HYNIX SEMICONDUCTOR
8-BIT SINGLE-CHIP MICROCONTROLLERS

HMS81020ET
HMS81032ET

User's Manual (Ver. 1.00)

---

hynix
Semiconductor

Version 1.00

Published by
MCU Application Team

Additional information of this manual may be served by Hynix Semiconductor offices in Korea or Distributors and Representatives listed at address directory.

Hynix 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, Hynix 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.

OVERVIEW 1
Description 1
Features 1
Development Tools 2
BLOCK DIAGRAM 3
PIN ASSIGNMENT (Top View) 4
PACKAGE DIMENSION 5
PIN FUNCTION 8
Standard mode pin function 9
PORT STRUCTURES 10
ELECTRICAL CHARACTERISTICS 12
Absolute Maximum Ratings 12
Recommended Operating Conditions 12
DC Electrical Characteristics 12
REMOUT Port Ioh Characteristics Graph 14
REMOUT Port Iol Characteristics Graph 14
AC Characteristics 15
MEMORY ORGANIZATION 16
Registers 16
Program Memory 19
Data Memory 22
List for Control Registers 23
Addressing Mode 25
I/O PORTS 29
R0 Ports 29
R1 Ports 29
R2 Port 31
CLOCK GENERATOR 32
Oscillation Circuit 33
BASIC INTERVAL TIMER 34
WATCH DOG TIMER 36
Timer0, Timer1, Timer2 37
INTERRUPTS 45
Interrupt priority and sources 46
Interrupt control register 46
Interrupt accept mode 47
Interrupt Sequence 48
BRK Interrupt 50
Multi Interrupt 50
External Interrupt 50
Key Scan Input Processing 51
STANDBY FUNCTION 53
Sleep Mode 53
Stop Mode 53
Standby mode release 54
Operation of standby mode release 56
RESET FUNCTION 59
External RESET 59
Power on RESET 59
Low voltage detection mode 61
EPROM MODE 64
PIN ASSIGNMENT (Top View) 64
EPROM mode pin function 65
EPROM Mode setting 66
General view of EPROM operation 67
EPROM write & verify 68
EPROM verify 69
LOCK bit write 70
LOCK bit read 71
Considerations on programming 72
Programming flow chart 73
Recommended circuit diagram of HMS81032ET 74

HMS81020ET/HMS81032ET

CMOS SINGLE- CHIP 8-BIT MICROCONTROLLER FOR UNIVERSAL REMOTE CONTROLLER

1. OVERVIEW

1.1 Description

The HMS81020ET/HMS81032ET is an advanced CMOS 8-bit microcontroller with 20K/32K bytes of EPROM. The device is one of GMS800 family. The HYNIX HMS81020ET/HMS81032ET is a powerful microcontroller which provides a highly flexible and cost effective solution to many UR applications.The HMS81020ET/HMS81032ET provides the following standard features: 20K/32K bytes of EPROM, 448 bytes of RAM, 8-bit timer/counter, on-chip oscillator and clock circuitry. In addition, the HMS81020ET/HMS81032ET supports power saving modes to reduce power consumption, low voltage indication circuit(LVIR) to display the consumption of batteries and low voltage detection control register(LVDC) to disable or enable LVD function during transmission.

Device Name	EPROM Size	RAM Size	Package
HMS81020ET	20K Bytes	448 Bytes (included 256 bytes stack memory)	20 SOP/PDIP 24 SOP/Skinny DIP 28 SOP/Skinny DIP
HMS81032ET	32K Bytes		

1.2 Features

- **Minimum instruction execution time: 1uS at 4MHz (2 cycle NOP Instruction)**
- **Programmable I/O pins**

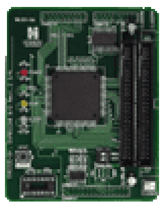
	20 PIN	24 PIN	28 PIN
INPUT	3	3	3
OUTPUT	2	2	2
I/O	13	17	21

- **Operating voltage**
  - 2.0 ~ 4.0 V @ 1~4MHz
- **One 16-bit Timer/Counter**
- **Two 8-bit Timer/Counter**
- **One 8-bit Basic interval timer**
- **One 6-bit Watch dog timer**
- **8 Interrupt sources**
  - Nested interrupt control is available.
  - External input: 2
  - Keyscan input: 1
  - Basic interval timer: 1
  - Watchdog timer: 1
  - Timer : 3
- **Power on reset**
- **Power saving operation modes**
  - STOP operation
  - SLEEP operation
- **Low voltage detection(LVD) circuit to get into back-up mode**
- **Watchdog timer auto start (during 1second after power on reset)**
- **Low voltage indication circuit(LVIR) to display the consumption of batteries**
  - LVIR0: 2.2V(Typ.) - 300mV ~ +200mV
  - LVIR1: 2.0V(Typ.) - 300mV ~ +200mV
- **Low voltage detection control register(LVDC)**

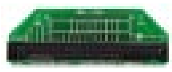
1.3 Development Tools

The HMS81020ET/HMS81032ET are supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr.^{TM} and OTP programmers. Macro assembler operates under the MS-Windows 95/98^{TM} /NT4/2000/XP.The OTP programmer can be supplied two types of programmer such as universal single programmer (CHOICE-SIGMA^{TM}) and stand alone gran4 programmer.

Software	- MS- Window base assembler - Linker / Editor / Debugger
Hardware (Emulator)	- CHOICE-Dr. (ICE main body) - EVA 81C50 (Evaluation board)
OTP programmer	- Universal single programmer - stand alone gang4 programmer



