1H/9H	2H/AH	3H/BH	4H/CH	5H/DH	6H/EH	7H/FH
8078_H								
8070_H								
8068_H								
8060_H								
8058_H								
8050_H								
8048_H								
8040_H								
8038_H	PBUF_38	PBUF_39	PBUF_3A	PBUF_3B	PBUF_3C	PBUF_3D	PBUF_3E	PBUF_3F
8030_H	PBUF_30	PBUF_31	PBUF_32	PBUF_33	PBUF_34	PBUF_35	PBUF_36	PBUF_37
8028_H	PBUF_28	PBUF_29	PBUF_2A	PBUF_2B	PBUF_2C	PBUF_2D	PBUF_2E	PBUF_2F
8020_H	PBUF_20	PBUF_21	PBUF_22	PBUF_23	PBUF_24	PBUF_25	PBUF_26	PBUF_27
8018_H	PBUF_18	PBUF_19	PBUF_1A	PBUF_1B	PBUF_1C	PBUF_1D	PBUF_1E	PBUF_1F
8010_H	PBUF_10	PBUF_11	PBUF_12	PBUF_13	PBUF_14	PBUF_15	PBUF_16	PBUF_17
8008_H	PBUF_08	PBUF_09	PBUF_0A	PBUF_0B	PBUF_0C	PBUF_0D	PBUF_0E	PBUF_0F
8000_H	PBUF_00	PBUF_01	PBUF_02	PBUF_03	PBUF_04	PBUF_05	PBUF_06	PBUF_07
$2F58_H$	FTR	-	FUSE_CAL 2	-	-	FUSE_CON F	TEST_B	TEST_A
$2F50_H$	P0BPC	P1BPC	P2BPC	P3BPC	-	-	-	PSR1
$2F08_H$	-	P1OD 0000_0000	P2OD ----_-000	P3OD 0000_0000	-	-	XBANK	CSUMH
$2F00_H$	P0PU 0000_0000	P1PU 0000_0000	P2PU ----_-000	P3PU 0011_1100	-	-	CSUML	CSUMM

Table 8-2 eXtended SFR Map

Caution : Writing to reserved registers may result in un-expected function.

8.4.3 Compiler Compatible SFR

Refer to section 8.2.3 for detailed description of these registers.

ACC (Accumulator) **$E0_H$**

7	6	5	4	3	2	1	0
ACC							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

ACC	Accumulator

B (B Register) **$F0_H$**

7	6	5	4	3	2	1	0
B							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

B	B Register

SP (Stack Pointer) **81_H**

7	6	5	4	3	2	1	0
SP							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 07_H

SP	Stack Pointer

DPL (Data Pointer Low Byte) **82_H**

7	6	5	4	3	2	1	0
DPL							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

DPL	Data Pointer Low Byte

DPH (Data Pointer High Byte) **83_H**

7	6	5	4	3	2	1	0
DPH							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

DPH	Data Pointer High Byte

PSW (Program Status Word) **$D0_H$**

7	6	5	4	3	2	1	0
CY	AC	F0	RS1	RS0	OV	F1	P

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

CY	Carry Flag. Receives carry out from bit 1 for ALU operands.
AC	Auxiliary Carry Flag. Receives carry out from bit 1 of addition operands.
F0	General Purpose Status Flag
RS1	Register Bank Selection bit 1
RS0	Register Bank Selection bit 0
OV	Overflow Flag. Set by arithmetic operations.
F1	User-definable Flag
P	Parity of ACC. Set by hardware to 1 if it contains an odd number of 1s, otherwise it is reset to 0.

EO (Extended Operation Register) $A2_H$

7	6	5	4	3	2	1	0
-	-	-	TRAP_EN	-	DPSEL.2	DPSEL.1	DPSEL.0
R	R	R	R/W	R	R/W	R/W	R/W

Initial value : 00_H

TRAP_EN Select the instruction between software TRAP and MOVC @(DPTR++), A.

0 Select MOVC @(DPTR++), A instruction.

1 Select software TRAP instruction.

DPSEL[2:0] Select DPT R.

DPSEL2	DPSEL1	DPSEL0	
0	0	0	DPTR0 selected.
0	0	1	DPTR1 selected.
0	1	0	DPTR2 selected (if included).
0	1	1	DPTR3 selected (if included).
1	0	0	DPTR4 selected (if included).
1	0	1	DPTR5 selected (if included).
1	1	0	DPTR6 selected (if included).
1	1	1	DPTR7 selected (if included).

MEX1 (Memory Extension Register 1) 94_H

7	6	5	4	3	2	1	0
CB19	CB18	CB17	CB16	NB19	NB18	NB17	NB16
R	R	R	R	R/W	R/W	R/W	R/W

Initial value : 00_H

CB[19:16]	Current Bank
NB[19:16]	Next Bank

This register records the ‘current’ and ‘next’ memory bank number for program code.

MEX2 (Memory Extension Register 2) 95_H

7	6	5	4	3	2	1	0
MCM	MCB18	MCB17	MCB16	IB19	IB18	IB17	IB16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

MCM	Memory Constant Mode. Set to '1' when Memory Bank used.
MCB[18:16]	Memory Constant Bank (with MEX3.7)
IB[19:16]	Interrupt Bank

This register controls the current memory bank numbers for interrupt service routine code and for memory constants.

MEX3 (Memory Extension Register 3) 96_H

7	6	5	4	3	2	1	0
MCB19	UB1	UB0	MXB19	MXM	MXB18	MXB17	MXB16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

MCB19	Memory Constant Bank MSB. See MEX2.
UB[1:0]	Bits available to the user.
MXM	XRAM Bank selector. When set to '1', the MOVX Bank bits MX19-MX16 are used as XRAMA 19-16 instead of the Current Bank(CB).
MXB[18:16]	XRAM Bank.

This register chiefly controls the current memory bank number for external data memory.

MEXSP (Memory Extension Stack Pointer) 97_H

7	6	5	4	3	2	1	0
-	MEXSP6	MEXSP5	MEXSP4	MEXSP3	MEXSP2	MEXSP1	MEXSP0
-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : $7F_H$

MEXSP[6:0]	Memory Extension Stack Pointer.
NB[19:16]	Next Bank

This register is the Memory Extension Stack Pointer. It provides for a stack depth of up to 128 bytes (bit 7 is always 0). It is pre-incremented by call instructions and post-decremented by return instructions.

9. I/O PORTS

9.1 Introduction

The MC96FR332A has four I/O ports (P0, P1, P2, P3). Each port can be easily configured by software whether to use internal pull up resistor or not, whether to use open drain output or not, or whether the pin is input or output. Also P0 includes function that can generate interrupt when the state of P0 changes.

9.2 Register Description

9.2.1 Data Register (Px)

Data Register is a bidirectional I/O port. If ports are configured as output ports, data can be written to the corresponding bit in the Px. If ports are configured as input ports, the port value can be read from the corresponding bit in the Px.

9.2.2 Direction Register (PxIO)

The PxnIO bit in the PxIO register selects the direction of this pin. If PxnIO is written logic one, Pxn is configured as an output pin. If PxnIO is written logic zero, Pxn is configured as an input pin. All bits are cleared by a system reset.

9.2.3 Pull-up Resistor Selection Register (PxPU)

All ports P0, P1, P2, P3 have optional internal pull-ups. The PxnPU bit in the PxPU register allows the use of pull-up resistor. If PxnPU is written logic one, the pull-up resistor is activated. If PxnPU is written logic zero, the pull-up resistor is deactivated. When the port is configured as an input port, internal pull-up is deactivated regardless of the PxnPU bit. After reset, all pull-up resistors are switched off except those of P3[5:2]. According to PKG types, some of these ports are omitted, so to maintain input status, the internal pull-ups for these ports are activated.

9.2.4 Open-drain Selection Register (PxOD)

The PxnOD bit in the PxOD register controls the port type when configured as an output port. If PxnOD is written logic one, the port becomes open-drain type. If PxnOD is written logic zero, the port becomes push-pull type. After reset, open-drain function is disabled.

Caution : Port 0 has no open drain control register.

9.2.5 Pull-up Control Register (PxBPC)

When the external VDD drops below the $V_{BODOUT0}$ level, the ports can be selectively configured as input ports with pull-up resistors activated regardless of the PxnIO. In this case, the port direction is changed by hardware automatically. If PxnBPC is written logic one, this function is enabled. If PxnBPC is written logic zero, the port maintain its status even if the device enters stop mode after the external VDD is fallen below the $V_{BODOUT0}$ level. After reset, PxnBPC is set to 1 allowing automatic port direction change due to voltage drop.

9.2.6 Pin Change Interrupt Enable Register (P0PC)

P0 port support Pin Change Interrupt (PCI) function. Pin Change Interrupt will trigger if any pin changes its status when P0nPC is set to 1. At reset, PCI function is disabled for all P0 pins.

9.2.7 Register Map

Name	Address	Dir	Default	Description
P0	80_H	R/W	00_H	P0 Data Register
P0IO	98_H	R/W	00_H	P0 Direction Register
P0PU	$2F00_H$	R/W	00_H	P0 Pull-up Resistor Selection Register
P0BPC	$2F50_H$	R/W	FF_H	P0 Pull-up Control Register
P0PC	$C1_H$	R/W	00_H	P0 Pin Change Interrupt Enable Register
P1	88_H	R/W	00_H	P1 Data Register
P1IO	$A0_H$	R/W	00_H	P1 Direction Register
P1PU	$2F01_H$	R/W	00_H	P1 Pull-up Resistor Selection Register
P1OD	$2F09_H$	R/W	00_H	P1 Open-drain Selection Register
P1BPC	$2F51_H$	R/W	FF_H	P1 Pull-up Control Register
P2	90_H	R/W	00_H	P2 Data Register
P2IO	$B0_H$	R/W	00_H	P2 Direction Register
P2PU	$2F02_H$	R/W	00_H	P2 Pull-up Resistor Selection Register
P2OD	$2F0A_H$	R/W	00_H	P2 Open-drain Selection Register
P2BPC	$2F52_H$	R/W	07_H	P2 Pull-up Control Register
P3	$9F_H$	R/W	00_H	P3 Data Register
P3IO	$9B_H$	R/W	00_H	P3 Direction Register
P3PU	$2F03_H$	R/W	$3C_H$	P3 Pull-up Resistor Selection Register
P3OD	$2F0B_H$	R/W	00_H	P3 Open-drain Selection Register
P3BPC	$2F53_H$	R/W	FF_H	P3 Pull-up Control Register
PSR0	92_H	R/W	00_H	Port Selection Register 0

Table 0-1 Register Map of Port

9.2.8 PORT 0

P0 (P0 Data Register) **80_H**

7	6	5	4	3	2	1	0
P07	P06	P05	P04	P03	P02	P01	P00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P0[7:0] I/O Data

P0IO (P0 Direction Register) **98_H**

7	6	5	4	3	2	1	0
P07IO	P06IO	P05IO	P04IO	P03IO	P02IO	P01IO	P00IO
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P0IO[7:0] P0 Direction

0 Input

1 Output

P0PU (P0 Pull-up Resistor Selection Register) **$2F00_H$**

7	6	5	4	3	2	1	0
P07PU	P06PU	P05PU	P04PU	P03PU	P02PU	P01PU	P00PU
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P0PU[7:0] P0 Pull-up Control

0 Disable pull-up

1 Enable pull-up

P0BPC (P0 Pull-up Control Register) **$2F50_H$**

7	6	5	4	3	2	1	0
P07BPC	P06BPC	P05BPC	P04BPC	P03BPC	P02BPC	P01BPC	P00BPC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

P0BPC[7:0] Control port direction and use of internal pull-up resistor when external VDD drops below $V_{BODOUT0}$ level.

0 Maintain its previous state (input or output)

1 Changed to input port and pull-up resistor is activated

P0PC (P0 Pin Change Interrupt Enable Register) **$C1_H$**

7	6	5	4	3	2	1	0
P07PC	P06PC	P05PC	P04PC	P03PC	P02PC	P01PC	P00PC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P0PC[7:0] Control Pin Change Interrupt function

0 Disable PCI function

1 Enable PCI function

9.2.9 PORT 1

P1 (P1 Data Register) **88_H**

7	6	5	4	3	2	1	0
P17	P16	P15	P14	P13	P12	P11	P10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P1[7:0] I/O Data

P1IO (P1 Direction Register) **$A0_H$**

7	6	5	4	3	2	1	0
P17IO	P16IO	P15IO	P14IO	P13IO	P12IO	P11IO	P10IO
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P1IO[7:0] P1 Direction

0 Input

1 Output

P1PU (P1 Pull-up Resistor Selection Register) **$2F01_H$**

7	6	5	4	3	2	1	0
P17PU	P16PU	P15PU	P14PU	P13PU	P12PU	P11PU	P10PU
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P1PU[7:0] P1 Pull-up Control

0 Disable pull-up

1 Enable pull-up

P1OD (P1 Open-drain Selection Register) **$2F09_H$**

7	6	5	4	3	2	1	0
P17OD	P16OD	P15OD	P14OD	P13OD	P12OD	P11OD	P10OD
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P1OD[7:0] Control P1 port type when configured as output port.

0 Push-pull type output drive

1 Open-drain type output drive

P1BPC (P1 Pull-up Control Register) **$2F51_H$**

7	6	5	4	3	2	1	0
P17BPC	P16BPC	P15BPC	P14BPC	P13BPC	P12BPC	P11BPC	P10BPC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

P1BPC[7:0] Control port direction and use of internal pull-up resistor when external VDD drops below $V_{BODOUT0}$ level.

0 Maintain its previous state (input or output)

1 Changed to input port and pull-up resistor is activated

9.2.10 PORT 2

P2 (P2 Data Register) **90_H**

7	6	5	4	3	2	1	0
-	-	-	-	-	P22	P21	P20
-	-	-	-	-	R/W	R/W	R/W

Initial value : 00_H

P2[2:0] I/O Data

P2IO (P2 Direction Register) **$B0_H$**

7	6	5	4	3	2	1	0
-	-	-	-	-	P22IO	P21IO	P20IO
-	-	-	-	-	R/W	R/W	R/W

Initial value : 00_H

P2IO[2:0] P2 Direction [NOTE1]

0 Input

1 Output

P2PU (P2 Pull-up Resistor Selection Register) **$2F02_H$**

7	6	5	4	3	2	1	0
-	-	-	-	-	P22PU	P21PU	P20PU
-	-	-	-	-	R/W	R/W	R/W

Initial value : 00_H

P2PU[2:0] P2 Pull-up Control [NOTE1 NOTE2]

0 Disable pull-up

1 Enable pull-up

[NOTE1] P20 is used as an external reset source when RSTDIS bit in FUSE_CONF register is cleared. In this case, the direction of P20 is input only and the internal pull-up resistor is always activated regardless of the P20IO or P20PU bits.

[NOTE2] P22 and P21 are used for OCD communication ports and the OCD mode is entered by toggling P22 and P21 in pre-defined manner while internal reset is being asserted. When a reset event occurs, P22 and P21 ports are switched to input state and internal pull up resistor is disabled. Because the floating input states can make the device to enter OCD-like mode, the internal pull up resistors of P22 and P21 ports are always activated while the device is in reset state to prevent wrong mode entering.

P2OD (P2 Open-drain Selection Register) **$2F0A_H$**

7	6	5	4	3	2	1	0
-	-	-	-	-	P22OD	P21OD	P20OD
-	-	-	-	-	R/W	R/W	R/W

Initial value : 00_H

P2OD[2:0] Control P2 port type when configured as output port.

0 Push-pull type output drive

1 Open-drain type output drive

P2BPC (P2 Pull-up Control Register) **$2F52_H$**

7	6	5	4	3	2	1	0
-	-	-	-	-	P22BPC	P21BPC	P20BPC
-	-	-	-	-	R/W	R/W	R/W

Initial value : 07_H

P2BPC[2:0] Control port direction and use of internal pull-up resistor when external VDD drops below $V_{BODOUT0}$ level.

0 Maintain its previous state (input or output)

1 Changed to input port and pull-up resistor is activated

9.2.11 PORT 3

P3 (P3 Data Register) **$9F_H$**

7	6	5	4	3	2	1	0
-	P36	P35	P34	P33	P32	P31	P30
-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P3[7:0] I/O Data

P3IO (P3 Direction Register) **$9B_H$**

7	6	5	4	3	2	1	0
P37IO	P36IO	P35IO	P34IO	P33IO	P32IO	P31IO	P30IO
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P3IO[7:0] P3 Direction

0 Input

1 Output

P3PU (P3 Pull-up Resistor Selection Register) **$2F03_H$**

7	6	5	4	3	2	1	0
P37PU	P36PU	P35PU	P34PU	P33PU	P32PU	P31PU	P30PU
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : $3C_H$

P3PU[7:0] P3 Pull-up Control

0 Disable pull-up

1 Enable pull-up

P3OD (P3 Open-drain Selection Register) **$2F0B_H$**

7	6	5	4	3	2	1	0

P37OD	P36OD	P35OD	P34OD	P33OD	P32OD	P31OD	P30OD
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

P3OD[7:0] Control P0 port type when configured as output port.

0 Push-pull type output drive

1 Open-drain type output drive

P3BPC (P3 Pull-up Control Register) $2F53_H$

7	6	5	4	3	2	1	0
P37BPC	P36BPC	P35BPC	P34BPC	P33BPC	P32BPC	P31BPC	P30BPC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

P3BPC[7:0] Control port direction and use of internal pull-up resistor when external VDD drops below $V_{BODOUT0}$ level.

0 Maintain its previous state (input or output)

1 Changed to input port and pull-up resistor is activated

PSR0 (Port Selection Register 0) 92_H

7	6	5	4	3	2	1	0
SDASWAP	SCLSWAP	SS0SWAP	XCK0SWAP	INT3SWAP	INT2SWAP	INT1SWAP	INT0SWAP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SDASWAP Select SDA port for I2C

0 SDA is P22

1 SDA is P37

SCLSWAP Select SCL port for I2C

0 SCL is P21

1 SCL is P36

SS0SWAP Select SS0 port for USART0

0 SS0 is P30

1 SS0 is P37

XCK0SWAP Select XCK0 port for USART0

0 XCK0 is P31

1 XCK0 is P36

INT3SWAP Select the source of External Interrupt 3

0 External Interrupt 3 is triggered on P22

1 External Interrupt 3 is triggered on P15

INT2SWAP Select the source of External Interrupt 2

0 External Interrupt 2 is triggered on P21

1 External Interrupt 2 is triggered on P14

INT1SWAP Select the source of External Interrupt 1

0 External Interrupt 1 is triggered on P37

1 External Interrupt 1 is triggered on P13

INT0SWAP Select the source of External Interrupt 0

0 External Interrupt 0 is triggered on P36

1 External Interrupt 0 is triggered on P12

10. Interrupt Controller

10.1 Overview

The interrupt controller has the following features to handle interrupt request from internal peripherals or external pins.

- support up to 21 interrupt sources NOTE
- 6 group of 4 priority level
- multiple interrupts handling
- global enable by EA bit and selective control by IEx bit
- Interrupt latency : 3~9 machine cycles in single interrupt system

NOTE Interrupt controller can accept up to 24 interrupt sources, but there are only 19 interrupt sources in MC96FR332A.

Interrupt controller has 4 Interrupt Enable Registers (IE, IE1, IE2, IE3) and 2 Interrupt Priority Registers (IP, IP1). There are 16 interrupt sources in MC96FR332A and overall control is done by EA bit in IE register. When EA is set to 0, all interrupt requests are ignored. When EA is set to 1, each interrupt request is accepted or not by INTnE bit in IEx registers. 16 interrupt sources are assigned to 6 groups and each group can have different priority according to IP and IP1 registers.

By default all interrupt sources are level-triggered, but external interrupts can be set to operate in edge-trigger mode. If more than 2 interrupts of different group priority are requested almost at the same time, the request of higher priority is serviced first. And among the requests of same priority, an internal polling sequence determines which request is serviced, ie, the interrupt having lower priority number in Table 10-2 is serviced first. Even in interrupt service routine, another interrupt of higher priority can interrupt the execution of service routine for the lower priority by software configuration.

Interrupt Group	**Highest** → **Lowest**			
0 (Bit0)	Interrupt0	Interrupt6	Interrupt12	Interrupt18
1 (Bit1)	Interrupt1	Interrupt7	Interrupt13	Interrupt19
2 (Bit2)	Interrupt2	Interrupt8	Interrupt14	Interrupt20
3 (Bit3)	Interrupt3	Interrupt9	Interrupt15	Interrupt21
4 (Bit4)	Interrupt4	Interrupt10	Interrupt16	Interrupt22
5 (Bit5)	Interrupt5	Interrupt11	Interrupt17	Interrupt23

Highest ↓ **Lowest**

Table 10-1 Interrupt Group and Default Priority

10.2 External Interrupt

The External Interrupts are triggered by the INT0, INT1, INT2, INT3 pins. The External Interrupts can be triggered by a falling or rising edge or a low or high level. The trigger mode and trigger level is controlled by External Interrupt Edge Register (EIEDGE) and External Interrupt Polarity Register (EIPOLA). When the external interrupt is enabled and is configured as level triggered, the interrupt will trigger as long as the pin is held low or high. External interrupts are detected asynchronously. This implies that these interrupts can be used for wake-up sources from stop mode. The interrupt requests from INT0, INT1, INT2, INT3 pins can be monitored through the External Interrupt Flag Register (EIFLAG).

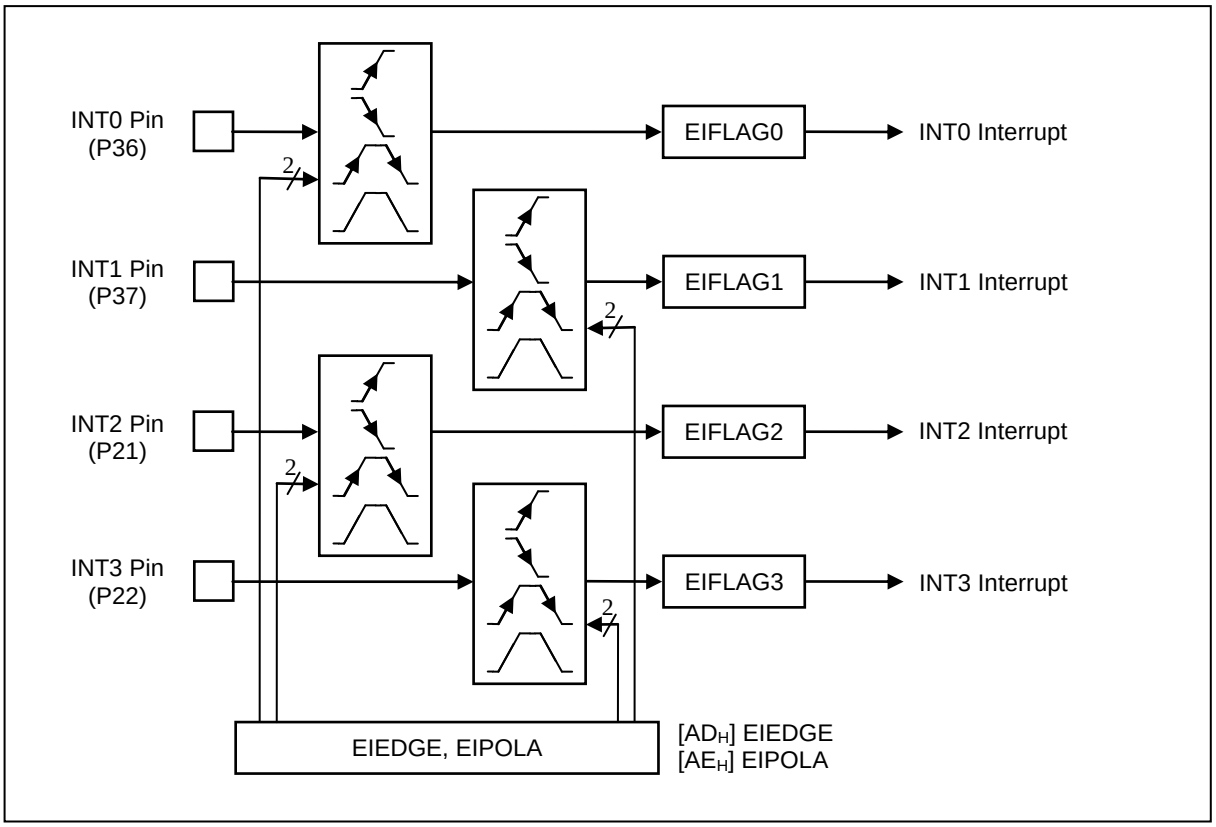


Figure 10-1 External Interrupt trigger condition

10.3 Block Diagram

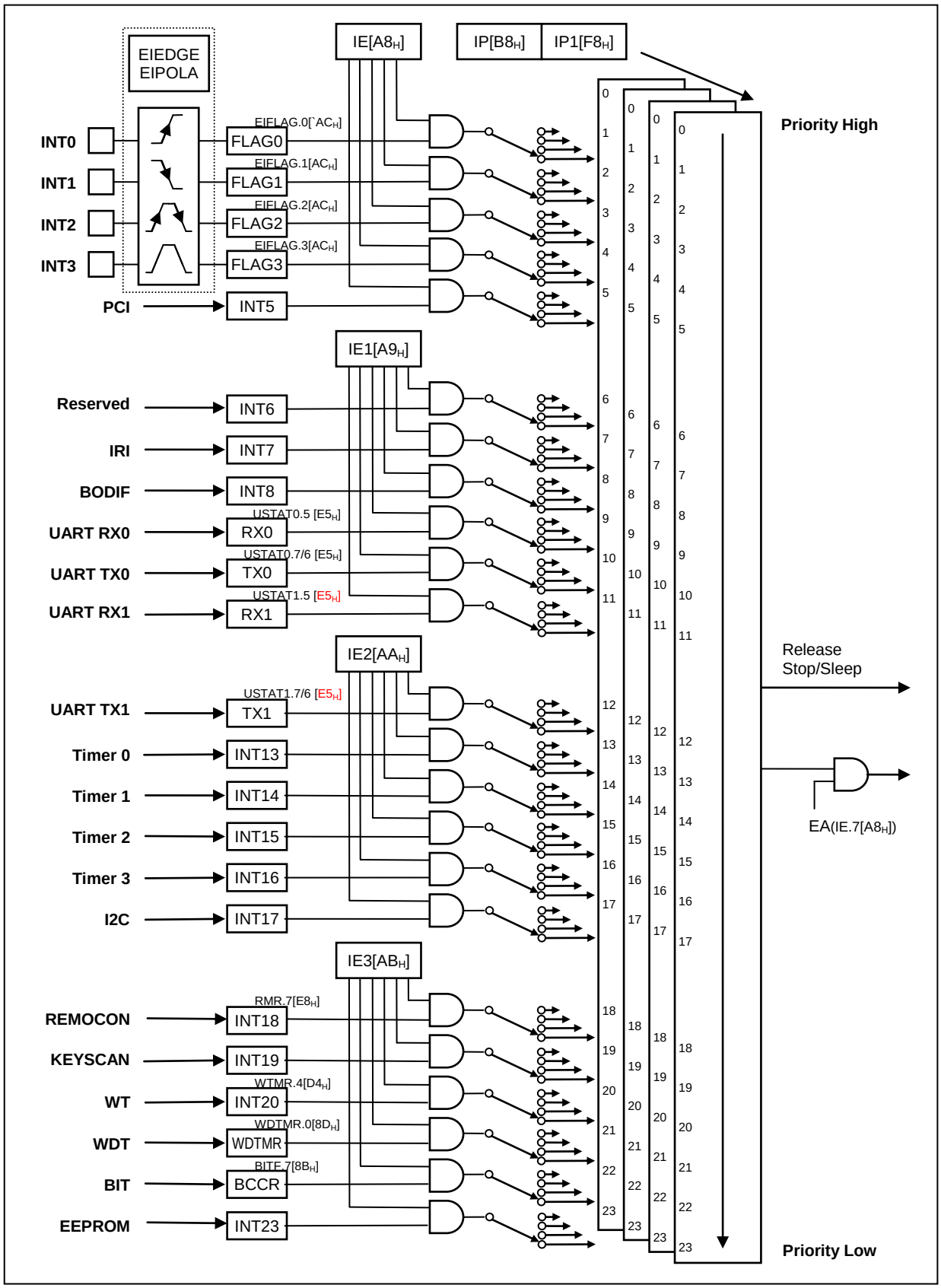


Figure 10-2 Block Diagram of Interrupt Controller

10.4 Interrupt Vectors

There are 16 interrupt sources which are from internal peripherals or from external pin inputs. When a interrupt is requested while EA bit in IE register and its individual enable bit INTnE in IEx register is set, the CPU executes a long call instruction (LCALL) to the vector address listed in Table 10-2. As can be seen in the table, all interrupt vector has 8 bytes address space except for reset vector. If priority level is not set by user software, the interrupt sources have default priority as in the following table, and the lower number has the higher priority.

Interrupt Source	Symbol	Interrupt Enable Bit	Priority	Mask	Vector Address
Hardware Reset	RESETB	Always	0	Non-Maskable	0000_H
-	INT0	IE0.0	1	Maskable	0003_H
External Interrupt 0	INT1	IE0.1	2	Maskable	$000B_H$
External Interrupt 1	INT2	IE0.2	3	Maskable	0013_H
External Interrupt 2	INT3	IE0.3	4	Maskable	$001B_H$
External Interrupt 3	INT4	IE0.4	5	Maskable	0023_H
Pin Change Interrupt (P0)	INT5	IE0.5	6	Maskable	$002B_H$
-	INT6	IE1.0	7	Maskable	0033_H
IRI	INT7	IE1.1	8	Maskable	$003B_H$
BOD Flag	INT8	IE1.2	9	Maskable	0043_H
USART RX0	INT9	IE1.3	10	Maskable	$004B_H$
USART TX0	INT10	IE1.4	11	Maskable	0053_H
USART RX1	INT11	IE1.5	12	Maskable	$005B_H$
USART TX1	INT12	IE2.0	13	Maskable	0063_H
T0	INT13	IE2.1	14	Maskable	$006B_H$
T1	INT14	IE2.2	15	Maskable	0073_H
T2	INT15	IE2.3	16	Maskable	$007B_H$
T3	INT16	IE2.4	17	Maskable	0083_H
I2C	INT17	IE2.5	18	Maskable	$008B_H$
REMOCON	INT18	IE3.0	19	Maskable	0093_H
KEYSCAN	INT19	IE3.1	20	Maskable	$009B_H$
WT	INT20	IE3.2	21	Maskable	$00A3_H$
WDT	INT21	IE3.3	22	Maskable	$00AB_H$
BIT	INT22	IE3.4	23	Maskable	$00B3_H$
FLASH	INT23	IE3.5	24	Maskable	$00BB_H$

Table 10-2 Reset and Interrupt Vectors Placement

To activate a interrupt request, both EA bit in IE register and INTnE bit in IEx register are enabled. When a interrupt is generated, the interrupt flag can be read through each status register except for KEYSCAN and Pin Change Interrupt which have no status register. And almost interrupt flags are automatically cleared when their interrupt is executed. These kinds of interrupts are BIT, WDT, TIMER0/1/2/3, I2C, Watch Timer, USART RX, REMOCON, External Interrupt 0/1/2/3 and Pin Change Interrupt. KEYSCAN, FLASH and Pin Change Interrupts have no flag bit,so these interrupts cannot be used in polling mode.

10.5 Interrupt Sequence

When a interrupt occurs, the flag is stored to the status register which belongs to the interrupt source. An interrupt request is preserved until the request is accepted by CPU or cleared to '0' by a reset or an instruction.[NOTE]. The CPU accepts a interrupt request at the last cycle of current instruction. So instead of executing the instruction being fetched, the CPU executes internally a LCALL instruction and saves the PC to the stack region. At the same time the interrupt controller hands over the address of LJMP instruction to the service routine, which is used by the CPU. It takes 3 to 9 cycles to

finish current instruction and jump to the interrupt service routine. After executing the service routine, the program address is retrieved from the stack by executing RETI instruction to restart from the position where the interrupt is accepted. The following figure shows the sequence.

NOTE Interrupt flags due to USART TX, KEYSCAN and FLASH are not auto-cleared when the CPU accepts the request. KEYSCAN module has no status register, so interrupt flag is not to be polled.

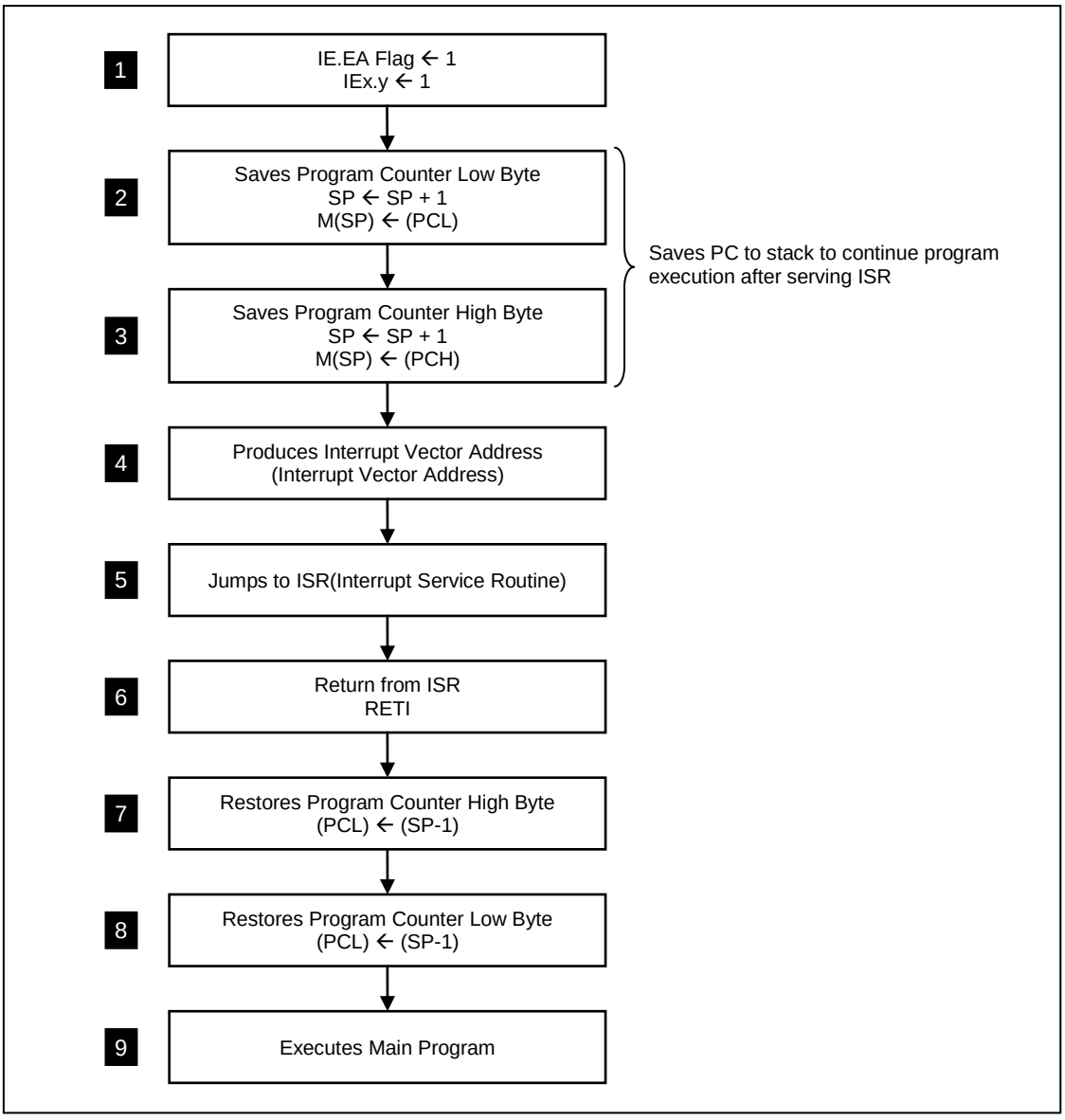


Figure 10-3 Sequence of Interrupt handling

10.6 Effective time of Interrupt Request

To activate interrupt request from interrupt sources, both EA bit in IE register and individual enable bit INTnE in IEx register must be enabled. At this time, the effective time of interrupt request after setting control registers is as follows.

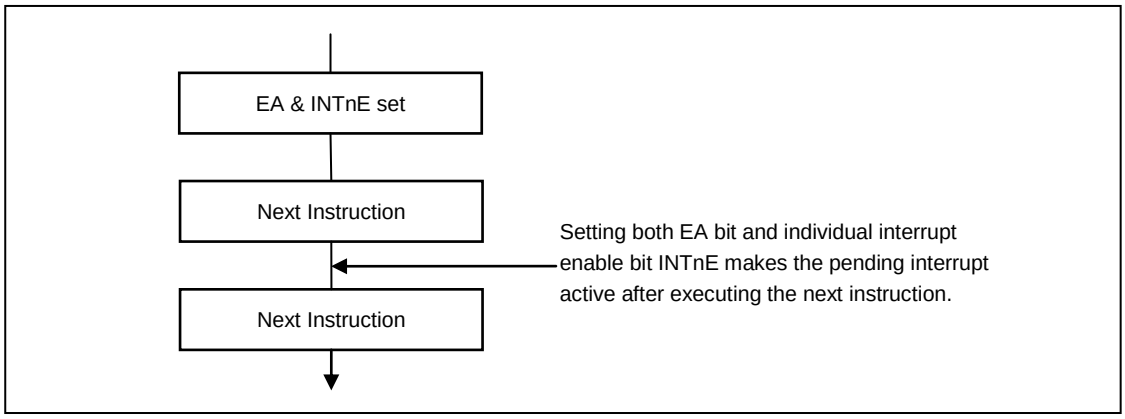


Figure 10-4 Effective time of interrupt request after setting IEx registers

10.7 Multiple Interrupts

If more than two interrupts are requested simultaneously, one of higher priority level is serviced first and others remain pending. Among pending interrupts, the interrupt of second highest priority is serviced next after executing current interrupt service.

In addition, as shown in Figure 10-6, another interrupt request can be serviced while servicing previously requested interrupt. In this case, interrupt requested later must have higher priority level and the interrupt handler should allow another interrupt request.

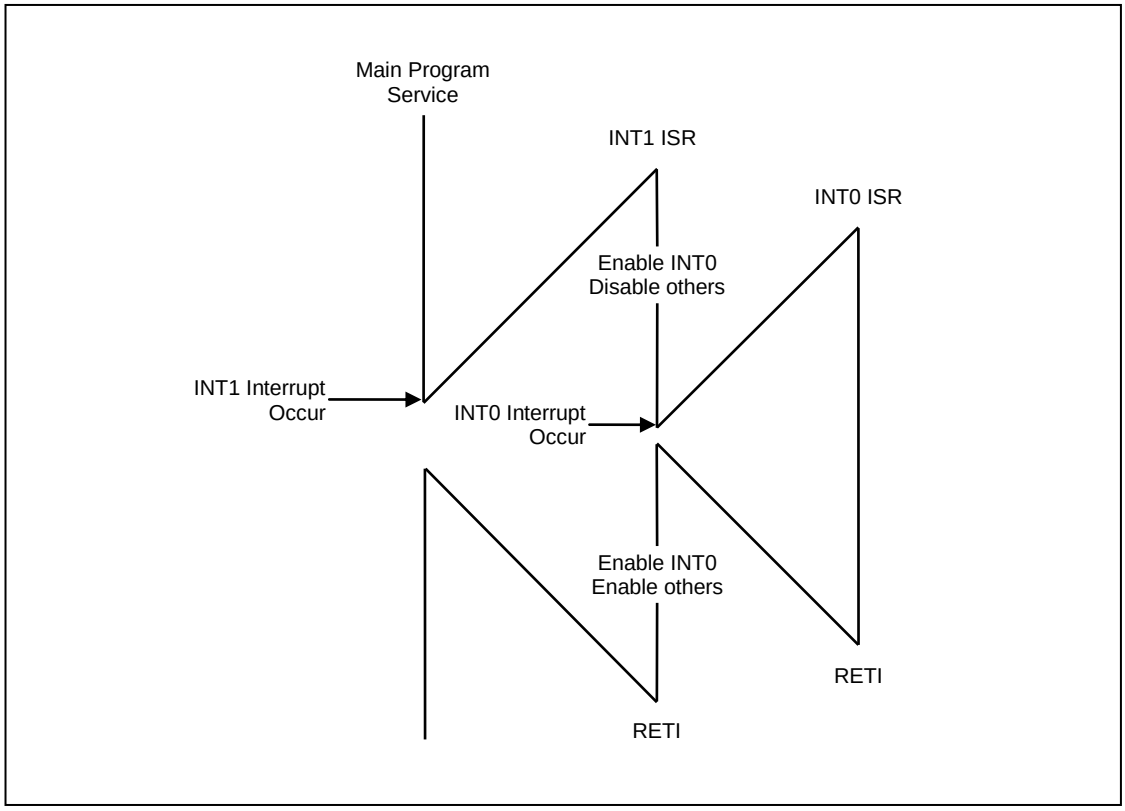


Figure 10-5 Accept of another interrupt request in interrupt service routine

The following example shows how to allow INT0 interrupt request while executing INT1 interrupt service routine. In this example, INT0 has higher group priority than INT1 interrupt according to IP0,

IP1 registers. Other interrupts having lower group priority than INT0 cannot be serviced until INT0 service routine is finished even if the INT0 interrupt handler allows those interrupt requests.

Example) Software Multi Interrupt

```
INT1 :   MOV    IE, #01H     ;Enable INT0 only
         MOV    IE1, #00H    ;Disable other interrupts
         :
         MOV    IE, #0FFH    ;Enable all Interrupts
         MOV    IE1, #0FFH
         RETI
```

In short, an interrupt service routine may only be interrupted by an interrupt of higher priority than being serviced. And when more than two interrupts are requested at the same time, the one of highest priority is serviced first.

10.8 Interrupt Service Procedure

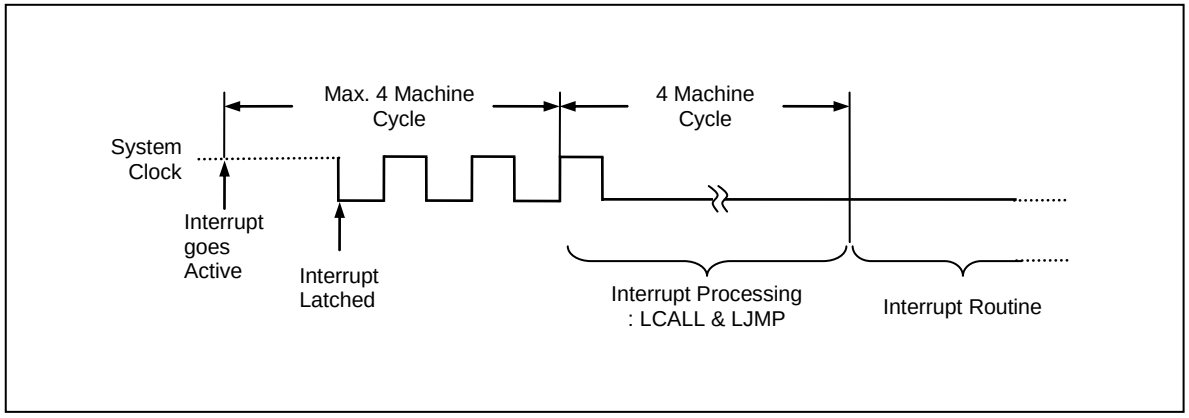


Figure 10-6 Interrupt Request and Service Procedure

10.9 Generation of Branch Address to Interrupt Service Routine(ISR)

The following figure shows the relationship between the vector address of BIT interrupt and the branch address to service routine.

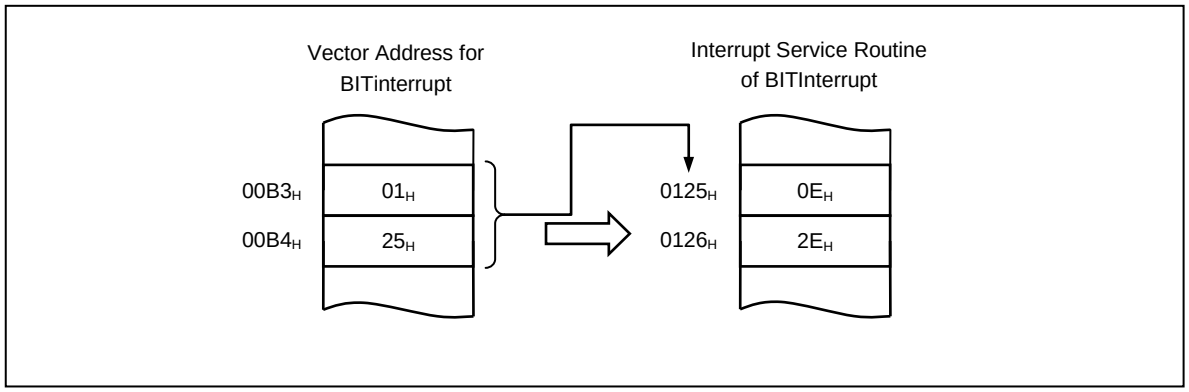


Figure 10-7 Generating branch address to BIT interrupt service routine from vector table

10.10 Saving and Restoring General Purpose Registers

```
INTxx :   PUSH  PSW
          PUSH  DPL
          PUSH  DPH
          PUSH  B
          PUSH  ACC
          :

Interrupt_Processing:
          :

          POP  ACC
          POP  B
          POP  DPH
          POP  DPL
          POP  PSW
          RETI
```

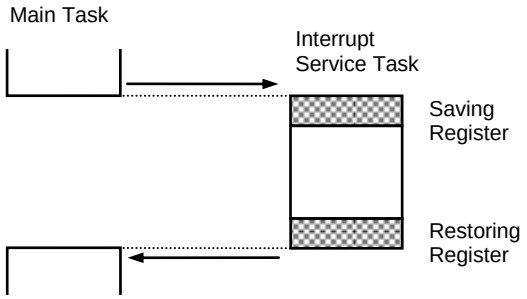


Figure 10-8 Processing General registers while an interrupt is serviced

10.11 Interrupt Timing

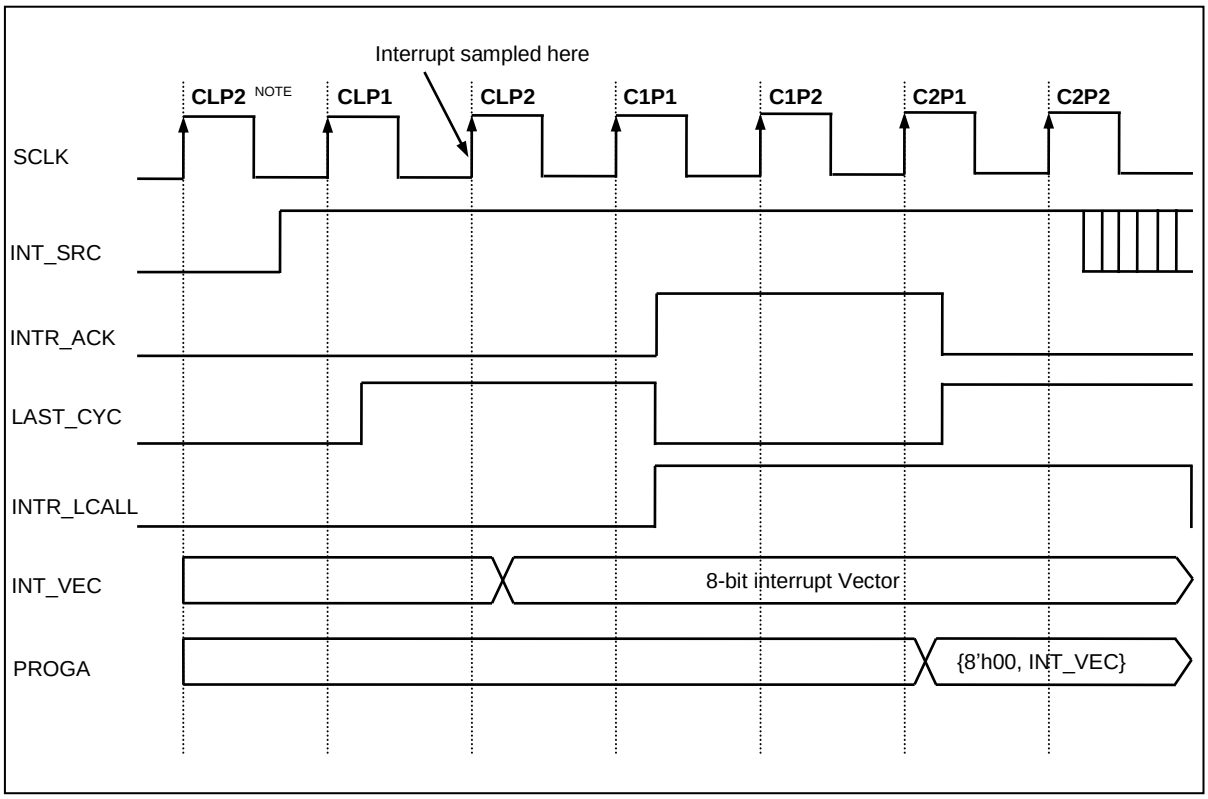


Figure 10-9 Timing chart for Interrupt Accept and Branch Address Generation

The interrupt request is sampled at the last cycle of the command currently being executed. On recognition of interrupt request, the interrupt controller hands over the corresponding lower 8-bit vector address to the CPU, M8051W and the CPU acknowledges the request at the first cycle of the next command to jump to the interrupt vector address.

NOTE command cycle C?P? : L=Last cycle, 1=1st cycle or 1st phase, 2=2nd cycle or 2nd phase

10.12 Interrupt Registers

10.12.1 Register Map

Name	Address	Dir	Default	Description
IE	$A8_H$	R/W	00_H	Interrupt Enable Register
IE1	$A9_H$	R/W	00_H	Interrupt Enable Register 1
IE2	AA_H	R/W	00_H	Interrupt Enable Register 2
IE3	AB_H	R/W	00_H	Interrupt Enable Register 3
IP	$B8_H$	R/W	00_H	Interrupt Priority Register
IP1	$F8_H$	R/W	00_H	Interrupt Proprity Register 1
EIFLAG	AC_H	R/W	00_H	External Interrupt Flag Register
EIEDGE	AD_H	R/W	00_H	External Interrupt Edge Register
EIPOLA	AE_H	R/W	00_H	External Interrupt Polarity Register
EIENAB	AF_H	R/W	00_H	External Interrupt Enable Register

Table 10-3 Register Map of Interrupt Controller

10.12.2 Interrupt Enable Register (IE, IE1, IE2, IE3)

There're 4 interrupt enable registers which are IE, IE1, IE2 and IE3. In IE register, there's two kinds of interrupt enable bits called the global interrupt enable bit, EA, and 6 individual interrupt enable bits, INTnE. Each IE1, IE2 and IE3 register only has 6 individual interrupt enable bits. Totally 16 peripheral and external interrupts are controlled by these registers.

10.12.3 Interrupt Priority Register (IP, IP1)

As described above, each interrupt enable register has 6 individual interrupt enable bits. So, interrupt controller itself can deal up to 24 interrupt sources. These 24 sources are classified into 6 groups by 4 sources. Each group can have 4 level of priority through IP and IP1 registers. The level 3 group interrupt is of the highest priority, and the level 0 group interrupt is of the lowest priority. The initial values of IP and IP1 registers are 00_H. By default, the lower numbered interrupt has the higher priority if group priority is the same. When the group priority is decided by configuring IP and IP1 registers, among 4 interrupt sources within the group, the lower numbered interrupt has the higher priority.

10.12.4 External Interrupt Flag Register (EIFLAG)

External Interrupt Flag Register shows the status of external interrupts. Each flag is set to '1' when a port is configured as a external interrupt source, and the port state changes to equal to the interrupt generating condition according to EIEDGE and EIPOLA register. To clear each flag, write '0' to corresponding bit position of this register.

10.12.5 External Interrupt Edge Register (EIEDGE)

External Interrupt Edge Register decides the trigger mode of external interrupt, edge or level mode. To make a external interrupt triggered by a falling or rising edge, write '00_B' to the corresponding bit position. And to make a external interrupt triggered by a low or high level, write '01_B', '10_B' or '11_B' to the corresponding bit position. Initially, all external interrupts are triggered by high level. Note there are 2 bits for each external interrupt pin.

10.12.6 External Interrupt Polarity Register (EIPOLA)

This register has different meaning according to the value set in EIEDGE register. When a external interrupt is configured to be triggered by a level, the high or low trigger level is selected through this register. When a external interrupt is configured to be triggered by a edge, the value in this register has nothing to do with the triggering edge.

10.12.7 External Interrupt Enable Register (EIENAB)

External Interrupt Enable Register selects each port pin, which has sub function for external interrupt, whether to use as external interrupt pin or normal port pin. When a bit in this register is written '0', the corresponding pin is used as general purpose I/O pin.

10.12.8 Register Description

IE (Interrupt Enable Register) **$A8_H$**

7	6	5	4	3	2	1	0
EA	-	INT5E	INT4E-	INT3E	INT2E	INT1E	INT0E
R/W	R	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

EA Globla Interrupt Enable Bit
0 Ignore interrupt request from any interrupt source.
1 Accept interrupt request

INT5E Enable or disable Pin Change Interrupt
0 Disable
1 Enable

INT4E Enable or disable External Interrupt 3
0 Disable
1 Enable

INT3E Enable or disable External Interrupt 2
0 Disable
1 Enable

INT2E Enable or disable External Interrupt 1
0 Disable
1 Enable

INT1E Enable or disable External Interrupt 0
0 Disable
1 enable

INT0E Reserved
0 Disable
1 enable

IE1 (Interrupt Enable Register 1) **$A9_H$**

7	6	5	4	3	2	1	0
-	-	INT11E	INT10E-	INT9E	INT8E	INT7E	INT6E
R	R	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

INT11E Enable or disable USART RX1 Interrupt
0 Disable
1 Enable

INT10E Enable or disable USART TX0 Interrupt
0 Disable
1 Enable

INT9E Enable or disable USART RX0 Interrupt
0 Disable
1 Enable

INT8E Enable or disable BOD Flag Interrupt
0 Disable
1 Enable

INT7E Enable or disable IRI Input Interrupt

0 Disable
1 Enable

INT6E Reserved
0 Disable
1 enable

IE2 (Interrupt Enable Register 2) AA_H

7	6	5	4	3	2	1	0
-	-	INT17E	INT16E-	INT15E	INT14E	INT13E	INT12E
R	R	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

INT17E Enable or disable I2C interrupt
0 Disable
1 Enable

INT16E Enable or disable Timer 3 Interrupt
0 Disable
1 Enable

INT15E Enable or disable Timer 2 Interrupt
0 Disable
1 Enable

INT14E Enable or disable Timer 1 Interrupt
0 Disable
1 Enable

INT13E Enable or disable Timer 0 Interrupt
0 Disable
1 enable

INT12E Enable or disable USART RX1 Interrupt
0 Disable
1 enable

IE3 (Interrupt Enable Register 3) AB_H

7	6	5	4	3	2	1	0
-	-	INT23E	INT22E-	INT21E	INT20E	INT19E	INT18E
R	R	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

INT23E Enable or disable FLASH Interrupt
0 Disable
1 Enable

INT22E Enable or disable BIT Interrupt
0 Disable
1 Enable

INT21E Enalbe or disable WDT Interrupt
0 Disable
1 Enable

INT20E Enable or disable Watch Timer Interrupt
0 Disable

	1	Enable
INT19E	Enable or disable KEYSCAN Interrupt	
	0	Disable
	1	Enable
INT18E	REMOCON (Carrier generator) Interrupt	
	0	Diable
	1	Enable

IP (Interrupt Priority Register) $B8_H$

7	6	5	4	3	2	1	0
-	-	IP5	IP4	IP3	IP2	IP1	IP0
R	R	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

IP1 (Interrupt Priority Register 1) $F8_H$

7	6	5	4	3	2	1	0
-	-	IP15	IP14	IP13	IP12	IP11	IP10
R	R	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

IP[5:0], IP1[5:0] Select Interrupt Group Priority

IP1x	IPx	Description
0	0	Group x is of level 0 priority (lowest)
0	1	Group x is of level 1 priority
1	0	Group x is of level 2 priority
1	1	Group x is of level 3 priority (highest)

EIFLAG (External Interrupt Flag Register) AC_H

7	6	5	4	3	2	1	0
-	-	-	–	FLAG3	FLAG2	FLAG1	FLAG0
R	R	R-	R	R/W	R/W	R/W	R/W

Initial value : 00_H

FLAG[3:0] External interrupt flag bit. To clear a flag, write '0' to each bit position.