EVA81C50 B/D

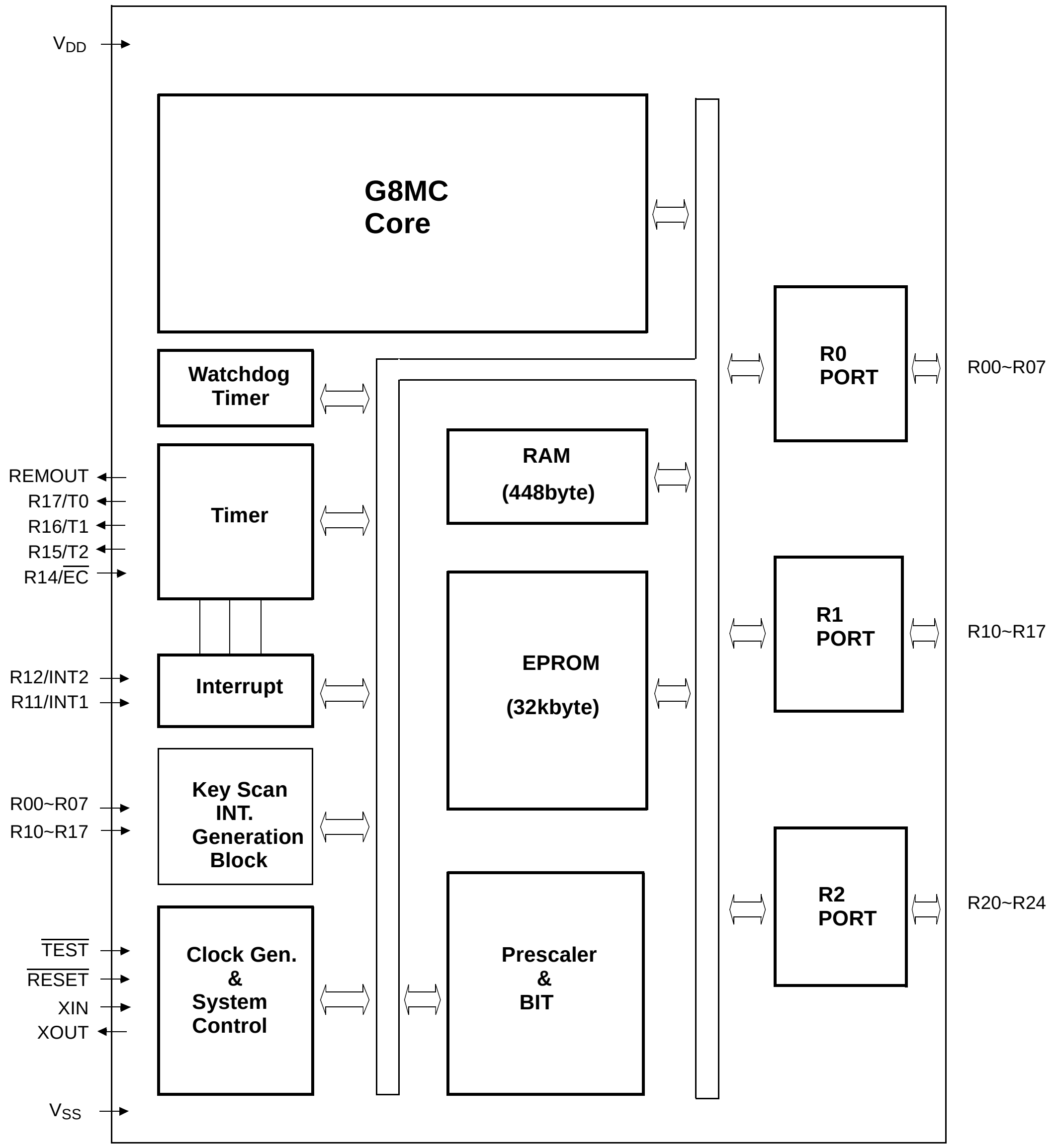


CHPOD810D-28SK

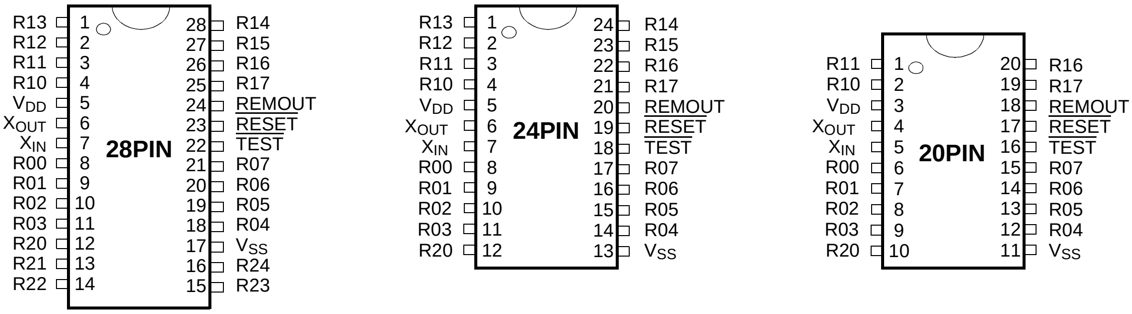
1.4 Ordering Information

Device name	Package	ROM Size (bytes)	RAM size
HMS81020ET 20 HMS81020ET 20D HMS81020ET 24SK HMS81020ET 24D HMS81020ET 28SK HMS81020ET 28D	20DIP 20SOP 24Skinny DIP 24SOP 28Skinny DIP 28SOP	20K bytes OTP	448 bytes
HMS81032ET 20 HMS81032ET 20D HMS81032ET 24SK HMS81032ET 24D HMS81032ET 28SK HMS81032ET 28D	20DIP 20SOP 24Skinny DIP 24SOP 28Skinny DIP 28SOP	32K bytes OTP	448 bytes

2. BLOCK DIAGRAM



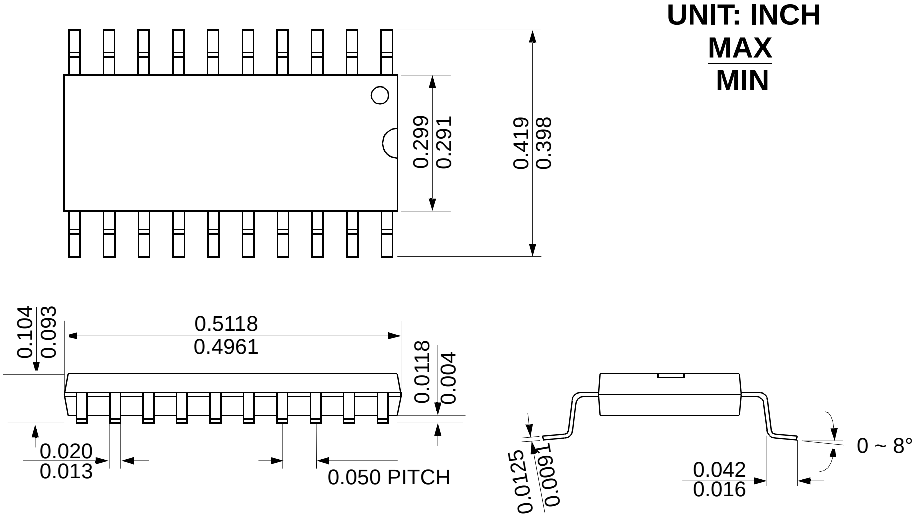
3. PIN ASSIGNMENT (Top View)



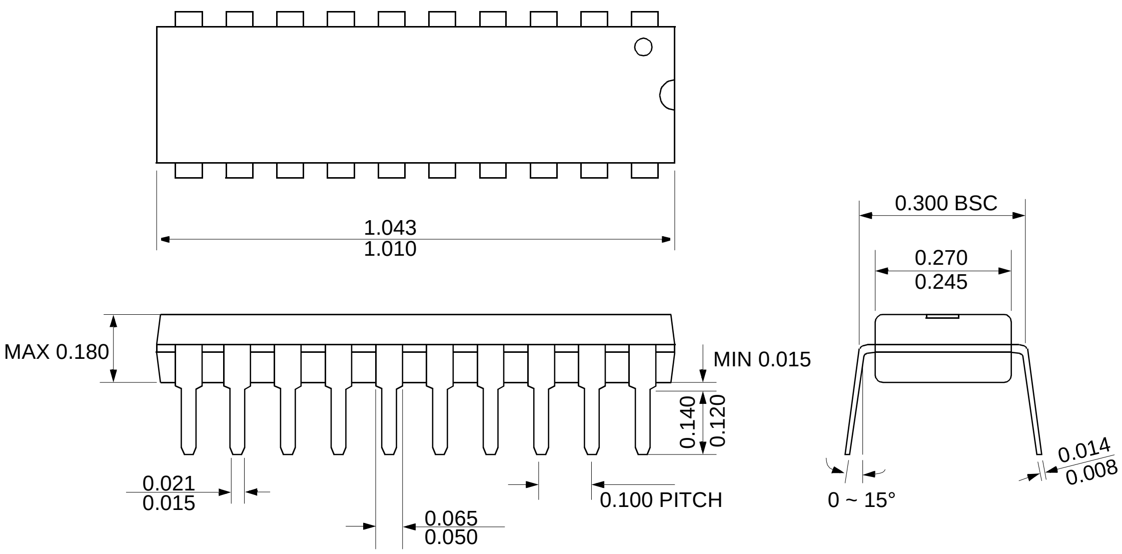
4. PACKAGE DIMENSION

UNIT: INCH
MAX
MIN

20 SOP

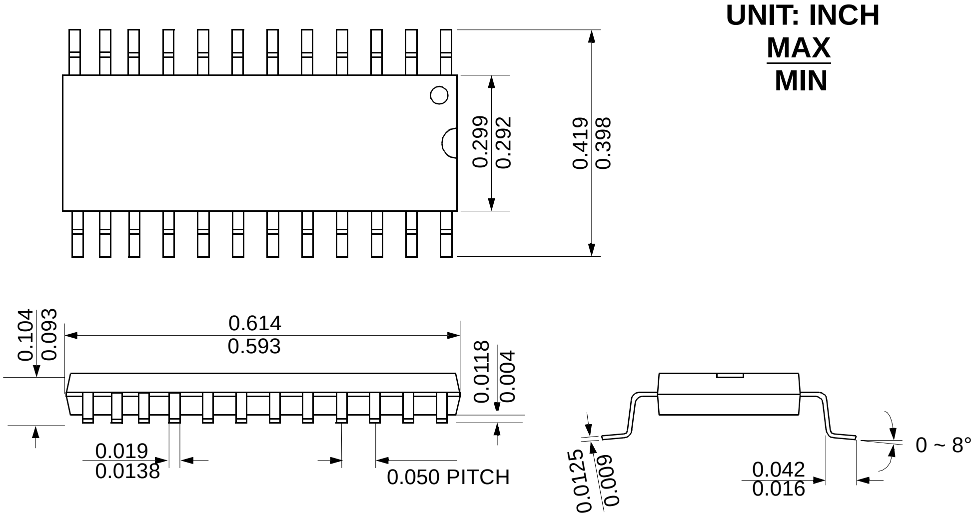


20 PDIP

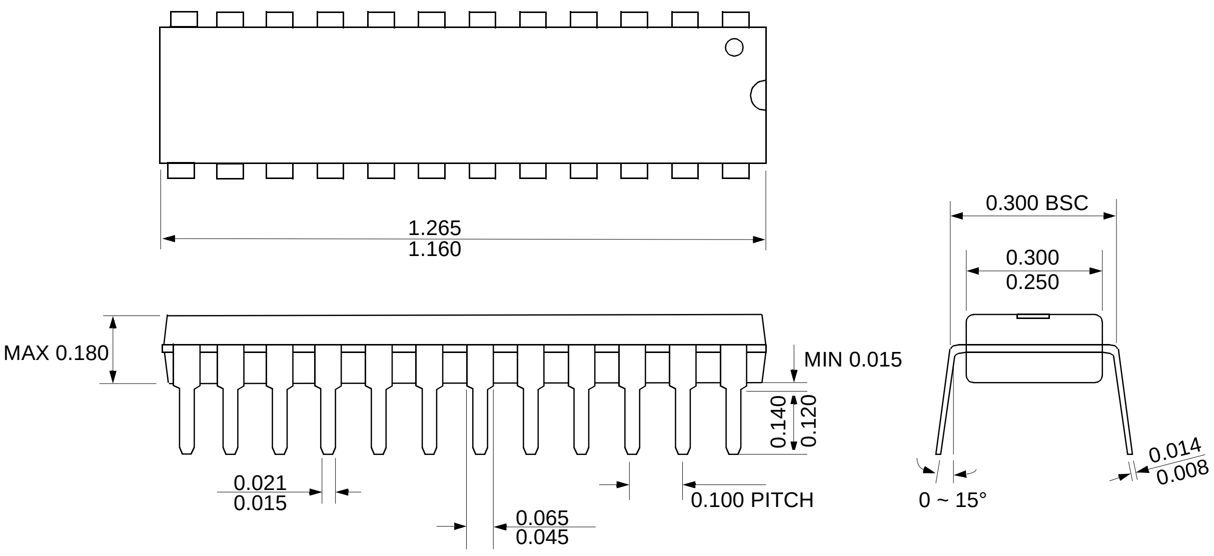


24 SOP

UNIT: INCH
MAX
MIN

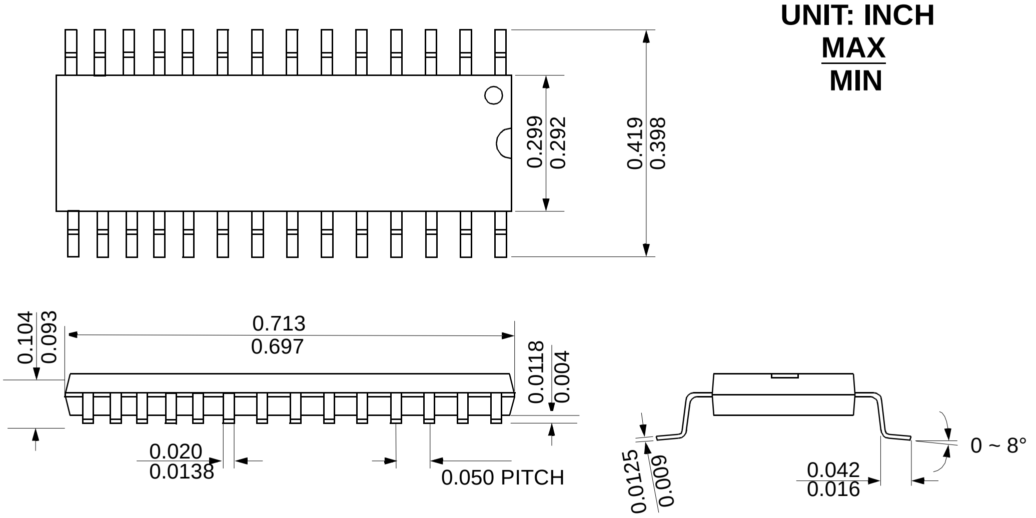


24 SKDIP

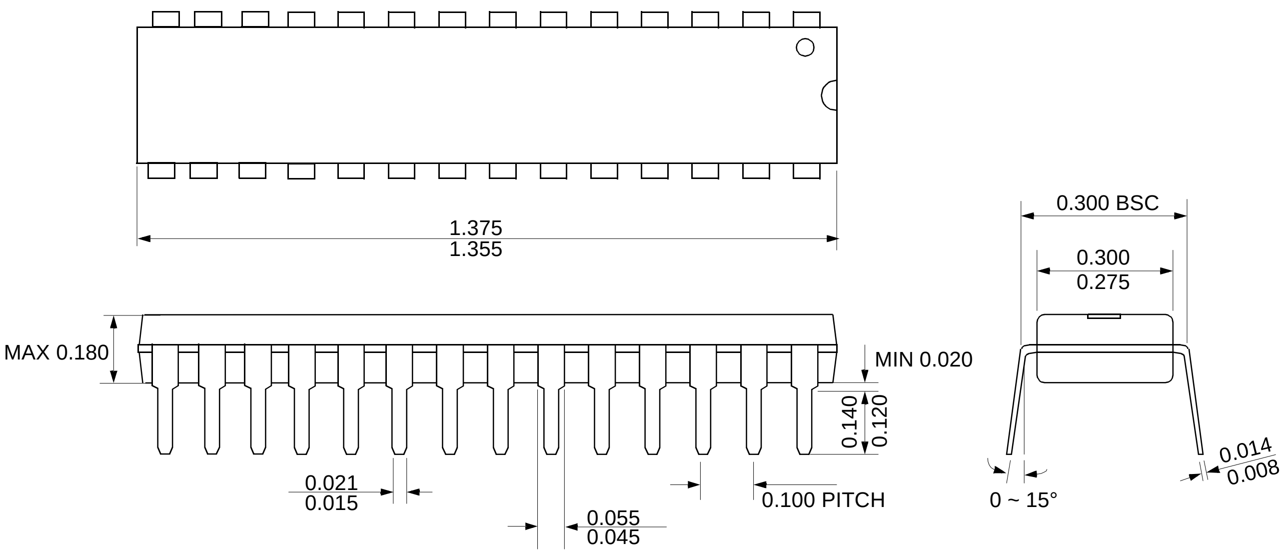


28 SOP

UNIT: INCH
MAX
MIN



28 SKDIP



5. PIN FUNCTION

V_{DD}: Supply voltage.

V_{SS}: Circuit ground.