0	External Interrupt not occurred
1	External Interrupt occurred

EIEDGE (External Interrupt Edge Register) AD_H

7	6	5	4	3	2	1	0
EDGE3R	EDGE3F	EDGE2R	EDGE2F	EDGE1R	EDGE1F	EDGE0R	EDGE0F
R/W-	R/W-	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

EDGEnR Selects the trigger mode of each external interrupt pin. Trigger mode is also affected by the EDGEnF bit.

0	External interrupt is triggered by level (default)

1 External interrupt is triggered by a rising edge

EDGEnF Selects the trigger mode of each external interrupt pin. Trigger mode is also affected by the EDGEnR bit.

0 External interrupt is triggered by level (default)

1 External interrupt is triggered by a falling edge

When EDGEnR and EDGEnF bits are set at the same time, an external interrupt is triggered by both rising and falling edge.

EIPOLA (External Interrupt Polarity Register) AE_H

7	6	5	4	3	2	1	0
-	-	-	–	POLA3	POLA2	POLA1	POLA0
R/W	R/W	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

POLA[3:0] Selects the trigger level of external interrupt, high or log level.

When configured as level trigger mode

0 External interrupt is triggered by a high level (default)

1 External interrupt is triggered by a low level

EIENAB (External Interrupt Enable Register) AF_H

7	6	5	4	3	2	1	0
-	-	-	–	ENAB3	ENAB2	ENAB1	ENAB0
R/W-	R/W-	R/W-	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

ENAB[3:0] Configure each port pin as external interrupt pin input

0 The port is not used for external interrupt (default)

1 The port is used for external interrupt

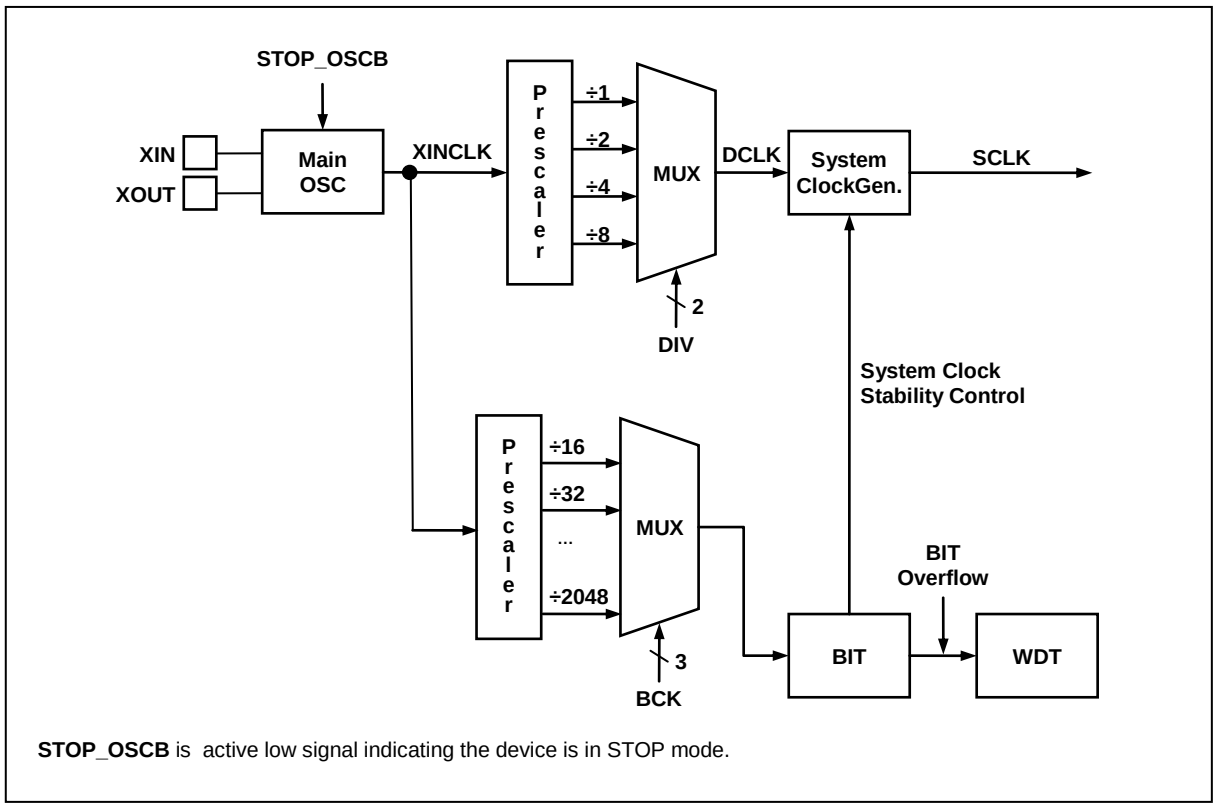
11. Peripheral Units

11.1 Clock Generator

11.1.1 Overview

The clock generator module plays a main role in making a stable operating clock, SCLK. There's only one clock source in MC96FR332A, which is the output of main oscillator, XINCLK, connected to the XIN and XOUT pins. The main clock input XINCLK is divided by 2, 4 or 8, and one of the divided clocks is used as internal operating clock, SCLK, according to the DIV[1:0] bits in SCCR register. By default, frequency of SCLK is same as that of XINCLK, ie, divided by 1.

11.1.2 Block Diagram



STOP_OSCB is active low signal indicating the device is in STOP mode.

Figure 11-1 Block Diagram of Clock Generator

11.1.3 Register Map

Name	Address	Dir	Default	Description
SCCR	$8A_H$	R/W	00_H	System and Clock Control Register

Table 11-1 Register Map of Clock Generator

11.1.4 Register Description

SCCR (System and Clock Control Register) **$8A_H$**

7	6	5	4	3	2	1	0
-	DIV1	DIV0	CBYS	-	-	-	-
-	R/W	R/W-	R/W	-	-	-	-

Initial value : 00_H

DIV[1:0] Selects the divide ratio of internal operating clock, SCLK.

DIV1	DIV0	Description (in case of f_{XIN}=8MHz)
0	0	f_{XIN}/1 (8MHz)
0	1	f_{XIN}/2 (4MHz)
1	0	f_{XIN}/4 (2MHz)
1	1	f_{XIN}/8 (1MHz)

CBYS This bit controls the moment when the SCLK is altered. When this bit is '0', the internal operating clock is updated after the device enters STOP mode by the command "PCON=03_H" and wakes-up from that mode. But when this bit is '1', the SCLK changes right after DIV[1:0] is modified.

0 Internal clock is altered during STOP mode.

1 Internal clock is altered while user program is running

11.2 Basic Interval Timer (BIT)

11.2.1 Overview

BIT module is a 8-bit counter used to guarantee oscillator stabilization time when MC96FR332A is reset or waken from STOP mode. The BIT counter is clocked by a clock divided from XINCLK and the divide ratio is selected from BCK[2:0] bits in BCCR register, from 16 to 2048. At reset, the BIT counter is clocked by a clock which is divided by 2048 from XINCLK.

BIT is a 8-bit binary counter and has the following features.

- Guarantees the oscillation stabilization time when a power-on or reset occurs
- Guarantees the oscillation stabilization time when this device wakes-from STOP mode
- Generates interval timer interrupt as a watch function

11.2.2 Block Diagram

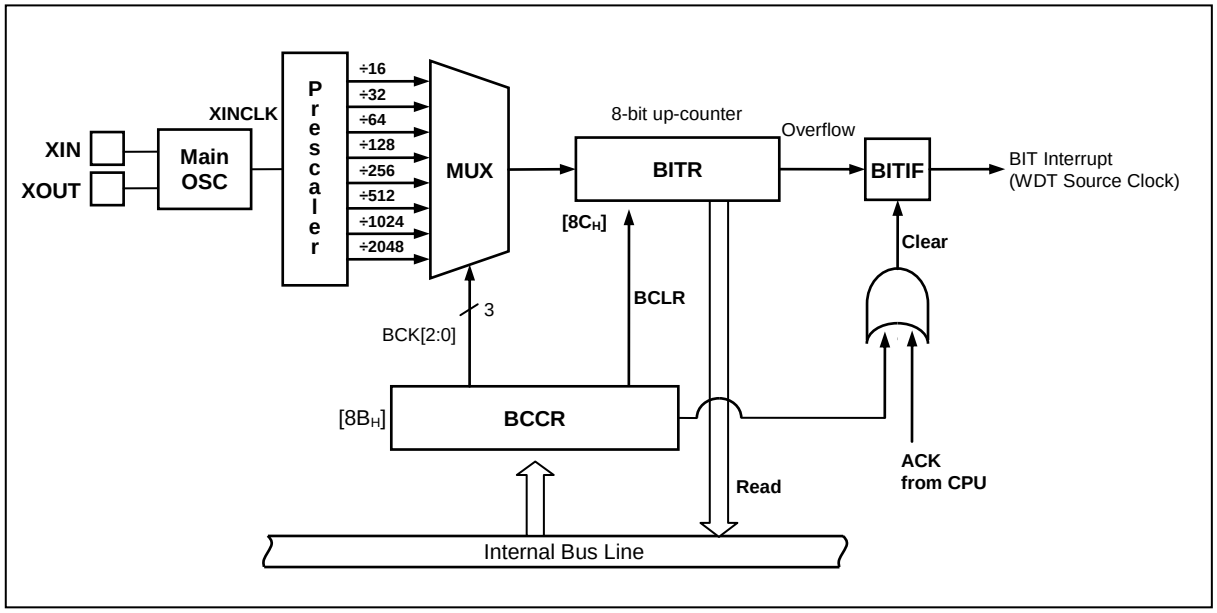


Figure 11-2 Block Diagram of BIT

11.2.3 Register Map

Name	Address	Dir	Default	Description
BCCR	$8B_H$	R/W	77_H	BIT Clock Control Register
BITR	$8C_H$	R	00_H	Basic Interval Timer Register

Table 11-2 Register Map of BIT

11.2.4 Register Description

BCCR (BIT Clock Control Register) **$8B_H$**

7	6	5	4	3	2	1	0
BITF	BCK2	BCK1	BCK0	BCLR	PRD2	PRD1	PRD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 77_H

BITF Reflects the state of BIT interrupt. To clear this flag, write '0' to this bit position. The BIT interrupt occurs when BIT counter reaches to the pre-defined value. The interrupt interval is decided from BCK[2:0] and PRD[2:0] bits.

0 BIT Interrupt not occurred

1 BIT Interrupt occurred

BCK[2:0]

BCK2	BCK1	BCK0	BIT Clock	BIT Interrupt Period [NOTE]
0	0	0	f_{XIN}/2^4	0.512ms
0	0	1	f_{XIN}/2^5	1.024ms
0	1	0	f_{XIN}/2^6	2.048ms
0	1	1	f_{XIN}/2^7	4.096ms
1	0	0	f_{XIN}/2^8	8.192ms
1	0	1	f_{XIN}/2^9	16.384ms
1	1	0	f_{XIN}/2^10	32.768ms
1	1	1	f_{XIN}/2^11	65.536ms (default)

BCLR Clears BIT Counter. Writing '1' to this bit resets BIT counter to 00_H. BCLR bit is auto cleared.

0 BIT counter free runs

1 BIT counter is cleared and counter re-starts

PRD[2:0] Selects BIT interrupt interval. When BIT counter reaches to the value listed below, an interrupt may be issued. The BIT interrupt period is same as the clock period for WDT counter.

PRD2	PRD1	PRD0	Interrupt condition
0	0	0	When BITR[0] = 1
0	0	1	When BITR[1:0] = 11
0	1	0	When BITR[2:0] = 111
0	1	1	When BITR[3:0] = 1111
1	0	0	When BITR[4:0] = 11111
1	0	1	When BITR[5:0] = 111111
1	1	0	When BITR[6:0] = 1111111
1	1	1	When BITR[7:0] = 11111111

[NOTE] This is the case when the frequency of main oscillator input clock, XIN, is 8MHz and the overflow period PRD[2:0] is set to 111_B, where f_{XIN} is the frequency of XIN clock.

The BIT interrupt period is acquired by multiplying clock period of BIT counter and the pre-defined value of BIT counter. That is, $T_{BIT\_INT} = T_{BIT\_CLK} \times 2^{(PRD[2:0]+1)}$, where T_{BIT_INT} is the interval of BIT interrupt and T_{BIT_CLK} is the clock period of BIT counter.

BITR (Basic Interval Timer Register) **$8C_H$**

7	6	5	4	3	2	1	0
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
R	R	R	R	R	R	R	R

Initial value : 00_H

BIT[7:0] BIT counter value

11.3 Watch Dog Timer (WDT)

11.3.1 Overview

The WDT, if enabled, generates an interrupt or a system reset when the WDT counter reaches the given time-out value set in WDTR. In normal operation mode, it is required that the user software clears the WDT counter by setting WDTCL bit in WDTMR register before the time-out value is reached. If the system doesn't restart the counter, an interrupt or a system reset will be issued.

The main features are :

- 2 operating modes : Interrupt or System Reset mode
- Selectable Time-out period

In Interrupt mode, the WDT gives an interrupt when the WDT counter expires. This interrupt can be used to wake the device from SLEEP mode (not from STOP mode[NOTE]), and also as a general system timer. One example is to limit the maximum time allowed for certain operations, giving an interrupt when the operation has run longer than expected. In System Reset mode, the WDT gives a reset when the timer expires. This is typically used to prevent system hang-up in case of runaway code.

The clock source of Watch Dog Timer is the BIT overflow. The interval of WDT interrupt is decided by BIT overflow period and WDTR value, and is calculated as follows.

WDT Interrupt Interval = (BIT overflow period) x (WDTR + 1)

NOTE MC96FR332A has only one clock source, XINCLK, and in STOP mode, the main oscillator stops. Also, the WDT/BIT module stops operation.

11.3.2 Block Diagram

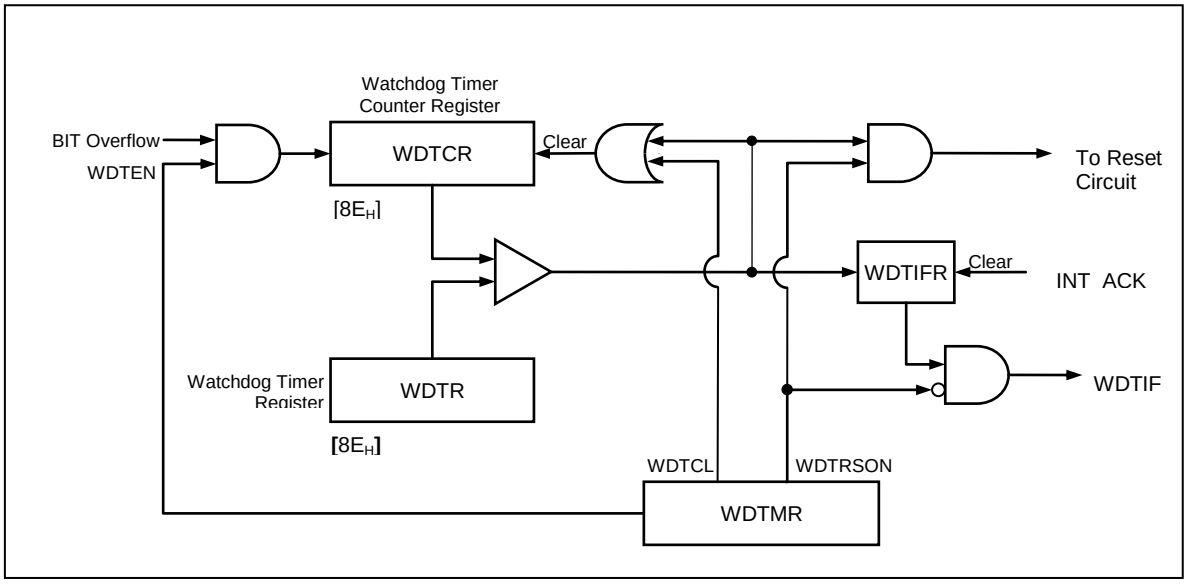


Figure 11-3 Block Diagram

11.3.3 Register Map

Name	Address	Dir	Default	Description
WDTR	$8E_H$	W	FF_H	Watch Dog Timer Register
WDTCR	$8E_H$	R	00_H	Watch Dog Timer Counter Register
WDTMR	$8D_H$	R/W	00_H	Watch Dog Timer Mode Register

Table 11-3 Register Map of WDT

11.3.4 Register Description

WDTR (Watch Dog Timer Register, Write Case) **$8E_H$**

7	6	5	4	3	2	1	0
WDTR7	WDTR6	WDTR5	WDTR4	WDTR3	WDTR2	WDTR1	WDTR0
W	W	W	W	W	W	W	W

Initial value : FF_H

WDTR[7:0] Time-out value of WDT counter (=the period of WDT interrupt)
WDT Interrupt Interval = (BIT Interrupt Interval) x (WDTR + 1)

Precaution must be taken when writing this register. To ensure proper operation, the written value, WDTR should be greater than 01_H.

WDTCR (Watch Dog Timer Counter Register, Read Case) **$8E_H$**

7	6	5	4	3	2	1	0
WDTCR7	WDTCR6	WDTCR5	WDTCR4	WDTCR3	WDTCR2	WDTCR1	WDTCR0
R	R	R	R	R	R	R	R

Initial value : 00_H

WDTCR[7:0] The value of WDT counter

WDTMR (Watch Dog Timer Mode Register) **$8D_H$**

7	6	5	4	3	2	1	0
WDTEN	WDTRSON	WDTCL	-	-	-	-	WDTIFR
R/W	R/W	R/W	-	-	-	-	R/W

Initial value : 00_H

WDTEN Enable or disable WDT module
0 Disable
1 Enable

WDTRSON Decides whether to use WDT interrupt as a reset source or not
0 WDT operates as a free-running 8-bit timer
1 WDT reset is generated when WDT counter overflows

WDTCL Initialize WDT counter
0 Free runs
1 Reset WDT counter. This bit is auto-cleared after 1 machine cycle.

WDTIFR This flag is set when WDT interrupt is generated. This bit is cleared when the CPU services or acknowledges WDT interrupt or s/w write'0' to this bit position.

0	WDT interrupt not occurred
1	WDT interrupt occurred

11.3.5 WDT Interrupt Timing

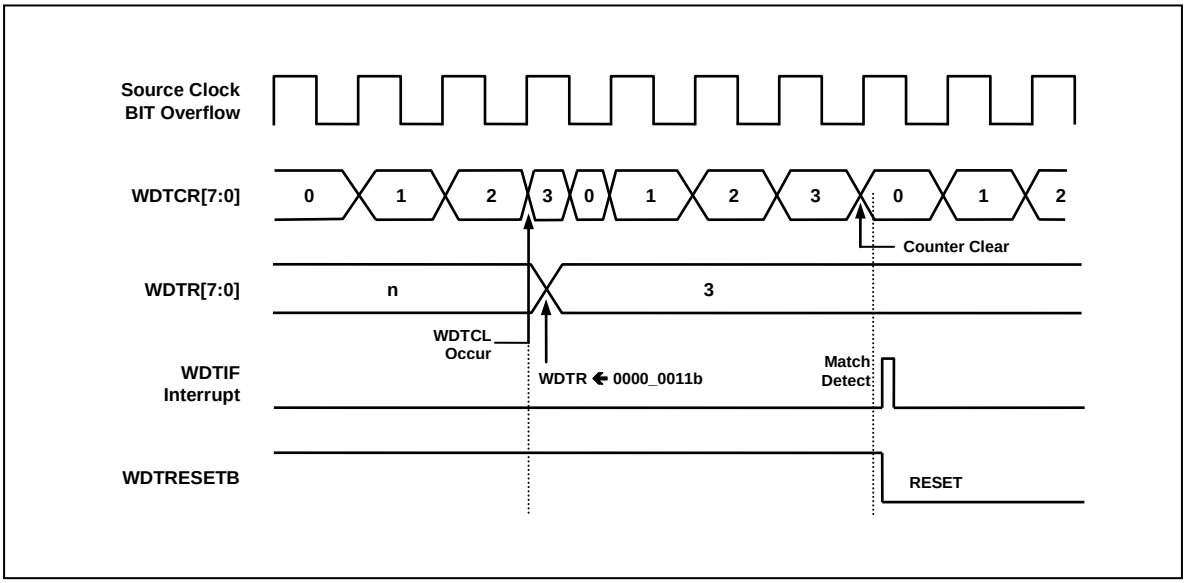


Figure 11-4 WDT Interrupt and Reset Timing

11.4 TIMER/PWM

11.4.1 8-bit Timer/Event Counter 0, 1

11.4.1.1 Overview

Timer 0 and Timer 1 can be used as either separate 8-bit Timer/Counter or one combined 16-bit Timer/Counter. Each 8-bit Timer/Event Counter module has a multiplexer, 8-bit timer data register, 8-bit counter register, mode control register, input capture register and comparator. For PWM mode of operation, Timer 1 has additional registers which are PWM1PR, PWM1DR and PWM1HR.

Timer 0 and Timer 1 have 5 operating modes as following.

- two separate 8-bit Timer/Counter Mode
- two separate 8-bit Capture Mode
- 16-bit Timer/Counter Mode
- 16-bit Capture Mode
- PWM Mode

Timer 0, 1 are clocked by an internal an external clock source (EC0). The clock source is selected by clock select logic which is controlled by the clock select bits(T0CK[2:0], T1CK[2:0]) located in the T0CR and T1CR registers. By configuring T1CK[2:0] bits, Timer 1 can be clocked by the clock source used for Timer 0 or by its own divided clock [NOTE]. Internal clock source is derived from the divider logic of each timer module. In Capture Mode, the counter value is captured into each Input Capture Register when a external interrupt condition is generated on INT0 or INT1 pins. In 8/16-bit Timer/Counter Mode, Timer 0 compares counter value with the value in timer data register and when counter reaches to the compare value, the timer output is toggled internally. When the T0_PE bit in T0CR register is set, the timer output overrides the normal port functionality of the I/O pin it is connected to. Timer 1 operates similar to Timer 0, and in addition can generate PWM wave form when configured as PWM mode. And the Timer 1 output or PWM output appears on T1/PWM1 pin.

[NOTE] SCLK is internal operating clock, which is the output of clock divider logic. The input source of clock divider is XINCLK, the output of main oscillator. The divide ratio can be selected from DIV[1:0] bits in SCCR register and the default frequency is that of main oscillator output, XINCLK. For more information about clock scheme, refer to chapter 11.1.

The next table shows register setting for each timer operating mode.

16 Bit	CAP0	CAP1	PWM1E	T0CK[2:0]	T1CK[1:0]	T0/1_PE	Timer 0	Timer 1
0	0	0	0	XXX	XX	00	8-bit Timer	8-bit Timer
0	0	1	0	111	XX	00	8-bit Event Counter	8-bit Capture
0	1	0	0	XXX	XX	01	8-bit Capture	8-bit Compare Output
0	0	0	1	XXX	XX	11	8-bit Timer/Counter	10-bit PWM
1	0	0	0	XXX	11	00	16-bit Timer	
1	0	0	0	111	11	00	16-bit Event Counter	
1	1	1	0	XXX	11	00	16-bit Capture	
1	0	0	0	XXX	11	01	16-bit Compare Output	

Table 11-4 Operating modes of Timer 0, 1

11.4.1.2 8-Bit Timer/Counter Mode

8-bit Timer/Counter Mode is selected when the T0CR and T1CR registers are configured as follows.

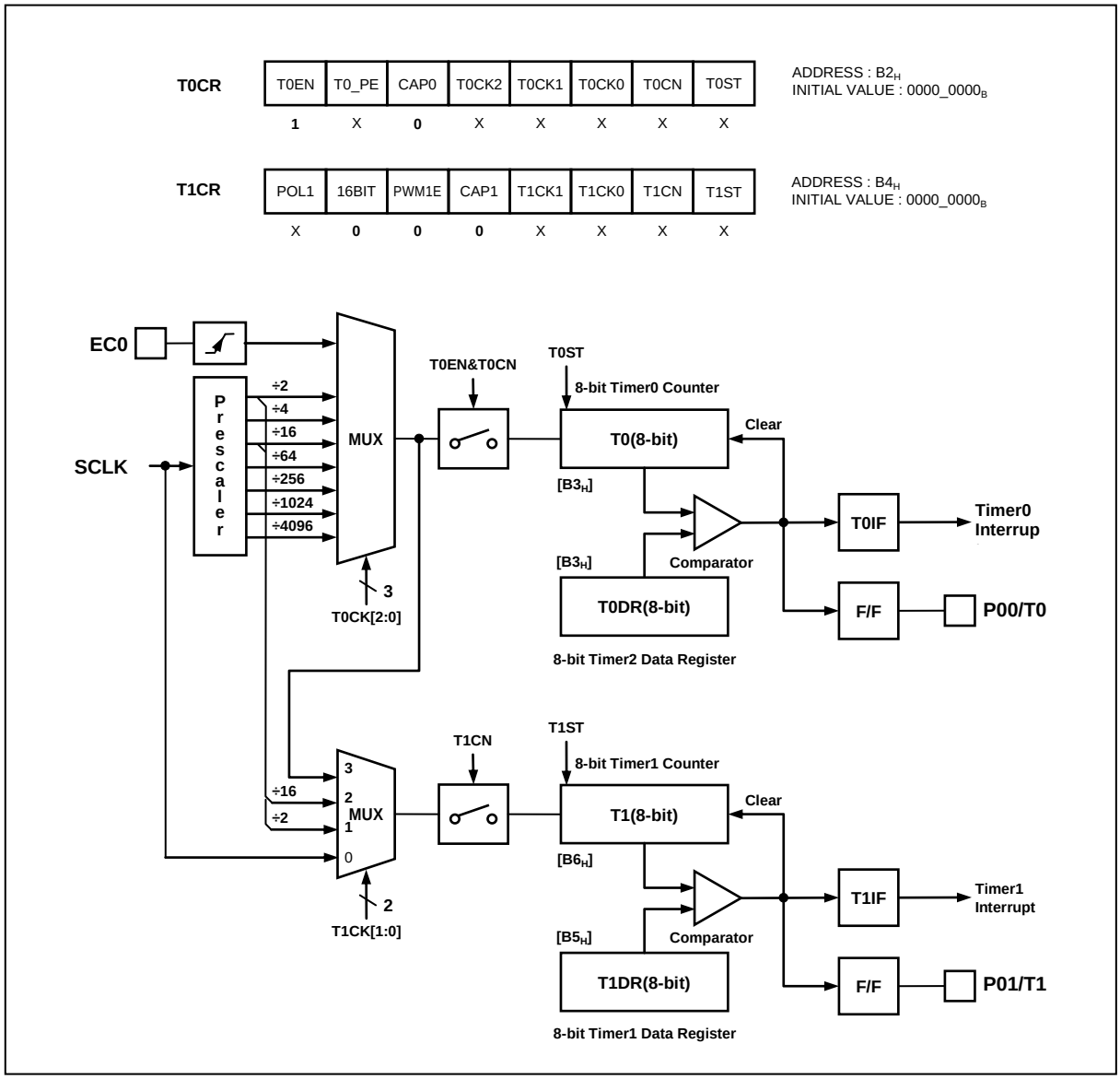


Figure 11-5 Block Diagram of Timer 0,1 in 8-bit timer/counter mode

Each Timer 0 and Timer 1 has its own counter register and data register. The counter is clocked by an internal or external clock source. Internal clock source comes from divider logic whose input clock is SCLK. For Timer 0 module, SCLK is divided by 2, 4, 16, 64, 256, 1024 and 4096. One of these divided clock is used as internal clock source of Timer 0. Divider logic of Timer 1 is much simpler. The SCLK is divided by 2 or 16. Along with these divided clock sources, the SCLK itself can be used as internal clock source of Timer 1. And Timer 1 can also be clocked by the clock source of Timer 0. Each divide ratio is decided by T0CK[2:0] and T1CK[2:0] bits. When the external clock , EC0 is selected as a clock source, the counter increases at rising edge of the clock. When the counter value of each 8-bit timer matches individual data register, an interrupt can be requested. The interrupt flags can be read through T3CR2 register.

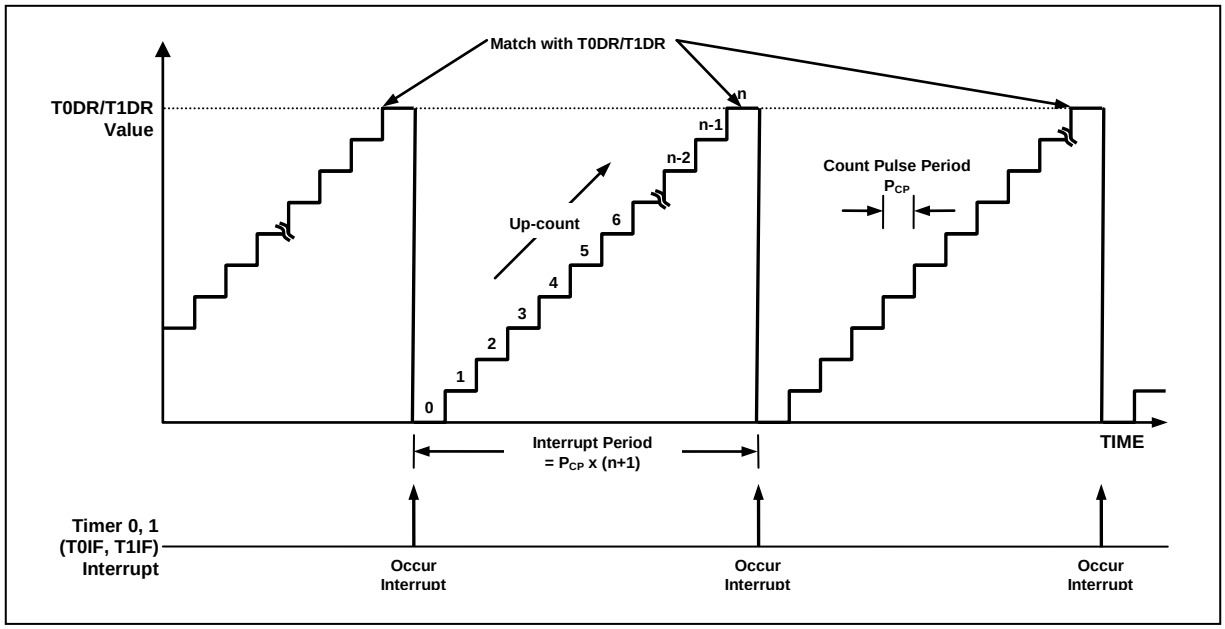


Figure 11-6 Interrupt Period of Timer 0, 1

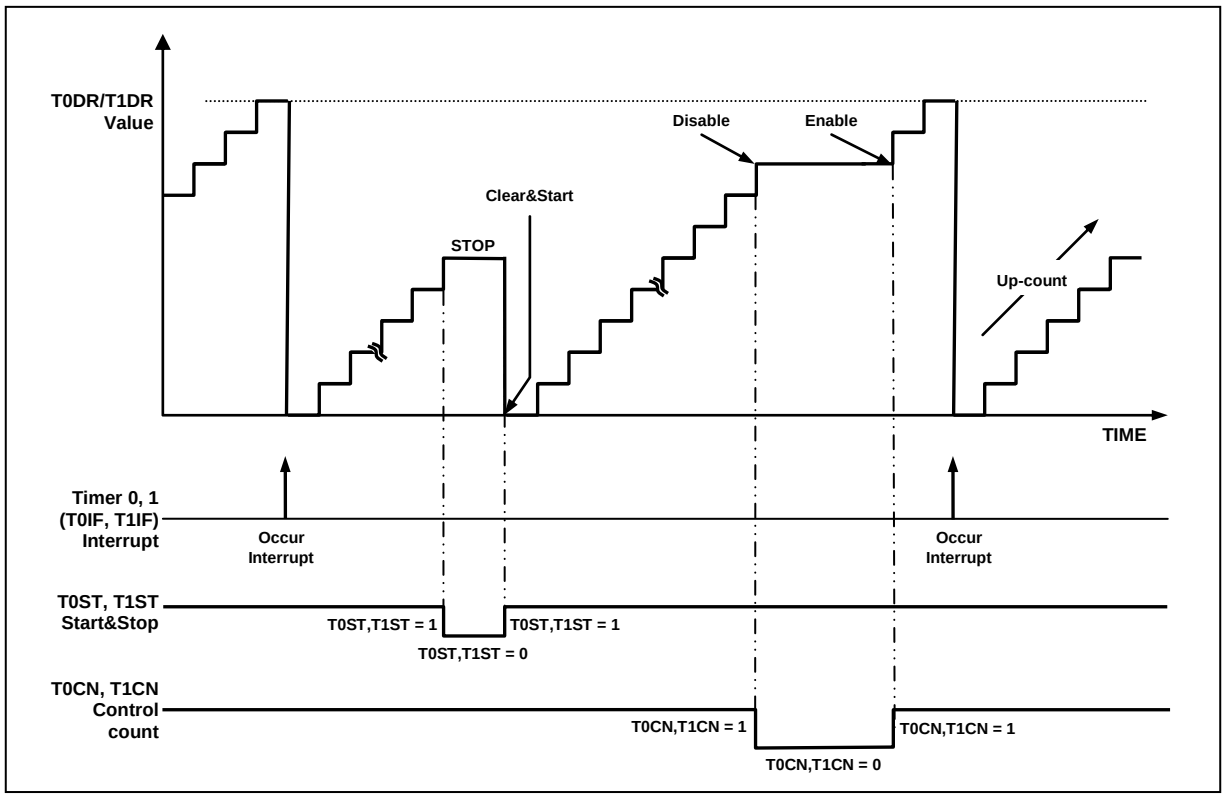


Figure 11-7 Counter Operation of Timer 0, 1

11.4.1.3 16-bit Timer/Counter Mode

When Timer 0, 1 are configured as 16-bit Timer/Counter Mode, Timer 0 becomes the lower part of the new 16-bit counter. When the lower 8-bit counter T0 matches T0DR and higher 8-bit counter T1 matches T1DR simultaneously, a 16-bit timer interrupt is issued via Timer 0 interrupt(not Timer 1). Both T0 and T1 should use the same clock source, which leads to the configuration, T1CK1=1, T1CK0=1 and 16BIT=1 in T1CR register. This means to use two separate 8-bit counters(T0, T1) as a single 16-bit counter, T1 must be clocked by the clock source of T0. This is shown in the following figure.

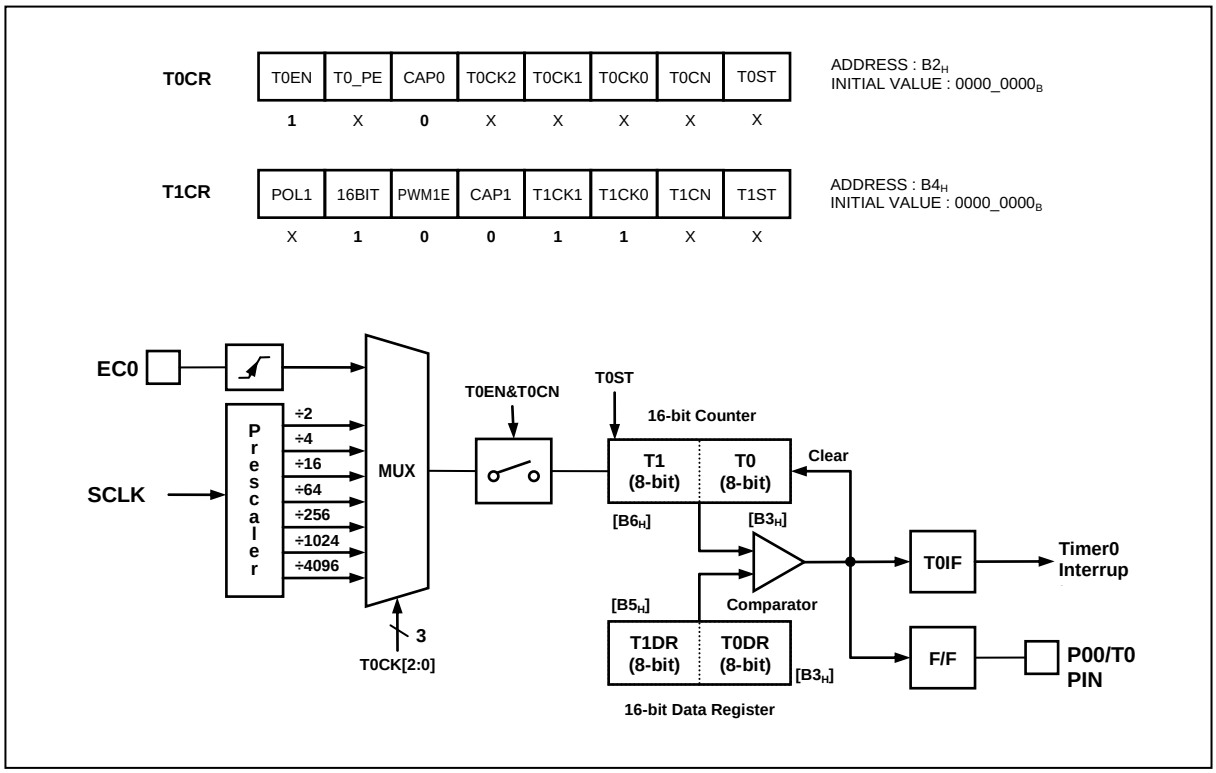


Figure 11-8 Block Diagram of Timer 0, 1 in 16-bit Timer/ Counter mode

In 8-bit Timer/Counter Mode, timer output is toggled and appears on P00(P01) port whenever T0(T1) matches T0DR(T1DR). In 16-bit Timer/Counter Mode, timer output is toggled and appears on P01 port whenever T1+T0 matches T1DR+T0DR. The initial value of each timer's output is '0' and output frequency is calculated by the following equation.

$$f_{COMP} = \frac{\text{Timer Clock Frequency}}{2 \times \text{Prescaler Value} \times (TnDR + 1)}$$

where f_{COMP} is the frequency of timer output, TnDR is T0DR or T1DR in 8-bit timer mode or concatenated T1DR+T0DR in 16-bittimer mode.

To observe timer output via port, T0_PE in T0CR register or T1_PE in PWM1HR register must be set.

11.4.1.4 8-bit Capture Mode

By setting CAP0(CAP1) to '1' in T0CR(T1CR) register, Timer 0(Timer 1) operates in Capture Mode. Basic timer function is still effective even in capture mode. So when the counter value reaches to the pre-defined data value in data register, an interrupt can be issued. When an external interrupt generating condition is detected on port P36(P37), the counter value is captured into capture register CDR0(CDR1). At the same time the counter T0(T1) is cleared to 00_H and counts up again.

The timer interrupt in Capture Mode is very useful when the interval of capture event on port P36(P37) is longer than the interrupt period of timer. That is, by counting number of timer interrupt, user software can figure out the time interval of external event. As you know, external interrupt is triggered by a falling edge, a rising edge or both edge according to the setting of EDEDGE register(Interrupt Edge Selection Register, AD_H).

CDR0, T0 and T0DR registers share peripheral address. Reading T0DR gives the value of CDR0 in Capture Mode, T0 in Timer/Count Mode. Writing T0DR alters the contents of T0DR in any mode. CDR1, T1 and T1DR is all the same as above.

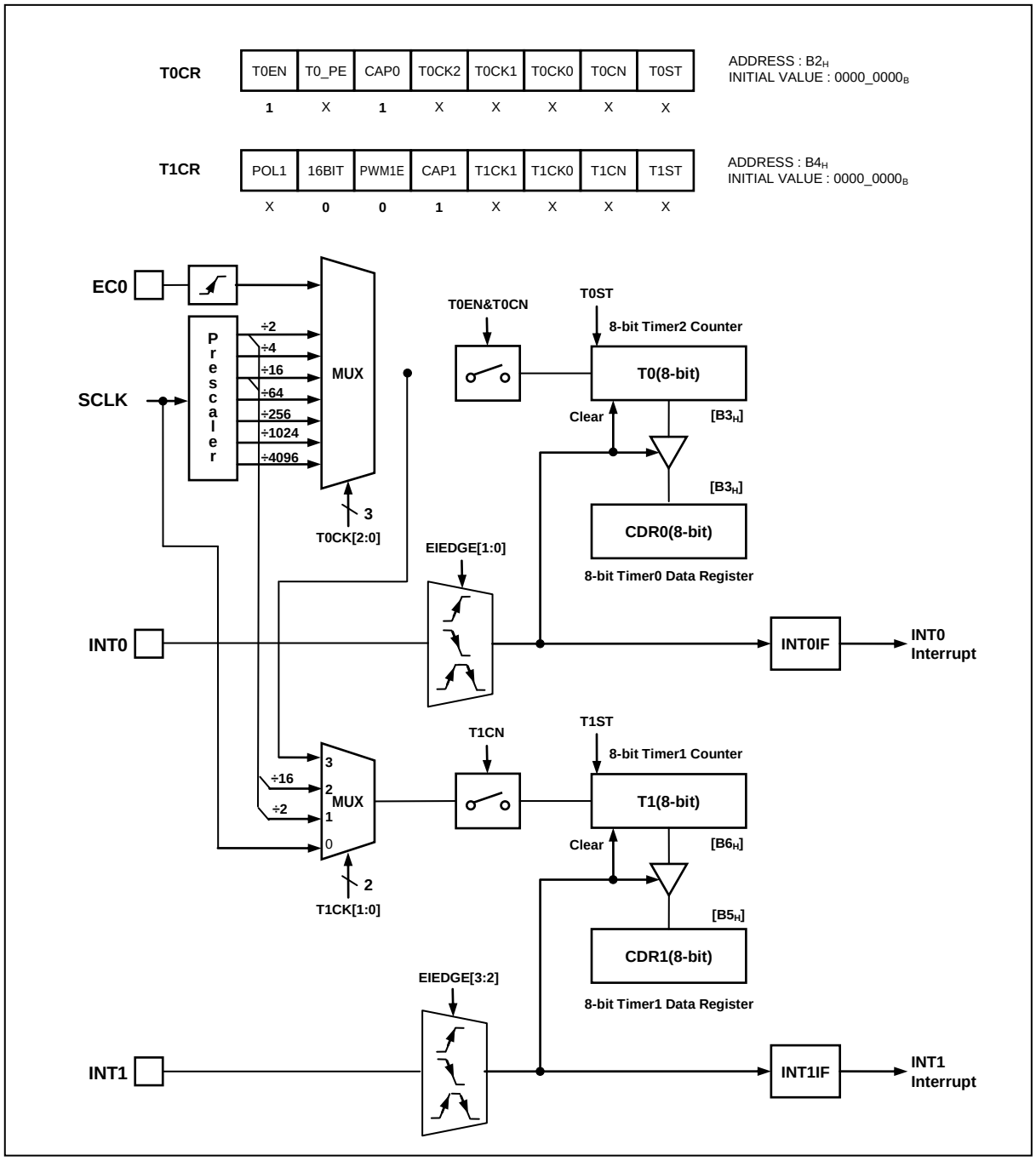


Figure 11-9 Block Diagram of Timer 0, 1 in 8-bit Capture mode

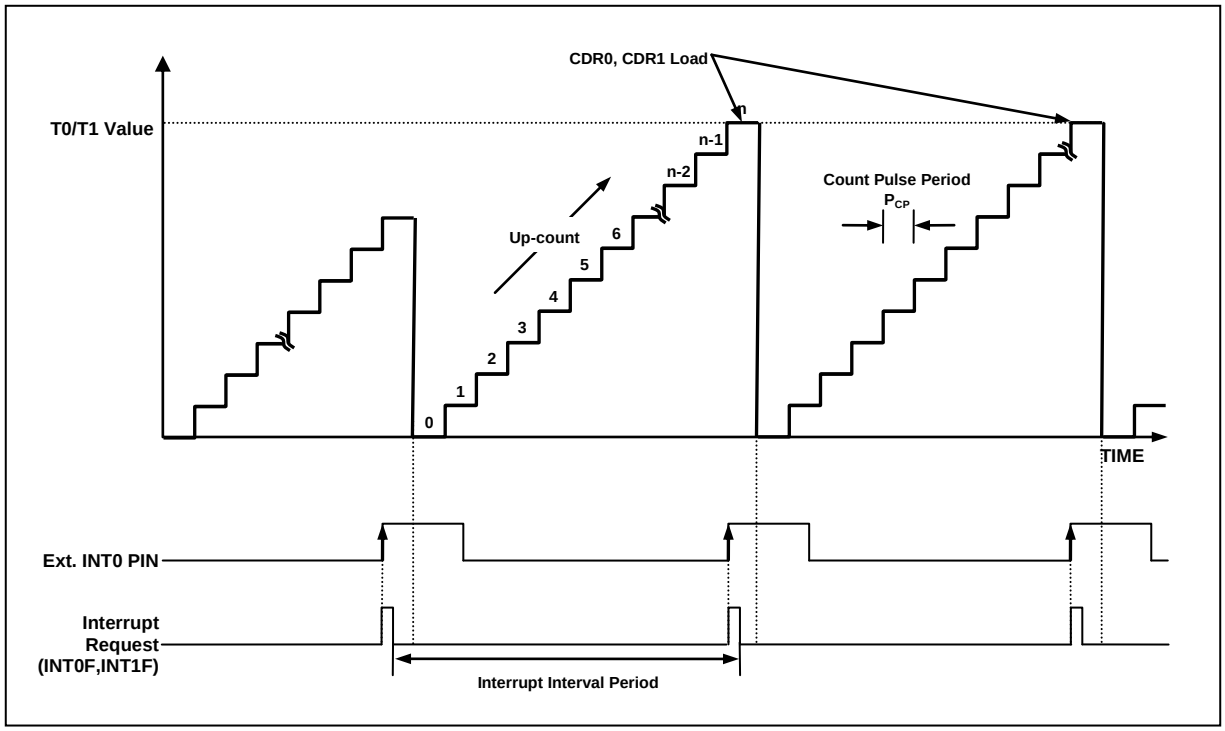


Figure 11-10 Timer 0,1 Operation in 8-bit Input Capture Mode

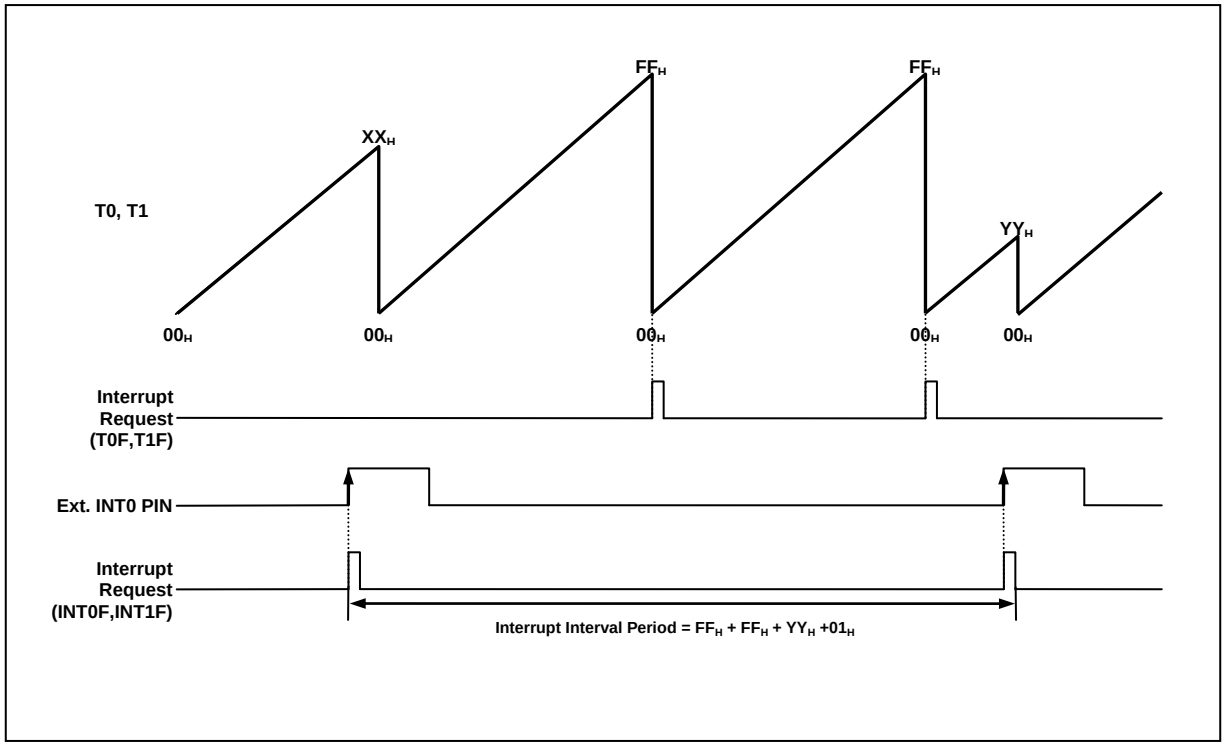


Figure 11-11 Example of Capture Interval Calculation in 8-bit Input Capture Mode

11.4.1.5 16-bit Capture Mode

If two 8-bit timers are combined to operate as a single 16-bit timer, this new timer can be in 16-bit Capture Mode. The operating mechanism is just like a 8-bit timer in capture mode except counter and capture register is 16-bit wide which are concatenated T0+T1 and CDR0+CDR1. The 16-bit counter T0+T1 is clocked by a clock source selected by T0CK[2:0] bits in T0CR register. And the T1CK1, T1CK0 and 16BIT bits in T1CR register must be set to '1' to operate correctly. The following figure shows how the Timer 0, 1 operate in 16-bit Capture Mode.

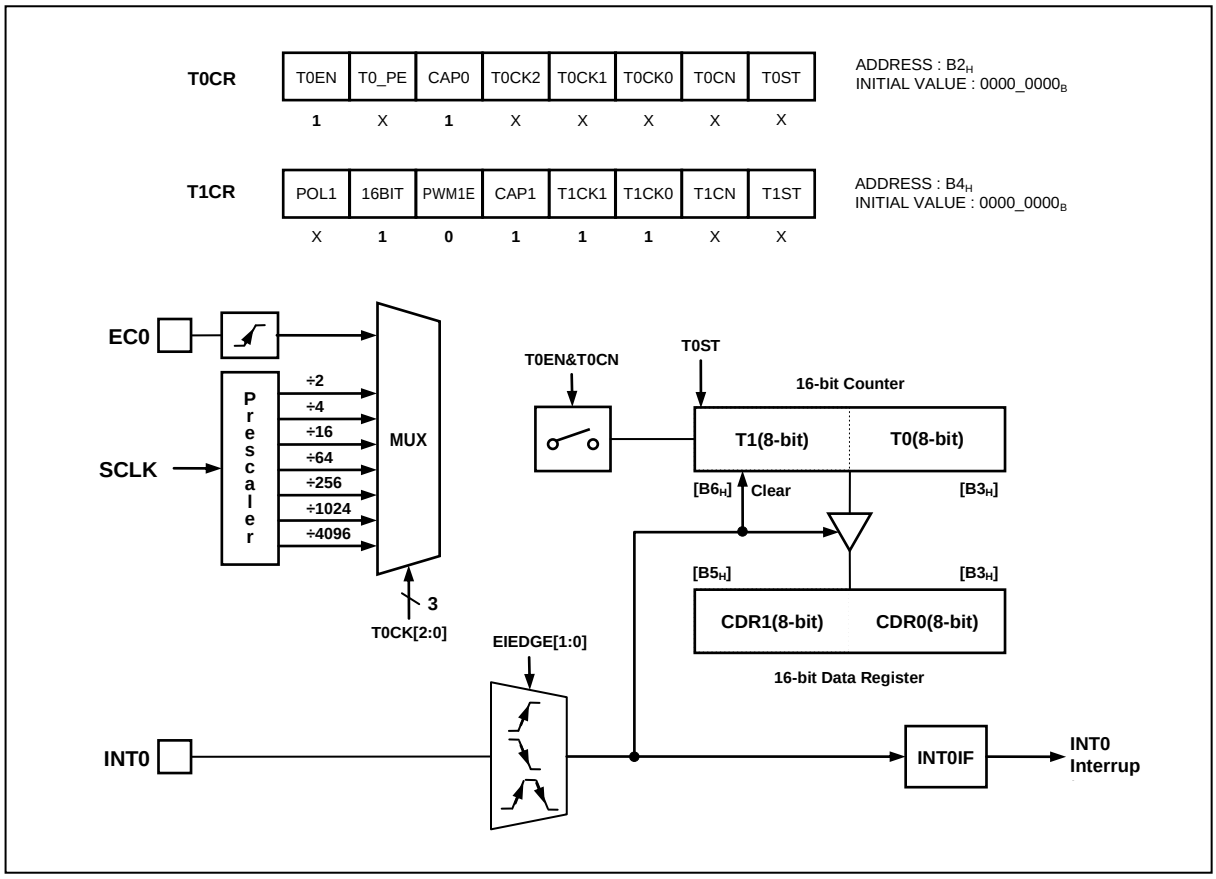


Figure 11-12 Block Diagram of Timer 0, 1 in 16-bit Capture Mode

11.4.1.6 PWM Mode (Timer 1)

Timer 1 supports simple PWM waveform generating function by setting PWM1E bit in T1CR register. To output the PWM waveform through T1/PWM1 pin, the T1_PE bit in PWM1HR register is to be set. The period and duty of PWM waveform are decided by PWM1PR(PWM Period Register), PWM1DR(PWM Duty Register) and PWM1HR registers. Note the PWM resolution is 10-bit depth, the period and duty is calculated by next equation.