$\overline{\text{TEST}}$: Used for shipping inspection of the IC. For normal operation, it should be connected to V_{DD}.

$\overline{\text{RESET}}$: Reset the MCU.

X_{IN}: Input to the inverting oscillator amplifier and input to the internal main clock operating circuit.

X_{OUT}: Output from the inverting oscillator amplifier.

R00~R07: R0 is an 8-bit CMOS bidirectional I/O port. R0 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs.

R10~R17: R1 is an 8-bit CMOS bidirectional I/O port. R1 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs.

In addition, R1 serves the functions of the various following special features.

Port pin	Alternate function
R11	INT1 (External Interrupt input 1)
R12	INT2 (External Interrupt input 2)
R14	$\overline{\text{EC}}$ (Event Counter input)
R15	T2 (Timer / Counter output 2)
R16	T1 (Timer / Counter output 1)
R17	T0 (Timer / Counter output 0)

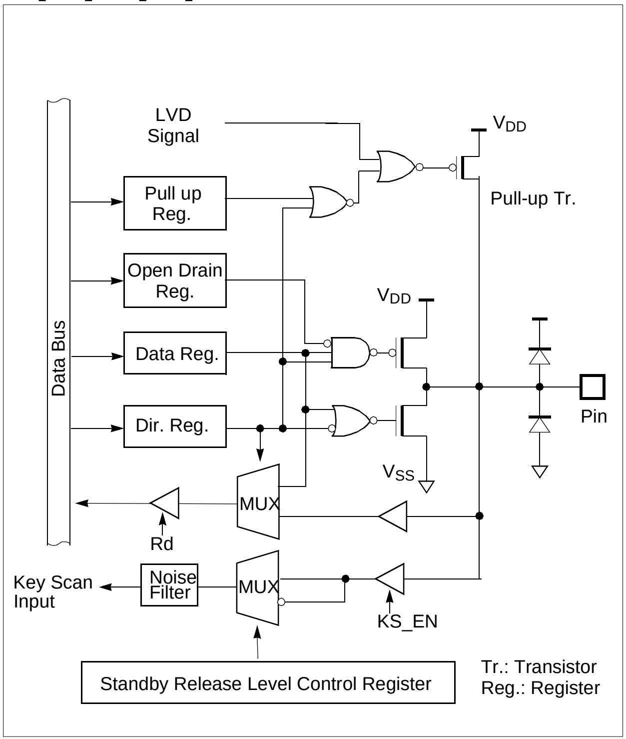
R20~R24: R2 is an 8-bit CMOS bidirectional I/O port. R2 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs.

5.1 Standard mode pin function

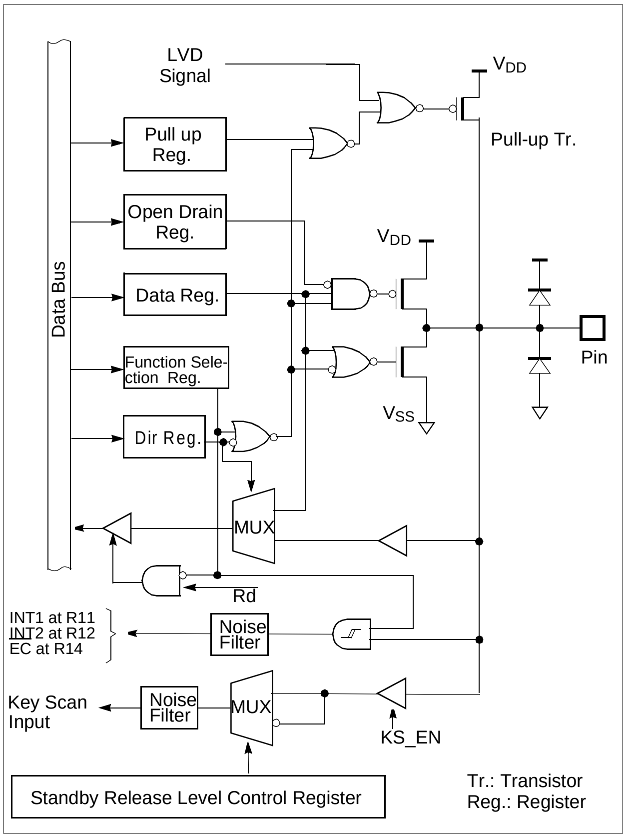
PIN NAME	INPUT/ OUTPUT	Function	@RESET	@STOP
R00	I/O	- Each bit of the port can be individually configured as an input or an output by user software - 8bit Push-pull output - 8bit CMOS input with pull-up resister (can be selectable by user software) - Can be programmable as key scan input or open drain output - Pull-up resisters are automatically disabled at output mode	INPUT	State of before Stop
R01	I/O			
R02	I/O			
R03	I/O			
R04	I/O			
R05	I/O			
R06	I/O			
R07	I/O			
R10	I/O	- Each bit of the port can be individually configured as an input or an output by user software - 8bit Push-pull output - 8bit CMOS input with pull-up resister (can be selectable by user software) - Can be programmable as key scan input or open drain output - Pull-up resisters are automatically disabled at output mode - Direct driving of LED(N-Tr.)	INPUT	State of before Stop
R11/INT1	I/O			
R12/INT2	I/O			
R13	I/O			
R14/$\overline{\text{EC}}$	I/O			
R15/T2	I/O			
R16/T1	I/O			
R17/T0	I/O			
R20	I/O	- Each bit of the port can be individually configured as an input or an output by user software - 5bit Push-pull output - 5bit CMOS input with pull-up resister (can be selectable by user software) - Pull-up resisters are automatically disabled at output mode - Direct driving of LED(N-Tr.)	INPUT	State of before Stop
R21	I/O			
R22	I/O			
R23	I/O			
R24	I/O			
X_{IN}	I	Oscillator input		Low
X_{OUT}	O	Oscillator output		Low
REMOUT	O	High current output	'L' output	'L' output
$\overline{\text{RESET}}$	I	Low active, includes pull-up resistor	'L' level	state of before stop
$\overline{\text{TEST}}$	I	Low active, includes pull-up resistor		
V_{DD}	P	Positive power supply		
V_{SS}	P	Ground		

6. PORT STRUCTURES

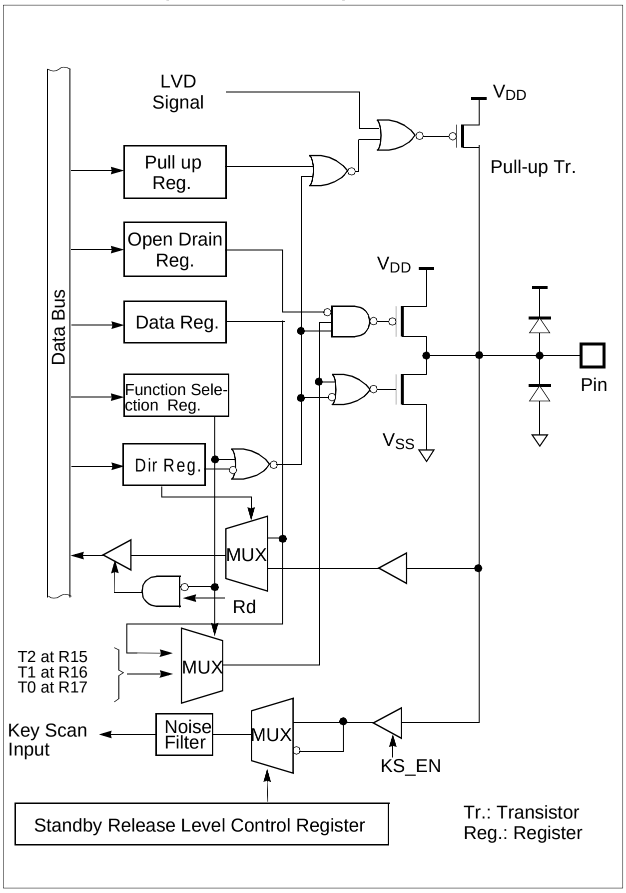
R0[0:7]/KS[0:7]



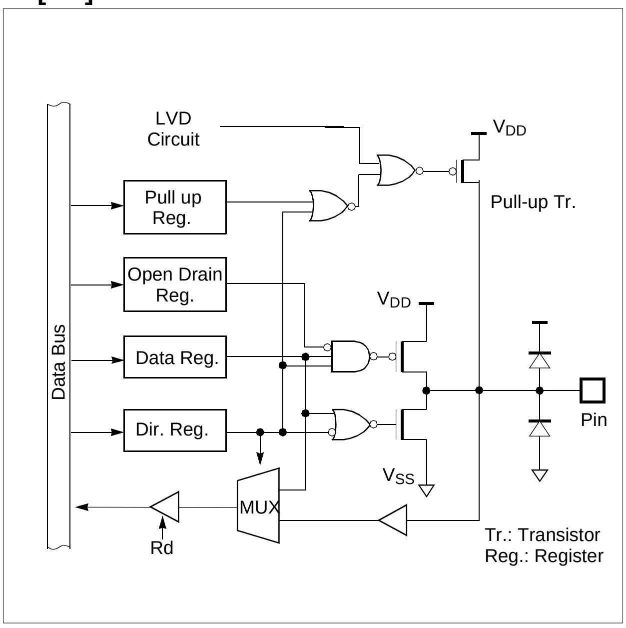
R10/KS10,R11/KS11/INT1, R12/KS12/INT2, R13/KS13,R14/KS14/EC



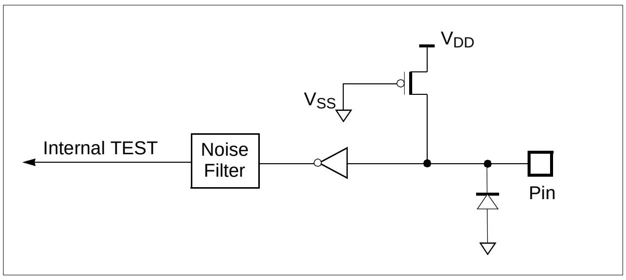
R15/KS15/T2, R16/KS16/T1, R17/KS17/T0



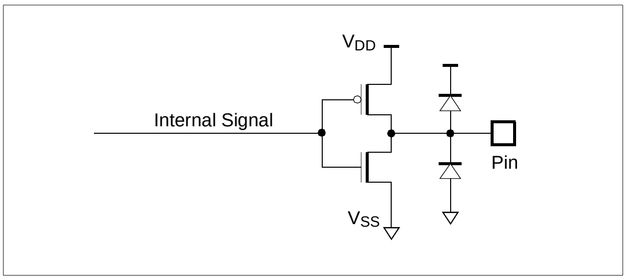
R2[0:4]



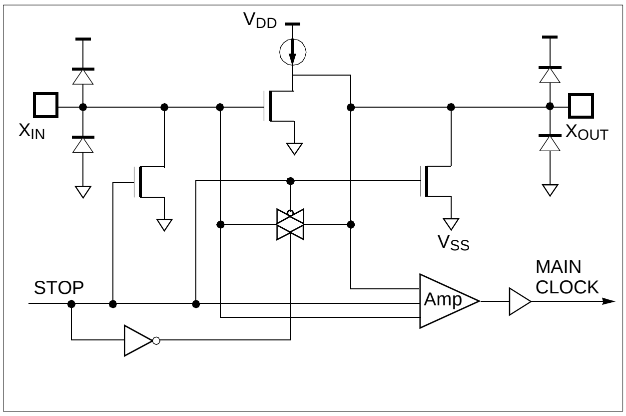
TEST



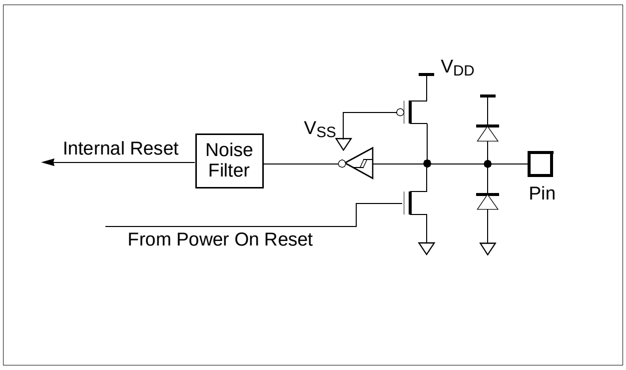
REMOUT



XIN, XOUT



RESET



7. ELECTRICAL CHARACTERISTICS

7.1 Absolute Maximum Ratings

Supply voltage .... -0.3 to +6.0 V

Input Voltage ....-0.3 to V_{DD}+0.3 V

Output Voltage ....-0.3 to V_{DD}+0.3 V

Operating Temperature.. 0~70°C

Storage Temperature .... -65~150°C

Power Dissipation...700 mW

Note: *Stresses above 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 above 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 Conditions

Parameter	Symbol	Specifications		Unit
		Min.	Max.	
Supply Voltage	V_{DD}	2.0	4.0	V
Operating Frequency	f_{XIN}	1.0	4.0	MHz
Operating Temperature	T_{OPR}	0	+70	°C

7.3 DC Electrical Characteristics

(T_A=-0~70°C, V_{DD}=2.0~4.0V, GND=0V)

Parameter	Symbol	Condition	Specifications			Unit
			Min.	Typ.	Max.	
High level input Voltage	V_{IH1}	R11,R12,R14(using the EXT. interrupt input),$\overline{RESET}$	0.8 V_{DD}	-	V_{DD}	V
	V_{IH2}	R0,R1,R2	0.7 V_{DD}	-	V_{DD}	V
Low level input Voltage	V_{IL1}	R11,R12,R14(using the EXT. interrupt input),$\overline{RESET}$	0	-	0.2 V_{DD}	V
	V_{IL2}	R0,R1,R2	0	-	0.3 V_{DD}	V
High level input Leakage Current	I_{IH}	R0,R1,R2,$\overline{RESET}$ (without pull-up),V_{IH}= V_{DD}	-	-	1	μA
Low level input Leakage Current	I_{IL}	R0,R1,R2,$\overline{RESET}$ (without pull-up),V_{IL}= 0	-	-	-1	μA
High level output Voltage	V_{OH1}	R0, I_{OH}=-0.5mA	V_{DD}-0.4	-	-	V
	V_{OH2}	R1, R2, I_{OH}=-1.0mA	V_{DD}-0.4	-	-	V
	V_{OH3}	XOUT,I_{OH}=-50μA	V_{DD}-0.9	-	-	V
Low level output Voltage	V_{OL1}	R0, I_{OL}=1mA	-	-	0.4	V
	V_{OL2}	R1, R2, I_{OL}=5mA	-	-	0.8	V
	V_{OL3}	XOUT,I_{OL}=50μA	-	-	0.8	V
High level output Leakage Current	I_{OHL}	R0,R1,R2, V_{OH}=V_{DD}	-	-	1	μA