PWM Period = [PWM1HR[3:2], PWM1PR] X Timer 1 Clock Period

PWM Duty = [PWM1HR[1:0], PWM1DR] X Timer 1 Clock Period

Resolution	Frequency		
	T1CK[1:0]=00 (125ns)	T1CK[1:0]=01 (250ns)	T1CK[1:0]=10 (2us)
10-bit	7.8KHz	3.9KHz	0.49KHz
9-bit	15.6KHz	7.8KHz	0.98KHz
8-bit	31.2KHz	15.6KHz	1.95KHz
7-bit	62.4KHz	31.2KHz	3.91KHz

Table 11-5 PWM Frequency vs. Resolution (In case frequency of SCLK(=f_{SCLK}) is 8MHz)

The POL bit in T1CR register determines the polarity of PWM waveform. Setting POL=1 makes PWM waveform high for duty value. In other case, PWM waveform is low for duty value.

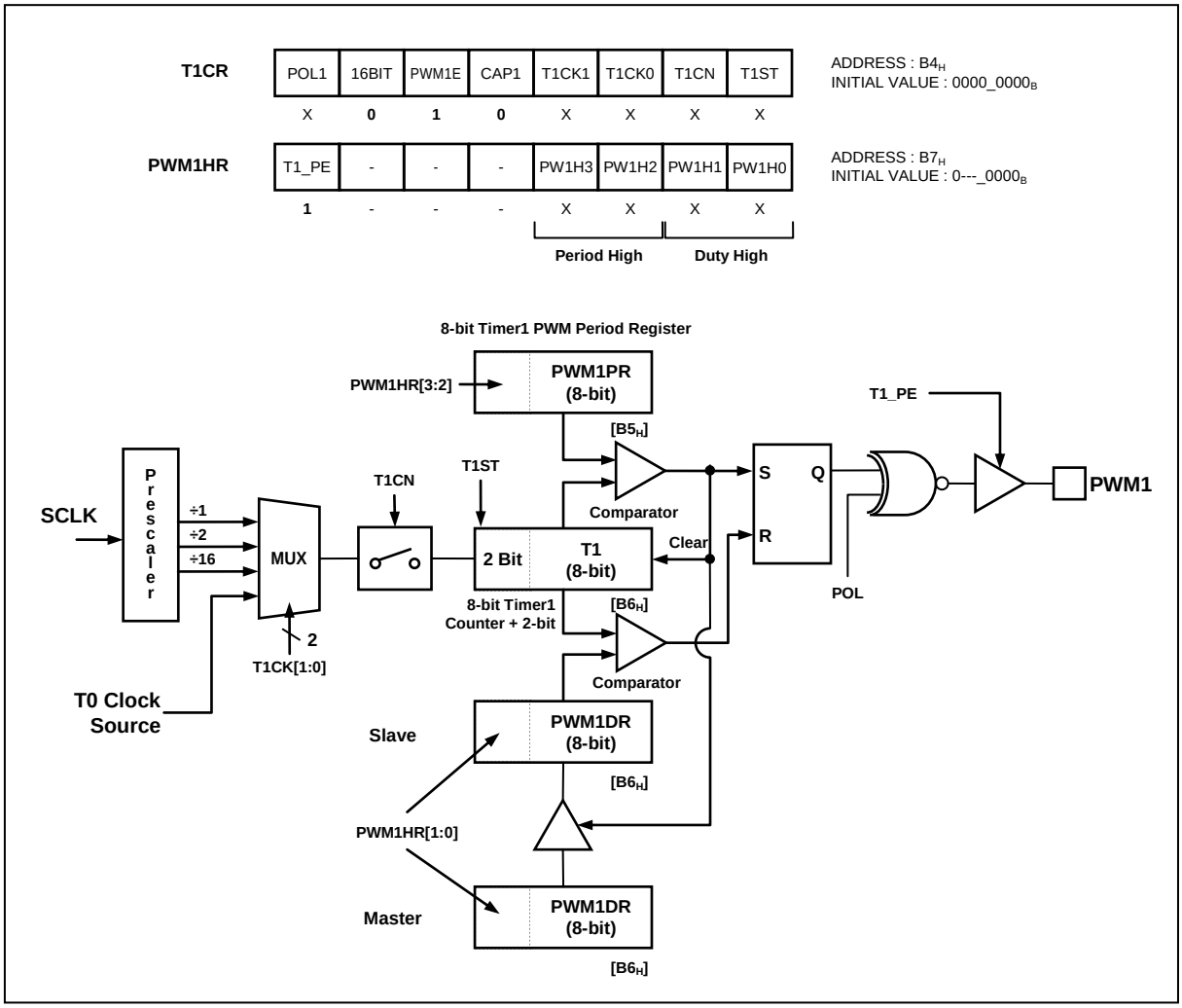


Figure 11-13 Block Diagram of Timer 1 in PWM mode

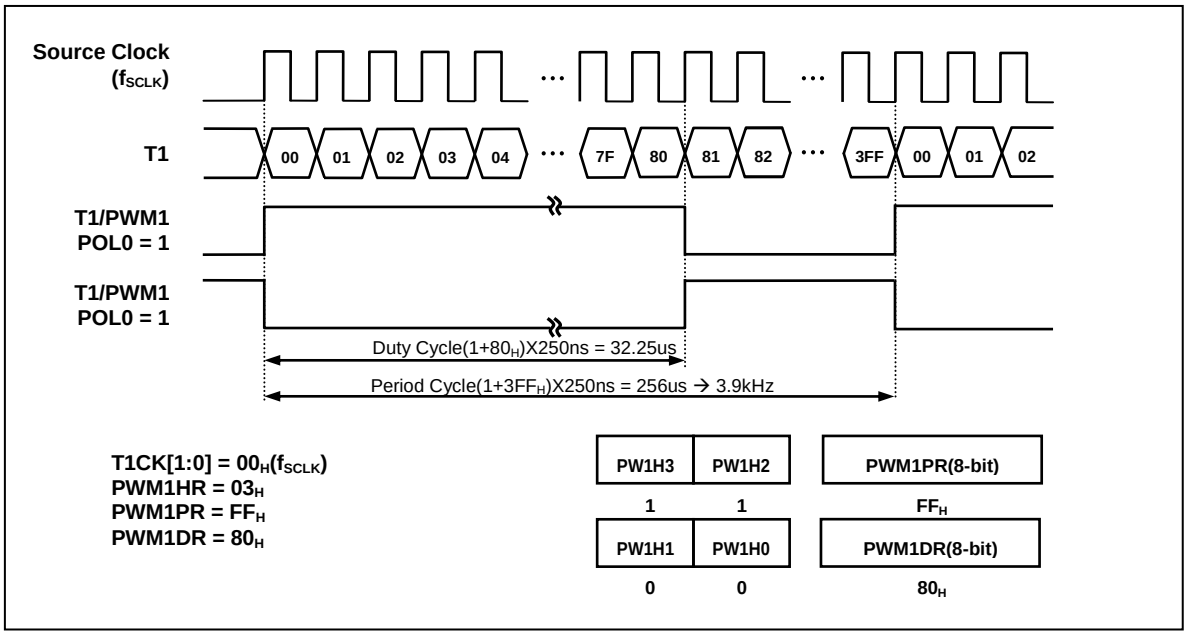


Figure 11-14 Example of PWM Waveform (In case frequency of SCLK(=f_{SCLK}) is 4MHz)

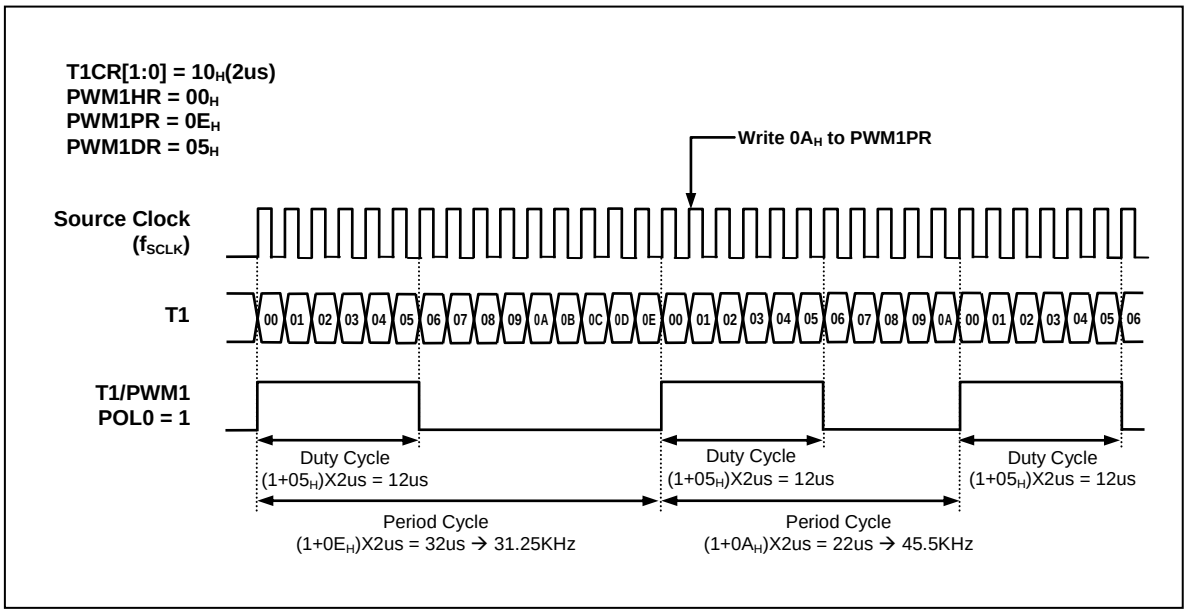


Figure 11-15 Behaviour of waveform when changing period (In case f_{SCLK} is 4MHz)

11.4.1.7 Register Map

Name	Address	Dir	Default	Description
T0CR	$B2_H$	R/W	00_H	Timer 0 Mode Control Register
T0	$B3_H$	R	00_H	Timer 0 Register
T0DR	$B3_H$	W	FF_H	Timer 0 Data Register
CDR0	$B3_H$	R	00_H	Capture 0 Data Register
T1CR	$B4_H$	R/W	00_H	Timer 1 Mode Control Register

T1DR	$B5_H$	W	FF_H	Timer 1 Data Register
PWM1PR	$B5_H$	W	FF_H	Timer 1 PWM Period Register
T1	$B6_H$	R	00_H	Timer 1 Register
PWM1DR	$B6_H$	R/W	00_H	Timer 1 PWM Duty Register
CDR1	$B6_H$	R	00_H	Capture 1 Data Register
PWM1HR	$B7_H$	W	00_H	Timer 1 PWM High Register

Table 11-6 Register Map of Timer 0, 1

11.4.1.8 Register Description

T0CR (Timer 0 Mode Control Register) **$B2_H$**

7	6	5	4	3	2	1	0
T0EN	T0_PE	CAP0	T0CK2	T0CK1	T0CK0	T0CN	T0ST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

T0EN Enables or disables Timer 0 module.
- 0 Disable Timer 0
- 1 Enable Timer 0

T0_PE Controls whether to output Timer 0 output or not through I/O pin.
- 0 Timer 0 output does not come out through I/O pin
- 1 Timer 0 output overrides the normal port functionality of I/O pin

CAP0 Selects operating mode of Timer 0.
- 0 Timer/Counter mode
- 1 Capture mode

T0CK[2:0] Selects clock source of Timer 0. NOTE

T0CK2	T0CK1	T0CK0	Timer 0 clock
0	0	0	$f_{SCLK}/2$
0	0	1	$f_{SCLK}/2^2$
0	1	0	$f_{SCLK}/2^4$
0	1	1	$f_{SCLK}/2^6$
1	0	0	$f_{SCLK}/2^8$
1	0	1	$f_{SCLK}/2^{10}$
1	1	0	$f_{SCLK}/2^{12}$
1	1	1	External Clock (EC0)

T0CN Decides whether to pause or continue counting
- 0 Pause counting temporarily
- 1 Continue to count

T0ST Decides whether to start or stop counter
- 0 Stops counting
- 1 Clear counter and starts up-counting

NOTE f_{SCLK} is the frequency of internal operating clock, SCLK.

T0 (Timer 0 Register, Read Case) **$B3_H$**

7	6	5	4	3	2	1	0
T07	T06	T05	T04	T03	T02	T01	T00

R	R	R	R	R	R	R	R

Initial value : 00_H

T0[7:0] T0 Counter value

T0DR (Timer 0 Data Register, Write Case) B3H

7	6	5	4	3	2	1	0
T0D7	T0D6	T0D5	T0D4	T0D3	T0D2	T0D1	T0D0
W	W	W	W	W	W	W	W

Initial value : FF_H

T0D[7:0] T0 Compare data

CDR0 (Capture 0 Data Register, Read Case) B3H

7	6	5	4	3	2	1	0
CDR07	CDR06	CDR05	CDR04	CDR03	CDR02	CDR01	CDR00
R	R	R	R	R	R	R	R

Initial value : 00_H

CDR0[7:0] T0 Capture value

T1CR (Timer 1 Mode Count Register) B4H

7	6	5	4	3	2	1	0
POL	16BIT	PWM1E	CAP1	T1CK1	T1CK0	T1CN	T1ST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

POL Selects polarity of PWM
- 0 PWM waveform is low for duty value
- 1 PWM waveform is high for duty value

16BIT Selects width of Timer 0,1
- 0 Timer 0,1 are two separate 8-bit timers
- 1 Timer 0+1 is combined single 16-bit timer

PWM1E Enable PWM function of Timer 1
- 0 Timer 1 is Normal Timer/Counter
- 1 Timer 1 is PWM

CAP1 Selects operating mode of Timer 1.
- 0 Timer/Counter mode
- 1 Capture mode

T1CK[1:0] Selects the clock source of Timer 1.

T1CK1	T1CK0	Timer 1 clock
0	0	f_{SCLK}
0	1	$f_{SCLK}/2$
1	0	$f_{SCLK}/2^4$
1	1	Timer 0 Clock

T1CN Decides whether to pause or continue counting
- 0 Pause counting temporarily
- 1 Continue to count

T1ST Decides whether to start or stop counter

0	Stops counting
1	Clear counter and starts up-counting

T1DR (Timer 1 Data Register, Write Case) **$B5_H$**

7	6	5	4	3	2	1	0
T1D7	T1D6	T1D5	T1D4	T1D3	T1D2	T1D1	T1D0
W	W	W	W	W	W	W	W

Initial value : FF_H

T1D[7:0] T1 Compare data

PWM1PR (Timer 1 PWM Period Register, Write Case) **$B5_H$**

7	6	5	4	3	2	1	0
T1PP7	T1PP6	T1PP5	T1PP4	T1PP3	T1PP2	T1PP1	T1PP0
W	W	W	W	W	W	W	W

Initial value : FF_H

T1PP[7:0] Period of PWM waveform

T1 (Timer 1 Register, Read Case) **$B6_H$**

7	6	5	4	3	2	1	0
T17	T16	T15	T14	T13	T12	T11	T10
R	R	R	R	R	R	R	R

Initial value : 00_H

T1[7:0] T1 Counter value

PWM1DR (Timer 1 PWM Duty Register, Write Case) **$B6_H$**

7	6	5	4	3	2	1	0
T1PD7	T1PD6	T1PD5	T1PD4	T1PD3	T1PD2	T1PD1	T1PD0
W	W	W	W	W	W	W	W

Initial value : 00_H

T1PD[7:0] Duty of PWM waveform. NOTE) This register is meaningful only when PWM1E bit in T1CR register is '1'.

CDR1 (Capture 1 Data Register, Read Case) **$B6_H$**

7	6	5	4	3	2	1	0
CDR17	CDR16	CDR15	CDR14	CDR13	CDR12	CDR11	CDR10
R	R	R	R	R	R	R	R

Initial value : 00_H

CDR1[7:0] T1 Capture value

PWM1HR (Timer 1 PWM High Register) **$B7_H$**

7	6	5	4	3	2	1	0
T1_PE	-	-	-	PW1H3	PW1H2	PW1H1	PW1H0

W	-	-	-	W	W	W	W

Initial value : 00_H

T1_PE	Controls whether to output Timer 1 output or not through I/O pin. Note this bit is write-only. 0 Timer 1 output does not come out through I/O pin 1 Timer 1 output overrides the normal port functionality of I/O pin
PW1H[3:2]	High (bit [9:8]) value of PWM period
PW1H[1:0]	High (bit [9:8]) value of PWM duty

When Timer 1 operates in PWM mode, PW1H[3:2] and T1PP constitute the period of PWM, PW1H[1:0] and T1PD constitute the duty of PWM.

11.4.2 16-bit Timer 2

11.4.2.1 Overview

16-bit Timer 2 is composed of Multiplexer, Timer Data Register High/Low, Timer Register High/Low and Mode Control Register.

Timer 2 is clocked by Carrier Signal (CRF) from Carrier Generator module or by an internal clock source deriving from clock divider logic where the base clock is SCLK. This timer supports output compare(Timer/Counter) and input capture function.

When IRCEN bit in IRCC1 is set, Timer 2 operates in IR capture mode. In this mode Timer 2 can detect the envelope of IR input signal or counts the number of input carrier signal. In envelop detection mode of operation, the counter value of Timer 2 is captured into timer capture register on first rising edge of IR input(normally amplified carrier signal) and overflow of WT. When Timer 2 is used to calculate the number of carrier signal, the rising edge of input carrier becomes the clock source of Timer 2. For more information about IR capture operation, refer to WT and IRCC section.

11.4.2.2 16-bit Output Compare or Event Counter Mode

When Timer 2 is in Output Compare or Event Counter Mode, timer output is toggled and appears on P02 port whenever T2(T2H+T2L) matches T2DR(T2DRH+T2DRL). An interrupt can be requested if enabled and the interrupt flag can be read through T3CR2 register. The initial value of timer output is ‘0’ and output frequency is calculated by the following equation.

$$f_{COMP} = \frac{\text{Timer Clock Frequency}}{2 \times \text{Prescaler Value} \times (T2DR + 1)}$$

where f_{COMP} is the frequency of timer output, T2DR is concatenated T2DRH+T2DRL. The clock source of Timer 2 is selected by T2CK[2:0] bits in T2CR register. To observe timer output via port, set T2_PE bit in T2CR register to ‘1’.

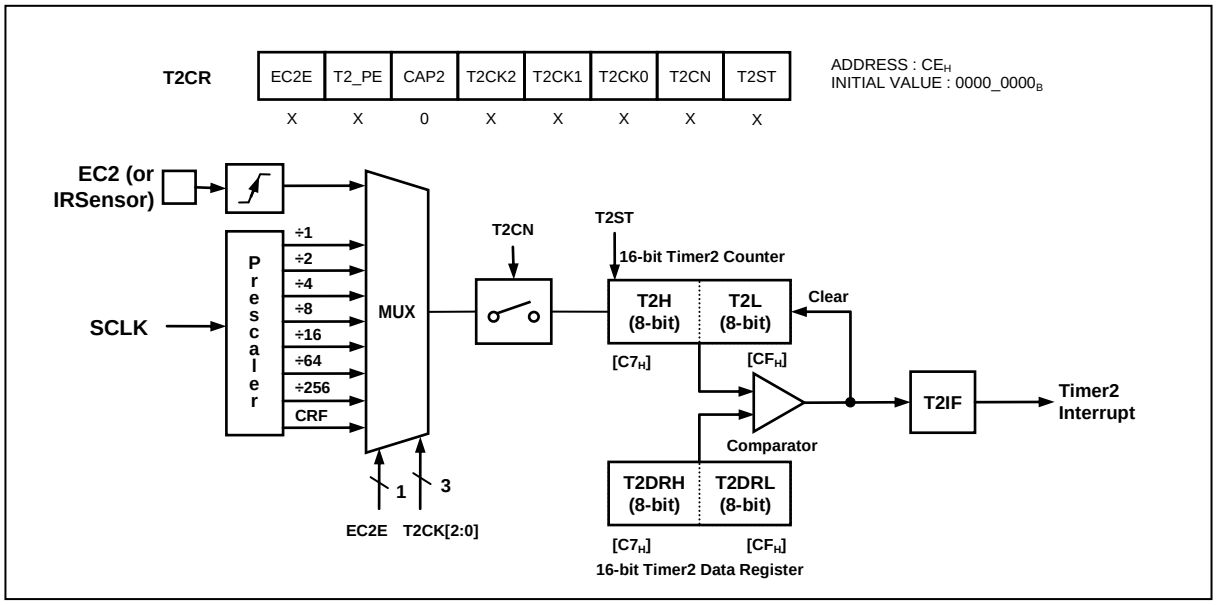


Figure 11-16 Block Diagram of 16-bit Timer 2 in Output Compare or Event Counter Mode

11.4.2.3 16-bit Capture Mode

Capture Mode is enabled by setting CAP2 bit in T2CR register. The clock source is the same as in output compare mode of operation.

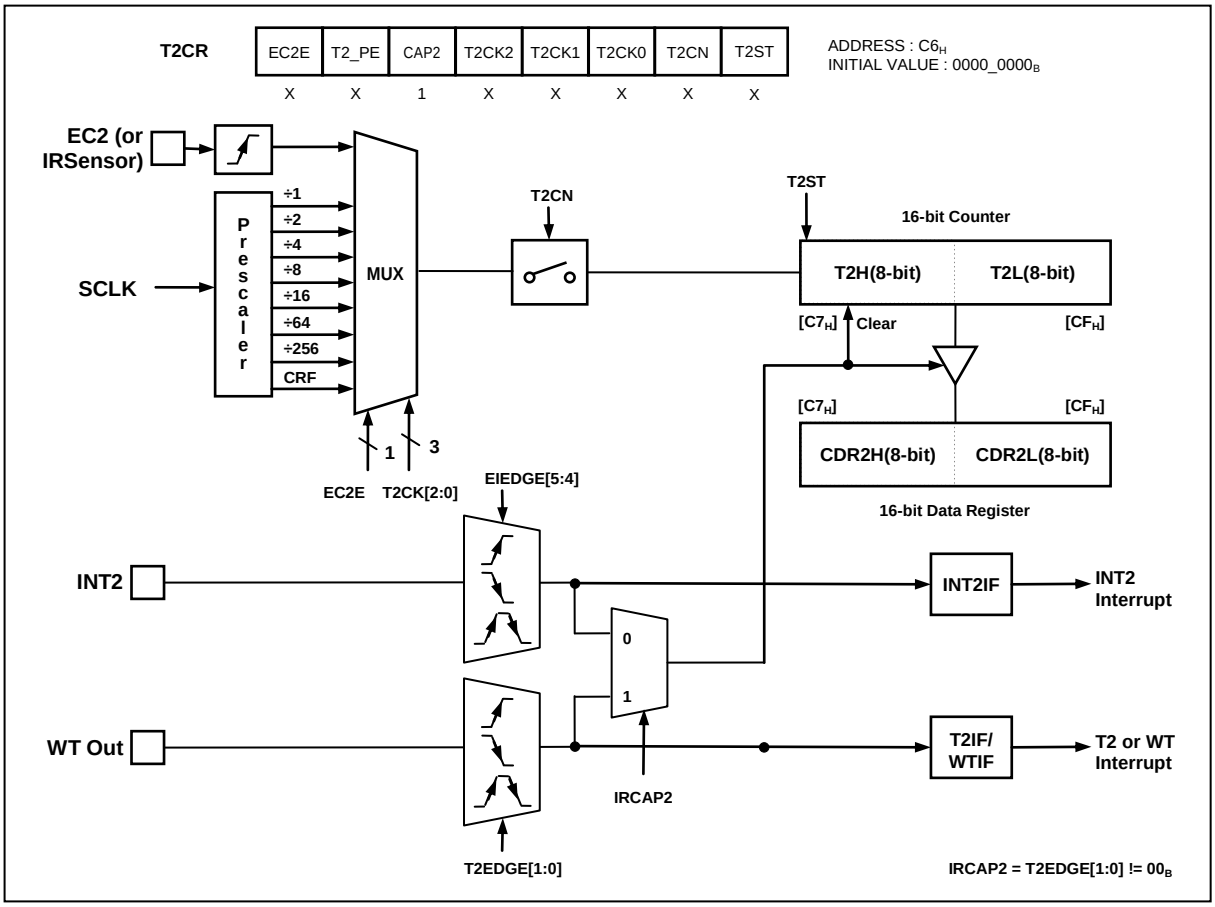


Figure 11-17 Block Diagram of Timer 2 in Capture Mode

When T2H+T2L reaches to the value of T2DRH+T2DRL, an interrupt is requested if enabled. When a compare-match occurs, the counter values T2H and T2L are captured into the capture registers CDR2H and CDR2L respectively. At the same time, the counter is cleared to 0000_H and starts up-counting.

Bit 4 and 5 in EIEDGE (External Interrupt Edge Selection Register, AD_H) register select the triggering condition of external interrupt 2(INT2), a falling edge, a rising edge or both edge.

When Timer 2 operates in IR capture mode, the capture source becomes the output of IR AMP. And the T2EDGE[1:0] bits in IRCC2 register select the triggering condition of Watch Timer output. In this mode, Timer 2 detects the envelop of input carrier signal, and the T2IR bit in IRCC2 register should be cleared to '0'

11.4.2.4 Carrier Counting Mode

Carrier Counting Mode is enabled by setting T2IR bit in IRCC2 register. This mode of operation is only available when IRCEN bit in IRCC1 register is set. The clock source is the rising edge of input carrier signal. Like output compare mode, when T2H+T2L reaches to the value of T2DRH+T2DRL, an interrupt is requested if enabled.

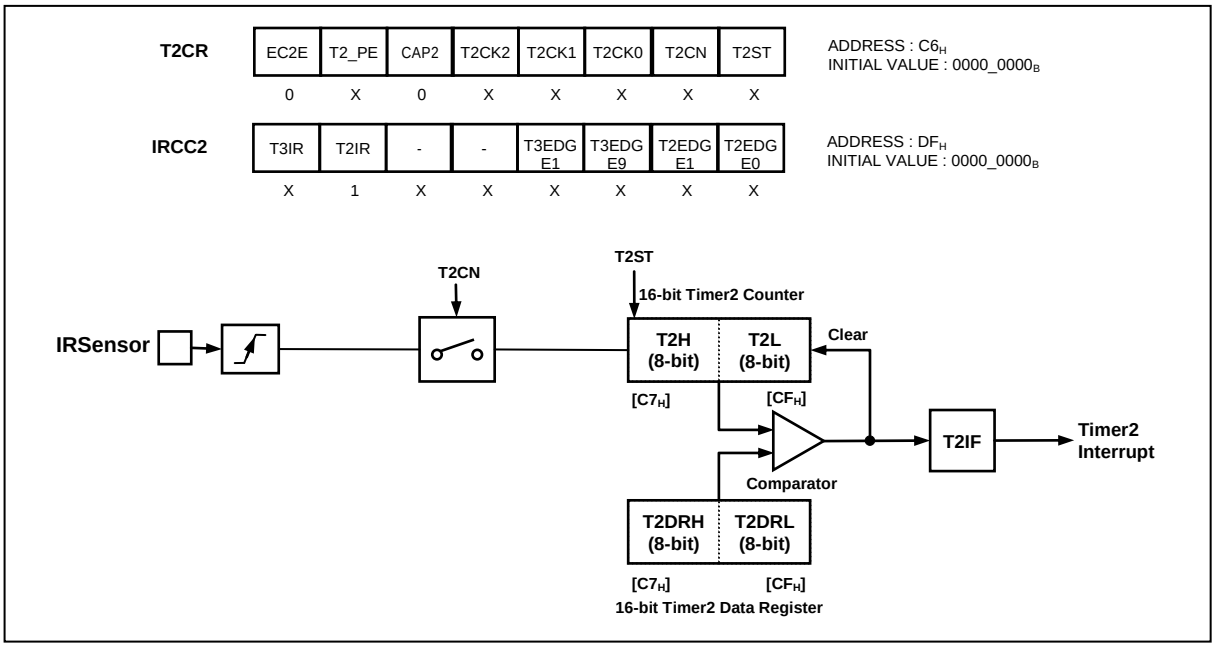


Figure 11-18 Block Diagram of Timer 2 in Carrier Counting Mode

The EC2E and CAP2 bit in T2CR register should be cleared to '0' for proper operation.

11.4.2.5 Register Map

Name	Address	Dir	Default	Description
T2CR	$C6_H$	R/W	00_H	Timer 2 Mode Control Register
T2H	$C7_H$	R	00_H	Timer 2 Counter High
T2DRH	$C7_H$	W	FF_H	Timer 2 Data Register High

CDR2H	$C7_H$	R	00_H	Timer 2 Capture Data Register High
T2L	CF_H	R/W	00_H	Timer 2 Counter Low
T2DRL	CF_H	W	FF_H	Timer 2 Data Register Low
CDR2L	CF_H	R	00_H	Timer 2 Capture Data Register Low

Table 11-7 Register Map of Timer 2

11.4.2.6 Register Description

CDR2H, T2DRH and T2H registers share peripheral address. Reading T2DRH gives CDR0 in Capture Mode, T2H in Output Compare Mode. Writing T2DRH alters the contents of T2DRH in any mode. This applies to the case of CDR2L, T2DRL and T2L registers.

T2CR (Timer 2 Mode Control Register) **$C6_H$**

7	6	5	4	3	2	1	0
EC2E	T2_PE	CAP2	T2CK2	T2CK1	T2CK0	T2CN	T2ST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

EC2E Enable event counter mode of Timer 2.
- 0 Timer 2 is a normal counter.
- 1 Timer 2 is an event counter clocked by EC2.

T2_PE Controls whether to output Timer 2 output or not through I/O pin.
- 0 Timer 2 output does not come out through I/O pin
- 1 Timer 2 output overrides the normal port functionality of I/O pin

CAP2 Selects operating mode of Timer 2.
- 0 Timer/Counter mode
- 1 Capture mode

T2CK[2:0] Selects clock source of Timer 2. NOTE

T2CK2	T2CK1	T2CK0	Timer 2 clock
0	0	0	f_{SCLK}
0	0	1	f_{SCLK}/2^1
0	1	0	f_{SCLK}/2^2
0	1	1	f_{SCLK}/2^3
1	0	0	f_{SCLK}/2^4
1	0	1	f_{SCLK}/2^6
1	1	0	f_{SCLK}/2^8
1	1	1	CRF (Carrier)

T2CN Decides whether to pause or continue counting.
- 0 Pause counting temporarily
- 1 Continue to count

T2ST Decides whether to start or stop counter
- 0 Stops counting
- 1 Clears counter and starts up-counting

NOTE f_{SCLK} is the frequency of internal operating clock, SCLK.

T2L (Timer 2 Counter Low, Read Case) **CF_H**

7	6	5	4	3	2	1	0
T2L7	T2L6	T2L5	T2L4	T2L3	T2L2	T2L1	T2L0
R	R	R	R	R	R	R	R

Initial value : 00_H

T2L[7:0] T2 Counter Low

T2DRL (Timer 2 Data Register Low, Write Case) CF_H

7	6	5	4	3	2	1	0
T2DRL7	T2DRL6	T2DRL5	T2DRL4	T2DRL3	T2DRL2	T2DRL1	T2DRL0
W	W	W	W	W	W	W	W

Initial value : FF_H

T2DRL[7:0] T2 Compare Data Low

CDR2L (Capture Data Register 2 Low, Read Case) CF_H

7	6	5	4	3	2	1	0
CDR2L7	CDR2L6	CDR2L5	CDR2L4	CDR2L3	CDR2L2	CDR2L1	CDR2L0
R	R	R	R	R	R	R	R

Initial value : 00_H

CDR2L[7:0] T2 Capture Data Low

T2H (Timer 2 Counter High, Read Case) $C7_H$

7	6	5	4	3	2	1	0
T2H7	T2H6	T2H5	T2H4	T2H3	T2H2	T2H1	T2H0
R	R	R	R	R	R	R	R

Initial value : 00_H

T2H[7:0] T2 Counter High

T2DRH (Timer 2 Data Register High, Write Case) $C7_H$

7	6	5	4	3	2	1	0
T2DRH7	T2DRH6	T2DRH5	T2DRH4	T2DRH3	T2DRH2	T2DRH1	T2DRH0
W	W	W	W	W	W	W	W

Initial value : FF_H

T2DRH[7:0] T2 Compare Data High

CDR2H (Capture Data Register 2 High, Read Case) $C7_H$

7	6	5	4	3	2	1	0
CDR2H7	CDR2H6	CDR2H5	CDR2H4	CDR2H3	CDR2H2	CDR2H1	CDR2H0
R	R	R	R	R	R	R	R

Initial value : 00_H

CDR2H[7:0] T2 Capture Data High

11.4.3 16-bit Timer 3

11.4.3.1 Overview

16-bit Timer 3 is composed of Multiplexer, Timer Data Register High/Low, Timer Register High/Low, Input Capture Register High/Low, Mode Control Register, PWM Duty High/Low and PWM Period High/Low Register.

Timer 3 is can be clocked by Carrier Signal(CRF) from Carrier Generator module or by an internal clock source deriving from clock divider logic where the base clock is SCLK.

When IRCEN bit in IRCC1 is set, Timer 3 operates in IR capture mode. In this mode Timer 3 can detect the envelope of IR input signal or counts the number of input carrier signal. In envelop detection mode of operation, the counter value of Timer 3 is captured into timer capture register on first rising edge of IR input(normally amplified carrier signal) and overflow of WT. When Timer 3 is used to calculate the number of carrier signal, the rising edge of input carrier becomes the clock source of Timer 3. For more information about IR capture operation, refer to WT and IRCC section.

11.4.3.2 16-bit Output Compare or Event Counter Mode

When Timer 3 is in Output Compare or Event Counter Mode, timer output is toggled and appears on P03 port whenever T3(T3H+T3L) matches T3DR(T3DRH+T3DRL). An interrupt can be requested if enabled and the interrupt flag can be read through T3CR2 register. The initial value of timer output is ‘0’ and output frequency is calculated by the following equation.

$$f_{COMP} = \frac{\text{Timer Clock Frequency}}{2 \times \text{Prescaler Value} \times (T3DR + 1)}$$

where f_{COMP} is the frequency of timer output. T3DR is concatenated T3DRH+T2DRL. The clock source of Timer 3 is selected by T3CK[2:0] bits in T3CR register. To observe timer output via port, set T3_PE bit in T3CR2 register to ‘1’.

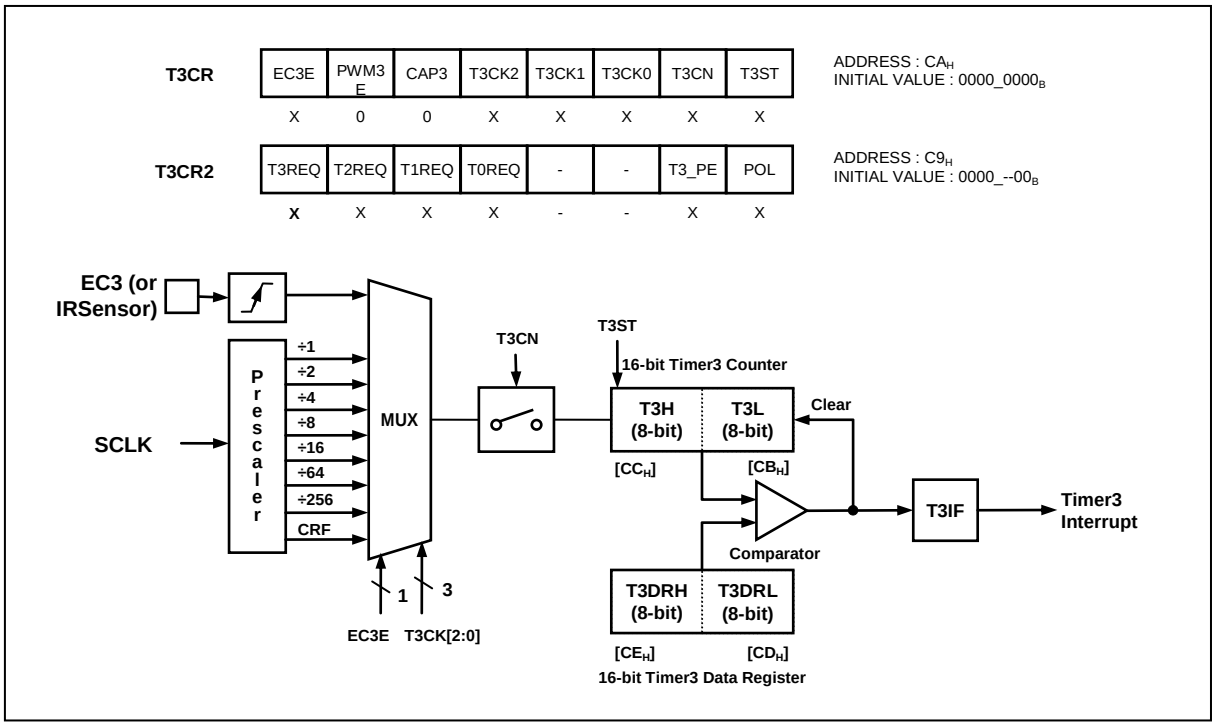


Figure 11-19 Block Diagram of Timer 3 in Output Compare or Event Counter Mode

11.4.3.3 16-bit Capture Mode

Capture Mode is enabled by setting CAP3 bit in T3CR register. The clock source is the same as in output compare mode of operation. When T3H+T3L reaches to the value of T3DRH+T3DRL, an interrupt is requested if enabled. When a compare-match occurs, the counter values T3H and T3L are captured into the capture registers CDR3H and CDR3L respectively. At the same time, the counter is cleared to 0000_H and starts up-counting.

Bit 6 and 7 in EIEDGE(External Interrupt Edge Selection Register, AD_H) register select the triggering condition of external interrupt 3(INT3), a falling edge, a rising edge or both edge.

When Timer 3 operates in IR capture mode, the capture source becomes the output of IR AMP. And the T3EDGE[1:0] bits in IRCC2 register select the triggering condition of Watch Timer output. In this mode, Timer 3 detects the envelop of input carrier signal, and the T3IR bit in IRCC2 register should be cleared to ‘0’

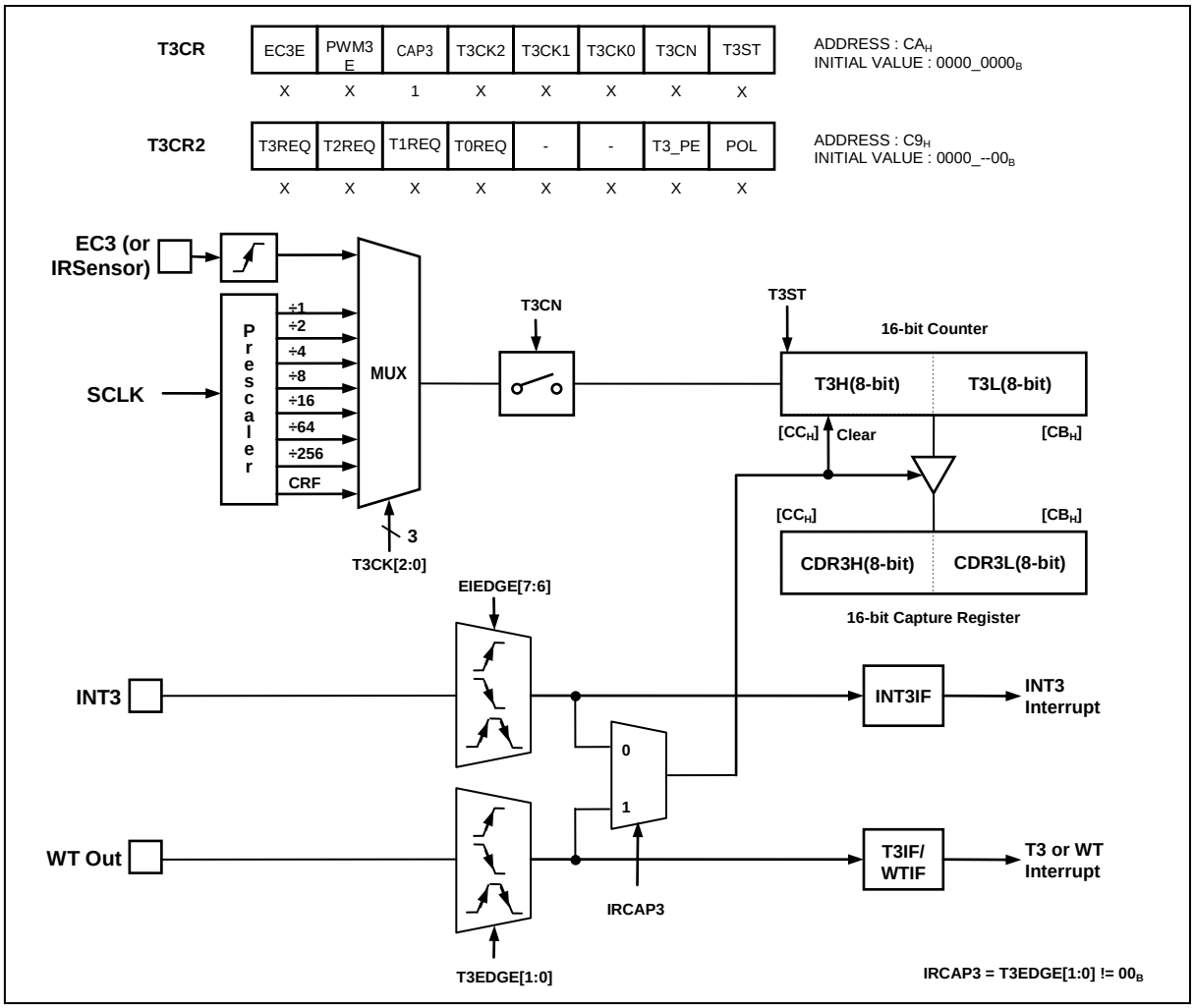


Figure 11-20 Block Diagram of Timer 3 in Capture Mode

11.4.3.4 Carrier Counting Mode

Carrier Counting Mode is enabled by setting T3IR bit in IRCC2 register. This mode of operation is only available when IRCEN bit in IRCC1 register is set. The clock source is the rising edge of input carrier signal. Like output compare mode, when T3H+T3L reaches to the value of T3DRH+T3DRL, an interrupt is requested if enabled.

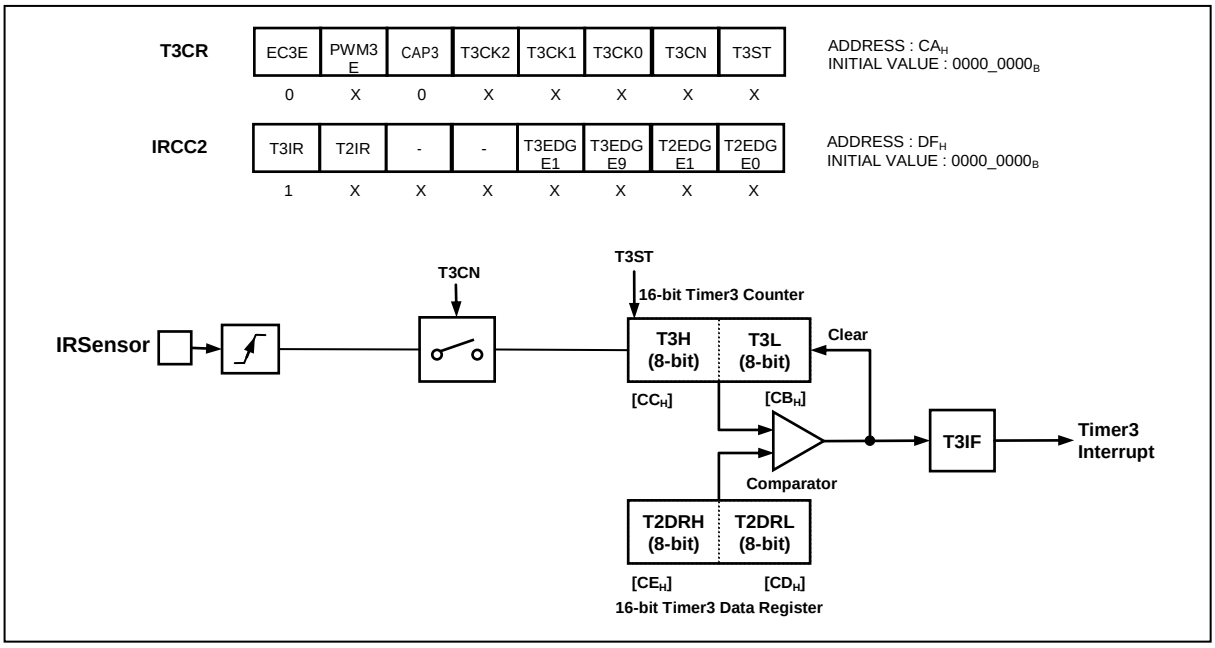


Figure 11-21 Block Diagram of Timer 3 in Carrier Counting Mode

The EC3E and CAP3 bit in T3CR register should be cleared to '0' for proper operation.

11.4.3.5 PWM Mode

Timer 3 supports simple PWM waveform generating function by setting PWM3E bit in T3CR register. As Timer 3 is 16-bit wide, the PWM resolution is also 16-bit depth. To output the PWM waveform through T3/PWM3 pin, the T3_PE bit in T3CR2 register is to be set. The period and duty of PWM waveform are decided by PWM3PRH, PWM3PRL, PWM3DRH and PWM3DRL registers. The equation to calculate period and duty is as follows.

PWM Period = [PWM3PRH, PWM3PRL] X Timer 3 Clock Period

PWM Duty = [PWM3DRH, PWM3DRL] X Timer 3 Clock Period

Resolution	Frequency		
	T3CK[2:0]=000 (250ns)	T3CK[2:0]=001 (500ns)	T3CK[2:0]=011 (2us)
16-bit	60.938Hz	30.469Hz	7.617Hz
15-bit	121.87Hz	60.938Hz	15.234Hz
10-bit	3.9KHz	1.95KHz	0.49KHz
9-bit	7.8KHz	3.9KHz	0.98KHz
8-bit	15.6KHz	7.8KHz	1.95KHz

Table 11-8 PWM Frequency vs. Resolution (In case of f_{SCLK}=4MHz)

The POL bit in T3CR register determines the polarity of PWM waveform. Setting POL=1 makes the PWM waveform high for duty value. In other case, PWM waveform is low for duty value.

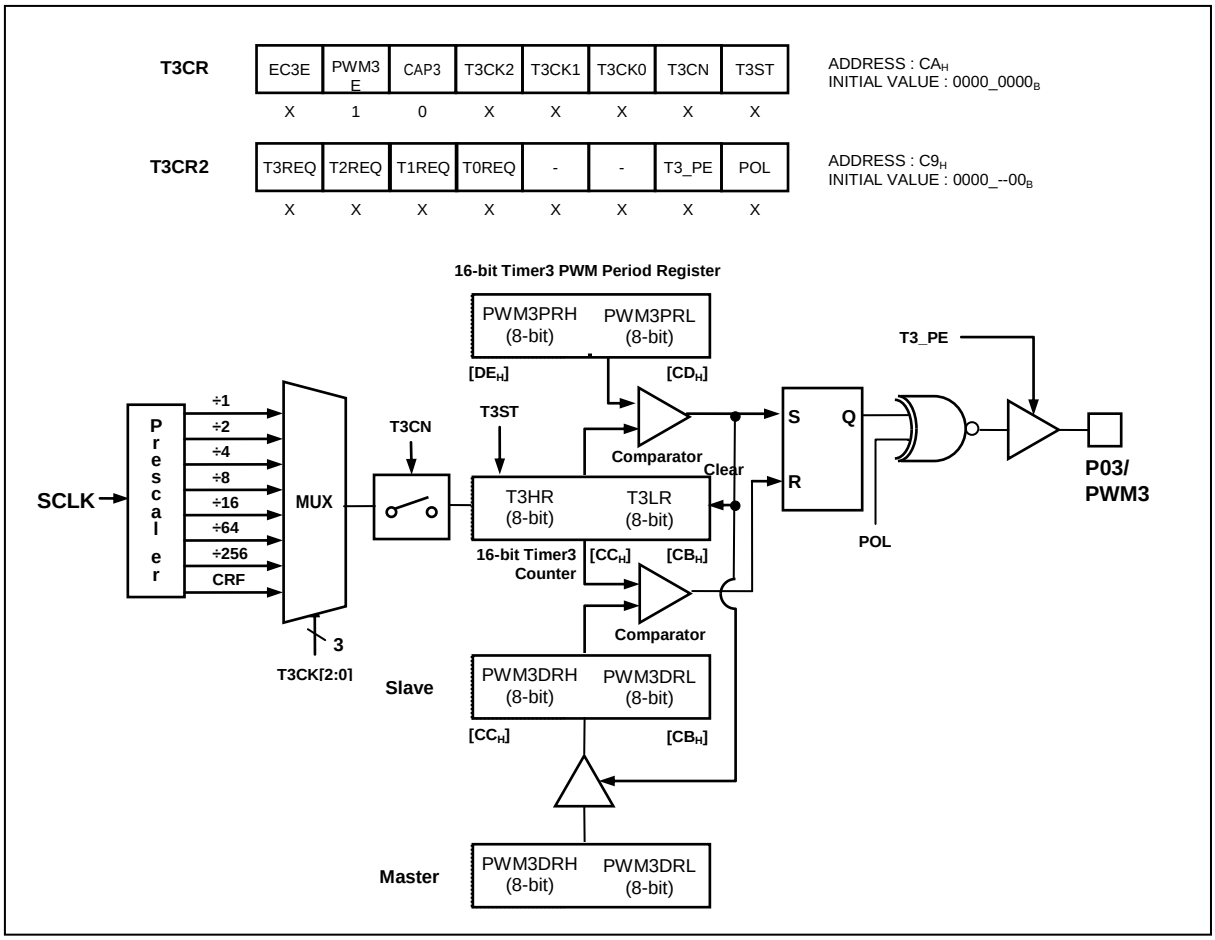


Figure 11-22 Block Diagram of Timer 3 in PWM Mode

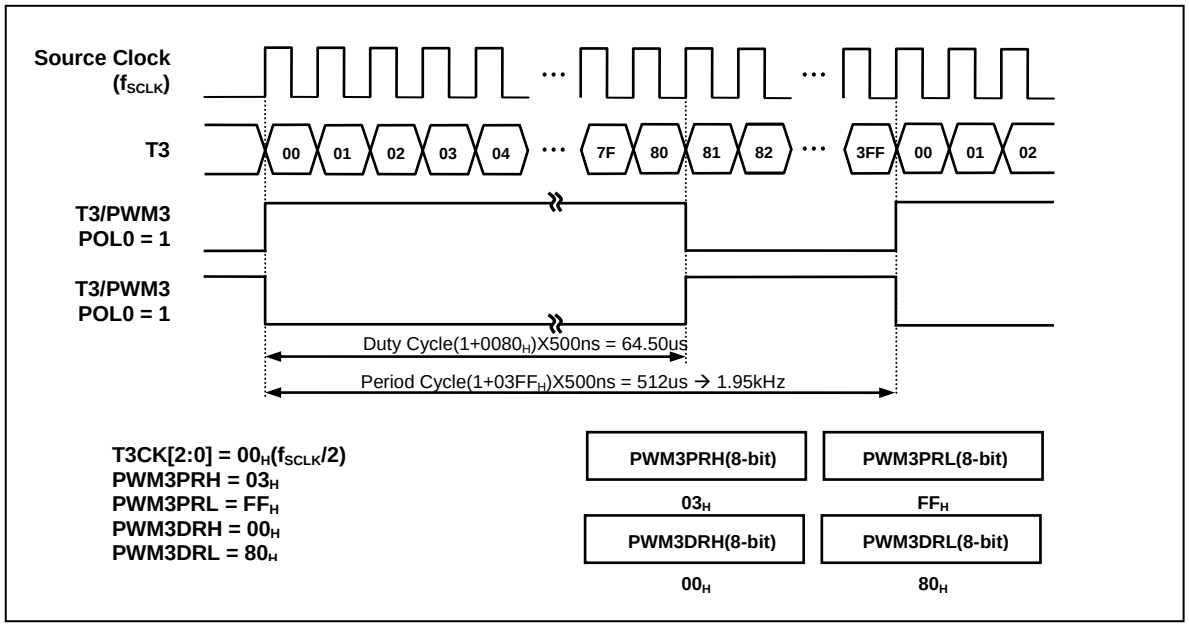


Figure 11-23 Example of PWM waveform (In case of f_{SCLK}=4MHz)

11.4.3.6 Register Map

Name	Address	Dir	Default	Description
T3CR2	$C9_H$	R/W	00_H	Timer 3 Mode Control Register 2
T3CR	CA_H	R/W	00_H	Timer 3 Mode Control Register
T3L	CB_H	R	00_H	Timer 3 Counter Low
PWM3DRL	CB_H	R/W	00_H	PWM 3 Duty Register Low
CDR3L	CB_H	R	00_H	Timer 3 Capture Data Register Low
T3H	CC_H	R	00_H	Timer 3 Counter High
PWM3DRH	CC_H	R/W	00_H	PWM 3 Duty Register High
CDR3H	CC_H	R	00_H	Timer 3 Capture Data Register High
T3DRL	CD_H	W	FF_H	Timer 3 Data Register Low
PWM3PRL	CD_H	W	FF_H	PWM 3 Period Register Low
T3DRH	CE_H	W	FF_H	Timer 3 Data Register High
PWM3PRH	CE_H	W	FF_H	PWM 3 Peiord Register High

Table 11-9 Register Map of Timer 3

11.4.3.7 Register Description

Timer3 can generate PWM output of 16-bit resolution. The period of PWM3 is decided by PWM3PRH and PWM3PRL registers and the duty of PWM3 is decided by PWM3DRH and PWM3DRL registers. PWM3PRH and PWM3PRL registers are write-only. Note that the value of period and duty registers can be changed only when PWM3E bit in T3CR register is set.

CDR3H, PWM3DRH and T3H registers share peripheral address. When PWM mode is enabled, reading this address gives PWM3DRH. When PWM mode is disabled, reading this address gives

CDR3H in Capture Mode or T3H in Output Compare Mode. Writing this address alters PWM3DRH when PWM3E bit is '1'. When PWM mode is disabled, writing this address alters T3DRH.

CDR3L, PWM3DRL and T3L registers share peripheral address. When PWM mode is enabled, reading this address gives PWM3DRL. When PWM mode is disabled, reading this address gives CDR3L in Capture Mode or T3L in Output Compare Mode. Writing this address alters PWM3DRL when PWM3E bit is '1'. When PWM mode is disabled, writing this address alters T3DRL.

T3CR (Timer 3 Mode Control Register) **CA_H**

7	6	5	4	3	2	1	0
EC3E	PWM3E	CAP3	T3CK2	T3CK1	T3CK0	T3CN	T3ST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

EC3E Enable event counter mode of Timer 3.
- 0 Timer 3 is a normal counter.
- 1 Timer 3 is an event counter clocked by EC3.