Parameter	Symbol	Condition	Specifications			Unit
			Min.	Typ.	Max.	
Low level output Leakage Current	I_{OLL}	R0,R1,R2, V_{OL}= 0	-	-	-1	µA
High Level output current	I_{OH}	REMOUT, V_{OH} =2V	-24	-12	-5	mA
Low Level output current	I_{OL}	REMOUT, V_{OL} =1V	0.3	-	3	mA
Input pull-up current	I_p	R0,R1,R2, $\overline{RESET}$, $\overline{TEST}$,V_{DD}=3V	15	30	60	µA
Feedback resistance	RFD	X_{IN}, V_{DD} =3V	0.1	0.4	1.0	MΩ
Power Supply Current	I_{DD1}	Operating current, fxin=4Mhz, V_{DD}=4.0V	-	4	10	mA
	I_{DD2}	Operating current, fxin=4Mhz, V_{DD}=2.0V	-	2.4	6	mA
	I_{SLP1}	Sleep mode current, fxin=4Mhz, V_{DD}=4.0V	-	2	3	mA
	I_{SLP2}	Sleep mode current, fxin=4Mhz, V_{DD}=2.0V	-	1	2	mA
	I_{STP1}	Stop mode current, Oscillator Stop V_{DD}=4.0V	-	3	10	µA
	I_{STP2}	Stop mode current, Oscillator Stop V_{DD}=2.0V	-	2	8	µA
RAM retention supply voltage	V_{RET}	-	0.7	-	-	V
Low voltage detection voltage	LVD	To get into back-up mode, Ta=25°C	1.55	1.70	1.85	V

7.4 REMOUT Port Ioh Characteristics Graph

(T_A= 25°C)

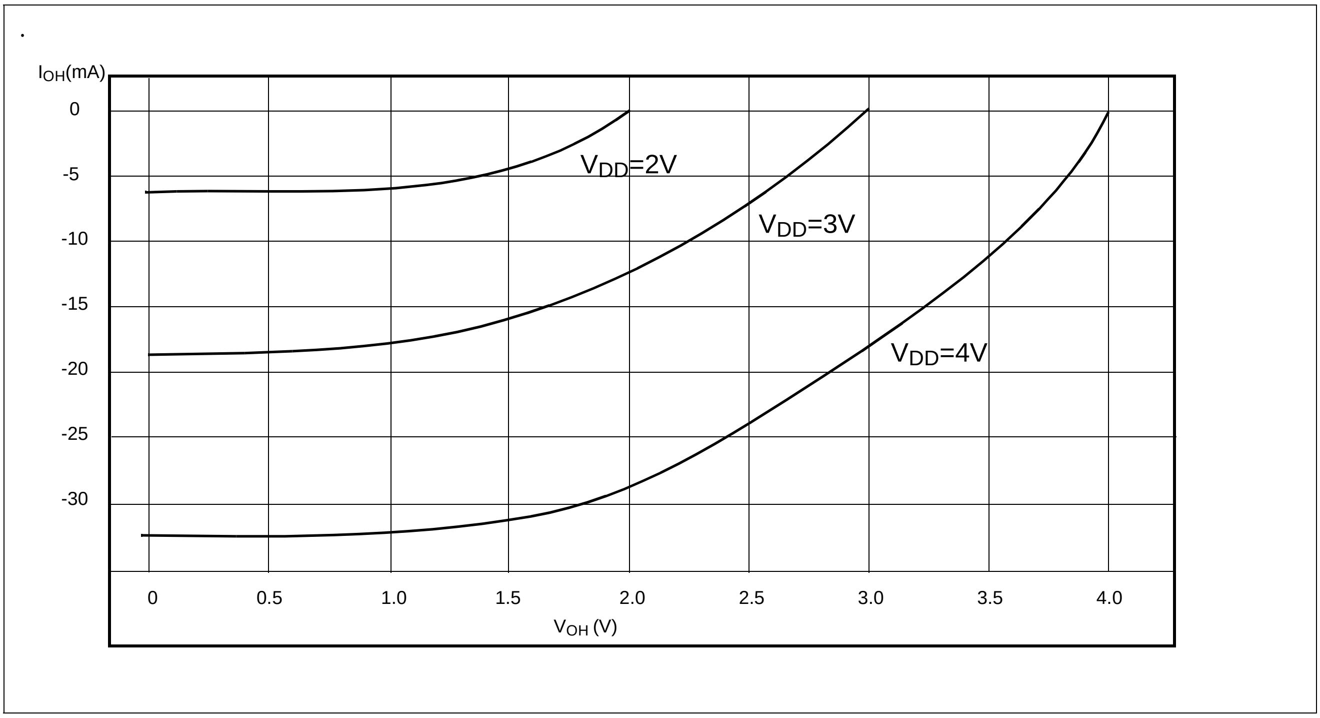


Figure 7-1 I_{OH} vs V_{OH}

7.5 REMOUT Port Iol Characteristics Graph

(T_A= 25°C)

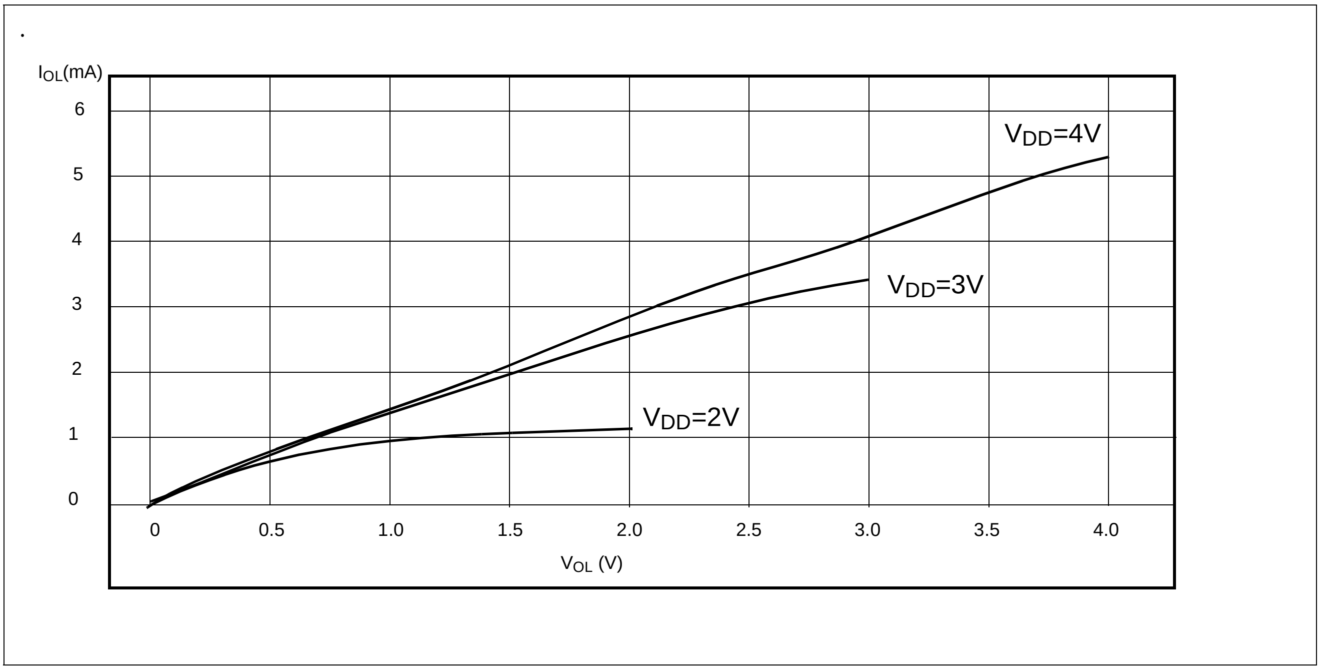


Figure 7-2 I_{OL} vs V_{OL}

7.6 AC Characteristics

(T_A=0~+70°C, V_{DD}=2.0~4.0V, V_{SS}=0V)

Parameter	Symbol	Pins	Specifications			Unit
			Min.	Typ.	Max.	
External clock input cycle time	t_{CP}	X_{IN}	250	500	1000	ns
System clock cycle time	t_{SYS}		500	1000	2000	ns
External clock pulse width High	t_{CPH}	X_{IN}	40	-	-	ns
External clock pulse width Low	t_{CPL}	X_{IN}	40	-	-	ns
External clock rising time	t_{RCP}	X_{IN}	-	-	40	ns
External clock falling time	t_{FCP}	X_{IN}	-	-	40	nS
Interrupt pulse width High	t_{IH}	INT1, INT2	2	-	-	t_{SYS}
Interrupt pulse width Low	t_{IL}	INT1, INT2	2	-	-	t_{SYS}
$\overline{\text{RESET}}$ Input pulse width low	t_{RSTL}	$\overline{\text{RESET}}$	8	-	-	t_{SYS}
Event counter input pulse width high	t_{ECH}	$\overline{\text{EC}}$	2	-	-	t_{SYS}
Event counter input pulse width low	t_{ECL}	$\overline{\text{EC}}$	2	-	-	t_{SYS}
Event counter input pulse rising time	t_{REC}	$\overline{\text{EC}}$	-	-	40	ns
Event counter input pulse falling time	t_{FEC}	$\overline{\text{EC}}$	-	-	40	ns

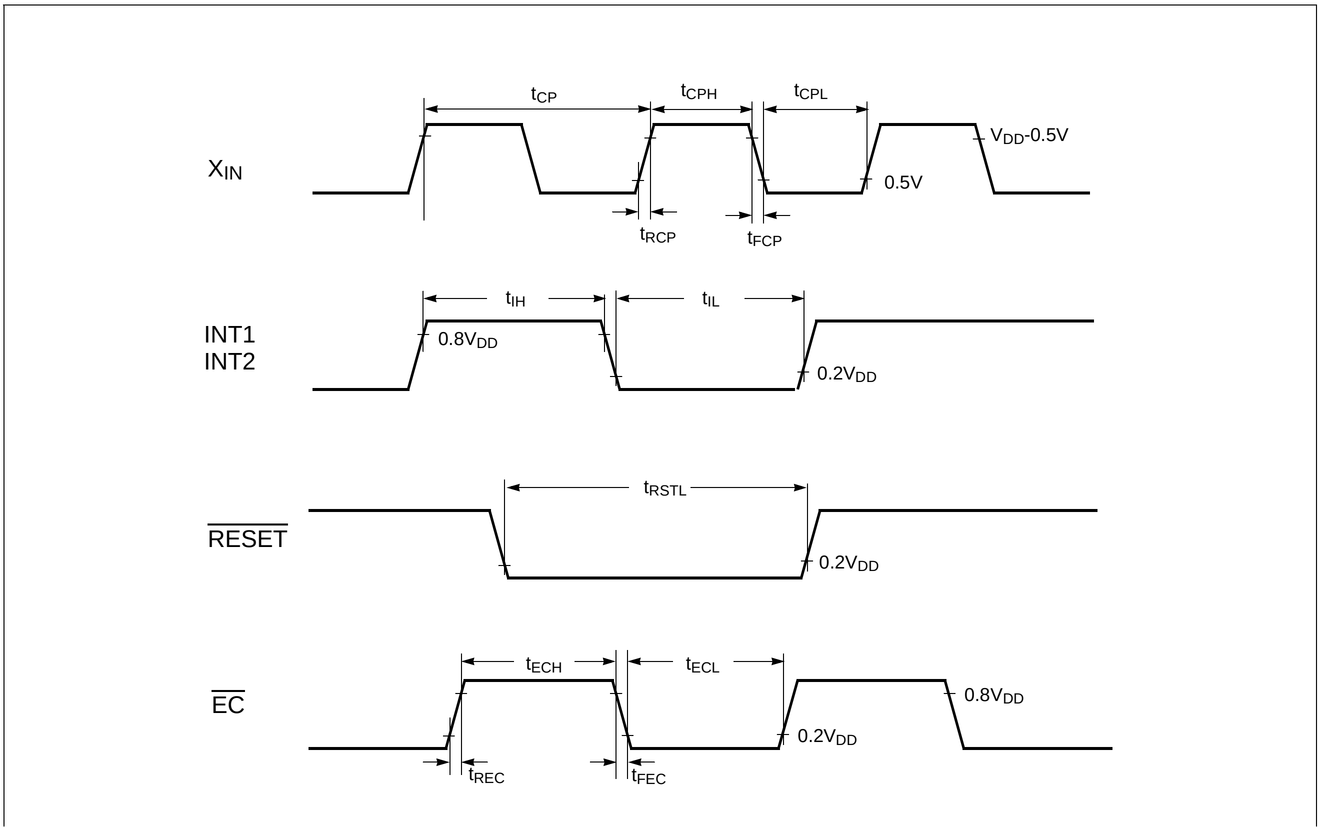


Figure 7-3 Timing Diagram

8. MEMORY ORGANIZATION

The HMS81020ET/HMS81032ET has separate address spaces for Program memory and Data Memory. Program memory can only be read, not written to. It can be up to 32K bytes of Program memory. Data memory can be read and written to up to 448 bytes including the stack area.

8.1 Registers

This device has six registers that are the Program Counter (PC), an Accumulator (A), two index registers (X, Y), the Stack Pointer (SP), and the Program Status Word (PSW). The Program Counter consists of 16-bit register.

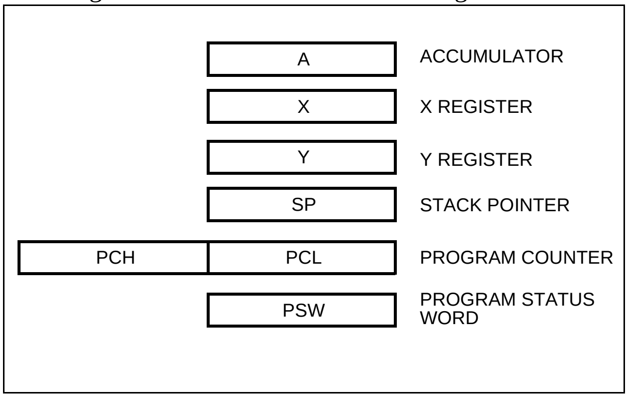


Figure 8-1 Configuration of Registers

Accumulator:

The Accumulator is the 8-bit general purpose register, used for data operation such as transfer, temporary saving, and conditional judgement, etc. The Accumulator can be used as a 16-bit register with Y Register as shown below.

In the case of multiplication instruction, execute as a multiplier register. After multiplication operation, the lower 8-bit of the result enters. (Y*A => YA). In the case of division instruction, execute as the lower 8-bit of dividend. After division operation, quotient enters.

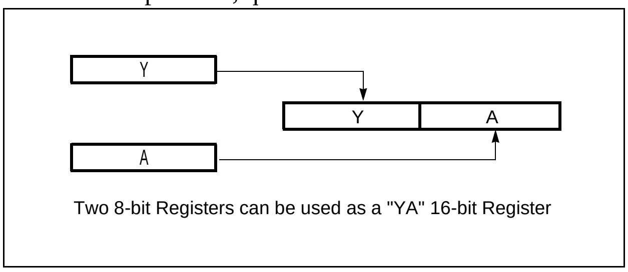


Figure 8-2 Configuration of YA 16-bit Register

X, Y Registers:

In the addressing mode which uses these index registers, the register contents are added to the specified address, which becomes the actual address. These modes are extremely effective for referencing subroutine tables and memory tables. The index registers also have increment, decrement, comparison and data transfer functions, and they can be used as simple accumulators.

- **X Register**

In the case of division instruction, execute as register.