PWM3E Enable PWM function of Timer 3
- 0 Timer 3 is Normal Timer/Counter
- 1 Timer 1 is PWM

CAP3 Selects operating mode of Timer 3
- 0 Timer/Counter mode
- 1 Capture mode

T3CK[2:0] Selects the clock source of Timer 3. NOTE

T3CK2	T3CK1	T3CK0	Timer 3 clock
0	0	0	f_{SCLK}
0	0	1	f_{SCLK}/2^1
0	1	0	f_{SCLK}/2^2
0	1	1	f_{SCLK}/2^3
1	0	0	f_{SCLK}/2^4
1	0	1	f_{SCLK}/2^6
1	1	0	f_{SCLK}/2^8
1	1	1	CRF (Carrier)

T3CN Decides whether to pause or continue counting
- 0 Pause counting temporarily
- 1 Continue to count

T3ST Decides whether to start or stop counter
- 0 Stops counting
- 1 Clear counter and starts up-counting

NOTE f_{SCLK} is the frequency of internal operating clock, SCLK.

T3CR2 (Timer 3 Mode Control Register 2) **$C9_H$**

7	6	5	4	3	2	1	0
T3REQ	T2REQ	T1REQ	T0REQ	-	-	T3_PE	POL3
R	R	R	R	-	-	R/W	R/W

Initial value : 00_H

T3REQ Timer 3 Interrupt Flag NOTE

0 Timer 3 interrupt not occurred
1 Timer 3 interrupt occurred

T2REQ Timer 2 Interrupt Flag [NOTE]
0 Timer 2 interrupt not occurred
1 Timer 2 interrupt occurred

T1REQ Timer 1 Interrupt Flag [NOTE]
0 Timer 1 interrupt not occurred
1 Timer 1 interrupt occurred

T0REQ Timer 0 Interrupt Flag [NOTE]
0 Timer 0 interrupt not occurred
1 Timer 0 interrupt occurred

POL3 Selects polarity of PWM
0 PWM waveform is low for duty value
1 PWM waveform is high for duty value

T3_PE Controls whether to output Timer 3 output or not through I/O pin.
0 Timer 3 output does not come out through I/O pin
1 Timer 3 output overrides the normal port functionality of I/O pin

[NOTE] Writing '0' to this bit position clears interrupt flag of each timer.

T3L (Timer 3 Counter Low, Read Case) — CB_H

7	6	5	4	3	2	1	0
T3L7	T3L6	T3L5	T3L4	T3L3	T3L2	T3L1	T3L0
R	R	R	R	R	R	R	R

Initial value : 00_H

T3L[7:0] T3 Counter Low

CDR3L (Capture Data Register 3 Low, Read Case) — CB_H

7	6	5	4	3	2	1	0
CDR3L7	CDR3L6	CDR3L5	CDR3L4	CDR3L3	CDR3L2	CDR3L1	CDR3L0
R	R	R	R	R	R	R	R

Initial value : 00_H

CDR3L[7:0] T3 Capture Data Low

PWM3DRL (PWM3 Duty Register Low, Write Case) — CB_H

7	6	5	4	3	2	1	0
T3PDL7	T3PDL6	T3PDL5	T3PDL4	T3PDL3	T3PDL2	T3PDL1	T3PDL0
W	W	W	W	W	W	W	W

Initial value : 00_H

T3PDL[7:0] PWM3 Duty Low
[NOTE] Writing is effective only when PWM3E = 1 and T3ST = 0.

T3H (Timer 3 Counter High, Read Case) — CC_H

7	6	5	4	3	2	1	0
T3H7	T3H6	T3H5	T3H4	T3H3	T3H2	T3H1	T3H0

R	R	R	R	R	R	R	R

Initial value : 00_H

T3H[7:0] T3 Counter High

CDR3H (Capture Data Register 3 High, Read Case) CC_H

7	6	5	4	3	2	1	0
CDR3H7	CDR3H6	CDR3H5	CDR3H4	CDR3H3	CDR3H2	CDR3H1	CDR3H0
R	R	R	R	R	R	R	R

Initial value : 00_H

CDR3H[7:0] T3 Capture Data High

PWM3DRH (PWM3 Duty Register High, Write Case) CC_H

7	6	5	4	3	2	1	0
T3PDH7	T3PDH6	T3PDH5	T3PDH4	T3PDH3	T3PDH2	T3PDH1	T3PDH0
W	W	W	W	W	W	W	W

Initial value : 00_H

T3PDH[7:0] PWM3 Duty High
NOTE Writing is effective only when PWM3E = 1 and T3ST = 0.

T3DRL (Timer 3 Data Register Low, Write Case) CD_H

7	6	5	4	3	2	1	0
T3DRL7	T3DRL6	T3DRL5	T3DRL4	T3DRL3	T3DRL2	T3DRL1	T3DRL0
W	W	W	W	W	W	W	W

Initial value : FF_H

T3DRL[7:0] T3 Compare Data Low
NOTE Be sure to clear PWM3E in T3CR register before loading this register.

PWM3PRL (PWM3 Period Register Low, Write Case) CD_H

7	6	5	4	3	2	1	0
T3PPL7	T3PPL6	T3PPL5	T3PPL4	T3PPL3	T3PPL2	T3PPL1	T3PPL0
W	W	W	W	W	W	W	W

Initial value : FF_H

T3PPL[7:0] PWM3 Period Low
NOTE Writing is effective only when PWM3E = 1 and T3ST = 0.

T3DRH (Timer 3 Data Register High, Write Case) CE_H

7	6	5	4	3	2	1	0
T3DRH7	T3DRH6	T3DRH5	T3DRH4	T3DRH3	T3DRH2	T3DRH1	T3DRH0
W	W	W	W	W	W	W	W

Initial value : FF_H

T3DRH[7:0] T3 Compare Data High
NOTE Be sure to clear PWM3E in T3CR register before loading this register.

PWM3PRH (PWM3 Period Register High, Write Case) **CE_H**

7	6	5	4	3	2	1	0
P3PPH7	P3PPH6	P3PPH5	P3PPH4	P3PPH3	P3PPH2	P3PPH1	P3PPH0
W	W	W	W	W	W	W	W

Initial value : FF_H

P3PPH[7:0] PWM3 Period High

NOTE Writing is effective only when PWM3E = 1 and T3ST = 0.

11.5 Watch Timer with event capture function (WT)

11.5.1 Overview

The watch timer (WT) has the function for RTC (Real Time Clock) operation. This module consists of the clock source select circuit, timer counter circuit, output select circuit and control registers. To activate watch timer, determine the input clock source, output interval and then set WTEN bit in Watch Timer Mode Register (WTMR). Control bits can be set individually or at a time. To stop or reset WT, clear the WTEN bit in WTMR. To obtain high resolution, the counter of WT is composed of low 14-bit binary counter(=WTIR) and high 7-bit counter(=WT_TMR), that makes the WT counter to become 21-bit wide. The high and low counters are auto-cleared when each counter reaches to their pre-defined data values. The WT Interrupt Interval is determined by writing to WTDRH, WTDR1 and WTDR0 registers. To read each WTDRH, WTDR1 and WTDR0 returns WT_TMR, high 6-bit of WTIR, and low 8-bit of WTIR counter value.

When Watch timer operates in IR capture mode, the WT is a simple 14-bit up counter and the counter is auto-cleared by the rising edge of an incoming event source. In this mode of operation, the 7-bit counter WT_TMR stops operation and the WTIR counter value is captured into WTCR0, WTCR1, WTCR2 registers on detecting the rising or falling edge of input carrier signal. The capture sequence is decided according to the setting of SINGLE and PHASE bits in IRCC1 register.

Note that the divide ratio of input clock applies in different manner whether WT is in normal WT mode or in IR capture mode.

11.5.2 Block Diagram

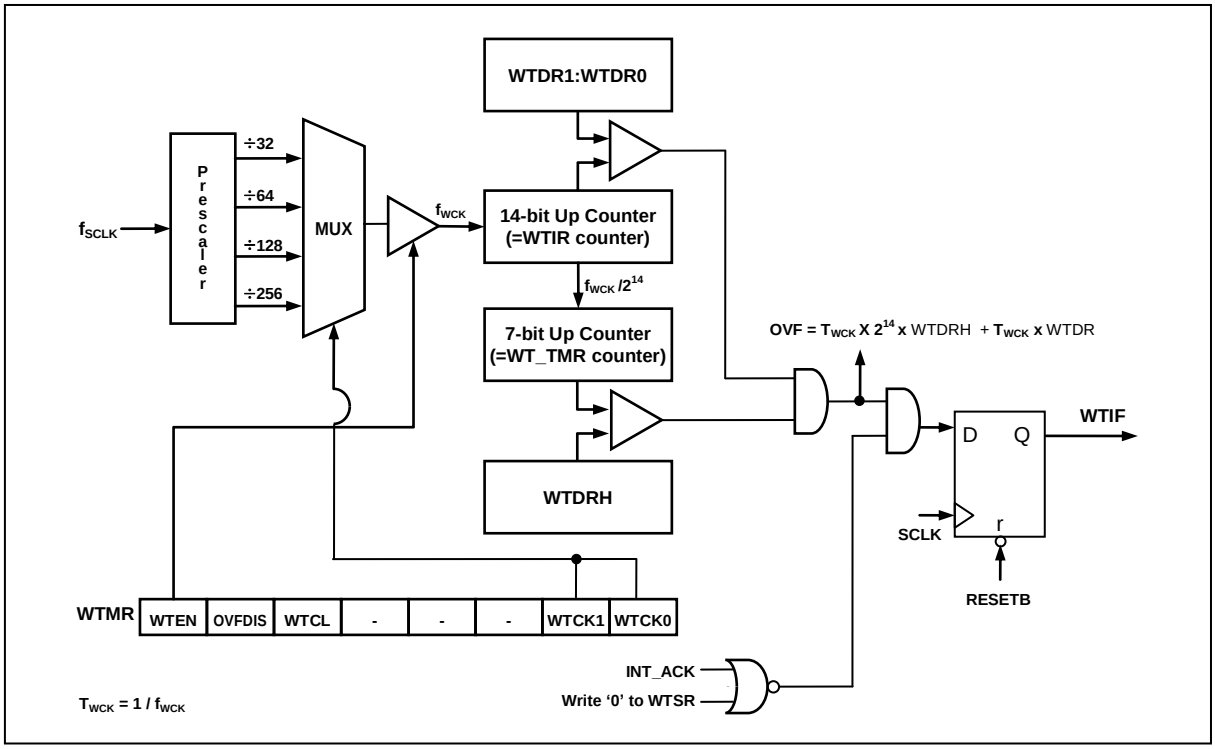


Figure 11-24 Block Diagram of Watch Timer in Normal mode

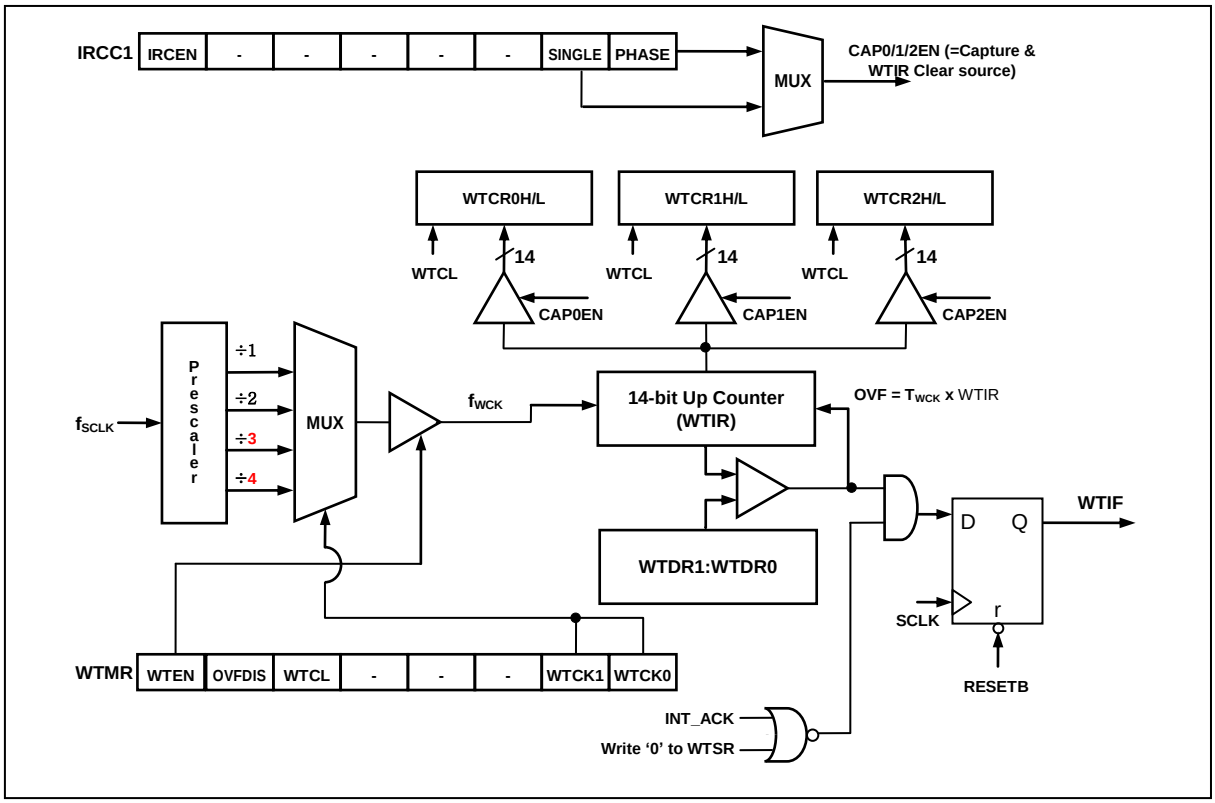


Figure 11-25 Block Diagram of Watch Timer in IR capture mode

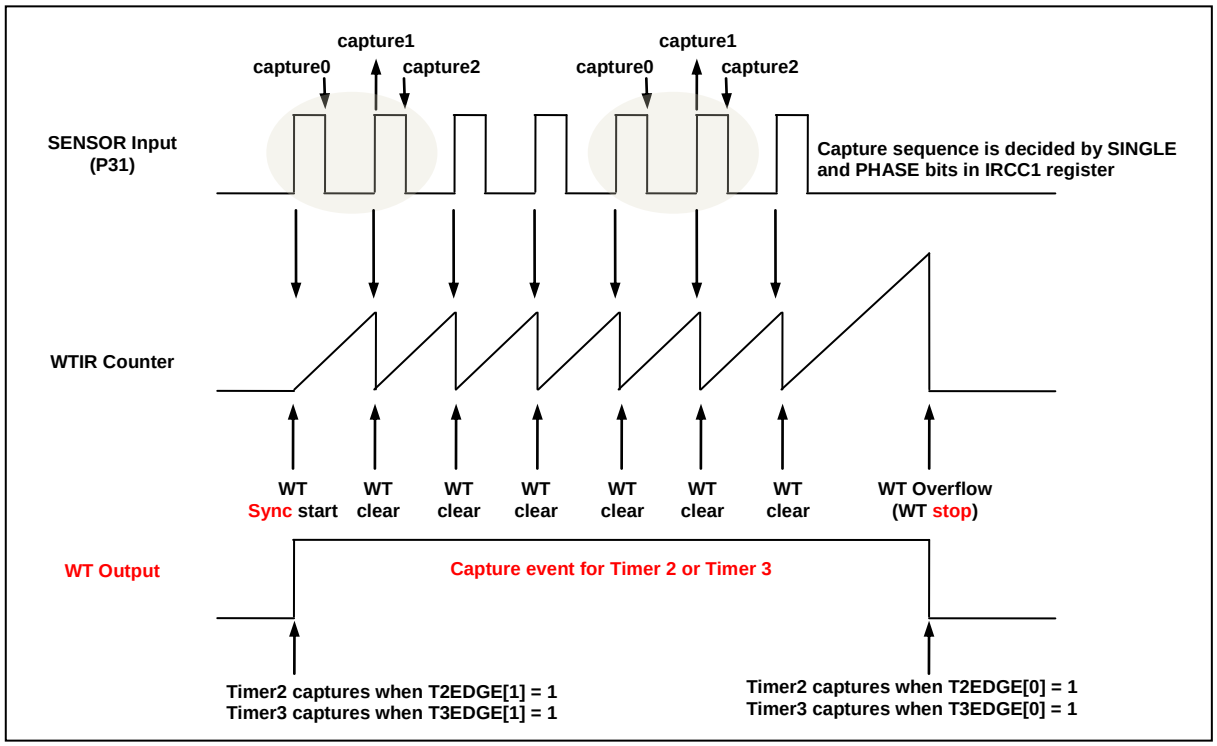


Figure 11-26 Timing Diagram of Watch Timer in IR capture mode

11.5.3 Register Map

Name	Address	Dir	Default	Description
WTMR	$D1_H$	R/W	00_H	Watch Timer Mode Register
WTDR1	$D4_H$	W	$3F_H$	Watch Timer Data Register 1
WTDR0	$D5_H$	W	FF_H	Watch Timer Data Register 0
WTSR	$D9_H$	R	00_H	Watch Timer Status Register
WTDRH	DC_H	W	$7F_H$	Watch Timer Data Register High
WTCR0H	$F1_H$	R	$3F_H$	Watch Timer Capture Register0 High
WTCR0L	$F2_H$	R	FF_H	Watch Timer Capture Register0 Low
WTCR1H	$F3_H$	R	$3F_H$	Watch Timer Capture Register1 High
WTCR1L	$F4_H$	R	FF_H	Watch Timer Capture Register1 Low
WTCR2H	$F5_H$	R	$3F_H$	Watch Timer Capture Register2 High
WTCR2L	$F6_H$	R	FF_H	Watch Timer Capture Register2 Low

Table 11-10 Register Map of Watch Timer

11.5.4 Register Description

WTMR (Watch Timer Mode Register) **$D1_H$**

7	6	5	4	3	2	1	0
WTEN	OVFDIS	WTCL	-	-	-	WTCK1	WTCK0
R/W	R/W	R/W	-	-	-	R/W	R/W

Initial value : 00_H

WTEN Enable Watch Timer
- 0 Disable WT
- 1 Enable WT

OVFDIS Control auto clear function of WT when counters overflow. NOTE1
- 0 Auto clear counters when overflow
- 1 Overflow event is ignored

WTCL Clear counter of Watch Timer
- 0 No operation (Free Run mode)
- 1 Clear WT counter (Auto-clear after 1 cycle)

WTCK[1:0] Select clock source of WT (=f_{WCK}) NOTE2

WTCK1	WTCK0	Watch Timer mode	IR Capture mode
0	0	$f_{SCLK}/32$	f_{SCLK}
0	1	$f_{SCLK}/64$	$f_{SCLK}/2$
1	0	$f_{SCLK}/128$	$f_{SCLK}/3$
1	1	$f_{SCLK}/256$	$f_{SCLK}/4$

NOTE1 The overflow means WT_TMR equals to WTDRH, and WTIR equals to WTCR1/0 when WT is in Watch Timer mode. In IR capture mode, the overflow condition occurs when WTIR equals to WTCR1/0.

NOTE2 f_{SCLK} is the frequency of system clock, SCLK.

f_{WCK} is the frequency of WTIR counter clock

WTDR1 (Watch Timer Data Register 1) **$D4_H$**

7	6	5	4	3	2	1	0

-	-	WTDR13	WTDR12	WTDR11	WTDR10	WTDR9	WTDR8
-	-	W	W	W	W	W	W

Initial value : $3F_H$

WTDR[13:8] Select WT overflow period. Reading this register returns the high 8-bit WTIR counter value.
WT Interrupt Interval = (Twck x 2^14) x (7-bit WTDRH) + (Twck x 14-bit WTDR)

WTDR0 (Watch Timer Data Register 0) $D5_H$

7	6	5	4	3	2	1	0
WTDR7	WTDR6	WTDR5	WTDR4	WTDR3	WTDR2	WTDR1	WTDR0
W	W	W	W	W	W	W	W

Initial value : FF_H

WTDR[7:0] Select WT overflow period. Reading this register returns the low 8-bit WTIR counter value.

WTSR (Watch Timer Status Register) $D9_H$

7	6	5	4	3	2	1	0
IRI	-	-	-	-	-	-	WTIFR
R	-	-	-	-	-	-	R

Initial value : 00_H

IRI IRI status (IRAMP output or Port input)

0 IRI is '0'

1 IRI is '1'

WTIFR Interrupt flag of WT. This flag bit is cleared when the interrupt is serviced or by writing '0' to this bit field.

0 No WT interrupt is generated

1 WT interrupt occurred

WTDRH (Watch Timer Data Register High) DC_H

7	6	5	4	3	2	1	0
-	WTDRH6	WTDRH5	WTDRH4	WTDRH3	WTDRH2	WTDRH1	WTDRH0
-	W	W	W	W	W	W	W

Initial value : $7F_H$

WTDRH[6:0] Select WT overflow period. Reading this register returns WT_TMR counter value, the high 7-bit counter.

WTCR0H (Watch Timer Capture Register 0 High) $F1_H$

7	6	5	4	3	2	1	0
-	-	WTCR013	WTCR012	WTCR011	WTCR010	WTCR009	WTCR008
-	-	R	R	R	R	R	R

Initial value : $3F_H$

WTCR0[13:8] When WT is in IR capture mode, the high 6-bit of WTIR counter is captured to this register at the first falling edge (when PHASE bit is '0') or first rising edge (when PHASE bit is '1') of input carrier signal. This register is initialized by setting WTCL bit in WTMR.

WTCR0L (Watch Timer Capture Register 0 Low) $F2_H$

7	6	5	4	3	2	1	0
WTCR007	WTCR006	WTCR005	WTCR004	WTCR003	WTCR002	WTCR001	WTCR000
R	R	R	R	R	R	R	R

Initial value : FF_H

WTCR0[7:0] When WT is in IR capture mode, the low 8-bit of WTIR counter is captured to this register at the first falling edge (when PHASE bit is '0') or first rising edge (when PHASE bit is '1') of input carrier signal. This register is initialized by setting WTCL bit in WTMR.

WTCR1H (Watch Timer Capture Register 1 High) $F3_H$

7	6	5	4	3	2	1	0
-	-	WTCR113	WTCR112	WTCR111	WTCR110	WTCR109	WTCR108
-	-	R	R	R	R	R	R

Initial value : $3F_H$

WTCR1[13:8] When WT is in IR capture mode, the high 6-bit of WTIR counter is captured to this register at the second rising edge (when PHASE bit is '0') or first falling edge (when PHASE bit is '1') of input carrier signal. This register is initialized by setting WTCL bit in WTMR.

WTCR1L (Watch Timer Capture Register 1 Low) $F4_H$

7	6	5	4	3	2	1	0
WTCR107	WTCR106	WTCR105	WTCR104	WTCR103	WTCR102	WTCR101	WTCR100
R	R	R	R	R	R	R	R

Initial value : FF_H

WTCR1[7:0] When WT is in IR capture mode, the low 8-bit of WTIR counter is captured to this register at the second rising edge (when PHASE bit is '0') or first falling edge (when PHASE bit is '1') of input carrier signal. This register is initialized by setting WTCL bit in WTMR.

WTCR2H (Watch Timer Capture Register 2 High) $F5_H$

7	6	5	4	3	2	1	0
-	-	WTCR213	WTCR212	WTCR211	WTCR210	WTCR209	WTCR208
-	-	R	R	R	R	R	R

Initial value : $3F_H$

WTCR2[13:8] When WT is in IR capture mode, the high 6-bit of WTIR counter is captured to this register at the second falling edge (when PHASE bit is '0') or second rising edge (when PHASE bit is '1') of input carrier signal. This register is initialized by setting WTCL bit in WTMR.

WTCR2L (Watch Timer Capture Register 2 Low) $F6_H$

7	6	5	4	3	2	1	0
WTCR207	WTCR206	WTCR205	WTCR204	WTCR203	WTCR202	WTCR201	WTCR200
R	R	R	R	R	R	R	R

Initial value : FF_H

WTCR2[7:0]	When WT is in IR capture mode, the low 8-bit of WTIR counter is captured to this register at the second falling edge (when PHASE bit is '0') or second rising edge (when PHASE bit is '1') of input carrier signal. This register is initialized by setting WTCL bit in WTMR.

The WT interrupt is requested only when overflow condition occurs. That is when WT is in IR capture mode, the interrupt is not issued even when capture event is generated.

11.6 IR Capture Control (IRCC)

11.6.1 Overview

MC96FR332A has an IR capture module which receives and captures the incoming digital IR signal to detect the IR carrier frequency and count the carrier number. With this module, the Watch Timer and Timer 2 can be configured to operate in IR capture mode by setting IRCEN bit in IRCC1 register. Also Timer 3 can support IR capture feature. Both Timer 2 and Timer 3 can detect the envelop of incoming carrier or count the number of input carrier signal according to the setting of IRCC2 register.

11.6.2 Block Diagram

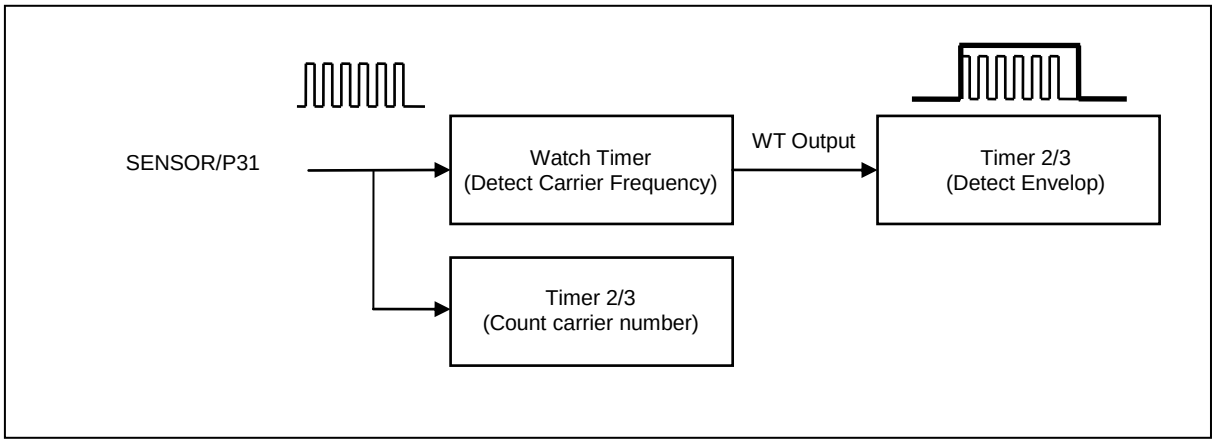


Figure 11-27 Block Diagram of IR Capture function

This IRCC module is closely related to WT module, so refer to Figure 11-26 for more information including counting mechanism of WTIR counter.

11.6.3 Register Map

Name	Address	Dir	Default	Description
IRCC1	DE_H	R/W	00_H	IR Capture Control Register 1
IRCC2	DF_H	R/W	00_H	IR Capture Control Register 2

Table 11-11 Register Map of IR Capture Control module

11.6.4 Register Description

IRCC1 (IR Capture Register 1) **DE_H**

7	6	5	4	3	2	1	0
IRCEN	IRIIF	IREDGE1	IREDGE0	-	IRPOL	SINGLE	PHASE
R/W	R	R/W	R/W	-	R/W	R/W	R/W

Initial value : 00_H

IRCEN Control operation mode of WT, T2 and T3

0 IR capture mode is disabled, normal timer function

1 IR capture mode is enabled (WT, T2 and T3 modules are under control of this bit)

IRIIF Interrupt flag of IRI input. This flag is cleared by writing ‘0’ to this bit field or interrupt is serviced.

0 No IRI input is generated

1 IRI interrupt is generated on the condition by IREDGE[1:0] bits

IREDGE[1:0] Select IRI interrupt triggering condition.

00 IRI interrupt is disabled

01 Interrupt is triggered on falling edge of IRI input

10 Interrupt is triggered on rising edge of IRI input

11 Interrupt is triggered on both edge of IRI input

IRPOL Select the polarity of WT input source.

0 The inverted signal of SENSOR/P36 input becomes the input source of WT.

1 The SENSOR/P36 input becomes the input source of WT.

SINGLE Select carrier capture numbers. Used with the PHASE bit.

0 Capture continuously until WTIR overflows(=WTIR reaches to pre-defined value, WTDR1 and WTDR0)

1 Capture first 3 edges of carrier signal

PHASE Select carrier capture sequence. Used with the SINGLE bit.

0 Capture sequence is 1st Falling→Rising→Falling edge

1 Capture sequence is 1st Rising→Falling→Rising edge

IRCC2 (IR Capture Register 2) DF_H

7	6	5	4	3	2	1	0
T3IR	T2IR	-	-	T3EDGE1	T3EDGE0	T2EDGE1	T2EDGE0
R/W	R/W	-	-	R/W	R/W	R/W	R/W

Initial value : 00_H

T3IR Make T3 to calculate the number of incoming carrier signal if CAP3 bit in T3CR bit is not '1'.

0 Timer 3 is in normal operation

1 Timer 3 calculates the number of incoming carrier signals.

T2IR Make T3 to calculate the number of incoming carrier signal if CAP3 bit in T3CR bit is not '1'.

0 Timer 2 is in normal operation

1 Timer 2 calculates the number of incoming carrier signals.

T3EDGE[1:0] Select capture edge when T3 is used for envelop detection of incoming carrier signal. These bits should be cleared to '00' when T3 operates in normal capture mode, or the WT output becomes capture source of Timer 3. The T3IR bit should be cleared to '0' also.

00 No capture

01 Falling edge

10 Rising edge

11 Both edge

T2EDGE[1:0] Select capture edge when T2 is used for envelop detection of incoming carrier signal. These bits should be cleared to '00' when T2 operates in normal capture mode, or the WT output becomes capture source of Timer 2. The T2IR bit should be cleared to '0' also.

00 No capture

01 Falling edge

10 Rising edge

11 Both edge

The next table shows register setting for Timer 2 and 3 for IR capture features.

IRCEN	CAP2(3)	T2(3)IR	T2(3)EDGE[1:0]	Timer 2(3) Operating Mode
0	0	0	XX	Normal 16-bit Counter
0	1	X	00	Normal 16-bit Capture
1	1	X	01, 10, 11	IR Capture (Envelop detect)
1	0	1	XX	Count IR Carrier

Table 11-12 Operating modes of Timer 2(3)

11.7 Carrier Generator

11.7.1 Overview

MC96FR332A has a specific module to generate carrier signal for remote control application. The internal carrier(CRF) signal is AND-ed with register value(RODR) and outputs through REMOUT port. The frequency and duty ratio of carrier signal is controlled by two 8-bit registers, CFRH and CFRL. Carrier signal can be on/off at previous stage of REMOUT port by the CEN bit in RMR register. When CEN=1, the remote out signal is generated by AND-ing carrier signal with RODR value. When CEN=0, the 8-bit counter for carrier generation(=CRC) stops and the remote out signal comes directly from RODR value. The RODR register is updated by ROB register when the 16-bit counter for data pulse generation(=RDC) reaches to RDRH or RDRL[NOTE]. In this case, the RDPE bit in RMR should be '1'. At each match event, an interrupt can be issued. The RODR register can also be altered by writing to this register. In this case, the RDPE bit is to be cleared to '0'. The base clock for RDC and CRC is system clock, SCLK or its divided clock. Note that the output clock of main oscillator, XINCLK, may differ from SCLK.

NOTE The concatenated RDRH and RDRL composes RDR value. And the RDR register is loaded with RDB(= concatenated RDBH and RDBL) when interrupt is generated. This is like the relationship between ROB and RODR.

11.7.2 Block Diagram

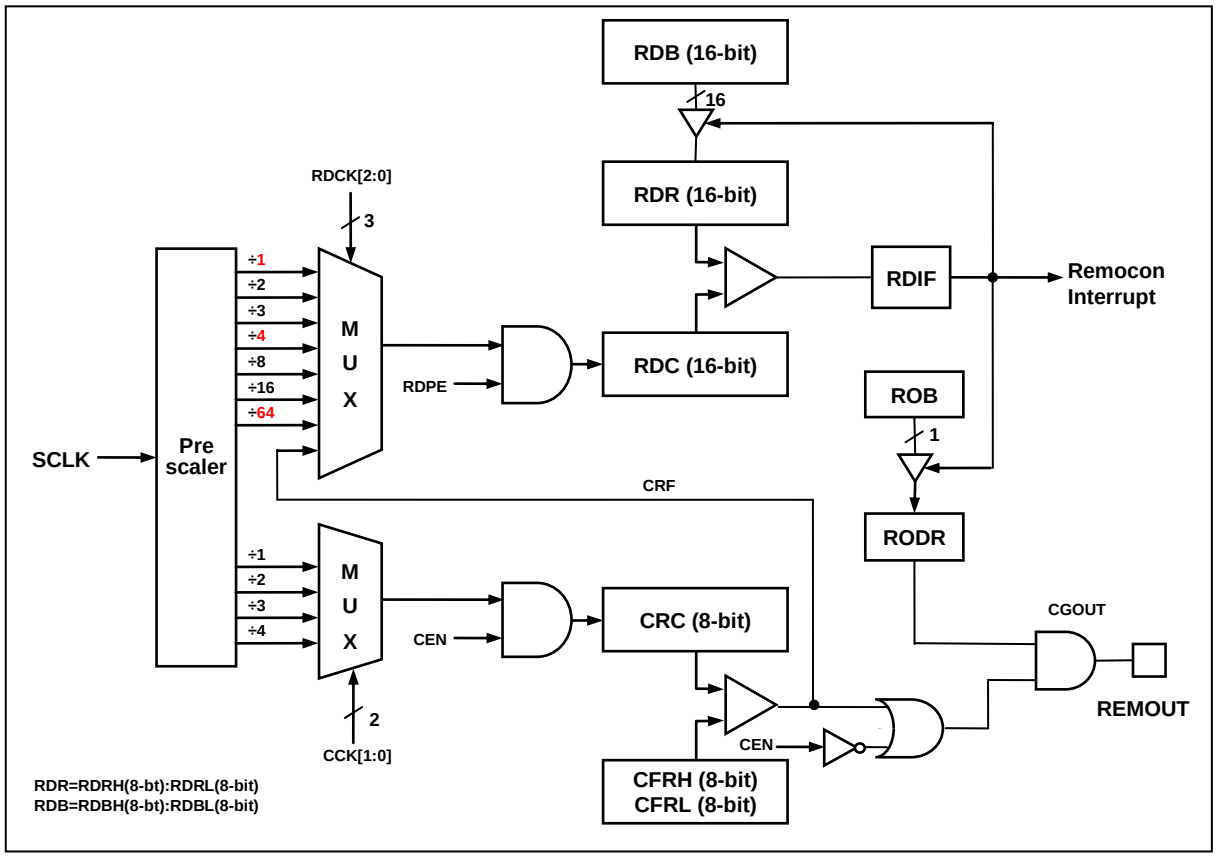


Figure 11-28 Block Diagram of Carrier Generator

11.7.3 Register Map

Name	Address	Dir	Default	Description
RMR	$E8_H$	R/W	00_H	Remocon Mode Register
RDCH	BA_H	R	00_H	Remocon Data Counter High
CFRH	BB_H	R/W	FF_H	Carrier Frequency Register High
CFRL	BC_H	R/W	FF_H	Carrier Frequency Register Low
RDCL	BD_H	R	00_H	Remocon Data Counter Low
RODR	BE_H	R/W	00_H	Remocon Output Data Register
ROB	BF_H	R/W	00_H	Remocon Output Buffer
RDBH	$C2_H$	R/W	FF_H	Remocon Data Buffer High
RDBL	$C3_H$	R/W	FF_H	Remocon Data Buffer Low
RDRH	$C4_H$	R/W	FF_H	Remocon Data Register High
RDRL	$C5_H$	R/W	FF_H	Remocon Data Register Low

Table 11-13 Register Map of Carrier Generator

11.7.4 Register Description

RMR (Remocon Mode Register) **$E8_H$**

7	6	5	4	3	2	1	0
RDIF	CEN	CCK1	CCK0	RDPE	RDCK2	RDCK1	RDCK0
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

RDIF Interrupt flag. This flag is cleared when the interrupt is serviced, RDPE bit is cleared or software writes '0' to this bit position. This flag has nothing to do with CEN bit. Writing '1' to this bit sets the interrupt flag.

RDCK[2:0] Selects clock source for 6-bit RDC counter. These bits are effective only when RDPE=1. NOTE

RDCK2	RDCK1	RDCK0	
0	0	0	$f_{SCLK}/1$
0	0	1	$f_{SCLK}/2$
0	1	0	$f_{SCLK}/3$
0	1	1	$f_{SCLK}/4$
1	0	0	$f_{SCLK}/8$
1	0	1	$f_{SCLK}/16$
1	1	0	$f_{SCLK}/64$
1	1	1	Carrier Signal(=CRF)

RDPE Remote Data Pulse Enable. Setting this bit enables RDC counter. Interrupt can only be issued when this bit is set.

0 Disable RDC counter

1 Enable RDC counter

CCK[1:0] Select clock source for 8-bit CRC counter. These bits are effective only when CEN=1. NOTE

0	0	$f_{SCLK}/1$
0	1	$f_{SCLK}/2$
1	0	$f_{SCLK}/3$
1	1	$f_{SCLK}/4$

CEN Carrier Frequency Enable. This bit enables CRC counter.

0 Carrier Frequency is not generated.

1 Carrier Frequency is generated and goes out through the REMOUT port with RODR value and-ed.

NOTE f_{SCLK} is the frequency of system clock, SCLK.

CFRH (Carrier Frequency Register High) BB_H

7	6	5	4	3	2	1	0
CFH7	CFH6	CFH5	CFH4	CFH3	CFH2	CFH1	CFH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

CFH[7:0] Carrier Frequency High

Carrier High Interval = CFH[7:0] X T_{CR_CLK}

T_{CR_CLK} is the period of clock source for CRC counter selected by CCK[1:0].

CFRL (Carrier Frequency Register Low) BC_H

7	6	5	4	3	2	1	0
CFL7	CFL6	CFL5	CFL4	CFL3	CFL2	CFL1	CFL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

CFL[7:0] Carrier Frequency Low

Carrier Low Interval = CFL[7:0] X T_{CR_CLK}

T_{CR_CLK} is the period of clock source for CRC counter selected by CCK[1:0].

RDBH (Remocon Data Buffer High) $C2_H$

7	6	5	4	3	2	1	0
RDB15	RDB14	RDB13	RDB12	RDB11	RDB10	RDB9	RDB8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

RDB[15:8] Remote Data High Buffer (Lower byte of RDB)

RDBL (Remocon Data Buffer Low) $C3_H$

7	6	5	4	3	2	1	0
RDB7	RDB6	RDB5	RDB4	RDB3	RDB2	RDB1	RDB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

RDB[7:0] Remote Data Low Buffer (Lower byte of RDB).

The RDB is transferred to RDR when interrupt occurs.

RDRH (Remocon Data Register High) $C4_H$

7	6	5	4	3	2	1	0
RDR15	RDR14	RDR13	RDR12	RDR11	RDR10	RDR9	RDR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

RDR[15:8] Remote Data High (Higher byte of RDH)
Remote Data High Interval = RDR[15:0] X T_{RD_CLK}
T_{RD_CLK} is the period of clock source for RDC counter selected by RDCK[2:0].

RDRL (Remocon Data Register Low) — $C5_H$

7	6	5	4	3	2	1	0
RDR15	RDR14	RDR13	RDR12	RDR11	RDR10	RDR9	RDR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

RDR[7:0] Remote Data Low (Lower byte of RDL)
Remote Data Low Interval = RDR[15:0] X T_{RD_CLK}
T_{RD_CLK} is the period of clock source for RDC counter selected by RDCK[2:0].

RDCH (Remocon Data Counter High) — BA_H

7	6	5	4	3	2	1	0
RDC15	RDC14	RDC13	RDC12	RDC11	RDC10	RDC9	RDC8
R	R	R	R	R	R	R	R

Initial value : 00_H

RDC[15:8] Data Counter Value High

RDCL (Remocon Data Counter Low) — BD_H

7	6	5	4	3	2	1	0
RDC7	RDC6	RDC5	RDC4	RDC3	RDC2	RDC1	RDC0
R	R	R	R	R	R	R	R

Initial value : 00_H

RDC[7:0] Data Counter Value High

RODR (Remocon Output Data Register) — BE_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ROD
-	-	-	-	-	-	-	R/W

Initial value : 00_H

ROD Remote Data Output

ROB (Remocon Output Buffer) — BF_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	ROB
-	-	-	-	-	-	-	R/W

Initial value : 00_H

ROB Remote Data Output Buffer

11.7.5 Carrier Signal and Data Pulse

The Remote Out signal(=CGOUT in Block Diagram) on REMOUT port is generated from carrier signal and RODR value. The carrier signal and RODR value are controlled independently. The CEN bit in RMR register makes the carrier signal on or off. The RODR register is updated by ROB on interrupt or by direct writing to this register. In this way, four kinds of signal muxing is supported using carrier signal and RODR value.

The period and frequency of carrier signal and remote data pulse is calculated by the following equation. The waveform is shown below.

t_H (Length of Carrier Signal's High Phase) = T_{CR_CLK} x CFRH[7:0]

t_L (Length of Carrier Signal's Low Phase) = T_{CR_CLK} x CFRL[7:0]

f_C (Carrier Frequency) = $1/(t_H + t_L)$

t_{DH} (Length of Data Pulse's High Phase) = T_{RD_CLK} x RDRH[15:0]

t_{DL} (Length of Data Pulse's Low Phase) = T_{RD_CLK} x RDRL[15:0]

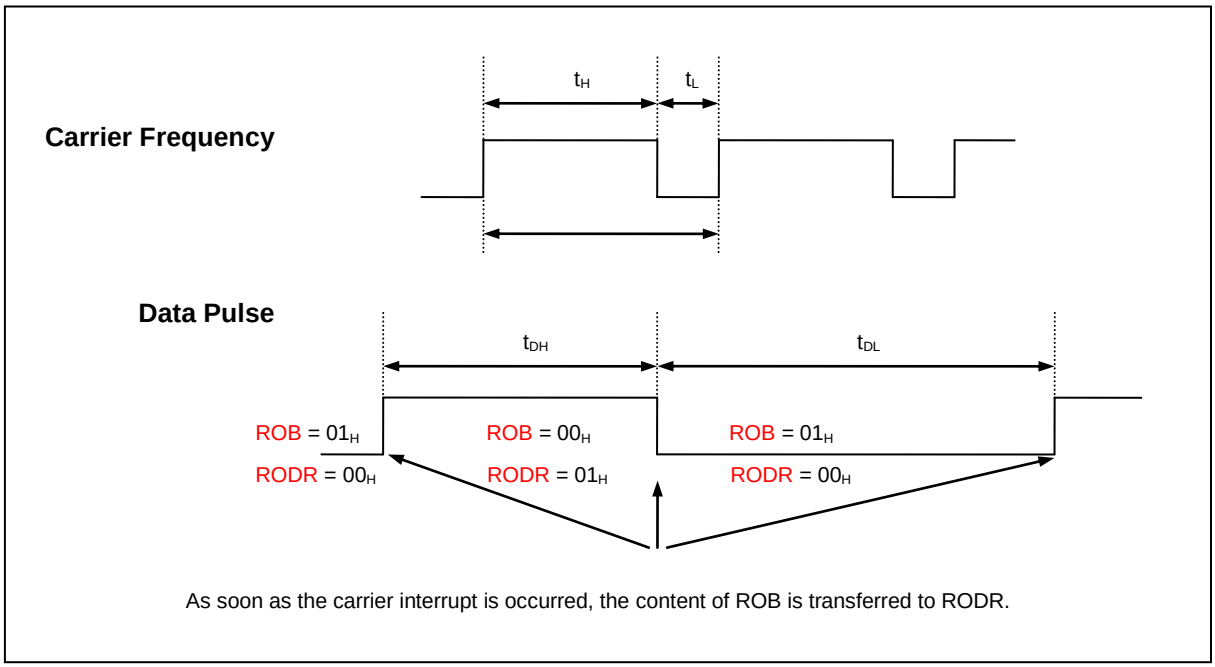


Figure 11-29 Period of Carrier signal and Remote data pulse

11.7.6 Examples of REMOUT control

Three examples of controlling REMOUT port are shown below.

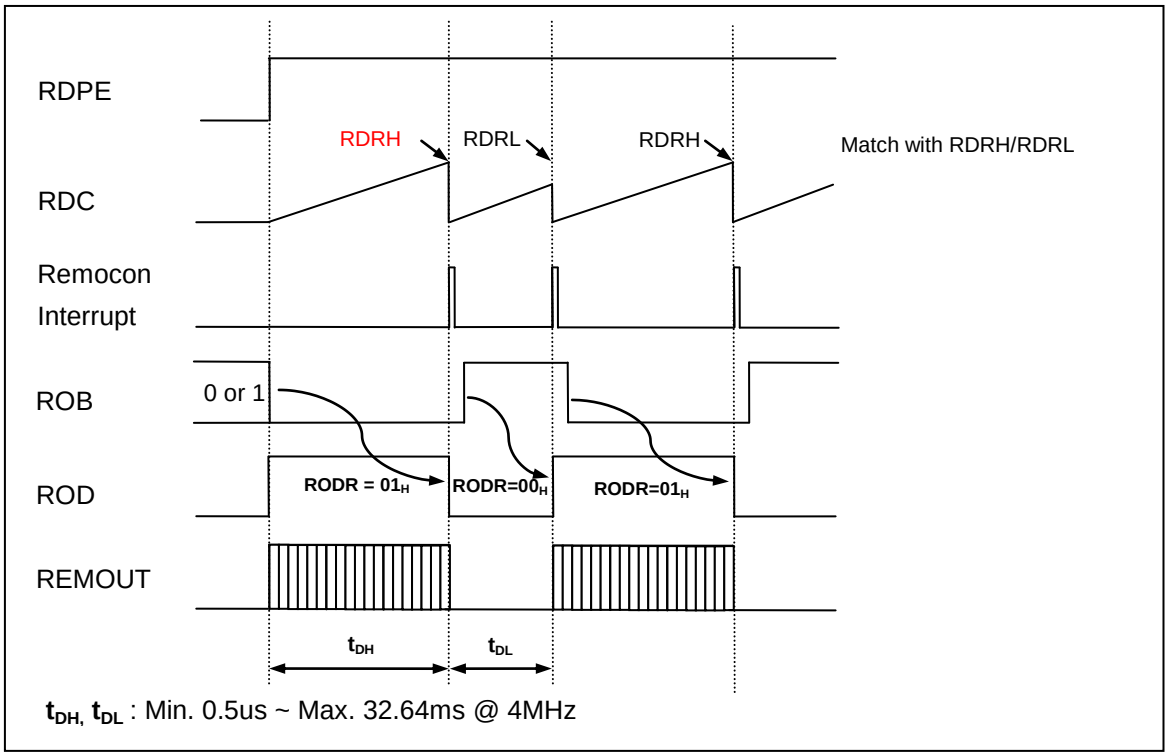


Figure 11-30 REMOUT by CRF & ROB (In case of CEN=1, RDPE=1)

The next figure shows the case carrier signal is off. As can be seen, only RODR value appears on REMOUT port. The difference between previous and below figure is apparent.

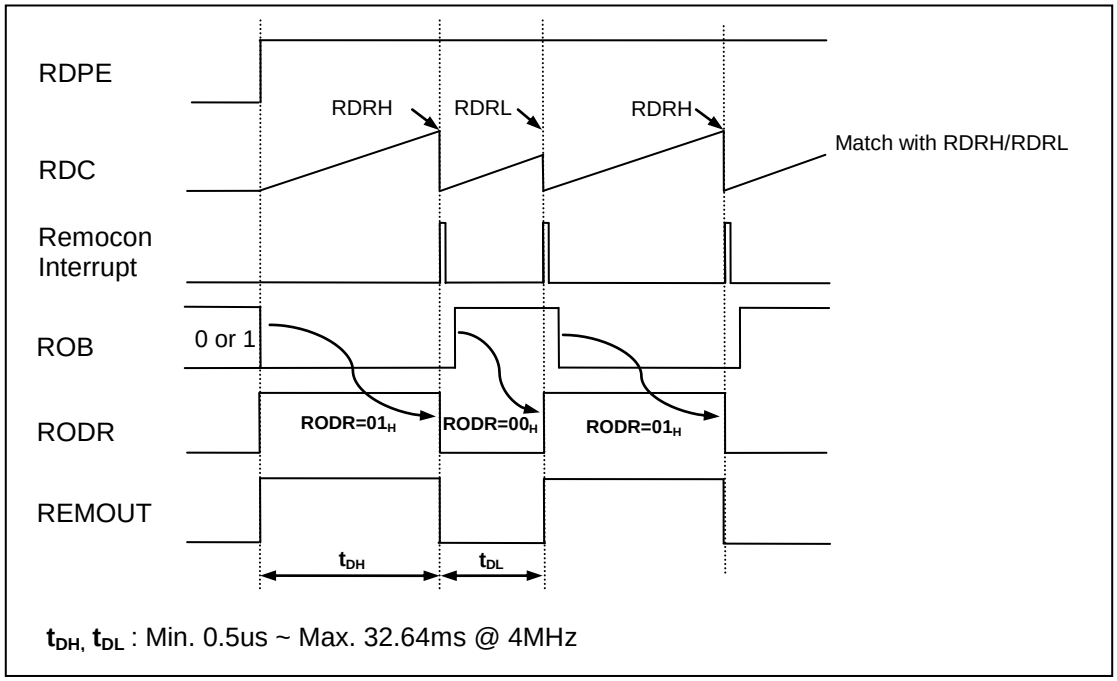


Figure 11-31 REMOUT by ROB only (In case of CEN=0, RDPE=1)

In the last figure, RODR is updated directly by writing to this register when the 16-bit Timer 2, 3 interrupts occur. As shown, the REMOUT waveforms are different according to CEN bit.

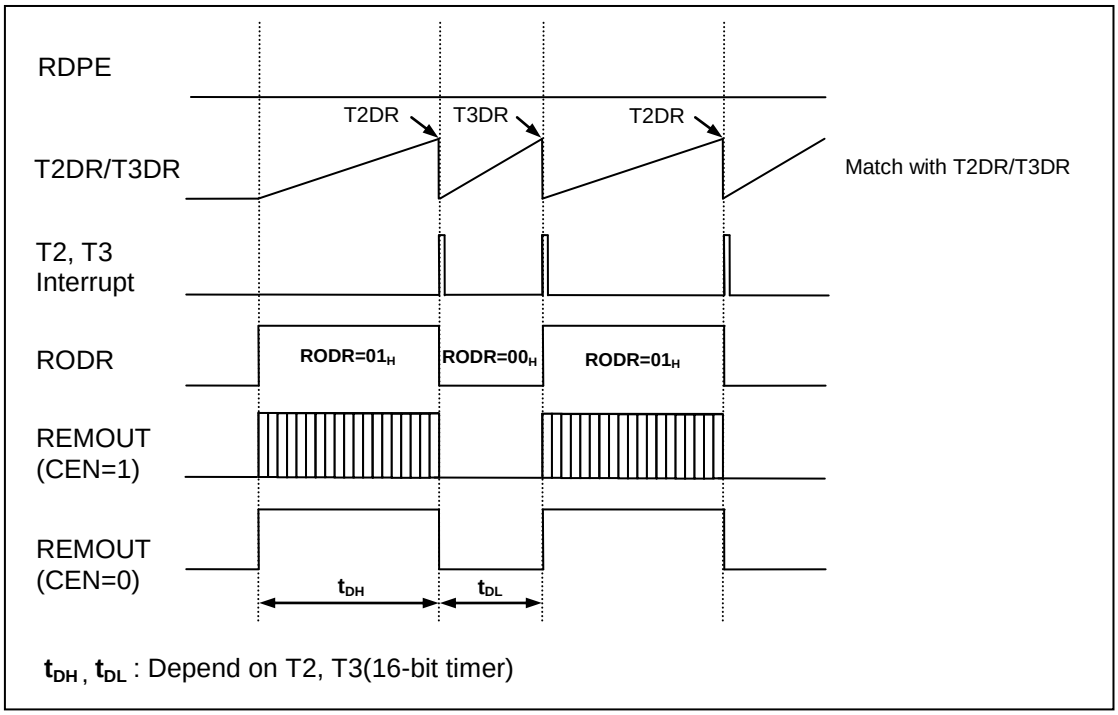


Figure 11-32 REMOUT by RODR

11.7.7 Carrier Generator Interrupt

When RDC counter reaches to RDRH or RDRL register, an interrupt can be requested. As the RDC counter functions when RDPE bit is '1', the interrupt is requested only when RDPE bit is '1'. Even if the interrupt is not required to be serviced by CPU, the flag can be read through RMR register. And this flag is cleared when the interrupt is serviced, RDPE bit is cleared or software writes '0' to the bit position.

11.7.8 Examples of Carrier Signal Selection

The next table shows examples of selecting carrier signal according to CFRH and CFRL registers for two kinds of carrier clocks.

Registers		CR_CLK=PS1		CR_CLK=PS3	
CFRH	CFRL	t_H(us)	t_L(us)	t_H(us)	t_L(us)
00_H	00_H	-	-	-	-
01_H	01_H	0.25	0.25	1.00	1.00
02_H	02_H	0.50	0.50	2.00	2.00
03_H	03_H	0.75	0.75	3.00	3.00
04_H	04_H	1.00	1.00	4.00	4.00
05_H	05_H	1.25	1.25	5.00	5.00
06_H	06_H	1.50	1.50	6.00	6.00
07_H	07_H	1.75	1.75	7.00	7.00
08_H	08_H	2.00	2.00	8.00	8.00
09_H	09_H	2.25	2.25	9.00	9.00
$0A_H$	$0A_H$	2.50	2.50	10.00	10.00
$0B_H$	$0B_H$	2.75	2.75	11.00	11.00
$0C_H$	$0C_H$	3.00	3.00	12.00	12.00
$0D_H$	$0D_H$	3.25	3.25	13.00	13.00

Registers		CR_CLK=PS1		CR_CLK=PS3	
CFRH	CFRL	t_H(us)	t_L(us)	t_H(us)	t_L(us)
20_H	20_H	8.00	8.00	32.00	32.00
21_H	21_H	8.25	8.25	33.00	33.00
22_H	22_H	8.50	8.50	34.00	34.00
23_H	23_H	8.75	8.75	35.00	35.00
24_H	24_H	9.00	9.00	36.00	36.00
25_H	25_H	9.25	9.25	37.00	37.00
26_H	26_H	9.50	9.50	38.00	38.00
27_H	27_H	9.75	9.75	39.00	39.00
28_H	28_H	10.00	10.00	40.00	40.00
29_H	29_H	10.25	10.25	41.00	41.00
$2A_H$	$2A_H$	10.50	10.50	42.00	42.00
$2B_H$	$2B_H$	10.75	10.75	43.00	43.00
$2C_H$	$2C_H$	11.00	11.00	44.00	44.00
$2D_H$	$2D_H$	11.25	11.25	45.00	45.00

$0E_H$	$0E_H$	3.50	3.50	14.00	14.00	$2E_H$	$2E_H$	11.50	11.50	46.00	46.00
$0F_H$	$0F_H$	3.75	3.75	15.00	15.00	$2F_H$	$2F_H$	11.75	11.75	47.00	47.00
10_H	10_H	4.00	4.00	16.00	16.00	30_H	30_H	12.00	12.00	48.00	48.00
11_H	11_H	4.25	4.25	17.00	17.00	31_H	31_H	12.25	12.25	49.00	49.00
12_H	12_H	4.50	4.50	18.00	18.00	32_H	32_H	12.50	12.50	50.00	50.00
13_H	13_H	4.75	4.75	19.00	19.00	33_H	33_H	12.75	12.75	51.00	51.00
14_H	14_H	5.00	5.00	20.00	20.00	34_H	34_H	13.00	13.00	52.00	52.00
15_H	15_H	5.25	5.25	21.00	21.00	35_H	35_H	13.25	13.25	53.00	53.00
16_H	16_H	5.50	5.50	22.00	22.00	36_H	36_H	13.50	13.50	54.00	54.00
17_H	17_H	5.75	5.75	23.00	23.00	37_H	37_H	13.75	13.75	55.00	55.00
18_H	18_H	6.00	6.00	24.00	24.00	38_H	38_H	14.00	14.00	56.00	56.00
19_H	19_H	6.25	6.25	25.00	25.00	39_H	39_H	14.25	14.25	57.00	57.00
$1A_H$	$1A_H$	6.50	6.50	26.00	26.00	$3A_H$	$3A_H$	14.50	14.50	58.00	58.00
$1B_H$	$1B_H$	6.75	6.75	27.00	27.00	$3B_H$	$3B_H$	14.75	14.75	59.00	59.00
$1C_H$	$1C_H$	7.00	7.00	28.00	28.00	$3C_H$	$3C_H$	15.00	15.00	60.00	60.00
$1D_H$	$1D_H$	7.25	7.25	29.00	29.00	$3D_H$	$3D_H$	15.25	15.25	61.00	61.00
$1E_H$	$1E_H$	7.50	7.50	30.00	30.00	$3E_H$	$3E_H$	15.50	15.50	62.00	62.00
$1F_H$	$1F_H$	7.75	7.75	31.00	31.00	$3F_H$	$3F_H$	15.75	15.75	63.00	63.00

Table 11-14 Period of carrier signal according to CFRH/CFRL

In above table, we assume the frequency of main oscillator, f_{XIN} is 8MHz and system clock is not divided from that clock, that is $f_{SCLK} = f_{XIN}$. PSn represents SCLK-divided clock, PS1=SCLK/2 and PS3=SCLK/8. CR_CLK is the clock source for CRC counter.

11.8 Key Scan

11.8.1 Overview

Port 0 and Port 1 can be used as key input sources. If KEY interrupt is enabled, this can be a wake-up source in STOP mode. Usually Port 0(Port 1) is used as output strobe lines, and Port 1(Port 0) is used as key input sources. The key interrupt triggering mode is selected by KITSR register.

11.8.2 Block Diagram

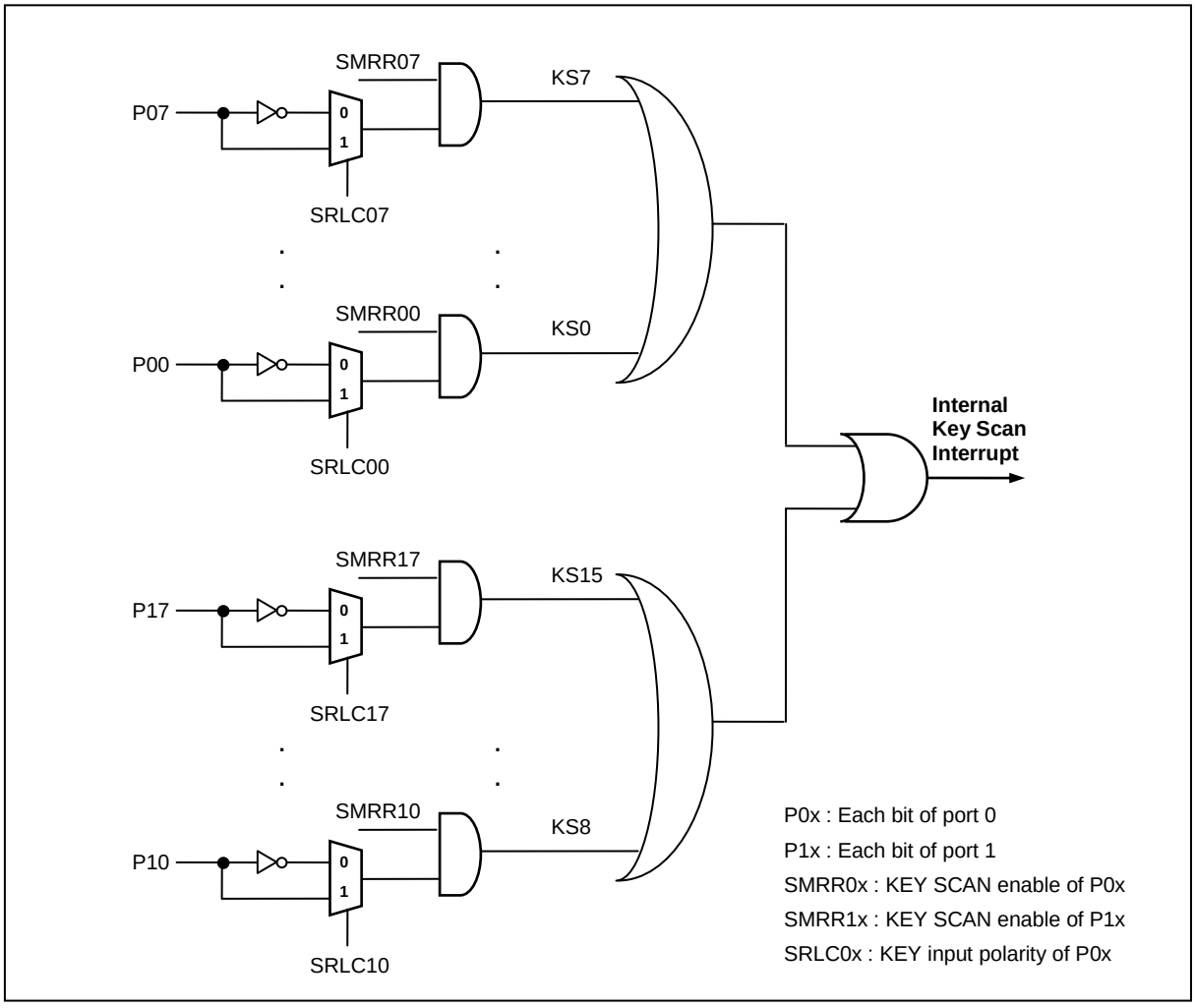


Figure 11-33 Block Diagram of KEYSCAN module

11.8.3 Register Map

Name	Address	Dir	Default	Description
SMRR0	$D2_H$	R/W	00_H	Standby Mode Release Register 0
SMRR1	$D3_H$	R/W	00_H	Standby Mode Release Register 1
SRLC0	$D6_H$	R/W	00_H	Standby Release Level Control Register 0
SRLC1	$D7_H$	R/W	00_H	Standby Release Level Control Register 1
KITSR	$F7_H$	R/W	00_H	Key Interrupt Trigger Selection Register

Table 11-15 Register Map of KEYSCAN module

11.8.4 Register Description

SMRR0 (Standby Mode Release Register 0) **$D2_H$**

7	6	5	4	3	2	1	0
SMRR07	SMRR06	SMRR05	SMRR04	SMRR03	SMRR02	SMRR01	SMRR00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SMRR0[7:0] Enables key function of Port 0 pins.

0 Key function is not used.

1 Key function overrides the normal port functionality of I/O pin.

SMRR1 (Standby Mode Release Register 1) **$D3_H$**

7	6	5	4	3	2	1	0
SMRR17	SMRR16	SMRR15	SMRR14	SMRR13	SMRR12	SMRR11	SMRR10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SMRR1[7:0] Enables key function of Port 1 pins.

0 Key function is not used.

1 Key function overrides the normal port functionality of I/O pin.

SRLC0 (Standby Release Level Control Register 0) **$D6_H$**

7	6	5	4	3	2	1	0
SRLC07	SRLC06	SRLC05	SRLC04	SRLC03	SRLC02	SRLC01	SRLC00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SRLC0[7:0] Selects the trigger level of key input & interrupt when Port 0 is used as key input source.

0 Triggered by a low level

1 Triggered by a high level

SRLC1 (Standby Release Level Control Register 1) $D7_H$

7	6	5	4	3	2	1	0
SRLC17	SRLC16	SRLC15	SRLC14	SRLC13	SRLC12	SRLC11	SRLC10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SRLC1[7:0] Selects the trigger level of key input & interrupt when Port 1 is used as key input source.

0 Triggered by a low level

1 Triggered by a high level

KITSR (Key Interrupt Trigger Select Register) $F7_H$

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	KITSR
-	-	-	-	-	-	-	R/W

Initial value : 00_H

KITSR Selects interrupt trigger mode.

0 Triggered by level detection

1 Triggered by edge detection

11.9 USART0/1

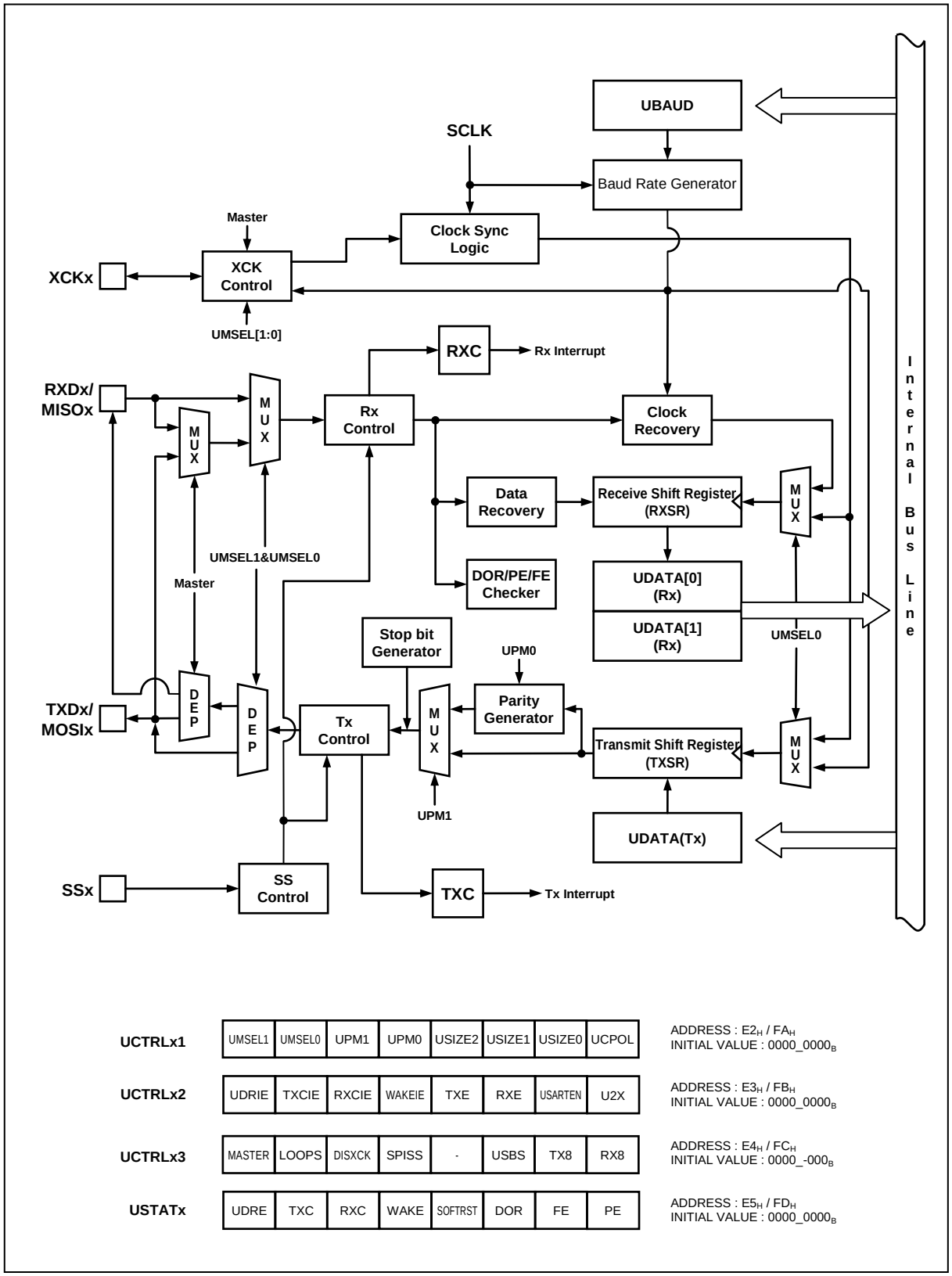
11.9.1 Overview

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device. The main features are listed below.

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous and SPI Operation
- Supports all four SPI Modes of Operation (Mode 0, 1, 2, 3)
- LSB First or MSB First Data Transfer @SPI mode
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5,6,7,8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete
- Double Speed Asynchronous Communication Mode

USART has three main parts of Clock Generator, Transmitter and Receiver. The Clock Generation logic consists of synchronization logic for external clock input used by synchronous or SPI slave operation, and the baud rate generator for asynchronous or master (synchronous or SPI) operation. The Transmitter consists of a single write buffer, a serial shift register, parity generator and control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery unit is used for asynchronous data reception. In addition to the recovery unit, the Receiver includes a parity checker, a shift register, a two level receive FIFO (UDATAn) and control logic. The Receiver supports the same frame formats as the Transmitter and can detect Frame Error, Data OverRun and Parity Errors.

11.9.2 Block Diagram



UCTRLx1	UMSEL1	UMSEL0	UPM1	UPM0	USIZE2	USIZE1	USIZE0	UCPOL
UCTRLx2	UDRIE	TXCIE	RXCIE	WAKEIE	TXE	RXE	USARTEN	U2X
UCTRLx3	MASTER	LOOPS	DISXCK	SPISS	-	USBS	TX8	RX8
USTATx	UDRE	TXC	RXC	WAKE	SOFTRST	DOR	FE	PE

ADDRESS : $E2_H$ / FA_H
INITIAL VALUE : 0000_0000_B

ADDRESS : $E3_H$ / FB_H
INITIAL VALUE : 0000_0000_B

ADDRESS : $E4_H$ / FC_H
INITIAL VALUE : 0000_-000_B

ADDRESS : $E5_H$ / FD_H
INITIAL VALUE : 0000_0000_B

Figure 11-34 The Block Diagram of USART

11.9.3 Clock Generation

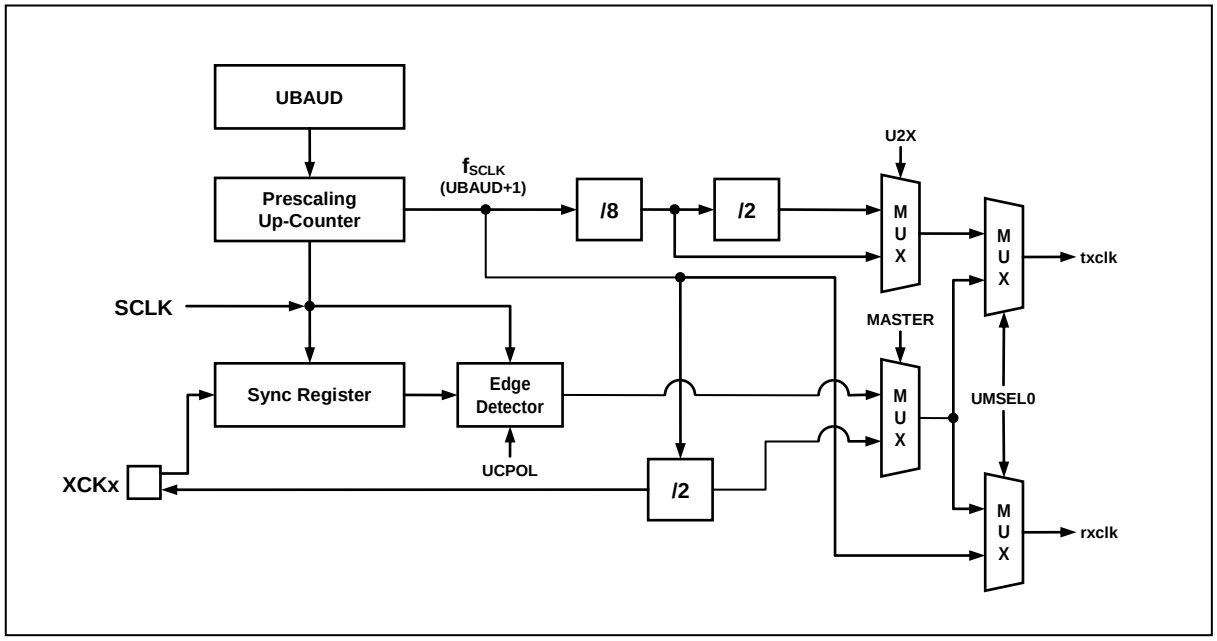


Figure 11-35 The Block Diagram of Clock Generation

The Clock generation logic generates the base clock for the Transmitter and Receiver. The USART supports four modes of clock operation and those are Normal Asynchronous, Double Speed Asynchronous, Master Synchronous and Slave Synchronous. The clock generation scheme for Master SPI and Slave SPI mode is the same as Master Synchronous and Slave Synchronous operation mode. The UMSELn bit in UCTRL1 register selects between asynchronous and synchronous operation. Asynchronous Double Speed mode is controlled by the U2X bit in the UCTRL2 register. The MASTER bit in UCTRL2 register controls whether the clock source is internal (Master mode, output port) or external (Slave mode, input port). The XCK pin is only active when the USART operates in Synchronous or SPI mode.

Table below contains equations for calculating the baud rate (in bps).

Operating Mode	Equation for Calculating Baud Rate
Asynchronous Normal Mode (U2X=0)	$\text{Baud Rate} = \frac{\text{fSCLK}}{16(\text{UBAUD}+1)}$
Asynchronous Double Speed Mode (U2X=1)	$\text{Baud Rate} = \frac{\text{fSCLK}}{8(\text{UBAUD}+1)}$
Synchronous or SPI Master Mode	$\text{Baud Rate} = \frac{\text{fSCLK}}{2(\text{UBAUD}+1)}$

Table 11-16 Equations for Calculation Baud Rate

11.9.4 External Clock (XCK)

External clocking is used by the synchronous or spi slave modes of operation.

External clock input from the XCK pin is sampled by a synchronization logic to remove meta-stability. The output from the synchronization logic must then pass through an edge detector before it can be used by the Transmitter and Receiver. This process introduces a two CPU clock period delay and therefore the maximum frequency of the external XCK pin is limited by the following equation.

$$fXCK = \frac{fSCLK}{4}$$

where fXCK is the frequency of XCK and fSCLK is the frequency of main system clock (SCLK).

11.9.5 Synchronous mode operation

When synchronous or spi mode is used, the XCK pin will be used as either clock input (slave) or clock output (master). The dependency between the clock edges and data sampling or data change is the same. The basic principle is that data input on RXD (MISO in spi mode) pin is sampled at the opposite XCK clock edge of the edge in the data output on TXD (MOSI in spi mode) pin is changed.

The UCPOL bit in UCTRL1 register selects which XCK clock edge is used for data sampling and which is used for data change. As shown in the figure below, when UCPOL is zero the data will be changed at rising XCK edge and sampled at falling XCK edge.

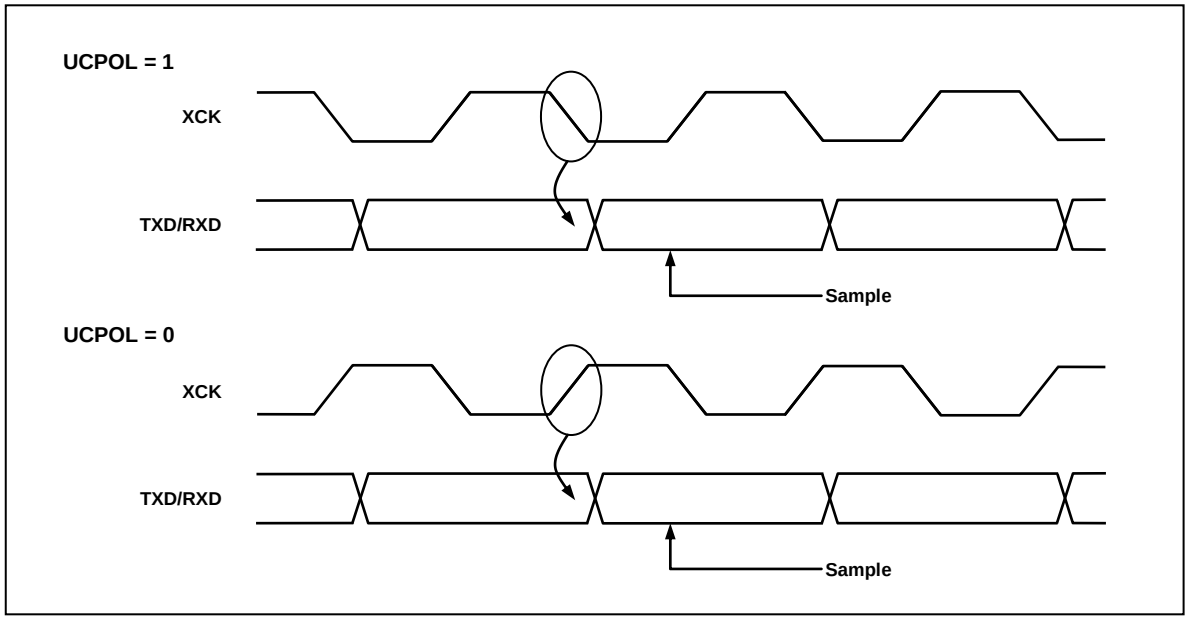


Figure 11-36 Synchronous Mode XCKn Timing.

11.9.6 Data format

A serial frame is defined to be one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error checking.

The USART supports all 30 combinations of the following as valid frame formats.

- 1 start bit
- 5, 6, 7, 8 or 9 data bits
- no, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by the least significant data bit (LSB). Then the next data bits, up to a total of nine, are succeeding, ending with the most significant bit (MSB). If enabled the parity bit is inserted after the data bits, before the stop bits. A high to low transition on data pin is considered as start bit. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle state. The idle means high state of data pin. The next figure shows the possible combinations of the frame formats. Bits inside brackets are optional.

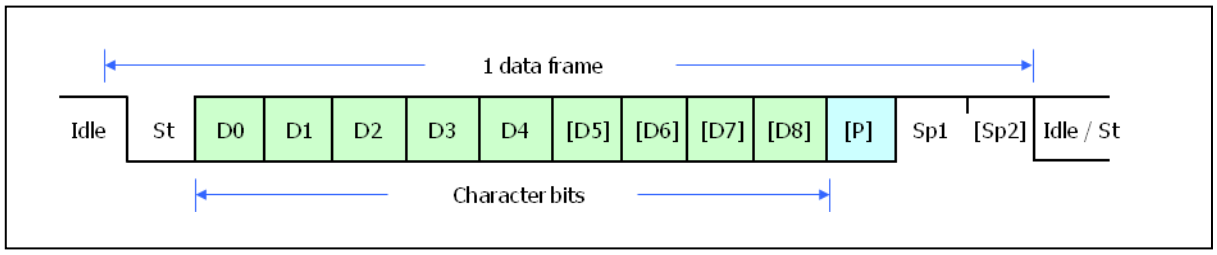


Figure 11-37 frame format

1 data frame consists of the following bits

- Idle No communication on communication line (TXD/RXD)
- St Start bit (Low)
- Dn Data bits (0~8)
- Parity bit ---------- Even parity, Odd parity, No parity
- Stop bit(s) ---------- 1 bit or 2 bits

The frame format used by the USART is set by the USIZE[2:0], UPM[1:0] and USBS bits in UCTRL1 register. The Transmitter and Receiver use the same setting.

11.9.7 Parity bit

The parity bit is calculated by doing an exclusive-or of all the data bits. If odd parity is used, the result of the exclusive-or is inverted. The parity bit is located between the MSB and first stop bit of a serial frame.

$P_{even} = D_{n-1} \wedge \ldots \wedge D_3 \wedge D_2 \wedge D_1 \wedge D_0 \wedge 0$

$P_{odd} = D_{n-1} \wedge \ldots \wedge D_3 \wedge D_2 \wedge D_1 \wedge D_0 \wedge 1$

P_{even} : Parity bit using even parity

P_{odd} : Parity bit using odd parity

D_n : Data bit n of the character

11.9.8 USART Transmitter

The USART Transmitter is enabled by setting the TXE bit in UCTRL1 register. When the Transmitter is enabled, the normal port operation of the TXD(=MOSI) pin is overridden by the serial output pin of USART. The baud-rate, operation mode and frame format must be setup once before doing any transmissions. If synchronous or spi operation is used, the clock on the XCK pin will be overridden and used as transmission clock. If USART operates in spi mode, SS pin is used as SS input pin in slave mode or can be configured as SS output pin in master mode. This can be done by setting SPISS bit in UCTRL3 register.

11.9.8.1 Sending TX data

A data transmission is initiated by loading the transmit buffer (UDATA register I/O location) with the data to be transmitted. The data written in transmit buffer is moved to the shift register when the shift register is ready to send a new frame. The shift register is loaded with the new data if it is in idle state or immediately after the last stop bit of the previous frame is transmitted. When the shift register is loaded with new data, it will transfer one complete frame at the settings of control registers. If the 9-bit characters are used in asynchronous or synchronous operation mode (USIZE[2:0]=7), the ninth bit must be written to the TX8 bit in UCTRL3 register before loading transmit buffer (UDATA register).

11.9.8.2 Transmitter flag and interrupt

The USART Transmitter has 2 flags which indicate its state. One is USART Data Register Empty (UDRE) and the other is Transmit Complete (TXC). Both flags can be interrupt sources.

UDRE flag indicates whether the transmit buffer is ready to be loaded with new data. This bit is set when the transmit buffer is empty and cleared when the transmit buffer contains data to be transmitted that has not yet been moved into the shift register. And also this flag can be cleared by writing ‘0’ to this bit position. Writing ‘1’ to this bit position is prevented.

When the Data Register Empty Interrupt Enable (UDRIE) bit in UCTRL2 register is set and the Global Interrupt is enabled, USART Data Register Empty Interrupt is generated while UDRE flag is set.

The Transmit Complete (TXC) flag bit is set when the entire frame in the transmit shift register has been shifted out and there are no more data in the transmit buffer. The TXC flag is automatically cleared when the Transmit Complete Interrupt service routine is executed, or it can be cleared by writing ‘0’ to TXC bit in USTAT register.

When the Transmit Complete Interrupt Enable (TXCIE) bit in UCTRL2 register is set and the Global Interrupt is enabled, USART Transmit Complete Interrupt is generated while TXC flag is set.

11.9.8.3 Parity Generator

The Parity Generator calculates the parity bit for the sending serial frame data. When parity bit is enabled (UPM[1]=1), the transmitter control logic inserts the parity bit between the MSB and the first stop bit of the sending frame.

11.9.8.4 Disabling Transmitter

Disabling the Transmitter by clearing the TXE bit will not become effective until ongoing transmission is completed. When the Transmitter is disabled, the TXD(=MOSI) pin is used as normal General Purpose I/O (GPIO) or primary function pin.

11.9.9 USART Receiver

The USART Receiver is enabled by setting the RXE bit in the UCTRL1 register. When the Receiver is enabled, the normal pin operation of the RXD(=MISO) pin is overridden by the USART as the serial input pin of the Receiver. The baud-rate, mode of operation and frame format must be set before starting serial reception. If synchronous or spi operation is used, the clock on the XCK pin will be used as transfer clock. If USART operates in spi mode, SS pin is used as SS input pin in slave mode or can be configured as SS output pin in master mode. This can be done by setting SPISS bit in UCTRL3 register.

11.9.9.1 Receiving RX data

When USART is in synchronous or asynchronous operation mode, the Receiver starts data reception when it detects a valid start bit (LOW) on RXD(=MISO) pin. Each bit after start bit is sampled at pre-defined baud-rate (asynchronous) or sampling edge of XCK (synchronous), and shifted into the receive shift register until the first stop bit of a frame is received. Even if there's 2^{nd} stop bit in the frame, the 2^{nd} stop bit is ignored by the Receiver. That is, receiving the first stop bit means that a complete serial frame is present in the receiver shift register and the contents of shift register are to be moved into the receive buffer. The receive buffer is read by reading the UDATA register.

If 9-bit characters are used (USIZE[2:0] = 7) the ninth bit is stored in the RX8 bit position in the UCTRL3 register. The 9^{th} bit must be read from the RX8 bit before reading the low 8 bits from the UDATA register. Likewise, the error flags FE, DOR, PE must be read before reading the data from UDATA register. This is because the error flags are stored in the same FIFO position of the receive buffer.

11.9.9.2 Receiver flag and interrupt

The USART Receiver has one flag that indicates the Receiver state.

The Receive Complete (RXC) flag indicates whether there are unread data present in the receive buffer. This flag is set when there are unread data in the receive buffer and cleared when the receive buffer is empty. If the Receiver is disabled (RXE=0), the receiver buffer is flushed and the RXC flag is cleared.

When the Receive Complete Interrupt Enable (RXCIE) bit in the UCTRL2 register is set and Global Interrupt is enabled, the USART Receiver Complete Interrupt is generated while RXC flag is set.

The USART Receiver has three error flags which are Frame Error (FE), Data OverRun (DOR) and Parity Error (PE). These error flags can be read from the USTAT register. As data received are stored in the 2-level receive buffer, these error flags are also stored in the same position of receive buffer. So, before reading received data from UDATA register, read the USTAT register first which contains error flags.

The Frame Error (FE) flag indicates the state of the first stop bit. The FE flag is zero when the stop bit was correctly detected as one, and the FE flag is one when the stop bit was incorrect, ie detected as zero. This flag can be used for detecting out-of-sync conditions between data frames.

The Data OverRun (DOR) flag indicates data loss due to a receive buffer full condition. A DOR occurs when the receive buffer is full, and another new data is present in the receive shift register which are to be stored into the receive buffer. After the DOR flag is set, all the incoming data are lost. To prevent data loss or clear this flag, read the receive buffer.

The Parity Error (PE) flag indicates that the frame in the receive buffer had a Parity Error when received. If Parity Check function is not enabled (UPM[1]=0), the PE bit is always read zero.

Caution : The error flags related to receive operation are not used when USART is in spi mode.

11.9.9.3 Parity Checker

If Parity Bit is enabled (UPM[1]=1), the Parity Checker calculates the parity of the data bits in incoming frame and compares the result with the parity bit from the received serial frame.

11.9.9.4 Disabling Receiver

In contrast to Transmitter, disabling the Receiver by clearing RXE bit makes the Receiver inactive immediately. When the Receiver is disabled the Receiver flushes the receive buffer and the remaining data in the buffer is all reset. The RXD(=MISO) pin is not overridden the function of USART, so RXD(=MISO) pin becomes normal GPIO or primary function pin.

11.9.9.5 Asynchronous Data Reception

To receive asynchronous data frame, the USART includes a clock and data recovery unit. The Clock Recovery logic is used for synchronizing the internally generated baud-rate clock to the incoming asynchronous serial frame on the RXD(=MISO) pin.

The Data recovery logic samples and low pass filters the incoming bits, and this removes the noise of RXD(=MISO) pin.

The next figure illustrates the sampling process of the start bit of an incoming frame. The sampling rate is 16 times the baud-rate for normal mode, and 8 times the baud rate for Double Speed mode (U2X=1). The horizontal arrows show the synchronization variation due to the asynchronous sampling process. Note that larger time variation is shown when using the Double Speed mode.

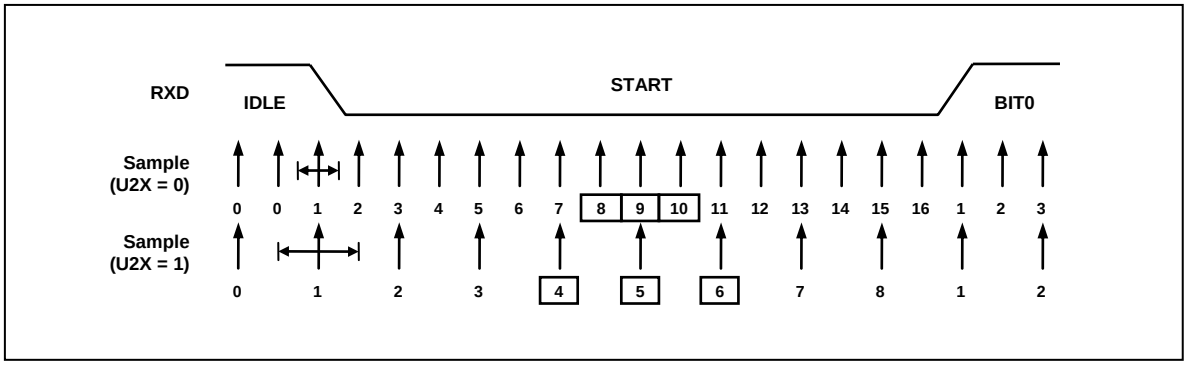


Figure 11-38 Start Bit Sampling

When the Receiver is enabled (RXE=1), the clock recovery logic tries to find a high to low transition on the RXD(=MISO) line, the start bit condition. After detecting high to low transition on RXD(=MISO) line, the clock recovery logic uses samples 8,9, and 10 for Normal mode, and samples 4, 5, and 6 for Double Speed mode to decide if a valid start bit is received. If more than 2 samples have logical low level, it is considered that a valid start bit is detected and the internally generated clock is synchronized to the incoming data frame. And the data recovery can begin. The synchronization process is repeated for each start bit.

As described above, when the Receiver clock is synchronized to the start bit, the data recovery can begin. Data recovery process is almost similar to the clock recovery process. The data recovery logic samples 16 times for each incoming bits for Normal mode and 8 times for Double Speed mode. And uses sample 8, 9, and 10 to decide data value for Normal mode, samples 4, 5, and 6 for Double Speed mode. If more than 2 samples have low levels, the received bit is considered to a logic 0 and more than 2 samples have high levels, the received bit is considered to a logic 1. The data recovery process is then repeated until a complete frame is received including the first stop bit. The decided bit value is stored in the receive shift register in order. Note that the Receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the Receiver is in idle state and waiting to find start bit.

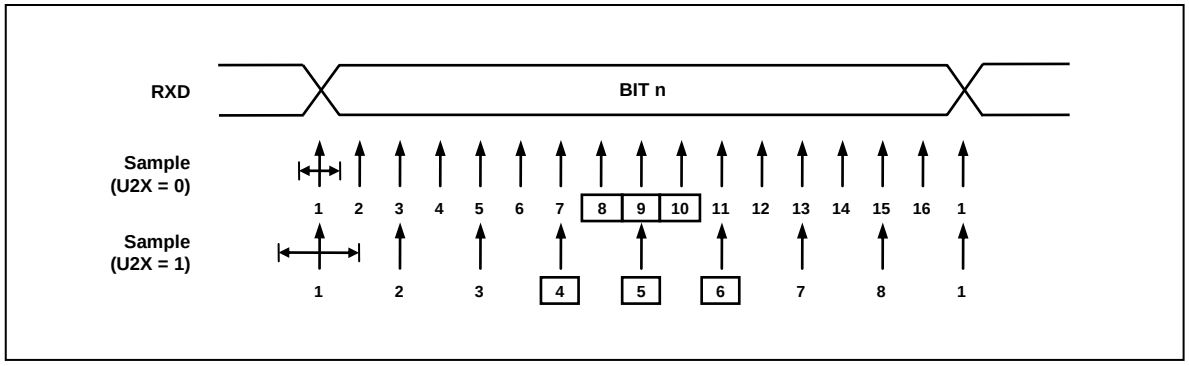


Figure 11-39 The Sampling of Data and Parity Bit

The process for detecting stop bit is like clock and data recovery process. That is, if 2 or more samples of 3 center values have high level, correct stop bit is detected, else a Frame Error flag is set. After deciding first stop bit whether a valid stop bit is received or not, the Receiver goes idle state and monitors the RXD(=MISO) line to check a valid high to low transition is detected (start bit detection).

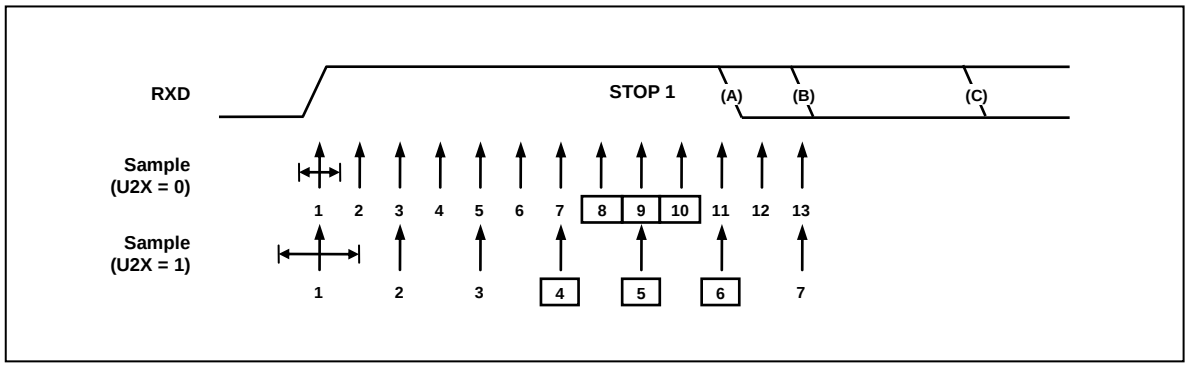


Figure 11-40 Stop Bit Sampling and Next Start Bit Sampling

11.9.10 SPI Mode

The USART can be set to operate in industrial standard SPI compliant mode. The SPI mode has the following features.

- Full duplex, three-wire synchronous data transfer
- Master or Slave operation
- Supports all four SPI modes of operation (mode0, 1, 2, and 3)
- Selectable LSB first or MSB first data transfer
- Double buffered transmit and receive
- Programmable transmit bit rate

When SPI mode is enabled (UMSEL[1:0]=11_B), the Slave Select (SS) pin becomes active low input in slave mode operation, or can be output in master mode operation if SPISS bit is set.

Note that during SPI mode of operation, the pin RXD is renamed as MISO and TXD is renamed as MOSI for compatibility to other SPI devices.

11.9.10.1 SPI Clock Formats and Timing

To accommodate a wide variety of synchronous serial peripherals from different manufacturers, the USART has a clock polarity bit (UCPOL) and a clock phase control bit (UCPHA) to select one of four clock formats for data transfers. UCPOL selectively insert an inverter in series with the clock. UCPHA chooses between two different clock phase relationships between the clock and data. Note that UCPHA and UCPOL bits in UCTRL1 register have different meaning according to the UMSEL[1:0] bits which decides the operating mode of USART.

Table below shows four combinations of UCPOL and UCPHA for SPI mode 0, 1, 2, and 3.

SPI Mode	UCPOL	UCPHA	Leading Edge	Trailing Edge
0	0	0	Sample (Rising)	Setup (Falling)
1	0	1	Setup (Rising)	Sample (Falling)
2	1	0	Sample (Falling)	Setup (Rising)
3	1	1	Setup (Falling)	Sample (Rising)

Table 11-17 SPI Mode by UCPOL & UCPHA

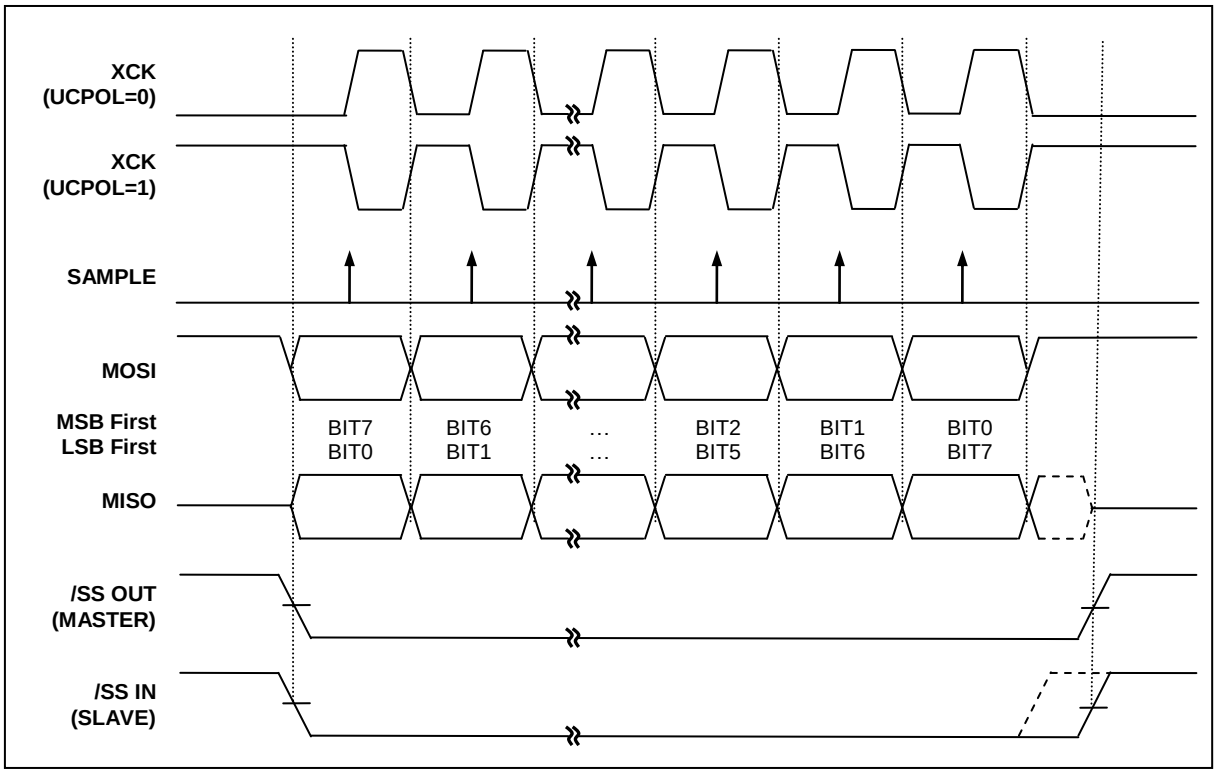


Figure 11-41 SPI Clock Formats when UCPHA=0

When UCPHA=0, the slave begins to drive its MISO output with the first data bit value when SS goes to active low. The first XCK edge causes both the master and the slave to sample the data bit value on their MISO and MOSI inputs, respectively. At the second XCK edge, the USART shifts the second data bit value out to the MOSI and MISO outputs of the master and slave, respectively. Unlike the case of UCPHA=1, when UCPHA=0, the slave's SS input must go to its inactive high level between transfers. This is because the slave can prepare the first data bit when it detects falling edge of SS input.

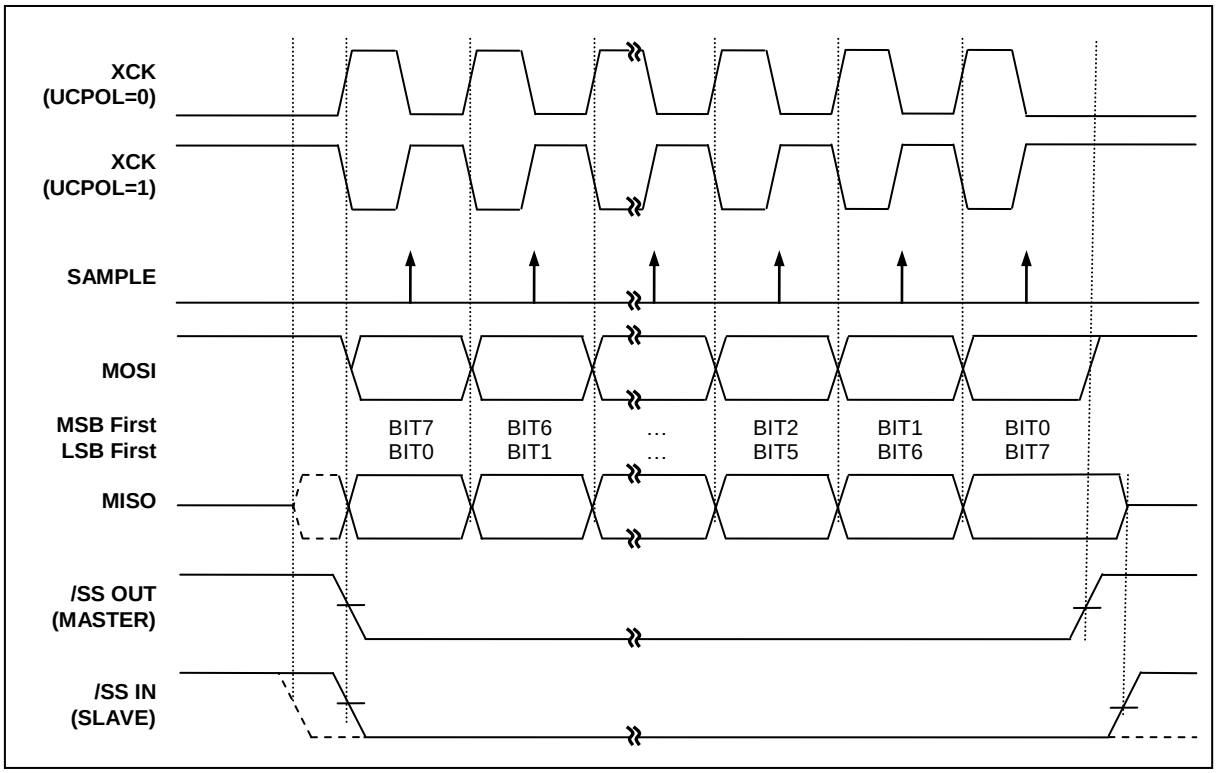


Figure 11-42 SPI Clock Formats when UCPHA=1

When UCPHA=1, the slave begins to drive its MISO output when SS goes active low, but the data is not defined until the first XCK edge. The first XCK edge shifts the first bit of data from the shifter onto the MOSI output of the master and the MISO output of the slave. The next XCK edge causes both the master and slave to sample the data bit value on their MISO and MOSI inputs, respectively. At the third XCK edge, the USART shifts the second data bit value out to the MOSI and MISO output of the master and slave respectively. When UCPHA=1, the slave's SS input is not required to go to its inactive high level between transfers.

Because the SPI logic reuses the USART resources, SPI mode of operation is similar to that of synchronous or asynchronous operation. An SPI transfer is initiated by checking for the USART Data Register Empty flag (UDRE=1) and then writing a byte of data to the UDATA Register.

Caution : In master mode of operation, even if transmission is not enabled (TXE=0), writing data to the UDATA register is necessary because the clock XCK is generated from transmitter block.

11.9.11 Register Map

Name	Address	Dir	Default	Description
UCTRL01	$E2_H$	R/W	00_H	USART0 Control 1 Register
UCTRL02	$E3_H$	R/W	00_H	USART0 Control 2 Register
UCTRL03	$E4_H$	R/W	00_H	USART0 Control 3 Register
USTAT0	$E5_H$	R	80_H	USART0 Status Register
UBAUD0	$E6_H$	R/W	FF_H	USART0 Baud Rate Generation Register
UDATA0	$E7_H$	R/W	FF_H	USART0 Data Register
UCTRL11	FA_H	R/W	00_H	USART1 Control 1 Register

UCTRL12	FB_H	R/W	00_H	USART1 Control 2 Register
UCTRL13	FC_H	R/W	00_H	USART1 Control 3 Register
USTAT1	FD_H	R	80_H	USART1 Status Register
UBAUD1	FE_H	R/W	FF_H	USART1 Baud Rate Generation Register
UDATA1	FF_H	R/W	FF_H	USART1 Data Register

Table 11-18 Register map of USART

11.9.12 Register Description

UCTRLx1 (USART0[1] Control 1 Register) **$E2_H$ / FA_H**

7	6	5	4	3	2	1	0
UMSEL1	UMSEL0	UPM1	UPM0	USIZE2	USIZE1 UDORD	USIZE0 UCPHA	UCPOL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

UMSEL[1:0] Selects operation mode of USART

UMSEL1	UMSEL0	Operating Mode
0	0	Asynchronous Mode (Normal Uart)
0	1	Synchronous Mode (Synchronous Uart)
1	0	Reserved
1	1	SPI Mode

UPM[1:0] Selects Parity Generation and Check methods

UPM1	UPM0	Parity mode
0	0	No Parity
0	1	Reserved
1	0	Even Parity
1	1	Odd Parity

USIZE[2:0] When in asynchronous or synchronous mode of operation, selects the length of data bits in frame.

USIZE2	USIZE1	USIZE0	Data length
0	0	0	5 bit
0	0	1	6 bit
0	1	0	7 bit
0	1	1	8 bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9 bit

UDORD This bit is in the same bit position with USIZE1. In SPI mode, when set to one the MSB of the data byte is transmitted first. When set to zero the LSB of the data byte is transmitted first.

0 LSB First

1 MSB First

UCPOL Selects polarity of XCK in synchronous or spi mode

0 TXD(=MOSI) change @Rising Edge, RXD(=MISO) change @Falling Edge

1 TXD(=MOSI) change @ Falling Edge, RXD(=MISO) change @ Rising Edge

UCPHA This bit is in the same bit position with USIZE0. In SPI mode, along

with UCPOL bit, selects one of two clock formats for different kinds of synchronous serial peripherals. Leading edge means first XCK edge and trailing edge means 2nd or last clock edge of XCK in one XCK pulse. And Sample means detecting of incoming receive bit, Setup means preparing transmit data.

UCPOL	UCPHA	Leading Edge	Trailing Edge
0	0	Sample (Rising)	Setup (Falling)
0	1	Setup (Rising)	Sample (Falling)
1	0	Sample (Falling)	Setup (Rising)
1	1	Setup (Falling)	Sample (Rising)

UCTRLx2 (USART0[1] Control 2 Register) **$E3_H$ / FB_H**

7	6	5	4	3	2	1	0
UDRIE	TXCIE	RXCIE	WAKEIE	TXE	RXE	USARTEN	U2X
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

UDRIE Interrupt enable bit for USART Data Register Empty.
- 0 Interrupt from UDRE is inhibited (use polling)
- 1 When UDRE is set, request an interrupt

TXCIE Interrupt enable bit for Transmit Complete.
- 0 Interrupt from TXC is inhibited (use polling)
- 1 When TXC is set, request an interrupt

RXCIE Interrupt enable bit for Receive Complete
- 0 Interrupt from RXC is inhibited (use polling)
- 1 When RXC is set, request an interrupt

WAKEIE Interrupt enable bit for Asynchronous Wake in STOP mode. When device is in stop mode, if RXD(=MISO) goes to LOW level an interrupt can be requested to wake-up system.
- 0 Interrupt from Wake is inhibited
- 1 When WAKE is set, request an interrupt

TXE Enables the transmitter unit.
- 0 Transmitter is disabled
- 1 Transmitter is enabled

RXE Enables the receiver unit.
- 0 Receiver is disabled
- 1 Receiver is enabled

USARTEN Activate USART module by supplying clock.
- 0 USART is disabled (clock is halted)
- 1 USART is enabled

U2X This bit only has effect for the asynchronous operation and selects receiver sampling rate.
- 0 Normal asynchronous operation
- 1 Double Speed asynchronous operation

UCTRLx3 (USART0[1] Control 3 Register) **$E4_H$ / FC_H**

7	6	5	4	3	2	1	0
MASTER	LOOPS	DISXCK	SPISS	-	USBS	TX8	RX8
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Initial value : 00_H

MASTER Selects master or slave in SPI or Synchronous mode operation and

controls the direction of XCK pin.

0 Slave mode operation and XCK is input pin.

1 Master mode operation and XCK is output pin

LOOPS Controls the Loop Back mode of USART, for test mode

0 Normal operation

1 Loop Back mode

DISXCK In Synchronous mode of operation, selects the waveform of XCK output.

0 XCK is free-running while USART is enabled in synchronous master mode.

1 XCK is active while any frame is on transferring.

SPISS Controls the functionality of SS pin in master SPI mode.

0 SS pin is normal GPIO or other primary function

1 SS output to other slave device

USBS Selects the length of stop bit in Asynchronous or Synchronous mode of operation.

0 1 Stop Bit

1 2 Stop Bit

TX8 The ninth bit of data frame in Asynchronous or Synchronous mode of operation. Write this bit first before loading the UDATA register.

0 MSB (9^{th} bit) to be transmitted is '0'

1 MSB (9^{th} bit) to be transmitted is '1'

RX8 The ninth bit of data frame in Asynchronous or Synchronous mode of operation. Read this bit first before reading the receive buffer.

0 MSB (9^{th} bit) received is '0'

1 MSB (9^{th} bit) received is '1'

USTATx (USART0[1] Status Register) $E5_H$ / FD_H

7	6	5	4	3	2	1	0
UDRE	TXC	RXC	WAKE	SOFTRST	DOR	FE	PE
R/W	R/W	R/W	R/W	R/W	R	R	R

Initial value : 80_H

UDRE The UDRE flag indicates if the transmit buffer (UDATA) is ready to be loaded with new data. If UDRE is '1', it means the transmit buffer is empty and can hold one or two new data. This flag can generate an UDRE interrupt. Writing '0' to this bit position will clear UDRE flag.

0 Transmit buffer is not empty.

1 Transmit buffer is empty.

TXC This flag is set when the entire frame in the transmit shift register has been shifted out and there is no new data currently present in the transmit buffer. This flag is automatically cleared when the interrupt service routine of a TXC interrupt is executed. It is also cleared by writing '0' to this bit position. This flag can generate a TXC interrupt.

0 Transmission is ongoing.

1 Transmit buffer is empty and the data in transmit shift register are shifted out completely.

RXC This flag is set when there are unread data in the receive buffer and cleared when all the data in the receive buffer are read. The RXC flag can be used to generate a RXC interrupt.

0 There is no data unread in the receive buffer

1 There are more than 1 data in the receive buffer

WAKE This flag is set when the RXD(=MISO) pin is detected low while the CPU is in stop mode. This flag can be used to generate a WAKE interrupt. This bit is set only when in asynchronous mode of operation. NOTE

0 No WAKE interrupt is generated.

1 WAKE interrupt is generated.

SOFTRST This is an internal reset and only has effect on USART. Writing '1' to this bit initializes the internal logic of USART and is auto cleared.

0 No operation

1 Reset USART

DOR This bit is set if a Data OverRun occurs. While this bit is set, the incoming data frame is ignored. This flag is valid until the receive buffer is read.

0 No Data OverRun

1 Data OverRun detected

FE This bit is set if the first stop bit of next character in the receive buffer is detected as '0'. This bit is valid until the receive buffer is read.

0 No Frame Error

1 Frame Error detected

PE This bit is set if the next character in the receive buffer has a Parity Error when received while Parity Checking is enabled. This bit is valid until the receive buffer is read.

0 No Parity Error

1 Parity Error detected

NOTE When the WAKE function of USART is used as a release source from STOP mode, it is required to clear this bit in the RX interrupt service routine. Else the device will not wake-up from STOP mode again by the change of RXD(=MISO) pin.

UBAUDx (USART0[1] Baud-Rate Generation Register) $E6_H$ / FE_H

7	6	5	4	3	2	1	0
UBAUD7	UBAUD6	UBAUD5	UBAUD4	UBAUD3	UBAUD2	UBAUD1	UBAUD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

UBAUD [7:0] The value in this register is used to generate internal baud rate in asynchronous mode or to generate XCK clock in synchronous or spi mode. To prevent malfunction, do not write '0' in asynchronous mode, and do not write '0' or '1' in synchronous or spi mode.

UDATAx (USART0[1] Data Register) $E7_H$ / FF_H

7	6	5	4	3	2	1	0
UDATA7	UDATA6	UDATA5	UDATA4	UDATA3	UDATA2	UDATA1	UDATA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

UDATA [7:0] The USART Transmit Buffer and Receive Buffer share the same I/O address with this DATA register. The Transmit Data Buffer is the destination for data written to the UDATA register. Reading the UDATA register returns the contents of the Receive Buffer.

Write this register only when the UDRE flag is set. In spi or synchronous master mode, write this register even if TX is not enabled to generate clock, XCK.

11.9.13 Baud Rate Setting (example)

Baud Rate	fOSC=1.00MHz				fOSC=1.8432MHz				fOSC=2.00MHz			
	U2X=0		U2X=1		U2X=0		U2X=1		U2X=0		U2X=1	
	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4K	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2K	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8K	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4K	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6K	-	-	1	8.5%	1	-25.0%	3	0.0%	1	8.5%	3	8.5%
76.8K	-	-	1	-18.6%	1	0.0%	2	0.0%	1	-18.6%	2	8.5%
115.2K	-	-	-	-	-	-	1	0.0%	-	-	1	8.5%
230.4K	-	-	-	-	-	-	-	-	-	-	-	-

Baud Rate	fOSC=3.6864MHz				fOSC=4.00MHz				fOSC=7.3728MHz			
	U2X=0		U2X=1		U2X=0		U2X=1		U2X=0		U2X=1	
	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR
2400	95	0.0%	191	0.0%	103	0.2%	207	0.2%	191	0.0%	-	-
4800	47	0.0%	95	0.0%	51	0.2%	103	0.2%	95	0.0%	191	0.0%
9600	23	0.0%	47	0.0%	25	0.2%	51	0.2%	47	0.0%	95	0.0%
14.4K	15	0.0%	31	0.0%	16	2.1%	34	-0.8%	31	0.0%	63	0.0%
19.2K	11	0.0%	23	0.0%	12	0.2%	25	0.2%	23	0.0%	47	0.0%
28.8K	7	0.0%	15	0.0%	8	-3.5%	16	2.1%	15	0.0%	31	0.0%
38.4K	5	0.0%	11	0.0%	6	-7.0%	12	0.2%	11	0.0%	23	0.0%
57.6K	3	0.0%	7	0.0%	3	8.5%	8	-3.5%	7	0.0%	15	0.0%
76.8K	2	0.0%	5	0.0%	2	8.5%	6	-7.0%	5	0.0%	11	0.0%
115.2K	1	0.0%	3	0.0%	1	8.5%	3	8.5%	3	0.0%	7	0.0%
230.4K	-	-	1	0.0%	-	-	1	8.5%	1	0.0%	3	0.0%
250K	-	-	1	-7.8%	-	-	1	0.0%	1	-7.8%	3	-7.8%
0.5M	-	-	-	-	-	-	-	-	-	-	1	-7.8%

Baud Rate	fOSC=8.00MHz				fOSC=11.0592MHz				fOSC=14.7456MHz			
	U2X=0		U2X=1		U2X=0		U2X=1		U2X=0		U2X=1	
	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR
2400	207	0.2%	-	-	-	-	-	-	-	-	-	-
4800	103	0.2%	207	0.2%	143	0.0%	-	-	191	0.0%	-	-
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4K	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2K	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8K	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4K	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%

57.6K	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8K	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2 K	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%
230.4 K	1	8.5%	3	8.5%	2	0.0%	5	0.0%	3	0.0%	7	0.0%
250K	1	0.0%	3	0.0%	2	-7.8%	5	-7.8%	3	-7.8%	6	5.3%
0.5M	-	-	1	0.0%	-	-	2	-7.8%	1	-7.8%	3	-7.8%
1M	-	-	-	-	-	-	-	-	-	-	1	-7.8%

11.10 I²C

11.10.1 Overview

The I²C is one of industrial standard serial communication protocols, and which uses 2 bus lines Serial Data Line (SDA) and Serial Clock Line (SCL) to exchange data. Because both SDA and SCL lines are open-drain output, each line needs pull-up resistor. The features are as shown below.

- Compatible with I²C bus standard
- Multi-master operation
- Up to 400 KHz data transfer speed
- 7 bit address
- Support two slave addresses
- Both master and slave operation
- Bus busy detection

11.10.2 Block Diagram

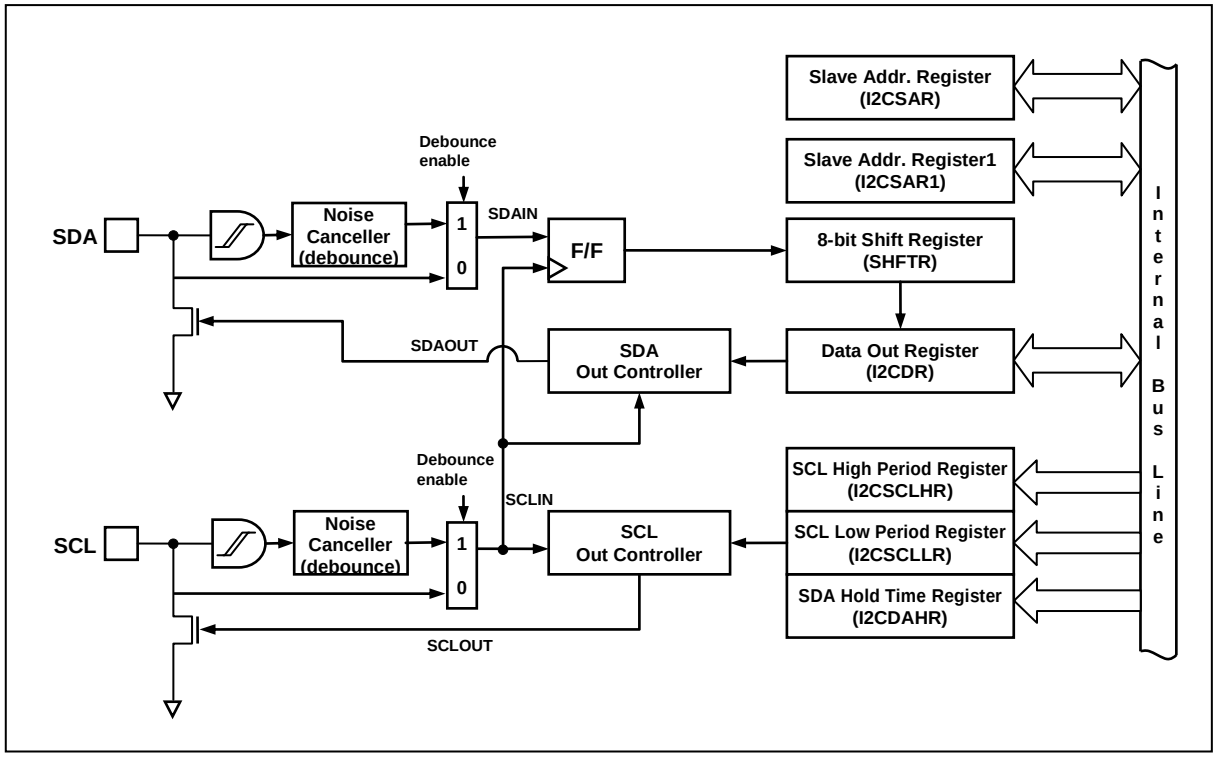


Figure 11-43 I²C Block Diagram

11.10.3 I²C Bit Transfer

The data on the SDA line must be stable during HIGH period of the clock, SCL. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW. The exceptions are START(S), repeated START(Sr) and STOP(P) condition where data line changes when clock line is high.

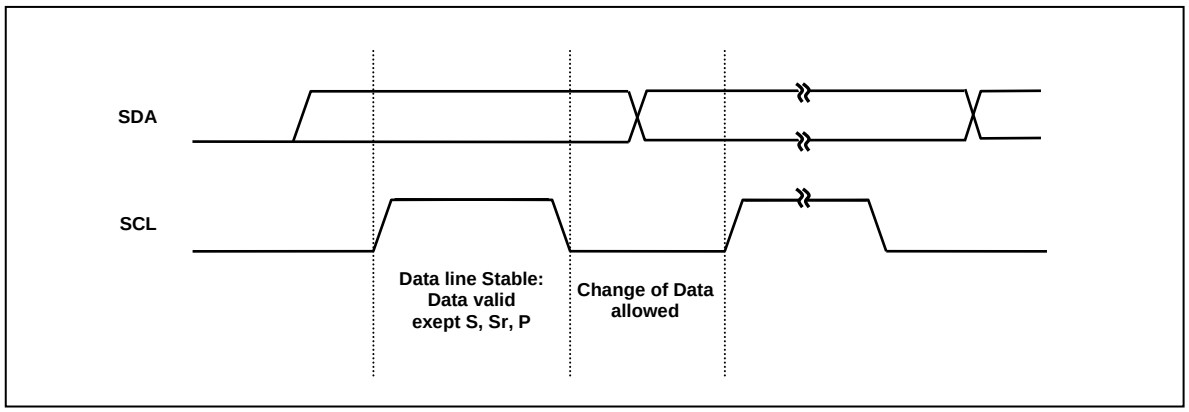


Figure 11-44 Bit Transfer on the I²C-Bus

11.10.4 Start / Repeated Start / Stop

One master can issue a START (S) condition to notice other devices connected to the SCL, SDA lines that it will use the bus. A STOP (P) condition is generated by the master to release the bus lines so that other devices can use it.

A high to low transition on the SDA line while SCL is high defines a START (S) condition.

A low to high transition on the SDA line while SCL is high defines a STOP (P) condition.

START and STOP conditions are always generated by the master. The bus is considered to be busy after START condition. The bus is considered to be free again after STOP condition, ie, the bus is busy between START and STOP condition. If a repeated START condition (Sr) is generated instead of STOP condition, the bus stays busy. So, the START and repeated START conditions are functionally identical.

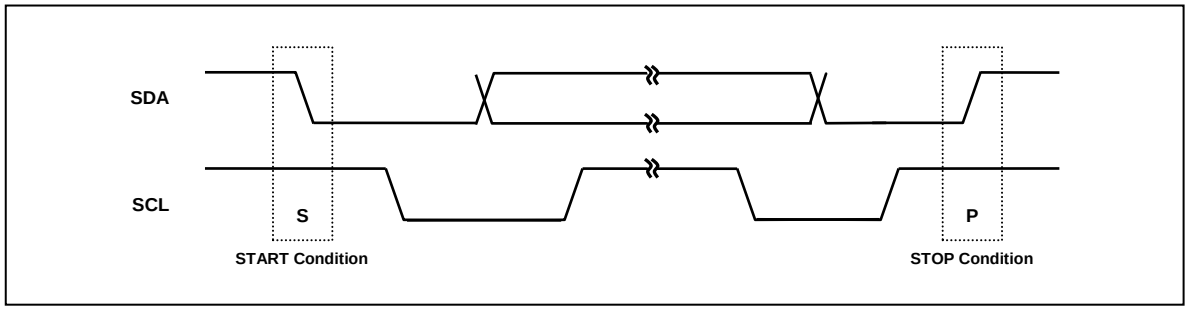


Figure 11-45 START and STOP Condition

11.10.5 Data Transfer

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unlimited. Each byte has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first. If a slave can't receive or transmit another complete byte of data until it has performed some other function, it can hold the clock line SCL LOW to force the master into a wait state. Data transfer then continues when the slave is ready for another byte of data and releases clock line SCL.

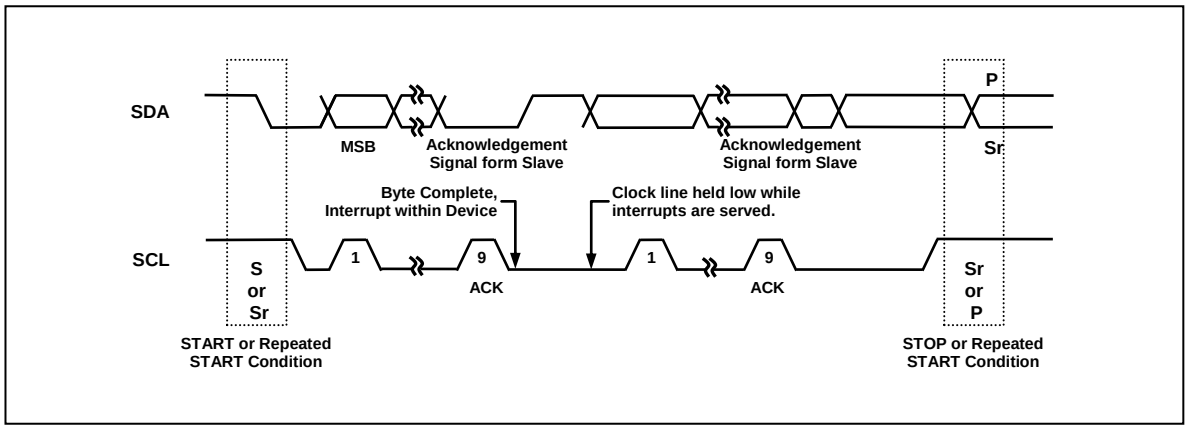


Figure 11-46 STOP or Repeated START Condition

11.10.6 Acknowledge

The acknowledge related clock pulse is generated by the master. The transmitter releases the SDA line (HIGH) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse. When a slave is addressed by a master (Address Packet), and if it is unable to receive or transmit because it's performing some real time function, the data line must be left HIGH by the slave. And also, when a slave addressed by a master is unable to receive more data bits, the slave receiver must release the SDA line (Data Packet). The master can then generate either a STOP condition to abort the transfer, or a repeated START condition to start a new transfer.

If a master receiver is involved in a transfer, it must signal the end of data to the slave transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

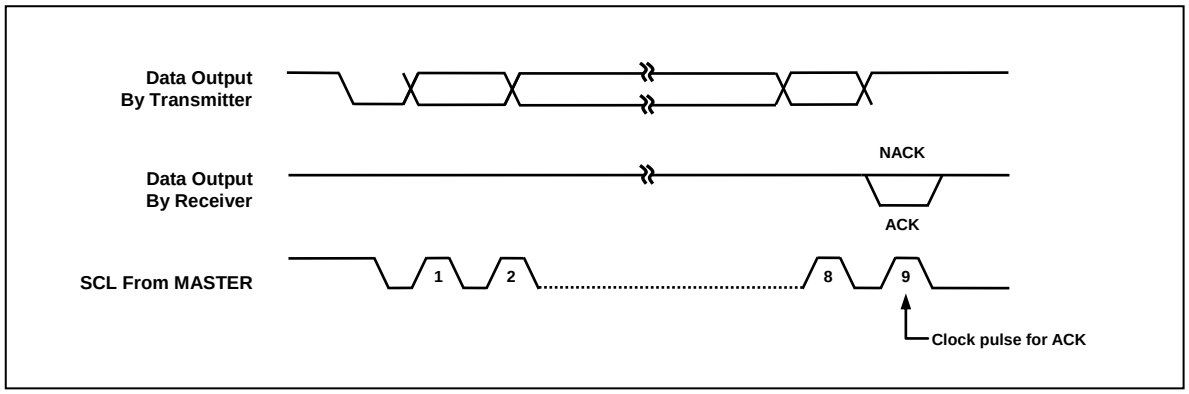


Figure 11-47 Acknowledge on the I^{2}C-Bus

11.10.7 Synchronization / Arbitration

Clock synchronization is performed using the wired-AND connection of I^2C interfaces to the SCL line. This means that a HIGH to LOW transition on the SCL line will cause the devices concerned to start counting off their LOW period and it will hold the SCL line in that state until the clock HIGH state is reached. However the LOW to HIGH transition of this clock may not change the state of the SCL line if another clock is still within its LOW period. In this way, a synchronized SCL clock is generated with its LOW period determined by the device with the longest clock LOW period, and its HIGH period determined by the one with the shortest clock HIGH period.

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition. Arbitration takes place on the SDA line, while the SCL line is at the HIGH level, in such a way that the master which transmits a HIGH level, while another master is transmitting a LOW level will switch off its DATA output state because the level on the bus doesn't correspond to its own level. Arbitration continues for many bits until a winning master gets the ownership of I^2C bus. Its first stage is comparison of the address bits.

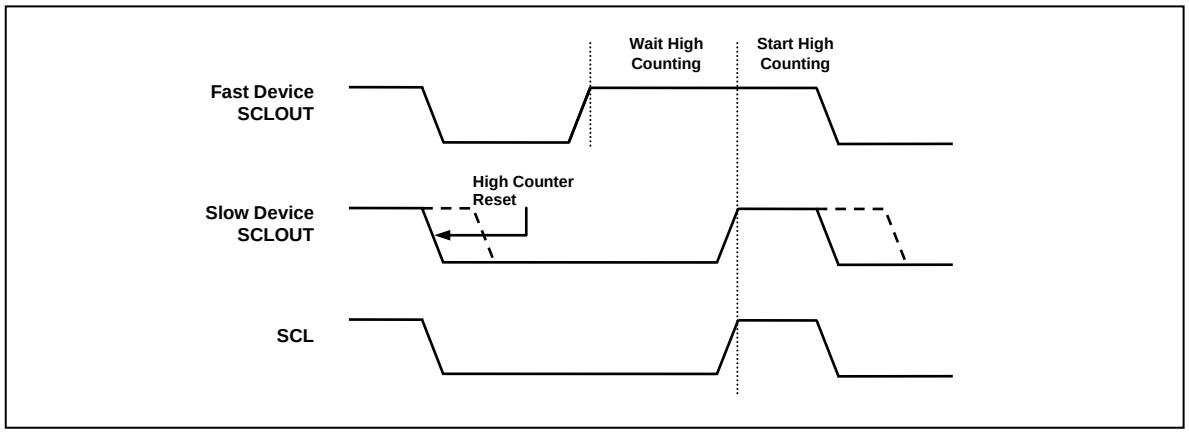


Figure 11-48 Clock Synchronization during Arbitration Procedure

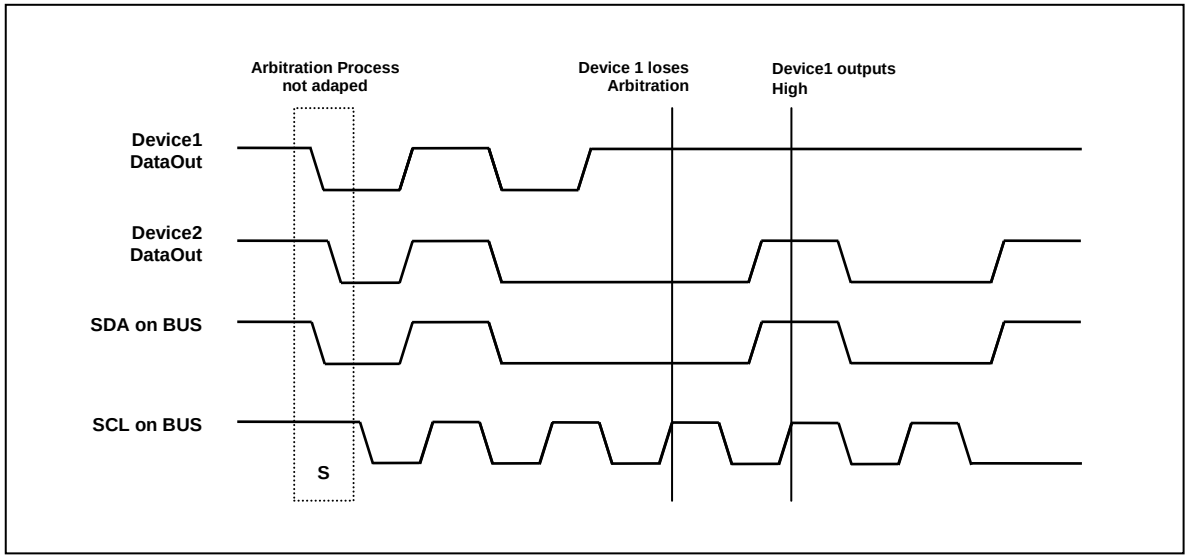


Figure 11-49 Arbitration Procedure of Two Masters

11.10.8 Operation

The I²C is byte-oriented and interrupt based. Interrupts are issued after all bus events except for a transmission of a START condition. Because the I²C is interrupt based, the application software is free to carry on other operations during a I²C byte transfer.

Note that when a I²C interrupt is generated, IIF flag in I2CMR register is set, it is cleared by writing an arbitrary value to I2CSR. When I²C interrupt occurs, the SCL line is hold LOW until writing any value to I2CSR. When the IIF flag is set, the I2CSR contains a value indicating the current state of the I²C bus. According to the value in I2CSR, software can decide what to do next.

I²C can operate in 4 modes by configuring master/slave, transmitter/receiver. The operating mode is configured by a winning master. A more detailed explanation follows below.

11.10.8.1 Master Transmitter

To operate I²C in master transmitter, follow the recommended steps below.

1. Enable I²C by setting IICEN bit in I2CMR. This provides main clock to the peripheral.
2. Load SLA+W into the I2CDR where SLA is address of slave device and W is transfer direction from the viewpoint of the master. For master transmitter, W is '0'. Note that I2CDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CSCLLR and I2CSCLHR for the Low and High period of SCL line.
4. Configure the I2CSDAHR to decide when SDA changes value from falling edge of SCL. If SDA should change in the middle of SCL LOW period, load half the value of I2CSCLLR to the I2CSDAHR.
5. Set the START bit in I2CMR. This transmits a START condition. And also configure how to handle interrupt and ACK signal. When the START bit is set, 8-bit data in I2CDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9th high period of SCL. If the master gains bus mastership, I²C generates GCALL interrupt regardless of the reception of ACK from the slave device. When I²C loses bus mastership during arbitration process, the MLOST bit in I2CSR is set, and I²C waits in idle state or can be operate as an addressed slave. To operate as a slave when the MLSOT bit in I2CSR is set, the ACKEN bit in I2CMR must be set and the received 7-bit address must equal to the SLA bits in I2CSAR. In this case I²C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I²C holds the SCL LOW. This is because to decide whether I²C continues serial transfer or stops communication. The following steps continue assuming that I²C does not lose mastership during first data transfer.

   I²C (Master) can choose one of the following cases regardless of the reception of ACK signal from slave.

   1) Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CDR.
   2) Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOP bit in I2CMR.
   3) Master transmits repeated START condition with not checking ACK signal. In this case, load SLA+R/W into the I2CDR and set START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of

3), move to step 6 after transmitting the data in I2CDR and if transfer direction bit is '1' go to master receiver section.

7. 1-Byte of data is being transmitted. During data transfer, bus arbitration continues.
8. This is ACK signal processing stage for data packet transmitted by master. I^2C holds the SCL LOW. When I^2C loses bus mastership while transmitting data arbitrating other masters, the MLOST bit in I2CSR is set. If then, I^2C waits in idle state. When the data in I2CDR is transmitted completely, I^2C generates TEND interrupt.

   I^2C can choose one of the following cases regardless of the reception of ACK signal from slave.

   1) Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CDR.
   2) Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOP bit in I2CMR.
   3) Master transmits repeated START condition with not checking ACK signal. In this case, load SLA+R/W into the I2CDR and set the START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR, and if transfer direction bit is '1' go to master receiver section.
9. This is the final step for master transmitter function of I^2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I^2C enters idle state.

The next figure depicts above process for master transmitter operation of I^2C.

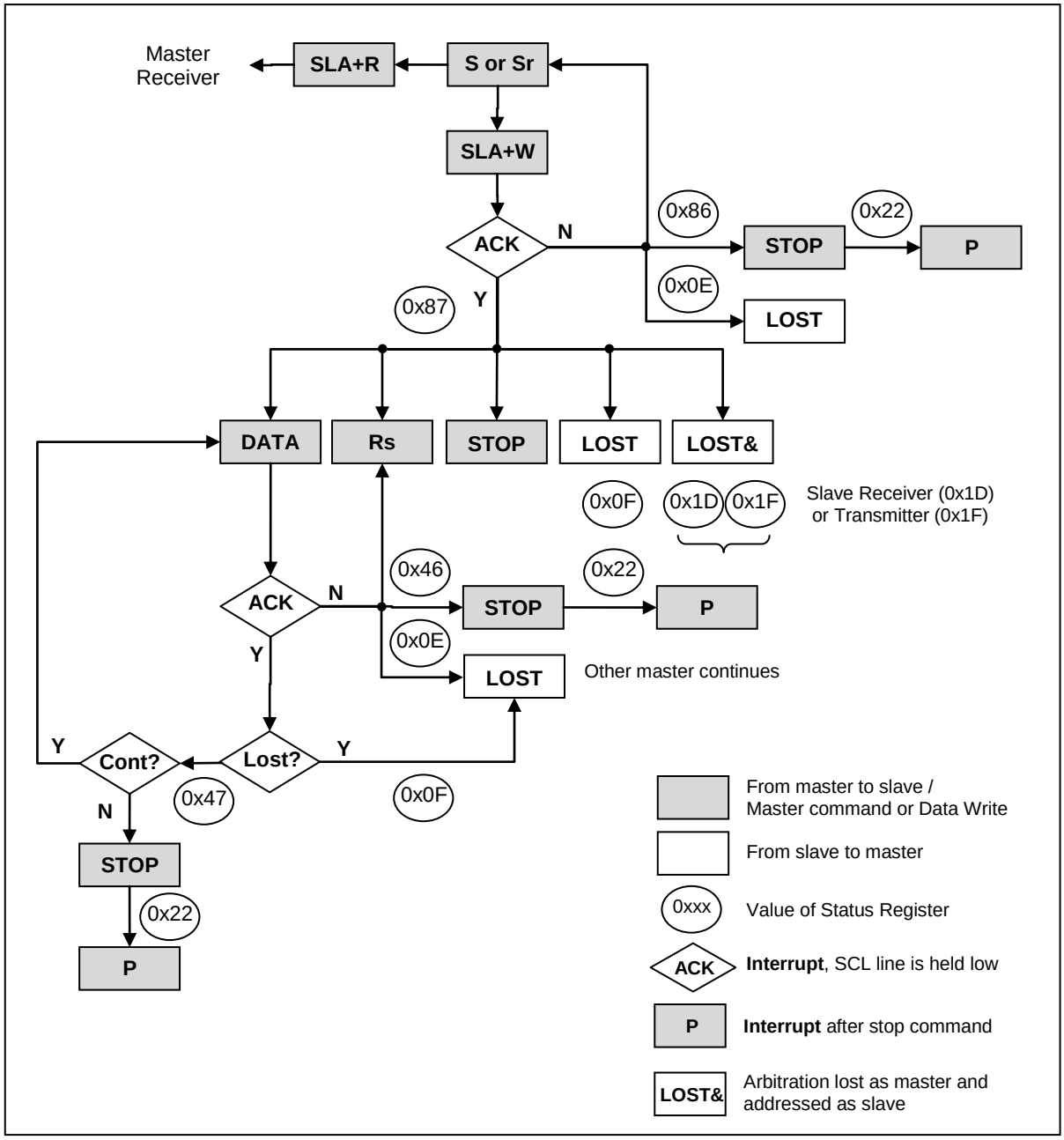


Figure 11-50 Formats and States in the Master Transmitter Mode

11.10.8.2 Master Receiver

To operate I^2C in master receiver, follow the recommended steps below.

1. Enable I^2C by setting IICEN bit in I2CMR. This provides main clock to the peripheral.
2. Load SLA+R into the I2CDR where SLA is address of slave device and R is transfer direction from the viewpoint of the master. For master receiver, R is '1'. Note that I2CDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CSCLLR and I2CSCLHR for the Low and High period of SCL line.
4. Configure the I2CSDAHR to decide when SDA changes value from falling edge of SCL. If SDA should change in the middle of SCL LOW period, load half the value of I2CSCLLR to the I2CSDAHR.
5. Set the START bit in I2CMR. This transmits a START condition. And also configure how to handle interrupt and ACK signal. When the START bit is set, 8-bit data in I2CDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9^{th} high period of SCL. If the master gains bus mastership, I^2C generates GCALL interrupt regardless of the reception of ACK from the slave device. When I^2C loses bus mastership during arbitration process, the MLOST bit in I2CSR is set, and I^2C waits in idle state or can be operate as an addressed slave. To operate as a slave when the MLSOT bit in I2CSR is set, the ACKEN bit in I2CMR must be set and the received 7-bit address must equal to the SLA bits in I2CSAR. In this case I^2C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I^2C holds the SCL LOW. This is because to decide whether I^2C continues serial transfer or stops communication. The following steps continue assuming that I^2C does not lose mastership during first data transfer.

   I^2C (Master) can choose one of the following cases according to the reception of ACK signal from slave.

   1) Master receives ACK signal from slave, so continues data transfer because slave can prepare and transmit more data to master. Configure ACKEN bit in I2CMR to decide whether I^2C ACKnowledges the next data to be received or not.
   2) Master stops data transfer because it receives no ACK signal from slave. In this case, set the STOP bit in I2CMR.
   3) Master transmits repeated START condition due to no ACK signal from slave. In this case, load SLA+R/W into the I2CDR and set START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR and if transfer direction bit is '0' go to master transmitter section.

7. 1-Byte of data is being received.
8. This is ACK signal processing stage for data packet transmitted by slave. I^2C holds the SCL LOW. When 1-Byte of data is received completely, I^2C generates TEND interrupt.

   I^2C can choose one of the following cases according to the RXACK flag in I2CSR.

   1) Master continues receiving data from slave. To do this, set ACKEN bit in I2CMR to ACKnowledge the next data to be received.
   2) Master wants to terminate data transfer when it receives next data by not generating ACK signal. This can be done by clearing ACKEN bit in I2CMR.
   3) Because no ACK signal is detected, master terminates data transfer. In this case, set the STOP bit in I2CMR.
   4) No ACK signal is detected, and master transmits repeated START condition. In this case,

load SLA+R/W into the I2CDR and set the START bit in I2CMR.

After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) and 2), move to step 7. In case of 3), move to step 9 to handle STOP interrupt. In case of 4), move to step 6 after transmitting the data in I2CDR, and if transfer direction bit is '0' go to master transmitter section.

9. This is the final step for master receiver function of I^2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I^2C enters idle state.

The processes described above for master receiver operation of I^2C can be depicted as the following figure.

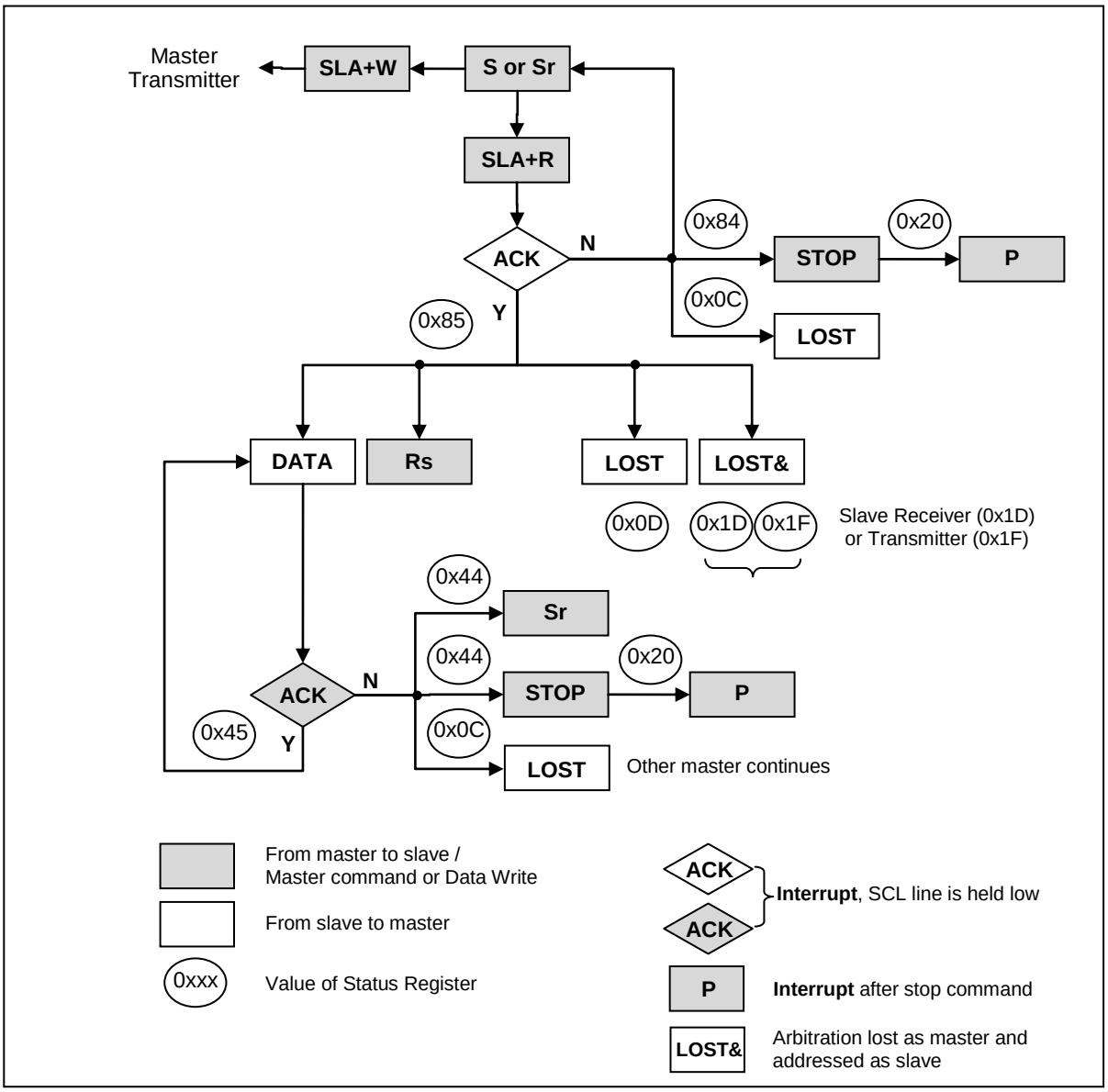


Figure 11-51 Formats and States in the Master Receiver Mode

11.10.8.3 Slave Transmitter

To operate I^2C in slave transmitter, follow the recommended steps below.

1. If the main operating clock (SCLK) of the system is slower than that of SCL, load value 0x00 into I2CSDAHR to make SDA change within one system clock period from the falling edge of SCL. Note that the hold time of SDA is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CSDAHR. When the hold time of SDA is longer than the period of SCLK, I^2C (slave) cannot transmit serial data properly.
2. Enable I^2C by setting IICEN bit and INTEN bit in I2CMR. This provides main clock to the peripheral.
3. When a START condition is detected, I^2C receives one byte of data and compares it with SLA bits in I2CSAR. If the GCALLEN bit in I2CSAR is enabled, I^2C compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLA bits in I2CSAR, I^2C enters idle state ie, waits for another START condition. Else if the address equals to SLA bits and the ACKEN bit is enabled, I^2C generates SSEL interrupt and the SCL line is held LOW. Note that even if the address equals to SLA bits, when the ACKEN bit is disabled, I^2C enters idle state. When SSEL interrupt occurs, load transmit data to I2CDR and write arbitrary value to I2CSR to release SCL line.
5. 1-Byte of data is being transmitted.
6. In this step, I^2C generates TEND interrupt and holds the SCL line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases.

   1) No ACK signal is detected and I^2C waits STOP or repeated START condition.
   2) ACK signal from master is detected. Load data to transmit into I2CDR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) move to step 7 to terminate communication. In case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move step 4.
7. This is the final step for slave transmitter function of I^2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I^2C enters idle state.

The next figure shows flow chart for handling slave transmitter function of I^2C.

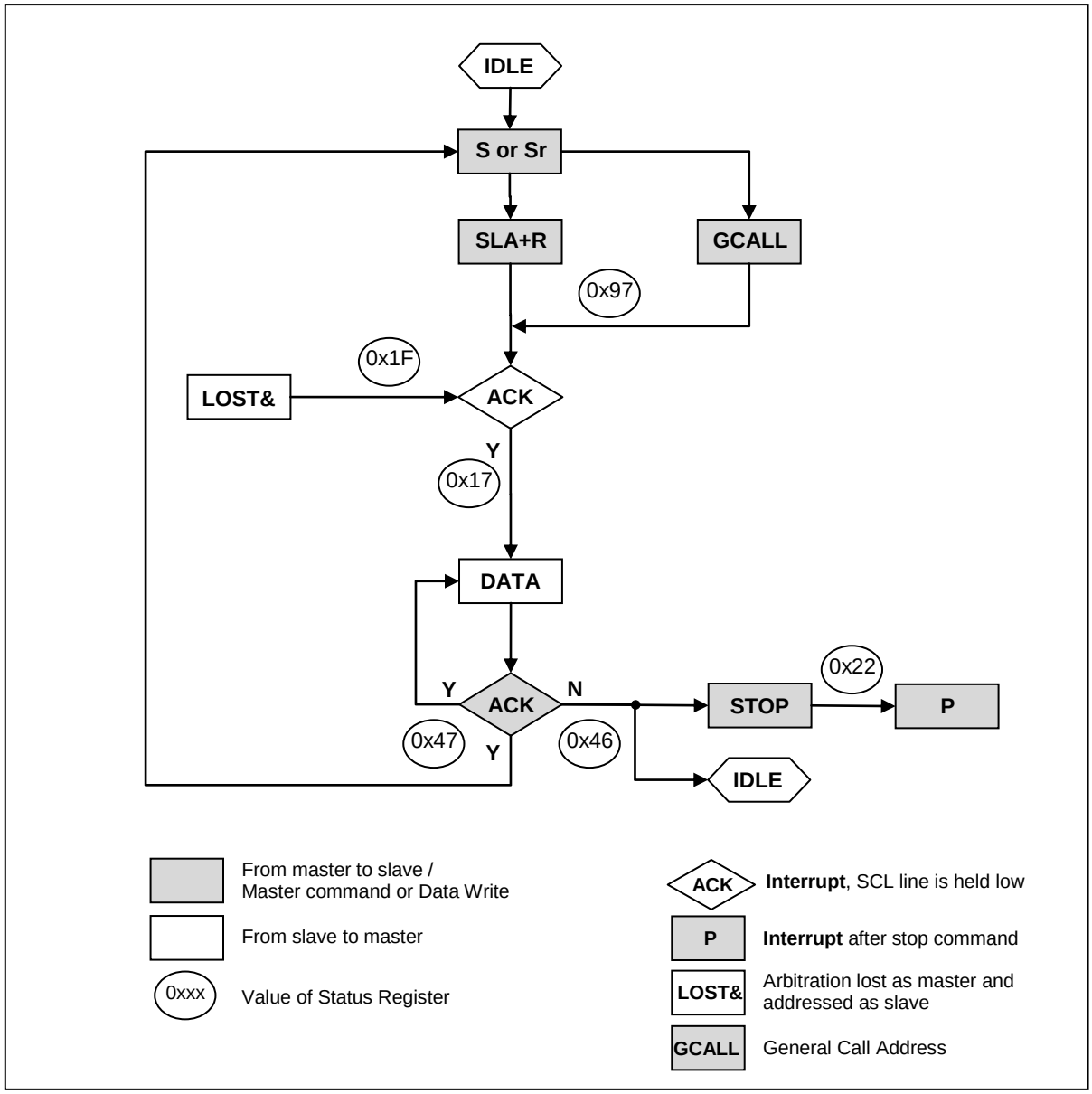


Figure 11-52 Formats and States in the Slave Transmitter Mode

11.10.8.4 Slave Receiver

To operate I^2C in slave receiver, follow the recommended steps below.

1. If the main operating clock (SCLK) of the system is slower than that of SCL, load value 0x00 into I2CSDAHR to make SDA change within one system clock period from the falling edge of SCL. Note that the hold time of SDA is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CSDAHR. When the hold time of SDA is longer than the period of SCLK, I^2C (slave) cannot transmit serial data properly.
2. Enable I^2C by setting IICEN bit and INTEN bit in I2CMR. This provides main clock to the peripheral.
3. When a START condition is detected, I^2C receives one byte of data and compares it with SLA bits in I2CSAR. If the GCALLEN bit in I2CSAR is enabled, I^2C compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLA bits in I2CSAR, I^2C enters idle state ie, waits for another START condition. Else if the address equals to SLA bits and the ACKEN bit is enabled, I^2C generates SSEL interrupt and the SCL line is held LOW. Note that even if the address equals to SLA bits, when the ACKEN bit is disabled, I^2C enters idle state. When SSEL interrupt occurs and I^2C is ready to receive data, write arbitrary value to I2CSR to release SCL line.
5. 1-Byte of data is being received.
6. In this step, I^2C generates TEND interrupt and holds the SCL line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases.

   1) No ACK signal is detected (ACKEN=0) and I^2C waits STOP or repeated START condition.
   2) ACK signal is detected (ACKEN=1) and I^2C can continue to receive data from master.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) move to step 7 to terminate communication. In case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move step 4.
7. This is the final step for slave receiver function of I^2C, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, I^2C enters idle state.

The process can be depicted as following figure when I^2C operates in slave receiver mode.

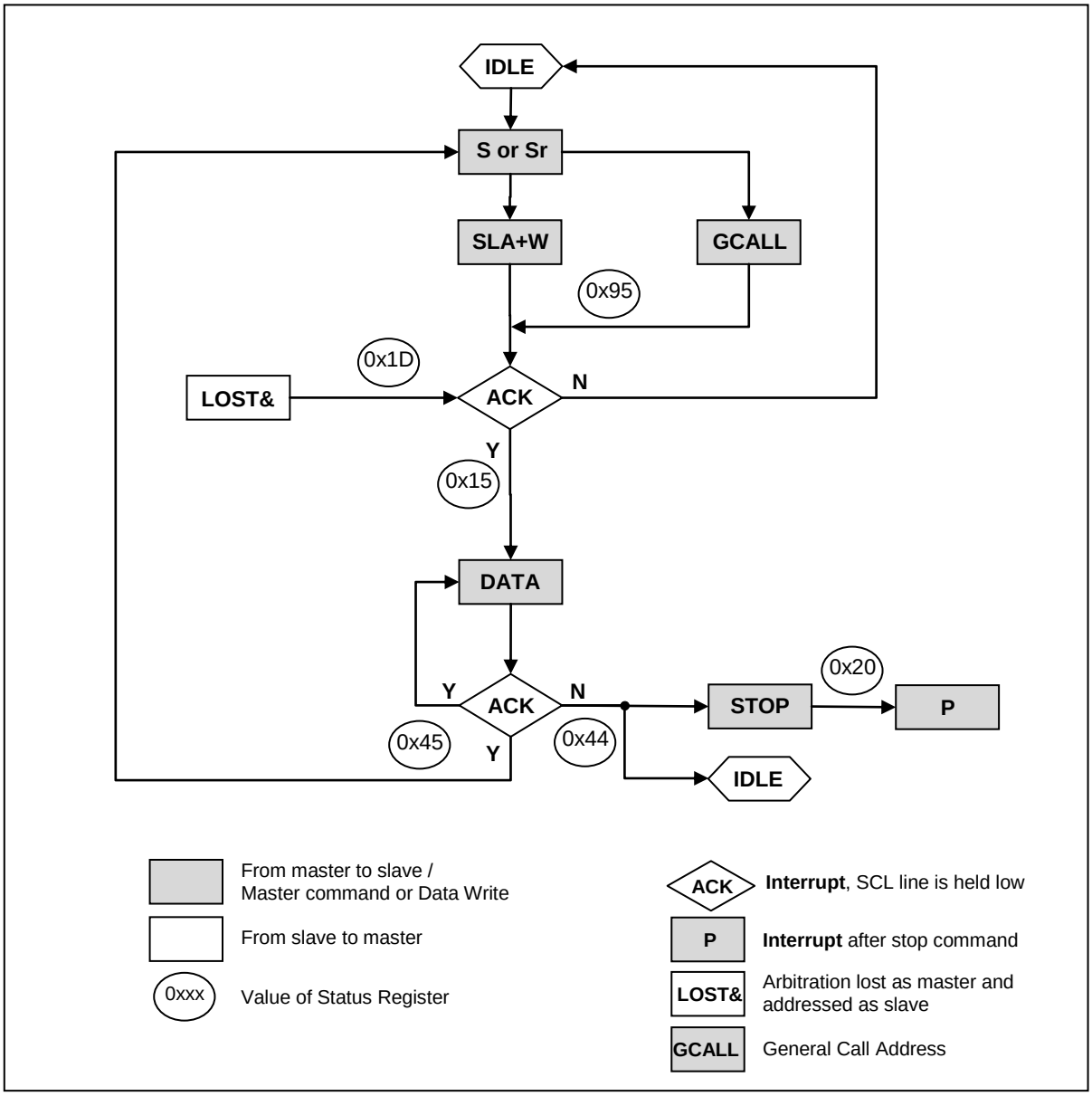


Figure 11-53 Formats and States in the Slave Receiver Mode

11.10.9 Register Map

Name	Address	Dir	Default	Description
I2CMR	$9C_H$	R/W	00_H	I^2C Mode Control Register
I2CSR	$9D_H$	R	00_H	I^2C Status Register
I2CSCLLR	$9E_H$	R/W	$3F_H$	SCL Low Period Register
I2CSCLHR	$9F_H$	R/W	$3F_H$	SCL High Period Register
I2CSDAHR	$A3_H$	R/W	01_H	SDA Hold Time Register
I2CDR	$A5_H$	R/W	FF_H	I^2C Data Register
I2CSAR	$A6_H$	R/W	00_H	I^2C Slave Address Register
I2CSAR1	$A7_H$	R/W	00_H	I^2C Slave Address Register 1

Table 11-19 Register map of I2C

11.10.10 I²C Register description

I²C Registers are composed of I²C Mode Control Register (I2CMR), I²C Status Register (I2CSR), SCL Low Period Register (I2CSCLLR), SCL High Period Register (I2CSCLHR), SDA Hold Time Register (I2CSDAHR), I²C Data Register (I2CDR), and I²C Slave Address Register (I2CSAR).

11.10.11 Register description for I²C

I2CMR (I²C Mode Control Register) **$9C_H$**

7	6	5	4	3	2	1	0
IIF	IICEN	RESET	INTEN	ACKEN	MASTER	STOP	START
R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

Initial value : 00_H

IIF This is interrupt flag bit.
- 0 No interrupt is generated or interrupt is cleared
- 1 An interrupt is generated

IICEN Enable I²C Function Block (by providing clock)
- 0 I²C is inactive
- 1 I²C is active

RESET Initialize internal registers of I²C.
- 0 No operation
- 1 Initialize I²C, auto cleared

INTEN Enable interrupt generation of I²C.
- 0 Disable interrupt, operates in polling mode
- 1 Enable interrupt

ACKEN Controls ACK signal generation at ninth SCL period.
Note) ACK signal is output (SDA=0) for the following 3 cases.
When received address packet equals to SLA bits in I2CSAR
When received address packet equals to value 0x00 with GCALL enabled
When I²C operates as a receiver (master or slave)
- 0 No ACK signal is generated (SDA=1)
- 1 ACK signal is generated (SDA=0)

MASTER Represent operating mode of I²C
- 0 I²C is in slave mode
- 1 I²C is in master mode

STOP When I²C is master, generates STOP condition.
- 0 No operation
- 1 STOP condition is to be generated

START When I²C is master, generates START condition.
- 0 No operation
- 1 START or repeated START condition is to be generated

I2CSR (I^2C Status Register) **$9D_H$**

7	6	5	4	3	2	1	0
GCALL	TEND	STOP	SSEL	MLOST	BUSY	TMODE	RXACK
R	R	R	R	R	R	R	R

Initial value : 00_H

GCALL This bit has different meaning depending on whether I^2C is master or slave. Note 1)
When I^2C is a master, this bit represents whether it received AACK (Address ACK) from slave.
When I^2C is a slave, this bit is used to indicate general call.
0 No AACK is received (Master mode)
1 AACK is received (Master mode)
0 Received address is not general call address (Slave mode)
1 General call address is detected (Slave mode)

TEND This bit is set when 1-Byte of data is transferred completely. Note 1)
0 1 byte of data is not completely transferred
1 1 byte of data is completely transferred

STOP This bit is set when STOP condition is detected. Note 1)
0 No STOP condition is detected
1 STOP condition is detected

SSEL This bit is set when I^2C is addressed by other master. Note 1)
0 I^2C is not selected as slave
1 I^2C is addressed by other master and acts as a slave

MLOST This bit represents the result of bus arbitration in master mode. Note 1)
0 I^2C maintains bus mastership
1 I^2C has lost bus mastership during arbitration process

BUSY This bit reflects bus status.
0 I^2C bus is idle, so any master can issue a START condition
1 I^2C bus is busy

TMODE This bit is used to indicate whether I^2C is transmitter or receiver.
0 I^2C is a receiver
1 I^2C is a transmitter

RXACK This bit shows the state of ACK signal.
0 No ACK is received
1 ACK is generated at ninth SCL period

Note 1) These bits can be source of interrupt.

When an I^2C interrupt occurs except for STOP interrupt, the SCL line is hold LOW. To release SCL, write arbitrary value to I2CSR. When I2CSR is written, the TEND, STOP, SSEL, LOST, RXACK bits are cleared.

I2CSCLLR (SCL Low Period Register) $9E_H$

7	6	5	4	3	2	1	0
SCLL7	SCLL6	SCLL5	SCLL4	SCLL3	SCLL2	SCLL1	SCLL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : $3F_H$

SCLL[7:0] This register defines the LOW period of SCL when I^2C operates in master mode. The base clock is SCLK, the system clock, and the period is calculated by the formula : $t_{SCLK} \times (4 \times SCLL + 1)$ where t_{SCLK} is the period of SCLK.

I2CSCLHR (SCL High Period Register) $9F_H$

7	6	5	4	3	2	1	0
SCLH7	SCLH6	SCLH5	SCLH4	SCLH3	SCLH2	SCLH1	SCLH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : $3F_H$

SCLH[7:0] This register defines the HIGH period of SCL when I^2C operates in master mode. The base clock is SCLK, the system clock, and the period is calculated by the formula : $t_{SCLK} \times (4 \times SCLH + 3)$ where t_{SCLK} is the period of SCLK.

So, the operating frequency of I^2C in master mode (fI2C) is calculated by the following equation.

$$fI2C = \frac{1}{tSCLK \times (4\,(SCLL + SCLH) + 4)}$$

I2CSDAHR (SDA Hold Time Register) $A3_H$

7	6	5	4	3	2	1	0
SDAH7	SDAH6	SDAH5	SDAH4	SDAH3	SDAH2	SDAH1	SDAH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 01_H

SDAH[7:0] This register is used to control SDA output timing from the falling edge of SCL. Note that SDA is changed after $t_{SCLK} \times$ SDAH. In master mode, load half the value of SCLL to this register to make SDA change in the middle of SCL. In slave mode, configure this register regarding the frequency of SCL from master. The SDA is changed after $t_{SCLK} \times (SDAH + 1)$. So, to insure normal operation in slave mode, the value $t_{SCLK} \times (SDAH + 1)$ must be smaller than the period of SCL.

I2CDR (I^2C Data Register) $A5_H$

7	6	5	4	3	2	1	0
ICD7	ICD6	ICD5	ICD4	ICD3	ICD2	ICD1	ICD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : FF_H

ICD[7:0] When I^2C is configured as a transmitter, load this register with data to be transmitted. When I^2C is a receiver, the received data is stored into this register.

I2CSAR (I²C Slave Address Register) A6H

7	6	5	4	3	2	1	0
SLA07	SLA06	SLA05	SLA04	SLA03	SLA02	SLA01	SLA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SLA0[7:1] These bits configure the slave address of this I²C module when I²C operates in slave mode.

GCALLEN This bit decides whether I²C allows general call address or not when I²C operates in slave mode.

0 Ignore general call address

1 Allow general call address

I2CSAR1 (I²C Slave Address Register 1) A7H

7	6	5	4	3	2	1	0
SLA17	SLA16	SLA15	SLA14	SLA13	SLA12	SLA11	SLA10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SLA1[7:1] These bits configure the slave address of this I²C module when I²C operates in slave mode.

GCALLEN This bit decides whether I²C allows general call address or not when I²C operates in slave mode.

0 Ignore general call address

1 Allow general call address

12. POWER MANAGEMENT

12.1 Overview

MC96FR332A supports two kinds of power saving modes, SLEEP and STOP. In these modes, the program execution is stopped. There's also BOD mode caused by voltage drop, which is almost the same as STOP mode.

12.2 PERIPHERAL OPERATION IN SLEEP/STOP/BOD MODE

Peripheral	SLEEP Mode	STOP Mode	BOD mode
CPU	ALL CPU Operations are disabled	ALL CPU Operations are disabled	ALL CPU Operations are disabled
RAM	Retain	Retain	Retain
Basic Interval Timer	Operates Continuously	Stop	Stop
Watch Dog Timer	Operates Continuously	Stop	Stop
Timer0~3	Operates Continuously	Halted	Halted
KEYSCAN	Operates Continuously	Stop	Stop
Carrier Generator	Operates Continuously	Stop	Stop
USART	Operates Continuously	Stop	Stop
BOD	Enabled	Disabled	Enabled
Main OSC (1~12MHz)	Oscillation	Stop	Stop
I/O Port	Retain	Retain	Retain or Input pull-up mode
Control Register	Retain	Retain	Retain
Address / Data Bus	Retain	Retain	Retain
Release Method	By RESET, all Interrupts	By RESET, Key interrupt, External Interrupt, UART by RX, PCI	By Reset, By power rise detect

Table 12-1 CPU and peripherals state in power saving modes

12.3 SLEEP mode

To enter SLEEP mode, write 01_H to Power Control Register(PCON, 87_H). In this mode, only the CPU halts and other peripherals including main oscillator operate normally. SLEEP mode can be released by a reset or interrupt. When SLEEP mode is released by a reset, all internal logics is initialized.

The following example shows the way to enter SLEEP mode.

```
(ex) MOV   PCON, #0000_0001b    ; Enter SLEEP mode : set bits of STOP and SLEEP Control register (PCON)
```

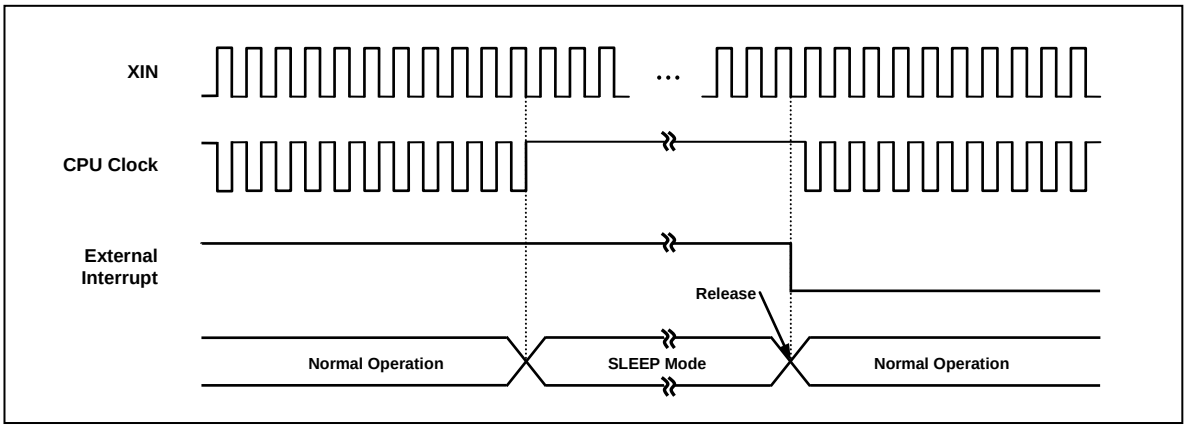


Figure 12-1 Wake-up from SLEEP mode by an interrupt

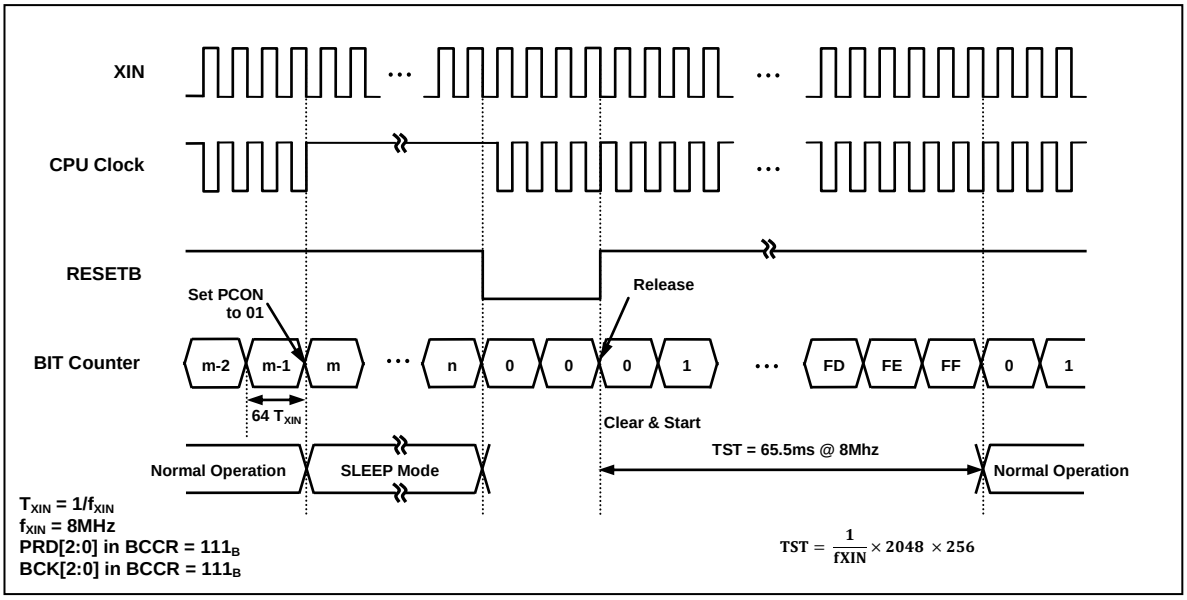


Figure 12-2 SLEEP mode release by an external reset

12.4 STOP mode

The least power consuming state called STOP mode is initiated by writing 03_H to Power Control Register (PCON, 87_H). In STOP mode, all analog and digital blocks including main oscillator stop operation. The analog block VDC also enters its own stop mode and BOD is auto-disabled, so power consumption is radically reduced. All registers value or RAM data are reserved.

STOP mode release is done by a reset or external pin interrupt request. When a reset is detected in STOP mode, the device is initialized, so all registers except for BODR register is reset to initial state. BODR register may or may not be initialized ‘cause it has reset flags which are affected only by it’s specific reset source. There’re three kinds of reset sources which can be used to release STOP mode, power on reset(nPOR), external reset(P20) and BOD reset. As main oscillator stops oscillation in STOP mode, WDT reset cannot be generated.

When a reset or interrupt occurs in STOP mode, the clock control logic makes system clock active when a pre-defined time is passed. This is needed because the oscillator needs oscillation stability time. So we recommend to ensure at least 20ms of stability time by configuring BCCR register before

entering STOP mode. The oscillation stability time can be calculated by the overflow period of BIT counter. NOTE

NOTE The oscillation stability time is up to the characteristic of an oscillator or a resonator connected to the device. So the 20ms of recommended stability time is not absolute.

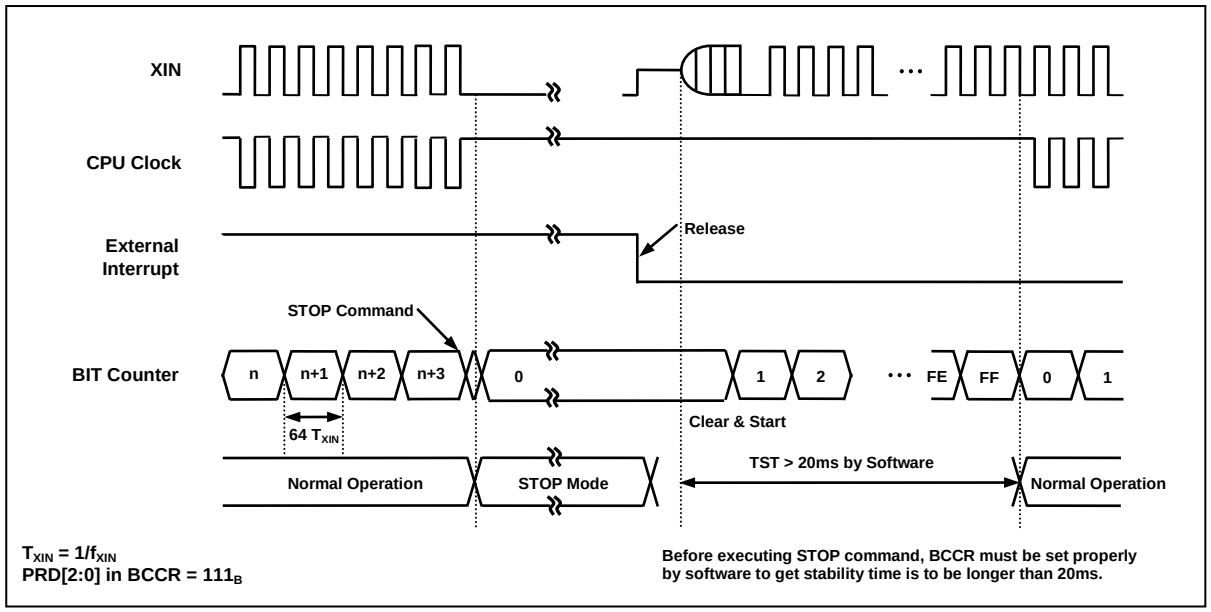


Figure 12-3 Wake-up from STOP mode by an interrupt

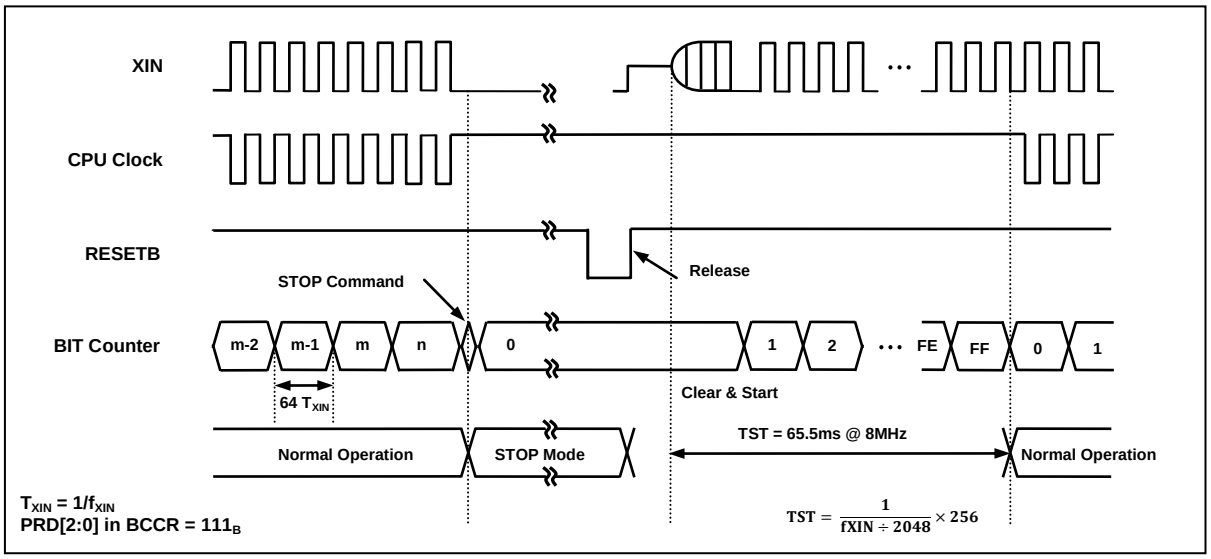


Figure 12-4 STOP mode release by an external reset

MC96FR332A acts in a little different manner after it awakes from STOP mode according to the external power condition after wake-up event.

In stop mode, the main oscillator is halted and any internal peripheral does not operate normally. So only an interrupt from external world can wake the device up from STOP mode. These kinds of interrupts are an external interrupt, key input or MISO(RXD) pin in UART mode. When an interrupt is detected, the wake-up logic enables BOD to check the external voltage level. If the voltage level is

higher than the BOD stop level(=$V_{BODOUT0}$), the device wakes up normally to resume program execution. Otherwise if the checked voltage level is below the $V_{BODOUT0}$, it remains in STOP mode. This continues unless the power level is recovered. Thereafter the device wakes up by another interrupt when the power level detected by BOD is sufficient. In this case, BOD reset is generated to initialize the device.

To wake up by an interrupt and accept interrupt request, the EA bit in IE register and the individual interrupt enable bit INTnE in IEx registers should be set.

12.5 BOD mode

When BOD is enabled and the external voltage drops below $V_{BODOUT0}$, the device enters BOD mode instantaneously. In BOD mode, all analog and digital blocks except for BOD stops operating. The internal state of the device is almost the same as in STOP mode. But there are 3 different points as follows. First, on entering BOD mode, the I/O ports can be set input ports with pull-up registers on if PxBPC registers are not altered from reset value. Second, BOD mode can only be released by voltage rise detected by BOD. And after mode exit, the device is initialized by a BOD reset event. Third, because BOD is enabled to detect voltage variation, the current consumption is larger than STOP mode of operation, typically maximum 40uA of current is consumed. Except the three different points described above, BOD mode is the same as STOP mode.

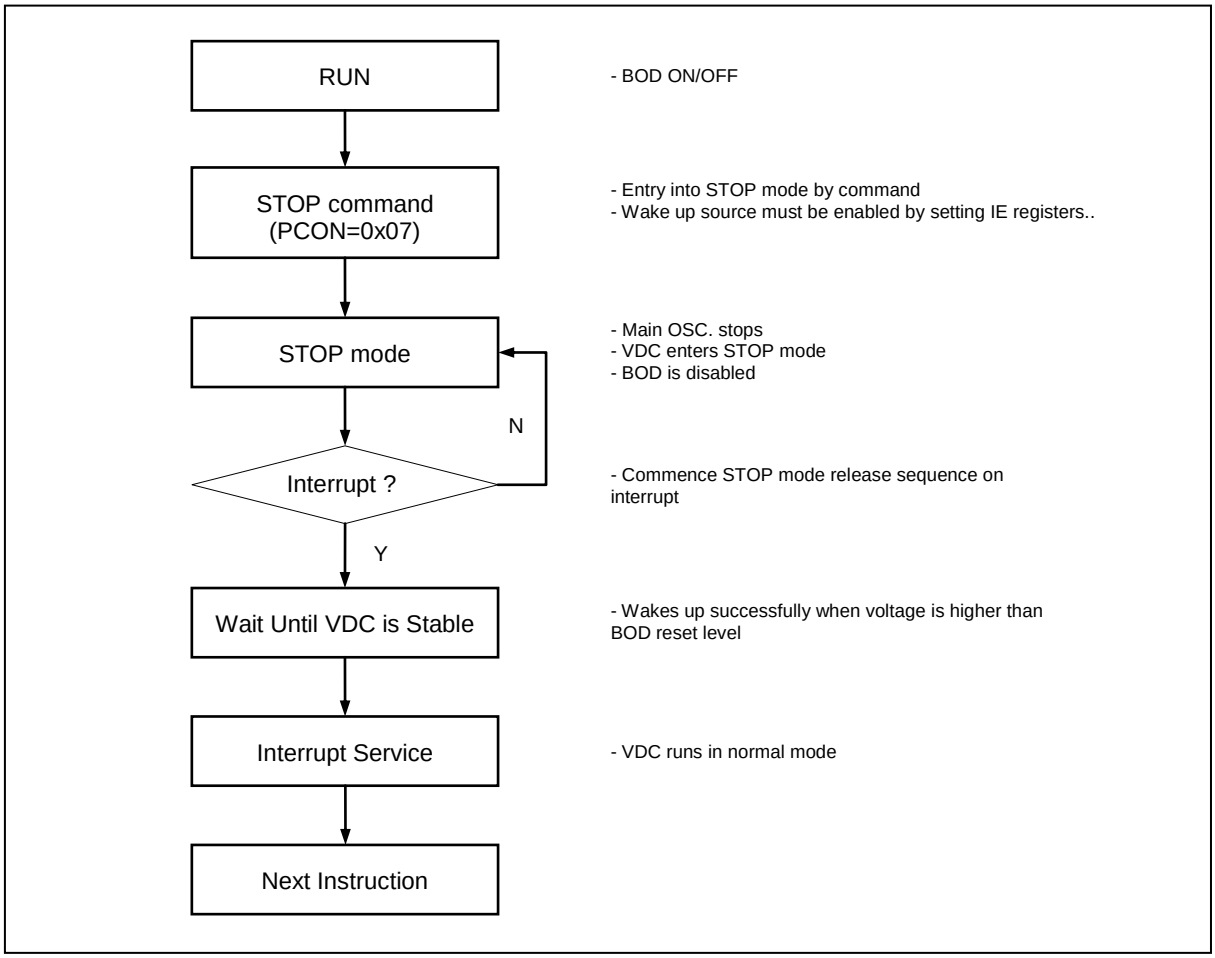


Figure 12-5 Entry into STOP mode and Release sequence

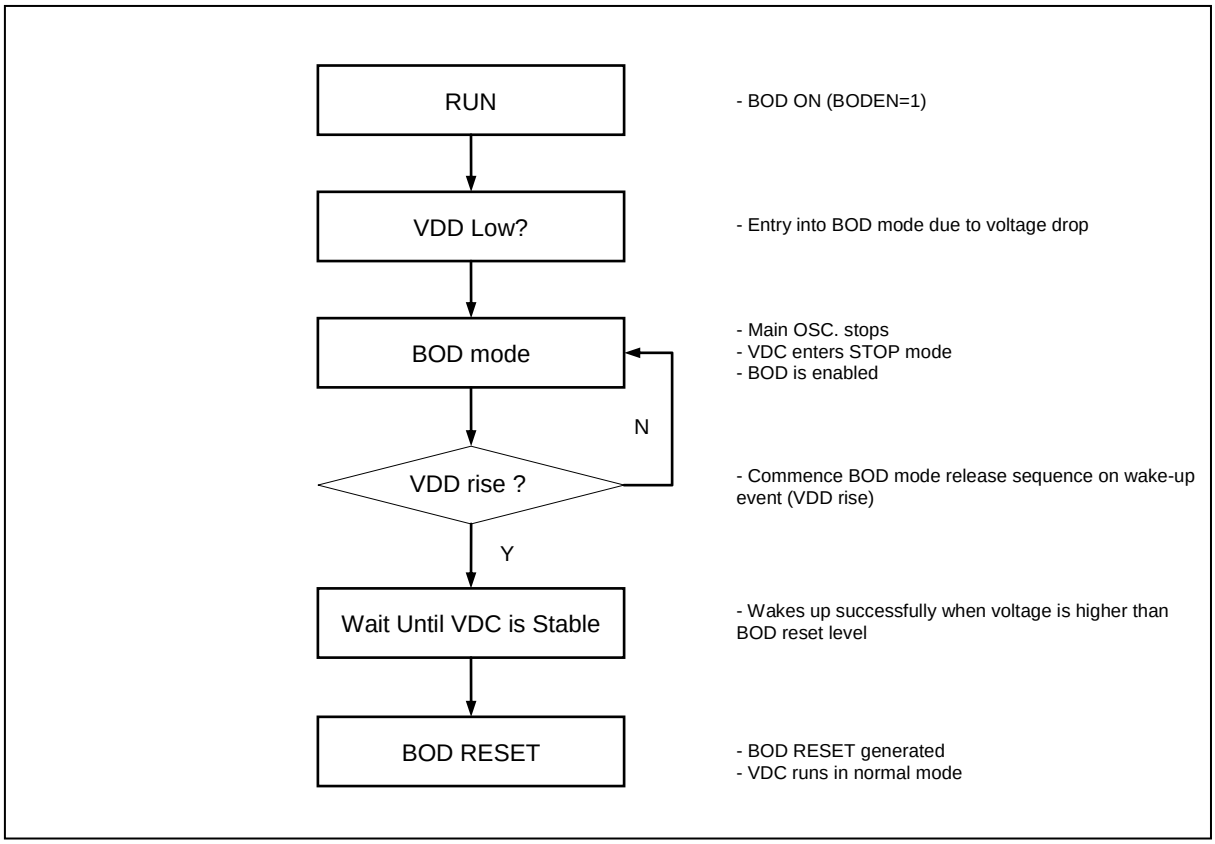


Figure 12-6 Entry into BOD mode and Release sequence

12.6 Register Map

Name	Address	Dir	Default	Description
PCON	87_H	R/W	00_H	Power Control Register

Table 12-2 Register Map of Power Control Logic

12.7 Register Description

PCON (Power Control Register) **87_H**

7	6	5	4	3	2	1	0
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

SLEEP mode

01_H Enters SLEEP mode

STOP mode

03_H Enters STOP mode

NOTE 1. Write PCON register 01_H or 03_H to enter SLEEP or STOP mode.
2. When mode exit from STOP or SLEEP is done successfully, the PCON register is auto-cleared.

13. RESET

13.1 Overview

When a reset event occurs, the CPU immediately stops whatever it is doing and all internal logics except for BODR register is initialized. The external reset pin(P20) shares normal I/O pin and the functionality is defined by fuse configuration(FUSE_CONF register). The hardware configuration right after reset event is as follows.

On Chip Hardware	Initial Value
Program Counter (PC)	0000_H
Accumulator	00_H
Stack Pointer (SP)	07_H
Peripheral Clocks	On
Control Registers	Refer to Peripheral Registers
Brown-Out Detector	Enabled on power-on-reset

Table 13-1 Internal status when a reset is asserted

13.2 Reset source

Reset can be caused by a power-on-reset (nPOR) event, configuration reset by software, watchdog overflow, voltage drop detection by BOD, OCD command, or by assertion of an external active-low reset pin. Five of reset sources except for power-on-reset can be configured whether to be used as a reset source or not.

-. External reset pin (P20) (Share with P20 pin. Active low)

-. Power-on reset (nPOR, Active low)

-. WDT overflow (when WDTEN is '1' and WDTRSON is '1')

-. Configuration reset by software (nCFGR)

-. BOD reset (when BODEN is '1') NOTE

-. OCD command (When debugger issues a command)

13.3 Block Diagram

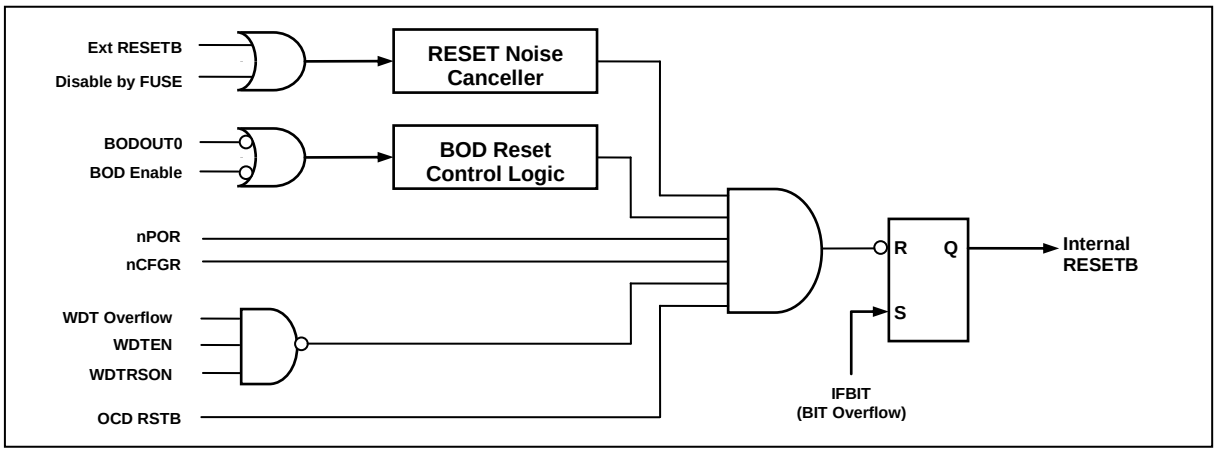


Figure 13-1 Block Diagram of Reset Circuit

NOTE Unlike other reset sources, BOD reset does not take place as soon as BODOUT0 goes HIGH(=voltage drops below BOD stop level). On detecting low voltage while the device is in normal run mode, the device enters BOD(STOP) mode first. And then by detecting voltage rise, the power control logic wakes the device up to give a reset signal.

Caution : When the device is in STOP mode by CPU command(PCON=0x03), the BOD cannot detect voltage drop because the BOD is disabled to reduce power consume. In this case, the BOD reset may be issued when a wake-up event or interrupt is generated with the external voltage below the BOD stop level.

13.4 Noise Canceller for External Reset Pin

A glitch-like or short pulse on external reset pin(P20) is ignored by the dedicated noise-canceller. To have an effect as a reset source, P20 port should be maintained low continuously at least 8us of time(T_{RNC}) in typical condition. The T_{RNC} may vary from 4.8us up to 13.8us according to the condition of manufacturing process.

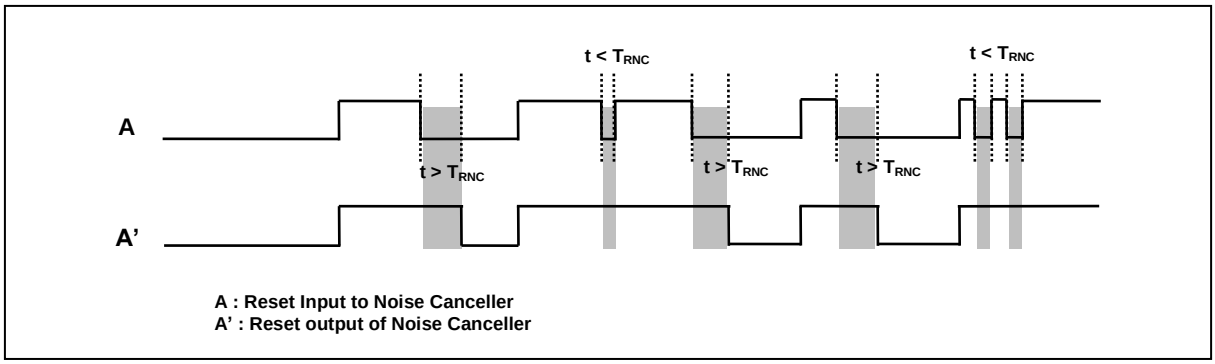


Figure 13-2 Noise Cancelling of External Reset Pin

13.5 Power-On-RESET

When power is initially applied to the MCU, or when the supply voltage drops below the V_{POR} level, the POR circuit will cause a reset condition. Owing to presence of POR, the external reset pin can be used as a normal I/O pin. Thus additional resistor and capacitor can be removed to be connected to reset pin.

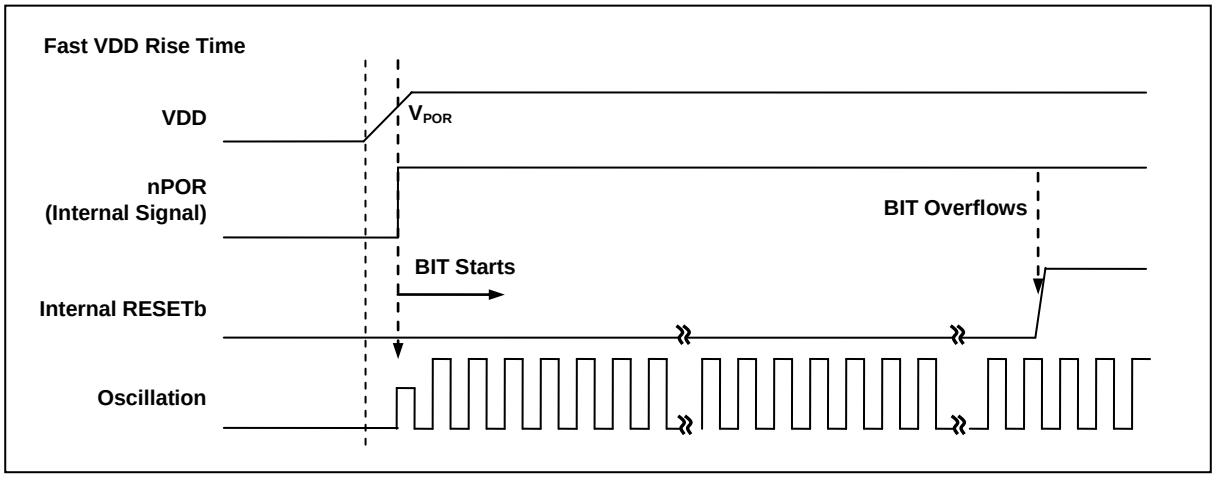


Figure 13-3 Reset Release Timing when Power is supplied (VDD Rises Rapidly)

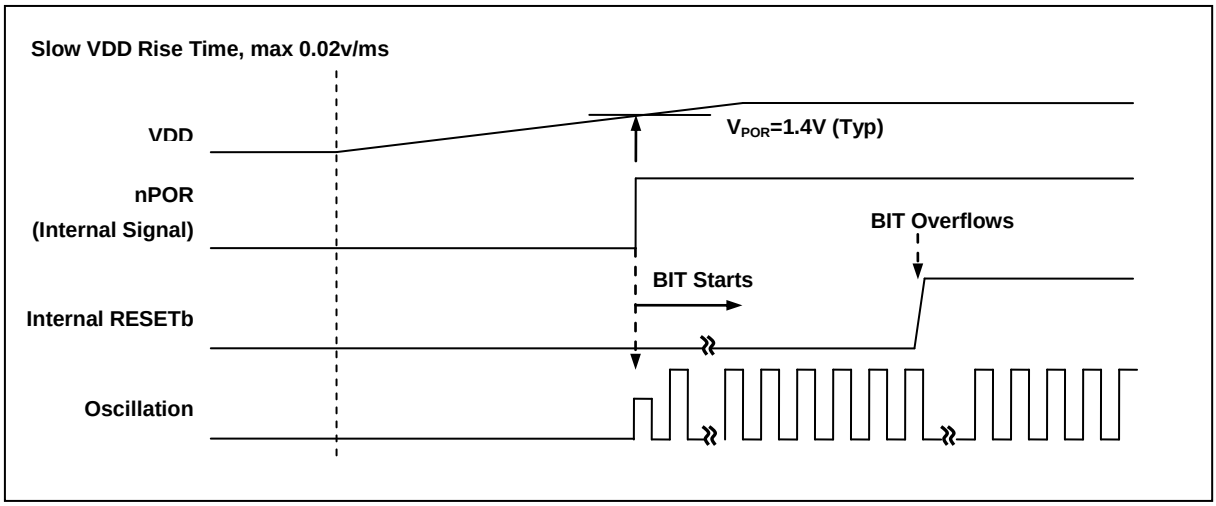


Figure 13-4 Reset Release Timing when Power is supplied (VDD Rises Slowly)

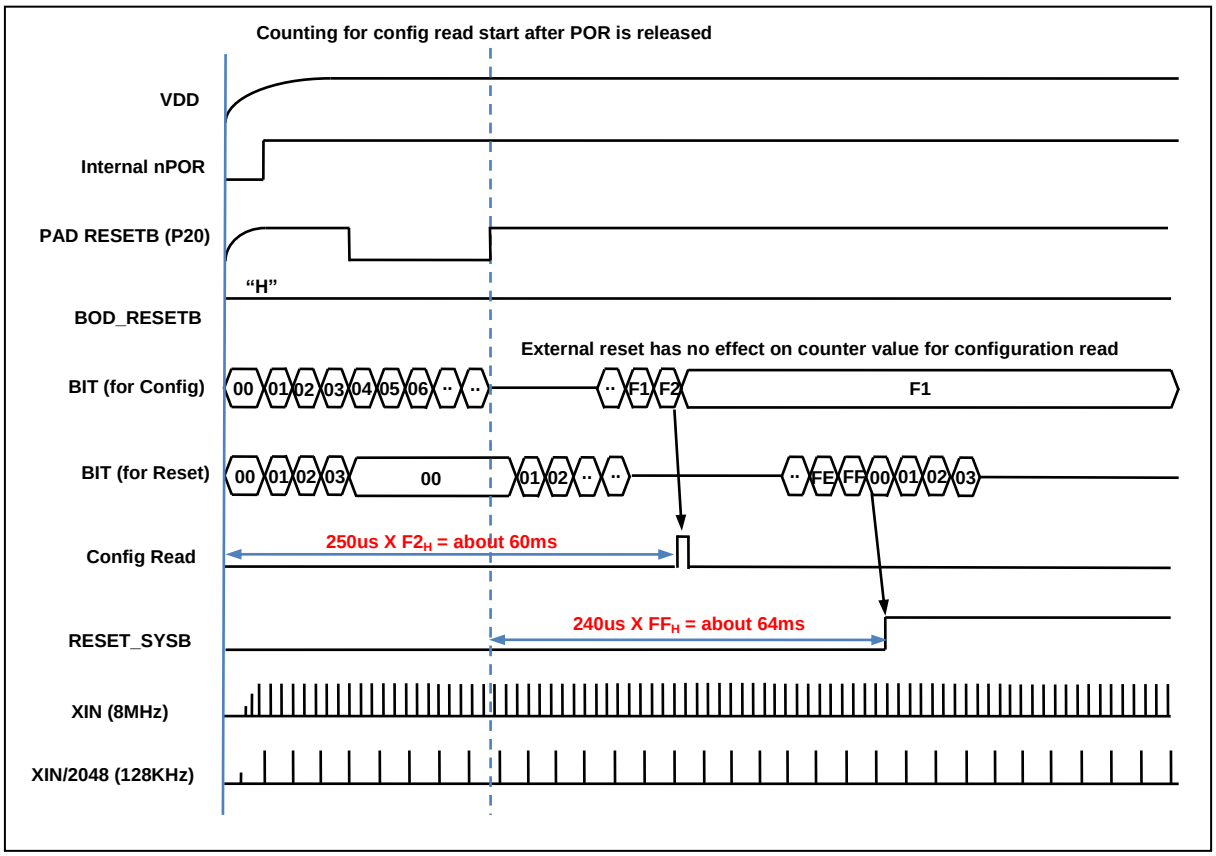


Figure 13-5 Fuse Configuration Value Read Timing after Power On (External 8MHz Clock)

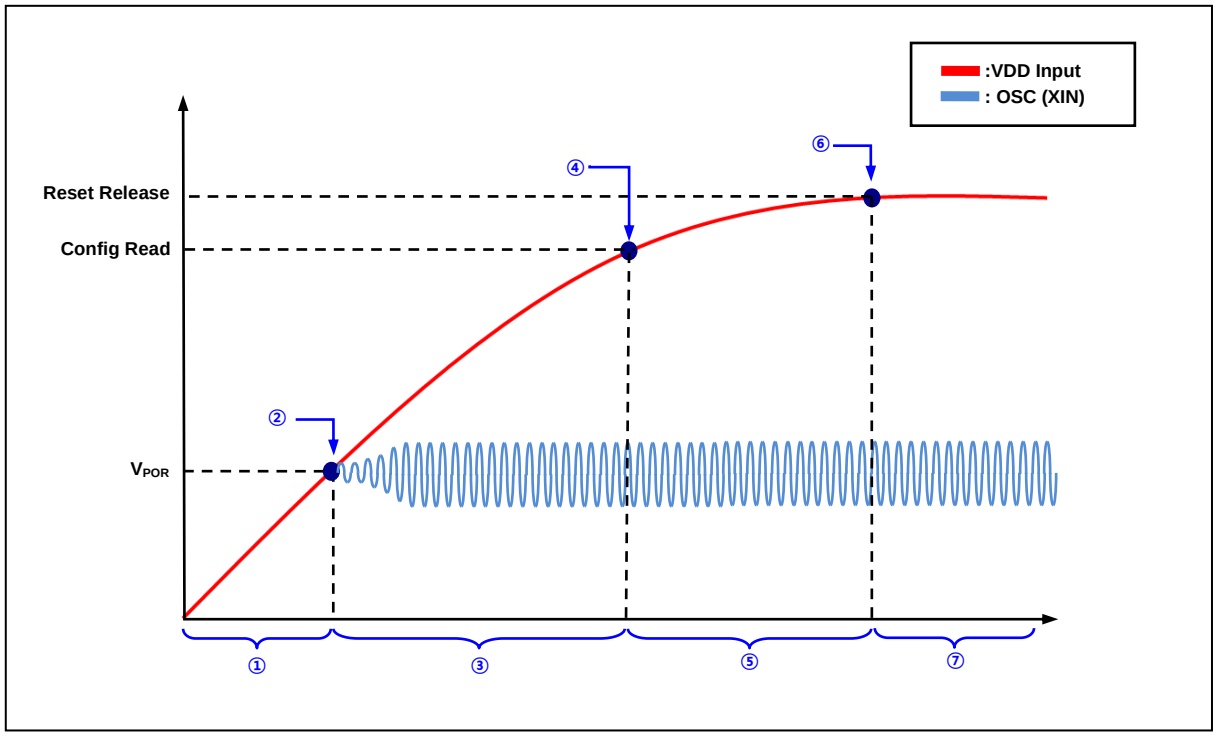


Figure 13-6 Operation according to Power Level

The above figure shows internal operation according to the voltage level and time. And the following table is short description about the figure.

Process	Description	Remarks
①	-POR	
②	-POR release point -Main OSC (Typically 8MHz) starts oscillation	-Around 1.4V ~ 1.6V
③	-(T_{XIN} X 2048) × $F2_H$ (60ms) delay section -VDD must rise above flash operating voltage	-T_{XIN} is period of XIN -Slew Rate >= 0.025V/ms
④	-Configuration value read point	-Around 1.5V ~ 1.6V -Config value is determined by writing option
⑤	-Rising section to reset release level	-64ms after power-on-reset or external reset is released
⑥	-Reset release point (BIT overflow)	-BIT is used to ensure oscillation stability time
⑦	-Normal operation	

Table 13-2 Power On Sequence

13.6 External RESETB Input

External reset pin is a Schmitt Trigger type input. External reset input should be asserted low at least for 8us(typically) for normal reset function when operating voltage and output of main oscillator are stable.

When the external reset input goes high, the internal reset is released after 64ms of stability time in case external clock frequency is 8MHz. For 5 clock periods from the point internal reset is released, an initialization procedure is performed. Thereafter the user program is executed from the address 0000_H.

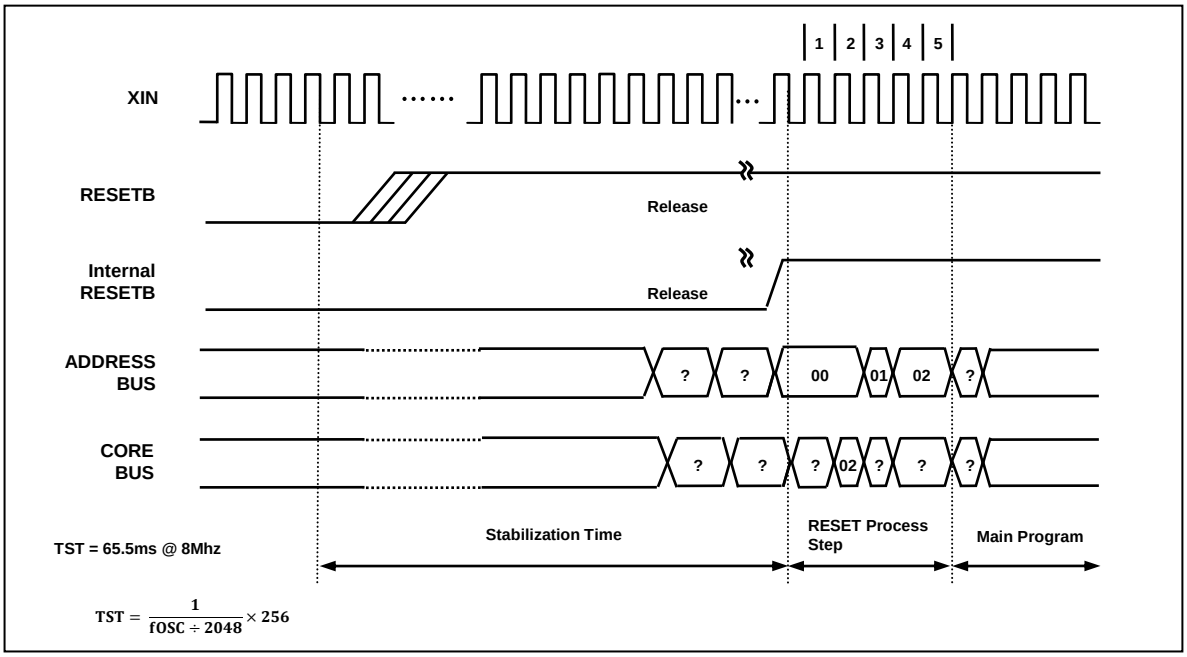


Figure 13-7 Reset procedure due to external reset input

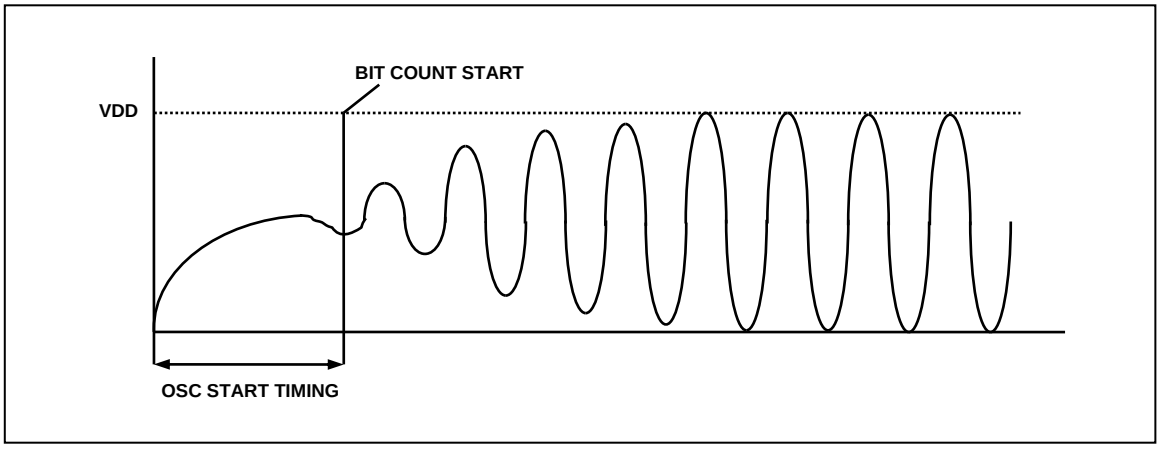


Figure 13-8 Example of oscillation

NOTE Oscillation start time does not belong to oscillation stability time.

13.7 Brown Out Detector

The MC96FR332A includes a system to protect against low voltage conditions in order to preserve memory contents and control MCU system states during supply voltage variations. The system is comprised of a power-on-reset(nPOR) and an BOD with 4 voltage level indicators. The BOD is enabled when BODEN in BODR is high. The BOD is auto-disabled upon entering STOP mode. This auto-disable function reduces operating power noticeably consumed by BOD itself.

The BOD has two main functions. One is to generate a BOD stop level(=BODOUT0) and the other is to indicate voltage levels above BOD stop level denoted by BODOUT1,2,3,4 each.

Remember that BOD itself does not generate a reset signal. When operating voltage drops below a pre-defined level($V_{BODOUT0}$), the BOD does not cause the whole system to be reset, but signals main chip to enter BOD(STOP) mode instantaneously. For more information about BOD stop, refer to section 12.4 POWER MANAGEMENT.

Besides BOD stop level, the BOD gives four low voltage warning flags to indicate to the user that the supply voltage is approaching, but is still above, the BOD stop level. These flags can be read through the BODSR register, also can be an interrupt source when BODIEN bit in BODSR is set.

Like external reset, BODOUT0 is also filtered by a dedicated noise cancelling logic. A pulse shorter than about 2us(typically) is ignored by the system in condition Vdd is between 1.75V and 5.5V.

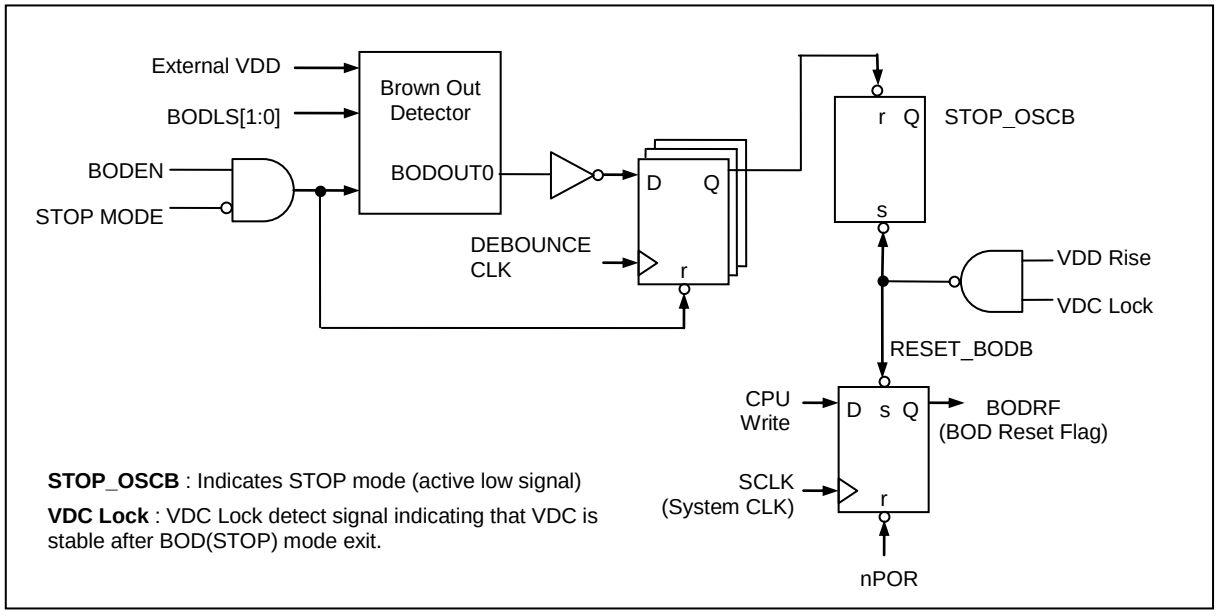


Figure 13-9 Block Diagram of BOD

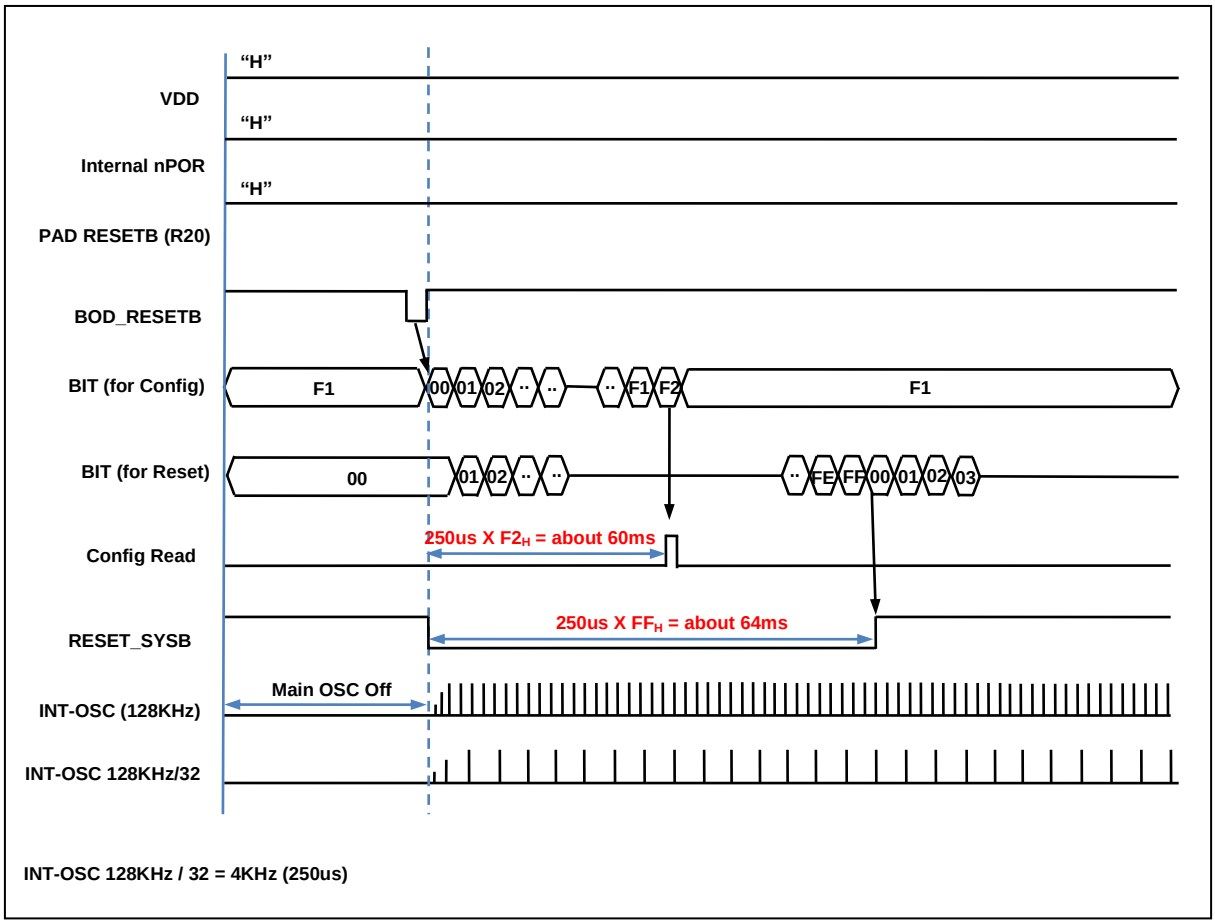


Figure 13-10 Configuration value read timing when BOD RESET is asserted

13.8 Register Map

Name	Address	Dir	Default	Description
BODR	86_H	R/W	81_H	BOD Control Register
BODSR	$8F_H$	R/W	00_H	BOD Status Register
CFGRR	$F9_H$	R/W	01_H	Configuration Reset Register

Table 13-3 Register Map of BOD

13.9 Register Description

BODR (BOD Control Register) 86_H

7	6	5	4	3	2	1	0
PORF	EXTRF	WDTRF	OCDRF	BODRF	BODLS1	BODLS0	BODEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 81_H

PORF Power-on reset or software reset event NOTE
- 0 No POR event detected after clear
- 1 POR occurred

EXTRF External Reset Event NOTE

0 No external reset detected after clear

1 External reset occurred

WDTRF Watchdog reset event NOTE

0 No WDT reset detected after clear

1 WDT reset occurred

OCDRF On-chip debugger reset event NOTE

0 No OCD reset detected after clear

1 OCD reset occurred

BODRF Brown-out detector reset event NOTE

0 No BOD reset detected after clear

1 BOD reset occurred

BODLS[1:0] Select bod flag level.

BODLS1	BODLS0	BOD flag level
0	0	BODOUT1 is bod flag.
0	1	BODOUT2 is bod flag.
1	0	BODOUT3 is bod flag.
1	1	BODOUT4 is bod flag.

BODEN Enables or disables BOD

0 Disable BOD

1 Enable BOD

NOTE To clear each reset flag, write '0' to associated bit position.

BODSR (BOD Status Register) $8F_H$

7	6	5	4	3	2	1	0
BODIF	-	-	-	BODOUT4	BODOUT3	BODOUT2	BODOUT1
R	-	-	-	R	R	R	R

Initial value : 00_H

BODIF BOD interrupt flag. To clear this flag, write '0' to this bit position.

0 BOD interrupt not requested

1 BOD interrupt requested

BODOUT4 VDD level indicator 4. After calibration, this flag turns on around 2.20V only when BODLS[1:0] = 11_B.

0 VDD level is higher than $V_{BODOUT4}$

1 VDD dropped below $V_{BODOUT4}$

BODOUT3 VDD level indicator 3. After calibration, this flag turns on around 2.10V only when BODLS[1:0] = 10_B.

0 VDD level is higher than $V_{BODOUT3}$

1 VDD dropped below $V_{BODOUT3}$

BODOUT2 VDD level indicator 2. After calibration, this flag turns on around 2.00V only when BODLS[1:0] = 01_B.

0 VDD level is higher than $V_{BODOUT2}$

1 VDD dropped below $V_{BODOUT2}$

BODOUT1 VDD level indicator 1. After calibration, this flag turns on around 1.90V only when BODLS[1:0] = 00_B.

0 VDD level is higher than $V_{BODOUT1}$

1 VDD dropped below $V_{BODOUT1}$

CFGRR (Configuration Reset Register) **$8F_H$**

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	nCFGRR
-	-	-	-	-	-	-	R/W

Initial value : 01_H

nSOFTR Generate software reset signal. When configuration reset is asserted, configuration option read sequence is started and the associated reset flag appears on this bit position. To generate configuration reset, write FA_H to this register, and to initialize this flag, write $F5_H$ to this register.

14. On-chip Debug System

14.1 Overview

14.1.1 Description

The On-chip debug system(OCD) of MC96FR332A is used to program/erase the non-volatile memory or debug the device. The main features are shown as follows.

14.1.2 Features

- Two-wire external interface : 1-wire serial clock input, 1-wire bi-directional serial data bus
- Debugger Access to :
  - All Internal Peripheral Units
  - Internal data RAM
  - Program Counter
  - Non-volatile Memories
- Extensive On-chip Debug Support for Break Conditions, Including
  - Break Instruction
  - Single Step Break
  - Program Memory Break Points on Single Address
  - Programming of Flash, Fuses, and Lock Bits through the two-wire Interface
  - On-chip Debugging Supported by Dr.Choice®
- Operating frequency : Supports the maximum frequency of the target MCU

Caution : When the device operates with OCD module connected, the main oscillator of MC96FR332A does not stop even if it is in STOP mode.

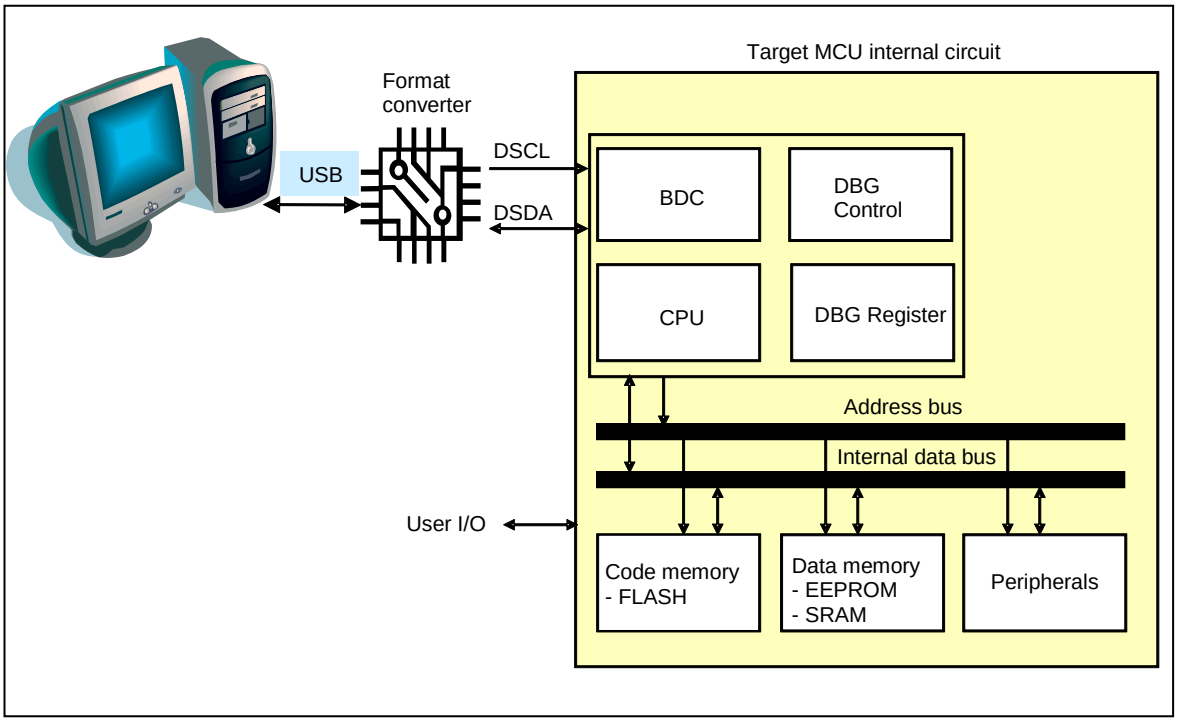


Figure 14-1 Block Diagram of On-Chip Debug System

14.2 Two-pin external interface

14.2.1 Basic transmission packet

- ✓ 10-bit packet transmission via two-wire interface.
- ✓ 1 packet consists of 8-bit data, 1-bit parity and 1-bit acknowledge.
- ✓ Even parity for 8-bit transmit data.
- ✓ Receiver gives acknowledge bit by pulling the data line low when 8-bit transmit data and parity bit has no error.
- ✓ When transmitter receives no acknowledge bit from the receiver, error process is done by transmitter.
- ✓ When acknowledge error is generated, host PC issues a stop condition and re-transmits the command.
- ✓ Background debugger command is composed of a bundle of packets.
- ✓ Each packet starts with a start condition and ends with a stop condition.

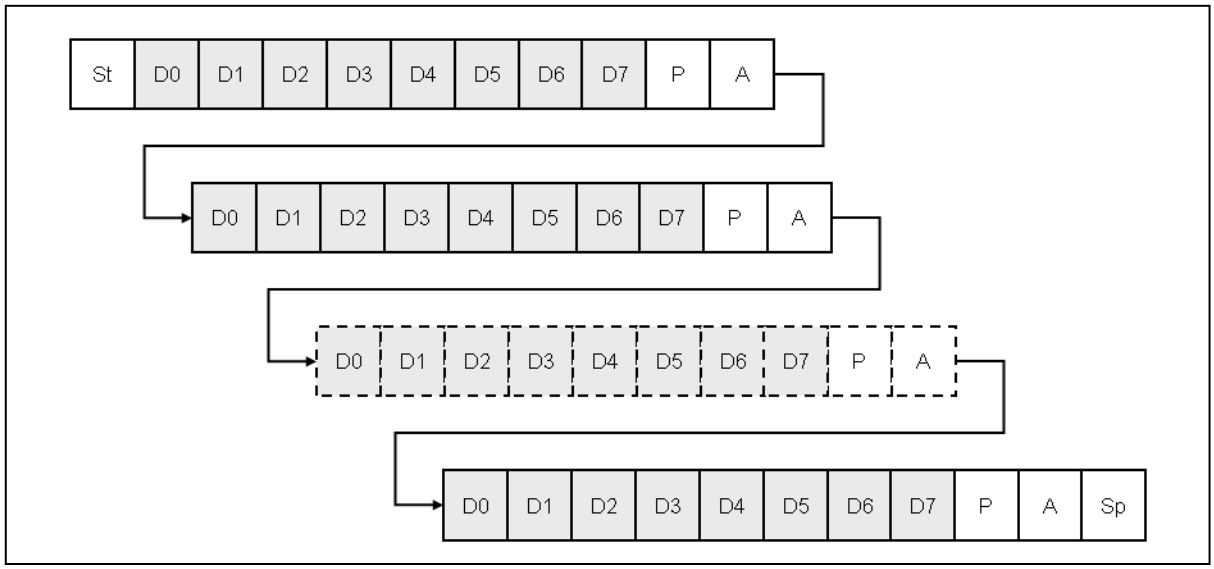


Figure 14-2 10-bit transmission packets

14.2.2 Packet transmission timing

14.2.2.1 Data transfer

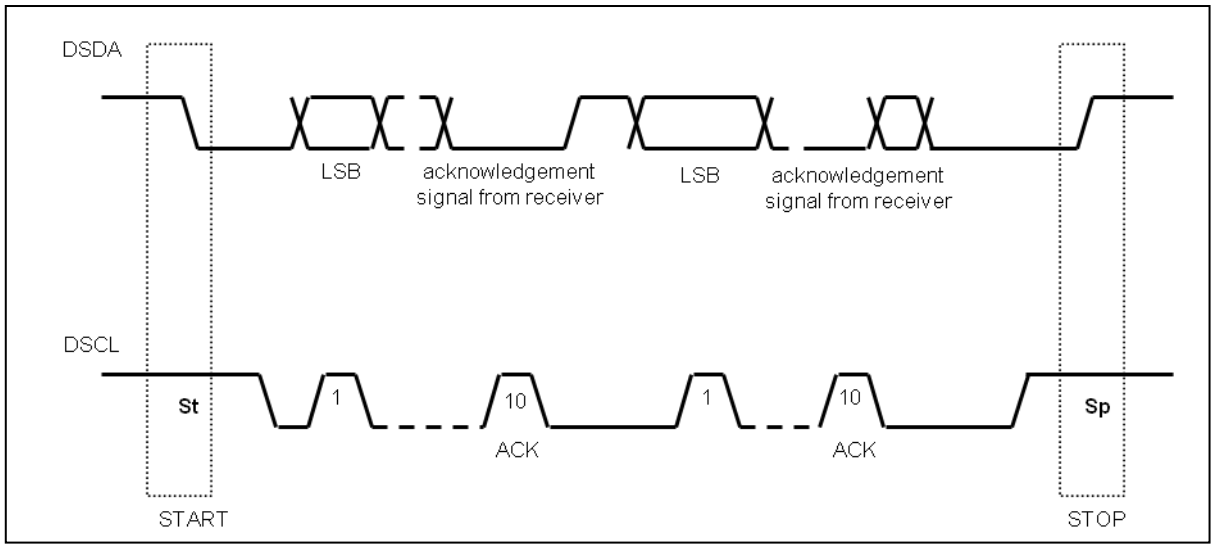


Figure 14-3 Data transfer on the twin bus

14.2.2.2 Bit transfer

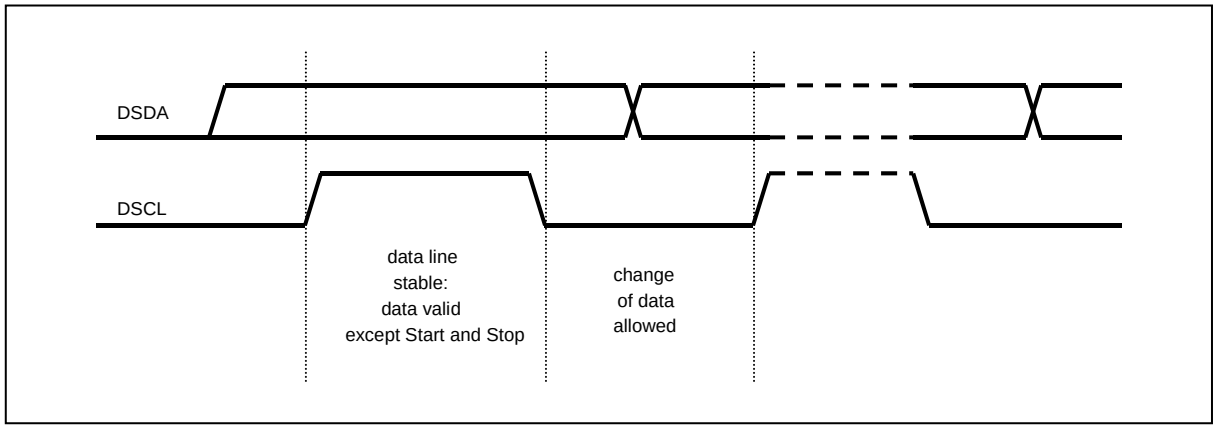


Figure 14-4 Bit transfer on the serial bus

14.2.2.3 Start and stop condition

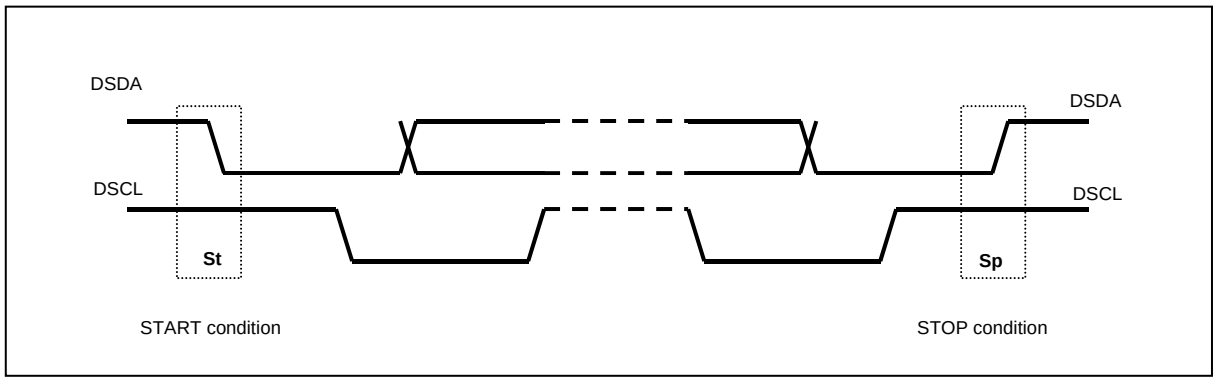


Figure 14-5 Start and stop condition

14.2.2.4 Acknowledge bit

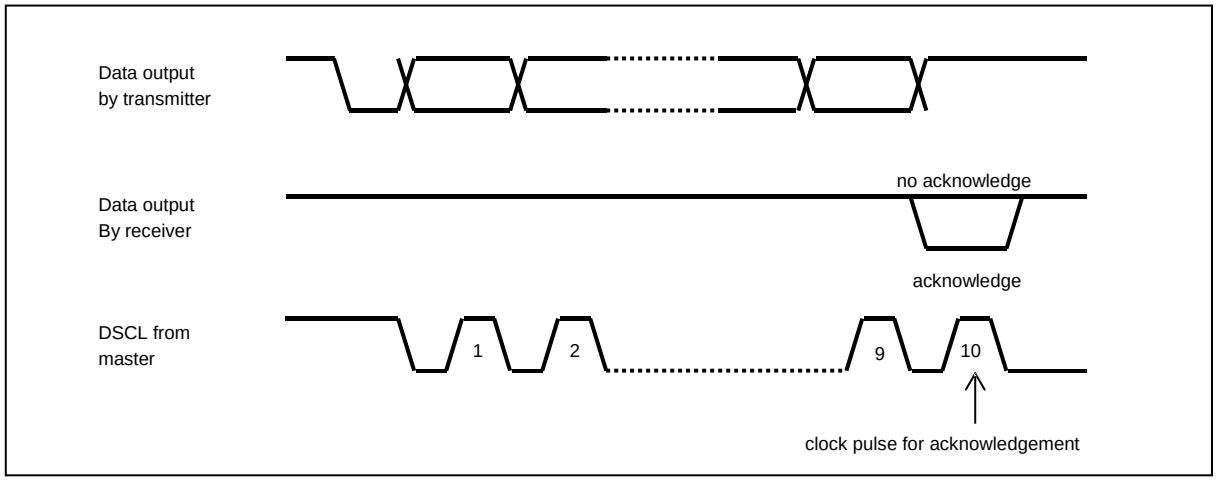


Figure 14-6 Acknowledge by receiver

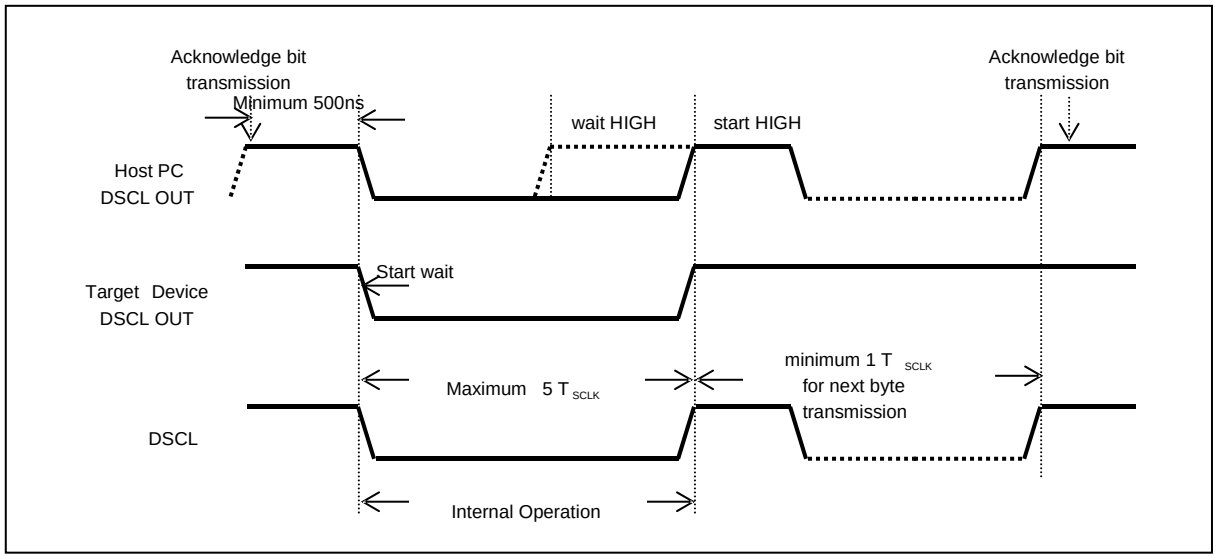


Figure 14-7 Clock synchronization during wait procedure

14.2.3 Connection of transmission

Two-pin interface connection uses open-drain (wired-AND bidirectional I/O).

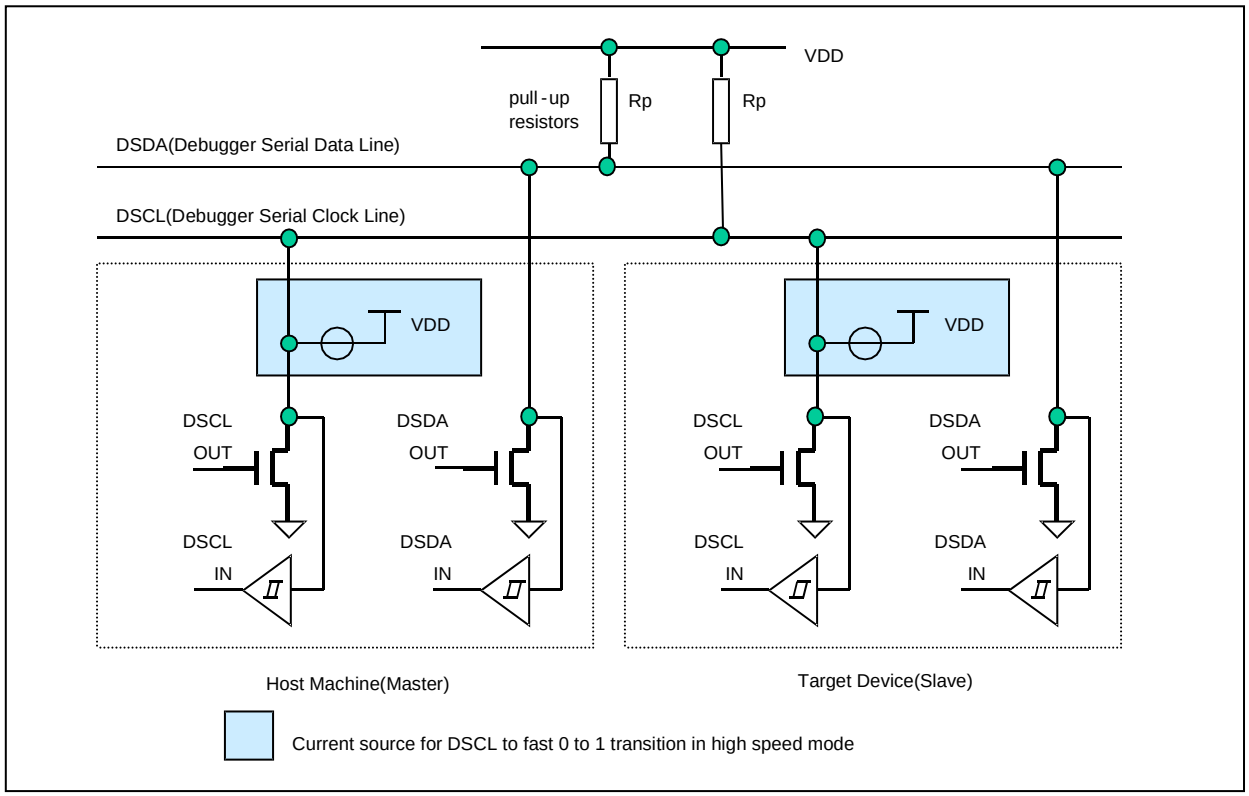


Figure 14-8 Wire connection for serial communication

15. FLASH Memory Controller

15.1 Overview

15.1.1 Description

The MC96FR332A has 32KB of embedded FLASH memory. On reset, this non-volatile memory is used as code memory. In user application program, parts of this non-volatile memory can be updated. Program and erase is performed by ISP via OCD or parallel ROM writer in byte size.

15.1.2 Features of FLASH

- Memory size : 32Kbytes
- Boot Area : Configurable boot area according to BSIZE[1:0]
- Can be updated through registers setting(ISP feature)
- PROGRAM or ERASE operation is performed with single power supply
- Command interface for fast program and erase operation
- Up to 10,000 program/erase cycles at typical voltage and temperature
- Security feature : Code and Boot Area
- Page (Buffer) Size : 64B (XDATA region, addresses 8000_H ~ $803F_H$)

15.2 Boot Area

Boot Area located in program memory area can store Boot program code for upgrading application code by interfacing with I/O port pins.

The Boot Area can't be erased or programmed by unless LOCKB in FUSE_CONF is cleared for the safety of boot code.

The size of Boot Area can be varied by BSIZE bits in FUSE_CONF. See chapter 16.

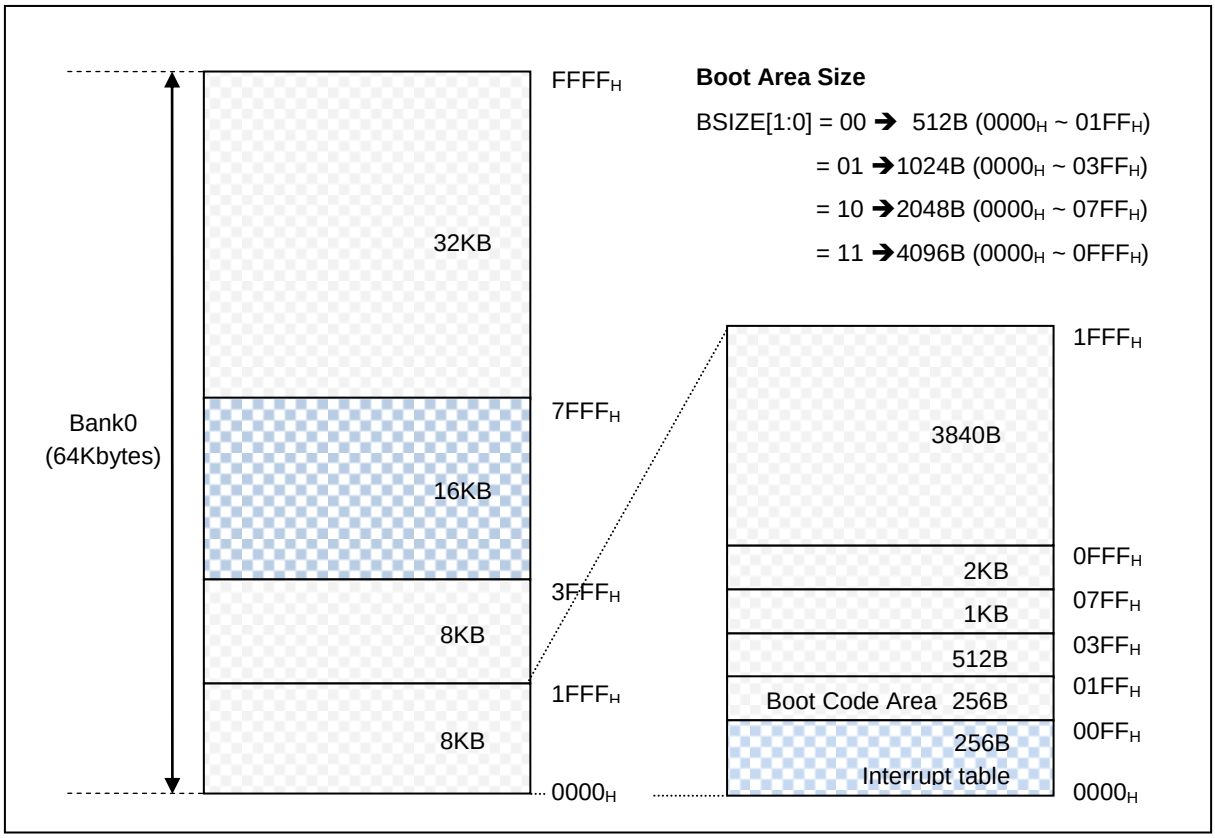


Figure 15-1 Program Memory Address Space

15.3 Register Map

Name	Address	Dir	Default	Description
FMR	$E1_H$	R/W	00_H	FLASH Mode Register
FARH	$E9_H$	R/W	00_H	FLASH Address Register High
FARM	EA_H	R/W	00_H	FLASH Address Register Middle
FARL	EB_H	R/W	00_H	FLASH Address Register Low
FCR	EC_H	R/W	03_H	FLASH Control Register
FSR	ED_H	R/W	80_H	F LASH Status Register
FTCR	EE_H	R/W	00_H	F LASH Time Control Register
CSUMH	$2F06_H$	R	00_H	FLASH Read Checksum Register Low
CSUMM	$2F07_H$	R	00_H	FLASH Read Checksum Register Middle
CSUML	$2F0F_H$	R	00_H	FLASH Read Checksum Register Low
FTR	$2F58_H$	R/W	00_H	F LASH Test Register @XSFR
PageBuffer	8000_H~$803F_H$	R/W		FLASH Page Buffer @XSFR

Table 15-1 Register Map of FLASH Memory Controller

15.4 Register Description

15.4.1 FLASH Control Registers Description

FMR (FLASH Mode Register) **$E1_H$**

7	6	5	4	3	2	1	0
AEF	PBUFF	FSEL	ESEL	OTPE	VFY	READ	nFERST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 01_H

AEF FLASH Bulk Erase Enable.
- 0 FLASH Bulk Erase Disabled
- 1 FLASH Bulk Erase Enabled

PBUFF Select Flash Page Buffer.
- 0 Main cell selected.
- 1 Page buffer selected.

OTPE Select OTP area
- 0 No operation
- 1 OTP READ/WRITE is enabled

VFY Enable verify mode with PGM or ERASE bit
- 0 No verify operation
- 1 Program Verify with PGM=1
- Erase Verify with ERASE=1

READ FLASH Read(VFY=0) or Write(program or erase) Verify(VFY=1). This bit initiates reading the entire FLASH area, and must be set in chip test mode, debugger mode or rom writing mode. Clear all FARH, FARM and FARL before setting this bit for proper operation.
- 0 No operation
- 1 Start Read or Verify operation

nFERST Reset FLASH/EEPROM Controller. This bit is auto-set after 1 system clock period.
- 0 No operation
- 1 Reset internal registers for FLASH/EEPROM Controller

Caution : The FEMR register is not used in normal operation including self programming. Do not alter the contents of this register if possible.

FARH (FLASH Address Register High) **$E9_H$**

7	6	5	4	3	2	1	0
-	-	-	-	FADDR19	FADDR17	FADDR17	FADDR16
-	-	-	-	R/W	R/W	R/W	R/W

Initial value : 00_H

FADDR[19:16] Flash Address High (Write)
Checksum result in auto verify mode (Read)

FARM (FLASH Address Register Middle) **EA_H**

7	6	5	4	3	2	1	0

FADDR15	FADDR14	FADDR13	FADDR12	FADDR11	FADDR10	FADDR9	FADDR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

FADDR[15:8] Flash Address Middle (Write)
Checksum result in auto verify mode (Read, PCRCRD=0)
CRC result in auto verify mode (Read, PCRCRD=1)

FARL (FLASH Address Register Low) — EB_H

7	6	5	4	3	2	1	0
FADDR7	FADDR6	FADDR5	FADDR4	FADDR3	FADDR2	FADDR1	FADDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

FADDR[7:0] Flash Address Low. As flash page buffer size is of 64 Bytes, only the MSB of this register is meaningful. (Write)
Checksum result in auto verify mode (Read, PCRCRD=0)
CRC result in auto verify mode (Read, PCRCRD=1)

FARH, FARM and FARL registers are used for program, erase, or auto-verify operation. In program or erase mode, these registers point to the page number to be programmed or erased.

FCR (FLASH Control Register) — EC_H

7	6	5	4	3	2	1	0
-	-	EXIT1	EXIT0	CMD3	PGM/ CMD2	ERASE/ CMD1	nPBRST/ CMD0
-	-	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 01_H

EXIT[1:0] Exit from program or erase mode. This bit is auto-cleared after 1 system clock period.

EXIT1	EXIT0	Description
0	0	No exit
0	1	No exit
1	0	No exit
1	1	Exit

CMD[3:0] FLASH Command. The CMD0 bit(=nPBRST) is auto-set after 1 system clock period.

0000	Page Buffer Reset
0011	Erase
0101	Program
1101	LOCKF Program
others	Prohibited (no operation)

FSR (FLASH Status Register) — ED_H

7	6	5	4	3	2	1	0
nPEVBSY	VFYGOOD	PCRCRD	-	ROMINT	PMODE	EMODE	VMODE
R	R/W	R/W	-	R/W	R	R	R

Initial value : 80_H

nPEVBSY BUSY flag. Represents that program, erase, or verify operation is on-

going, active low. This bit is auto-set when operation is done.

0 Busy (Operation processing)

1 Operation completed

VFYGOOD Auto-verification result flag

0 Auto-verification failed

1 Auto-verification succeeded

PCRCRD CRC calculation data read control. For correct operation, clear the FARH, FARM and FARL before starting CRC or setting READ bit in FEMR.

0 Reading FARH, FARM and FARL registers return checksum value (24-bit)

1 Reading FARM and FARL registers return CRC result (16-bit)

ROMINT FLASH Interrupt Flag. This bit is auto-cleared when program, erase, or verify operation is started.

0 No interrupt requested

1 Interrupt requested

PMODE Program mode flag

EMODE Erase mode flag

VMODE Verify mode flag

FTCR (FLASH Time control Register) **EE_H**

7	6	5	4	3	2	1	0
TCR7	TCR6	TCR5	TCR4	TCR3	TCR2	TCR1	TCR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Initial value : 00_H

TCR[7:0] Program or Erase Time control

Program or erase time is controlled by the value in FETCR register.

The FLASH Memory controller includes a 10-bit counter used to calculate program or erase time. The counter is clocked by a clock which is divided by 64 from XIN clock(XIN/64). It's a simple counter. When program or erase operation starts, the counter is cleared and start up-counting until it reaches the target value coming from FTCR. On matching, the counter stops and the PEVBSY flag is set.

In bulk erase mode, the TCR[7:0] becomes the most significant eight bits in counter target value, and the least significant two bits are filled with "11". In program or erase mode, the most significant two bit is filled with '01, the least significant bit is filled with '1', and the TCR[7:0] becomes the middle eight bits in counter target value. So the program or erase time is calculated by the following equation. In the following equation and description, it is assumed that the frequency of external clock source, f_{XIN} is 8MHz. In that case, the period of XIN/64 clock is about 8us.

Tpe = (TCR+1) * 2 * 8us

Tbe = (TCR+1) * 4 * 8us

where Tpe is time to be taken when program or erase operation is performed in byte- or page-size, Tbe is time for bulk erase operation.

Normally the maximum program or erase time can be 8us * 512 ≒ 4ms. And considering the error rate of ±10%, about 3.6~4.4ms of program/erase time can be ensured. Similarly in bulk erase mode, the maximum time can be 8ms, so 7.2~8.8ms of bulk erase time will be applied.

CSUMH (FLASH Read Check Sum Register High) **$2F06_H$**

7	6	5	4	3	2	1	0
CSUM23	CSUM22	CSUM21	CSUM20	CSUM19	CSUM18	CSUM17	CSUM16
R	R	R	R	R	R	R	R

Initial value : 00_H

CSUM[23:16] FLASH Read Checksum in auto-verify mode

The CSUMH, CSUMM and CSUML registers are test purpose only.

CSUMM (FLASH Read Check Sum Register Middle) **$2F07_H$**

7	6	5	4	3	2	1	0
CSUM15	CSUM14	CSUM13	CSUM12	CSUM11	CSUM10	CSUM9	CSUM8
R	R	R	R	R	R	R	R

Initial value : 00_H

CSUM[15:8] FLASH Read Checksum in auto-verify mode

CSUML (FLASH Read Check Sum Register Low) **$2F0F_H$**

7	6	5	4	3	2	1	0
CSUM7	CSUM6	CSUM5	CSUM4	CSUM3	CSUM2	CSUM1	CSUM0
R	R	R	R	R	R	R	R

Initial value : 00_H

CSUM[7:0] FLASH Read Checksum in auto-verify mode

In auto-verify mode, the FLASH address increases automatically by one. CSUMH, CSUMM and CSUML registers are read-only, and reading these registers returns the 24-bit checksum result.

15.5 Memory map

As described previously, MC96FR332A has 32KB of Program Memory called FLASH. It is needed to write page address into FARH, FARM and FARL registers to program or erase the non-volatile memory.

15.5.1 FLASH area division

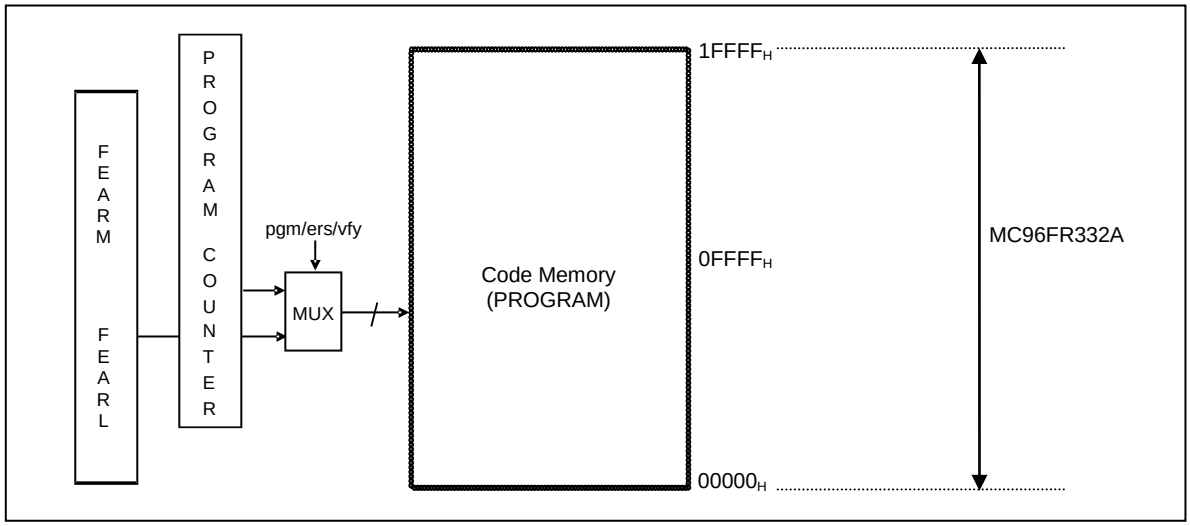


Figure 15-2 FLASH Memory Map

15.5.2 Address configuration of FLASH memory

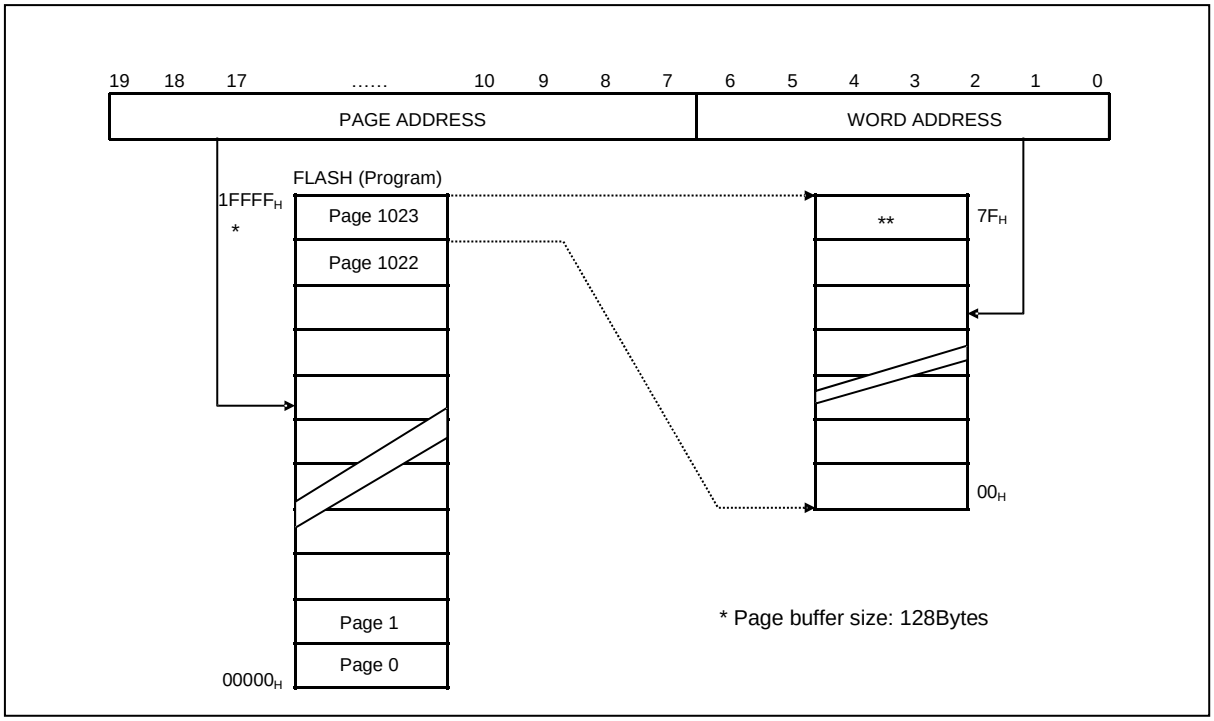


Figure 15-3 FLASH Memory Address generation

15.6 Serial In-System Program Mode

Serial In-System Program is performed via the interface of debugger which uses two wires. For more information about debugger, refer to chapter 14.

15.6.1 ISP or Self Programming Sequence

In MC96FR332A, the commands needed to update FLASH is commenced by FECR register only. PROGRAM or ERASE sequence is as follows :

1. Set Erase or Program time : FETCR(EE_H) = 0xxx [NOTE1]
2. Enter ISP or Self Program Mode [NOTE2]
3. Reset Page Buffer : FECR(EC_H) = 0x00
4. Load Page Buffer by “MOVX” instruction (up to 64Bytes).
5. Set Page Address to be programmed or erased by writing FARH($E9_H$), FARM(EA_H) and FARL(EB_H)

6-1. Set AEF and FSEL bits in FEMR. (This is used in ISP mode for Bulk Erase operation)

6-2. Set OTPE bit in FEMR. (OTPE bit is used to access OTP area. Set this bit if needed.)

6-3. Set VFY bit in FEMR. (VFY bit is used for read-verify operation. Set this bit if needed.)

7. Start Erase or Program : FECR(EC_H) = 0x03(Erase) or 0x05(Program)
8. Wait PEVBSY bit in FESR(ED_H) : for OCD mode
9. ISP or Self Program Mode Exit : FECR(EC_H) = 0x31

[NOTE1] Program or Erase time is only to be set before real program or erase operation. Normally, the FETCR value doesn’t have to be changed.

[NOTE2] Between flash program mode entry and exit, there can be several program or erase operation. It is only need to exit flash program mode when all program or erase operations are done.

Step / Operation	Page/Byte Program	Page/Byte Erase	Bulk Erase
Set Erase or Program Time	1) FETCR = 0xxx	1) FETCR = 0xxx	1) FETCR = 0xxx
Mode Entry	2) Mode Entry	2) Mode Entry	2) Mode Entry
Page Buffer Reset	3) FECR = 0x00	3) FECR = 0x00	3) FECR = 0x00
Load Page Buffer	4) Load bytes by MOVX instruction	4) Load bytes by MOVX instruction	4) Load page(64B) by movx instruction
Set Page Address	5) FARH = 0xxx	5) FARH = 0x00	
	FARM = 0xxx	FARM = 0xxx	
	FARL = 0xxx	FARL = 0xxx	
Set AEF/FSEL (Bulk Erase only)			6-1) FEMR = 0xA1
Set OTPE (OTP Access only)	6-2) FEMR |= 0x09	6-2) FEMR |= 0x09	6-2) FEMR |= 0x09
Set VFY (Verify operation only)	6-3) FEMR |= 0x02	6-3) FEMR |= 0x02	6-3) FEMR |= 0x02
Start Program or Erase	7) FECR = 0x05	7) FECR = 0x03	7) FECR = 0x03
Wait PEVBSY (OCD only)	8) Wait PEVBSY	8) Wait PEVBSY	8) Wait PEVBSY
Mode Exit	9) Mode Exit	9) Mode Exit	9) Mode Exit

Table 15-2 Program or Erase sequence in ISP or Self Program Mode

15.6.1.1 FLASH Read

Step 1. Enter OCD(=ISP) mode.

Step 2. Set ENBDM bit of BCR.

Step 3. Enable debug and Request debug mode.

Step 4. Read data from Flash.

15.6.1.2 Enable ISP or Self Programming Mode

Step 1. Enter OCD(=ISP) mode.[NOTE 1]

Step 2. Set ENBDM bit of BCR.

Step 3. Enable debug and Request debug mode.

Step 3. Enter program/erase mode sequence.[NOTE 2]

(1) Write AA_H to $F555_H$

(2) Write 55_H to $FAAA_H$

(3) Write $A5_H$ to $F555_H$

[NOTE 1]. Refer to chapter 14

[NOTE 2]. Command sequence to activate FLASH program/erase mode. It is composed of sequential write to fixed FLASH addresses.

15.6.2 Example of FLASH control in C language

The next example code shows how to program or erase a specific page area of FLASH using C language. The program or erase sequence used in the test code complies with above rules. In this example code, the page address $F000_H$ is erased and programmed.

Example:

```
////////////////////////////////////////////////////////
//  Project     : Write data to EEPROM at 0xF000
//  Device      :  MC96FR332A
//  Oscillator  :  4MHz
//  Compiler    :  Keil uvision C Compiler V7.20
////////////////////////////////////////////////////////

#include <intrins.h>
#include "MC96FR332A.h"

#define FLASH_PBUFF_SIZE       64

// PGM or ERASE Timing, normally the same values are set.
#define PGMTIME                0x4F    // 2.5ms @4MHz
#define ERSTIME                0x4F    // 2.5ms @4MHz
```

```
void page_buffer_reset();
void flash_page_write(unsigned int addr, unsigned char *wdata);
void flash_page_erase(unsigned int addr);
void flash_program_enter();
void flash_program_exit();

xdata unsigned char pagerom[FLASH_PBUFF_SIZE] _at_ 0x8000;  // page buffer
data   unsigned char page_data[FLASH_PBUFF_SIZE];                  // write data buffer

void main()
{
        unsigned p_index;

    // Step 2
         flash_program_entry();

        eeprom_page_erase(0xF000);

    // Tmp data for page write operation. Try other data!!!
        for (p_index=0; p_index < FLASH_PBUFF_SIZE; p_index++) {
            page_data[p_index] = p_index;
        }

        eeprom_page_write(0xF000, page_data);

    // Step 10
         flash_program_exit();
        while(1);
}

void flash_page_erase(unsigned int addr)
{
        int i;
        unsigned char temp;
        int addr_index;

    // Step 1
        FETCR = ERSTIME;

    // Step 3
        page_buffer_reset();

    // Step 4
         for (i=0; I < FLASH_PBUFF_SIZE; i++) {
              pagerom[i] = 0x00;
        }

    // Step 5
```

```
        FARL = (unsigned char) addr;
        FARM = (unsigned char) (addr>>8);

    // Step 8
        FECR = 0x0B;

    // Step 9 : It is optional because the CPU clock halts while in program or erase operation.
        while(FESR>>7 == 0x00);
}

void flash_page_write(unsigned int addr, unsigned char *wdata)
{
        int i;
        unsigned char temp;
        int addr_index;

    // Step 1
        FETCR = PGMTIME;

    // Step 3
        page_buffer_reset();

    // Step 4
        for (i=0; I < FLASH_PBUFF_SIZE; i++) {
            pagerom[i] = wdata[i];
        }

    // Step 5
        FARL = (unsigned char) addr;
        FARM = (unsigned char) (addr>>8);

    // Step 8
        FECR = 0x0D;

    // Step 9 : It is optional because the CPU clock halts while in program or erase operation.
        while(FESR>>7 == 0x00);
}

void page_buffer_reset()
{
        FECR  = 0x00;
}

void flash_program_exit()
{
        FECR  = 0x31;
}
```

15.6.3 Summary of FLASH Program/Erase Mode

Flash Operation	Description
FLASH read	Read cell by byte.
FLASH write	Write cell by bytes or page.
FLASH page erase	Erase cell by page.
FLASH bulk erase	Erase the whole cells.
FLASH program verify	Read cell in verify mode after programming.
FLASH erase verify	Read cell in verify mode after erase.
FLASH page buffer load	Load data to page buffer.

Table 15-3 FLASH operating mode

15.7 Security

The MC96FR332A provides one LOCKF bit to protect memory contents from illegal attempt to read. The LOCKF bit can be erased only by bulk erase operation.

LOCKF	USER MODE								OCD(ISP) / PMODE							
	FLASH				OTP				FLASH				OTP			
	R	W	PE	BE	R	W	PE	BE	R	W	PE	BE	R	W	PE	BE
0	O	O	O	X	X	X	X	X	O	O	O	O	O	O	O	O
1	O	O	O	X	X	X	X	X	X	X	X	O	O	X	X	O

Table 15-4 Memory protection by lock bit

- LOCKF : Lock bit of FLASH memory
- R : Read
- W : Write
- PE : Page Erase
- BE : Bulk Erase
- O : Operation is possible.
- X : Operation is impossible.

15.8 FLASH Memory operating mode

15.8.1.1 Electrical Characteristics

FLASH 32KB IP			Spec			
Description	**Symbol**	**Condition**	**Min**	**Typ**	**Max**	**Unit**
Operating Temperature	Temp	Commercial	-40	-	85	℃
Supply Voltage	VDD		1.62	1.8	1.98	V
Ground	VSS			0		V
Clock Frequency	fCLK	VDD=1.8V		-	10	MHz
Clock Period	tPER	VDD=1.8V	100	-		ns
Access Time	tAA	VDD=1.8V	100			ns
Setup Time	tSP	VDD=1.8V	2			ns
Address Hold Time	tAA	VDD=1.8V	10			ns
Address Setup Time	tAS	VDD=1.8V	2			ns
Set Down Time	tSD	VDD=1.8V	2			ns
Output Enable Access Time	tOE	VDD=1.8V			2	ns
Output Delay Time	tOD	VDD=1.8V	2			ns
Data Setup Time	tDS	VDD=1.8V	2			ns
Data Hold Time	tDH	VDD=1.8V	20			ns
Erase Time	tERS	VDD=1.8V	2.5			ms
Program Time	tPGM	VDD=1.8V	2.5			ms
Bulk Erase Time	tBERS	VDD=1.8V	5			ms
Power Down Time	tPD			5		us
Power Up Time	tPU			1		ms
Read Current	Icc1	VDD=1.8V			3	mA
Write Current	Icc2	VDD=1.8V			7	mA
Power Down Current	Ipd	VDD=1.8V,Clock Off			5	uA
Input Low Voltage	VIL	VDD=1.8V		VDD/2	0.2VDD	V
Input High Voltage	VIH	VDD=1.8V	0.8VDD	VDD/2		V
Output Low Voltage	VOL	VDD=1.8V		VDD/2	0.45	V
Output High Voltage	VOH	VDD=1.8V	VDD-0.2	VDD/2		V

Table 15-5 AC Timing Specification

16. Etc..

16.1 FUSE Control Register

FUSE_CONF (Pseudo-Configure Data) **FD_H**

7	6	5	4	3	2	1	0
BSIZE1	BSIZE0	-	-	RSTDIS	-	LOCKB	LOCKF
R	R	R	R	R	R	R	R

Initial value : 00_H

BSIZE[1:0]	Selects the size of Boot Area	
	00	512B
	01	1024B
	10	2048B
	11	4096B
RSTDIS	Enables or disables external reset function	
	0	P20/RESETB is used as a external reset input
	1	P20/RESETB is used as a normal I/O pin
LOCKB	Lock Boot Area	
	0	Boot Area protection disabled
	1	Boot Area protection enabled
LOCKF	Lock FLASH	
	0	LOCK Disable
	1	LOCK Enable

Caution : The reserved bits in FUSE_CONF register, actually OTP region, should not be altered by user. Please do not attempt to erase or program the OTP region except for the above bits or the device will not operate as expected.

17. APPENDIX

A. Instruction Table

The instruction length of M8051W can be 1, 2, or 3 bytes as listed in the following table. It takes 1, 2, or 4 cycles for the CPU to execute an instruction. The cycle is composed of two internal clock periods.

ARITHMETIC				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
ADD A,Rn	Add register to A	1	1	28-2F
ADD A,dir	Add direct byte to A	2	1	25
ADD A,@Ri	Add indirect memory to A	1	1	26-27
ADD A,#data	Add immediate to A	2	1	24
ADDC A,Rn	Add register to A with carry	1	1	38-3F
ADDC A,dir	Add direct byte to A with carry	2	1	35
ADDC A,@Ri	Add indirect memory to A with carry	1	1	36-37
ADDC A,#data	Add immediate to A with carry	2	1	34
SUBB A,Rn	Subtract register from A with borrow	1	1	98-9F
SUBB A,dir	Subtract direct byte from A with borrow	2	1	95
SUBB A,@Ri	Subtract indirect memory from A with borrow	1	1	96-97
SUBB A,#data	Subtract immediate from A with borrow	2	1	94
INC A	Increment A	1	1	04
INC Rn	Increment register	1	1	08-0F
INC dir	Increment direct byte	2	1	05
INC @Ri	Increment indirect memory	1	1	06-07
DEC A	Decrement A	1	1	14
DEC Rn	Decrement register	1	1	18-1F
DEC dir	Decrement direct byte	2	1	15
DEC @Ri	Decrement indirect memory	1	1	16-17
INC DPTR	Increment data pointer	1	2	A3
MUL AB	Multiply A by B	1	4	A4
DIV AB	Divide A by B	1	4	84
DA A	Decimal Adjust A	1	1	D4

LOGICAL				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
ANL A,Rn	AND register to A	1	1	58-5F
ANL A,dir	AND direct byte to A	2	1	55
ANL A,@Ri	AND indirect memory to A	1	1	56-57
ANL A,#data	AND immediate to A	2	1	54
ANL dir,A	AND A to direct byte	2	1	52
ANL dir,#data	AND immediate to direct byte	3	2	53
ORL A,Rn	OR register to A	1	1	48-4F
ORL A,dir	OR direct byte to A	2	1	45
ORL A,@Ri	OR indirect memory to A	1	1	46-47
ORL A,#data	OR immediate to A	2	1	44
ORL dir,A	OR A to direct byte	2	1	42
ORL dir,#data	OR immediate to direct byte	3	2	43
XRL A,Rn	Exclusive-OR register to A	1	1	68-6F
XRL A,dir	Exclusive-OR direct byte to A	2	1	65

XRL A, @Ri	Exclusive-OR indirect memory to A	1	1	66-67
XRL A,#data	Exclusive-OR immediate to A	2	1	64
XRL dir,A	Exclusive-OR A to direct byte	2	1	62
XRL dir,#data	Exclusive-OR immediate to direct byte	3	2	63
CLR A	Clear A	1	1	E4
CPL A	Complement A	1	1	F4
SWAP A	Swap Nibbles of A	1	1	C4
RL A	Rotate A left	1	1	23
RLC A	Rotate A left through carry	1	1	33
RR A	Rotate A right	1	1	03
RRC A	Rotate A right through carry	1	1	13

DATA TRANSFER				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
MOV A,Rn	Move register to A	1	1	E8-EF
MOV A,dir	Move direct byte to A	2	1	E5
MOV A,@Ri	Move indirect memory to A	1	1	E6-E7
MOV A,#data	Move immediate to A	2	1	74
MOV Rn,A	Move A to register	1	1	F8-FF
MOV Rn,dir	Move direct byte to register	2	2	A8-AF
MOV Rn,#data	Move immediate to register	2	1	78-7F
MOV dir,A	Move A to direct byte	2	1	F5
MOV dir,Rn	Move register to direct byte	2	2	88-8F
MOV dir,dir	Move direct byte to direct byte	3	2	85
MOV dir,@Ri	Move indirect memory to direct byte	2	2	86-87
MOV dir,#data	Move immediate to direct byte	3	2	75
MOV @Ri,A	Move A to indirect memory	1	1	F6-F7
MOV @Ri,dir	Move direct byte to indirect memory	2	2	A6-A7
MOV @Ri,#data	Move immediate to indirect memory	2	1	76-77
MOV DPTR,#data	Move immediate to data pointer	3	2	90
MOVC A,@A+DPTR	Move code byte relative DPTR to A	1	2	93
MOVC A,@A+PC	Move code byte relative PC to A	1	2	83
MOVX A,@Ri	Move external data(A8) to A	1	2	E2-E3
MOVX A,@DPTR	Move external data(A16) to A	1	2	E0
MOVX @Ri,A	Move A to external data(A8)	1	2	F2-F3
MOVX @DPTR,A	Move A to external data(A16)	1	2	F0
PUSH dir	Push direct byte onto stack	2	2	C0
POP dir	Pop direct byte from stack	2	2	D0
XCH A,Rn	Exchange A and register	1	1	C8-CF
XCH A,dir	Exchange A and direct byte	2	1	C5
XCH A,@Ri	Exchange A and indirect memory	1	1	C6-C7
XCHD A,@Ri	Exchange A and indirect memory nibble	1	1	D6-D7

BOOLEAN				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
CLR C	Clear carry	1	1	C3
CLR bit	Clear direct bit	2	1	C2
SETB C	Set carry	1	1	D3
SETB bit	Set direct bit	2	1	D2
CPL C	Complement carry	1	1	B3
CPL bit	Complement direct bit	2	1	B2

ANL C,bit	AND direct bit to carry	2	2	82
ANL C,/bit	AND direct bit inverse to carry	2	2	B0
ORL C,bit	OR direct bit to carry	2	2	72
ORL C,/bit	OR direct bit inverse to carry	2	2	A0
MOV C,bit	Move direct bit to carry	2	1	A2
MOV bit,C	Move carry to direct bit	2	2	92

BRANCHING				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
ACALL addr 11	Absolute jump to subroutine	2	2	11→F1
LCALL addr 16	Long jump to subroutine	3	2	12
RET	Return from subroutine	1	2	22
RETI	Return from interrupt	1	2	32
AJMP addr 11	Absolute jump unconditional	2	2	01→E1
LJMP addr 16	Long jump unconditional	3	2	02
SJMP rel	Short jump (relative address)	2	2	80
JC rel	Jump on carry = 1	2	2	40
JNC rel	Jump on carry = 0	2	2	50
JB bit,rel	Jump on direct bit = 1	3	2	20
JNB bit,rel	Jump on direct bit = 0	3	2	30
JBC bit,rel	Jump on direct bit = 1 and clear	3	2	10
JMP @A+DPTR	Jump indirect relative DPTR	1	2	73
JZ rel	Jump on accumulator = 0	2	2	60
JNZ rel	Jump on accumulator ≠ 0	2	2	70
CJNE A,dir,rel	Compare A,direct jne relative	3	2	B5
CJNE A,#d,rel	Compare A,immediate jne relative	3	2	B4
CJNE Rn,#d,rel	Compare register, immediate jne relative	3	2	B8-BF
CJNE @Ri,#d,rel	Compare indirect, immediate jne relative	3	2	B6-B7
DJNZ Rn,rel	Decrement register, jnz relative	3	2	D8-DF
DJNZ dir,rel	Decrement direct byte, jnz relative	3	2	D5

MISCELLANEOUS				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
NOP	No operation	1	1	00

ADDITIONAL INSTRUCTIONS (selected through EO[7:4])				
Mnemonic	**Description**	**Bytes**	**Cycles**	**Hex code**
MOVC @(DPTR++),A	M8051W/M8051EW-specific instruction supporting software download into program memory	1	2	A5
TRAP	Software break command	1	1	A5

In the above table, an entry such as E8-EF indicates a continuous block of hex opcodes used for 8 different registers, the register numbers of which are defined by the lowest three bits of the corresponding code. Non-continuous blocks of codes, shown as 11→F1 (for example), are used for absolute jumps and calls, with the top 3 bits of the code being used to store the top three bits of the destination address.

The CJNE instructions use the abbreviation #d for immediate data; other instructions use #data.

B. Instructions on how to use the input port.

- Error occur status
  - Using compare jump instructions with input port, it could cause error due to the timing conflict inside the MCU.
  - Compare jump Instructions which cause potential error used with input port condition:

```
JB    bit, rel        ; jump on direct bit=1
JNB   bit, rel        ; jump on direct bit=0
JBC   bit, rel        ; jump on direct bit=1 and clear
CJNE  A, dir, rel     ; compare A, direct jne relative
DJNZ  dir, rel        ; decrement direct byte, jnz relative
```

  - It is only related with Input port. Internal parameters, SFRs and output bit ports don't cause an y error by using compare jump instructions.
  - If input signal is fixed, there is no error in using compare jump instructions.

- Error status example

```
while(1){
  if (P00==1){ P10=1; }
  else { P10=0; }
  P11^=1;
}
```

```
zzz:  JNB    080.0, xxx  ; it possible to be error
      SETB   088.0
      SJMP   yyy
xxx:  CLR    088.0
yyy:  MOV    C,088.1
      CPL    C
      MOV    088.1,C
      SJMP   zzz
```

```
unsigned char ret_bit_err(void)
{
  return !P00;
}
```

```
      MOV    R7, #000
      JB     080.0, xxx  ; it possible to be error
      MOV    R7, #001
xxx:  RET
```

- Preventative measures (2 cases)
  - Do not use input bit port for bit operation but for byte operation. Using byte operation instead of bit operation will not cause any error in using compare jump instructions for input port.

```
while(1){
 if ((P0&0x01)==0x01) { P10=1; }
else { P10=0; }
  P11^=1;
}
```

```
zzz:  MOV    A, 080      ; read as byte
      JNB    0E0.0, xxx  ; compare
      SETB   088.0
      SJMP   yyy
xxx:  CLR    088.0
yyy:  MOV    C,088.1
      CPL    C
      MOV    088.1,C
      SJMP   zzz
```

- If you use input bit port for compare jump instruction, you have to copy the input port as intern al parameter or carry bit and then use compare jump instruction.

```
bit tt;
while(1){
  tt=P00;
  if (tt==0){ P10=1;}
  else {  P10=0;}
  P11^=1;
}
```

```
zzz:  MOV   C,080.0  ; input port use internal parameter
      MOV   020.0, C      ; move
      JB    020.0, xxx    ; compare
      SETB  088.0
      SJMP  yyy
xxx:  CLR   088.0
yyy:  MOV   C,088.1
      CPL   C
      MOV   088.1,C
      SJMP  zzz
```