- **Y Register**

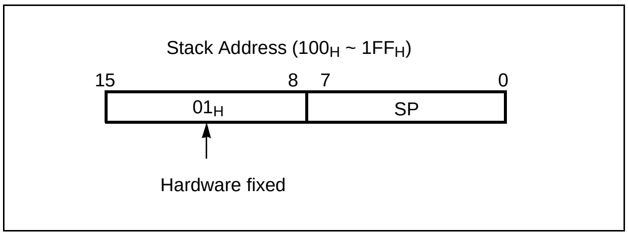
In the case of 16-bit operation instruction, execute as the upper 8-bit of YA. (16-bit accumulator). In the case of multiplication instruction, execute as a multiplicand register. After multiplication operation, the upper 8-bit of the result enters. In the case of division instruction, execute as the upper 8-bit of dividend. After division operation, remains enters. Y register can be used as loop counter of conditional branch command. (e.g.DBNE Y, rel)

Stack Pointer:

The Stack Pointer is an 8-bit register used for occurrence interrupts, calling out subroutines and PUSH, POP, RETI, RET instruction. Stack Pointer identifies the location in the stack to be accessed (save or restore).

Generally, SP is automatically updated when a subroutine call is executed or an interrupt is accepted. However, if it is used in excess of the stack area permitted by the data memory allocating configuration, the user-processed data may be lost. The SP is post-decremented when a subroutine call or a push instruction is executed, or when an interrupt is accepted. The SP is pre-incremented when a return or a pop instruction is executed.

The stack can be located at any position within 100_H to $1FF_H$ of the internal data memory. The SP is not initialized by hardware, requiring to write the initial value (the location with which the use of the stack starts) by using the initialization routine. Normally, the initial value of "FF_H" is used.



> ***Caution:***
>
> ***The Stack Pointer must be initialized by software because its value is undefined after RESET.***
>
> Example: To initialize the SP
>
> ```
> LDX    #0FFH
> TXSP                    ; SP ← FFH
> ```

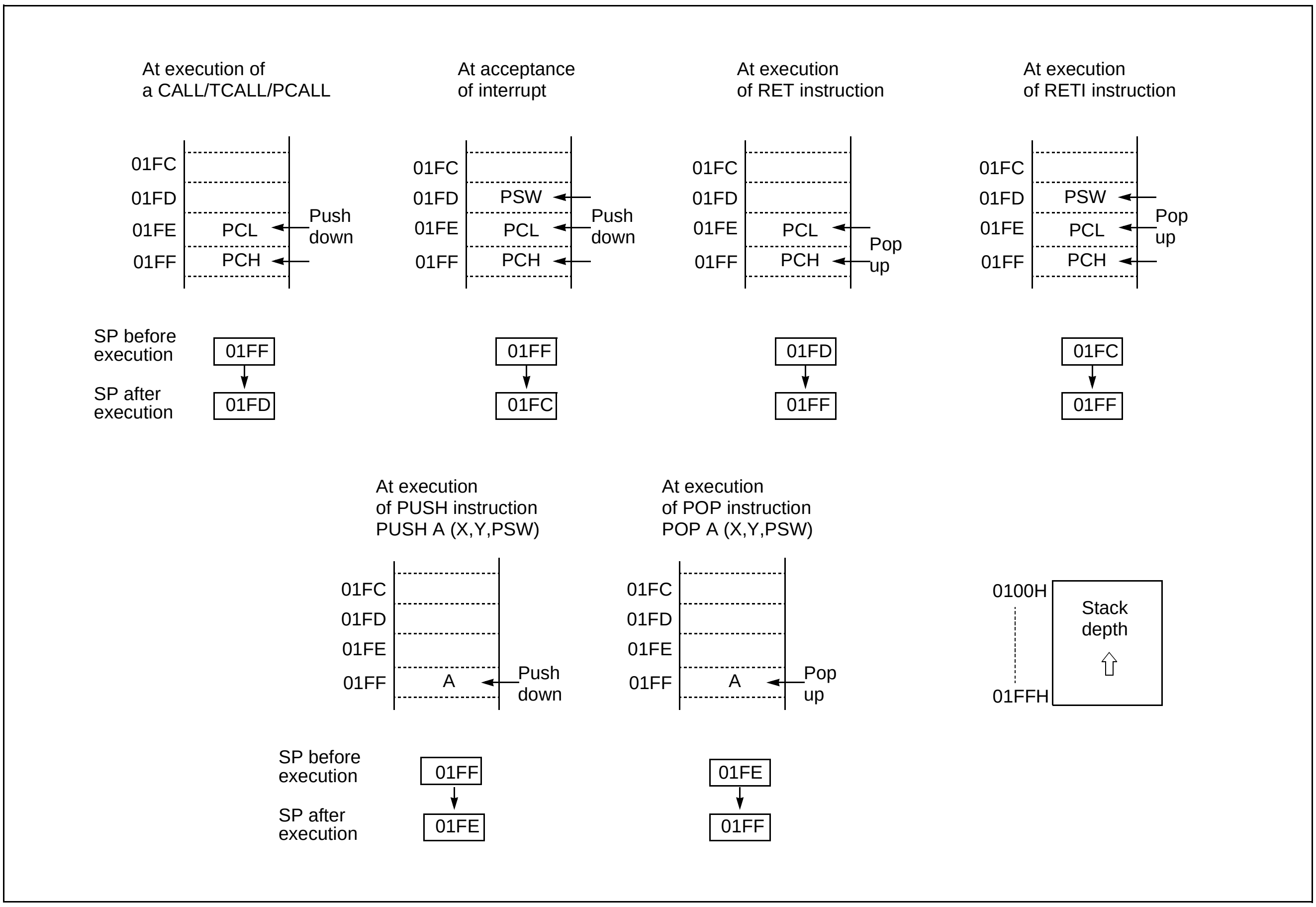


Figure 8-3 Stack Operation

Program Counter:

The Program Counter is a 16-bit wide which consists of two 8-bit registers, PCH and PCL. This counter indicates the address of the next instruction to be executed. In reset state, the program counter has reset routine address (PC_H:$0FF_H$, PC_L:$0FE_H$).

Program Status Word:

The Program Status Word (PSW) contains several bits that reflect the current state of the CPU. The PSW is described in Figure 8-4. It contains the Negative flag, the Overflow flag, the Break flag the Half Carry (for BCD operation), the Interrupt enable flag, the Zero flag, and the Carry flag.

[Carry flag C]

This flag stores any carry or borrow from the ALU of CPU after an arithmetic operation and is also changed by the Shift Instruction or Rotate Instruction.

[Zero flag Z]

This flag is set when the result of an arithmetic operation or data transfer is "0" and is cleared by any other result.

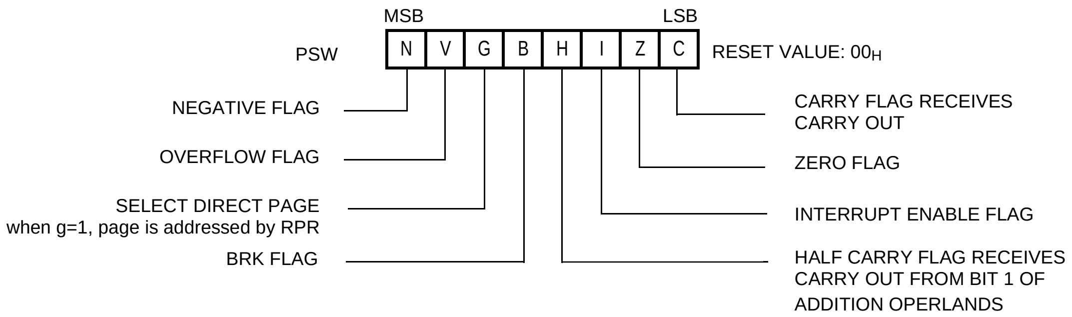


Figure 8-4 PSW (Program Status Word) Register

[Interrupt disable flag I]

This flag enables/disables all interrupts except interrupt caused by Reset or software BRK instruction. All interrupts are disabled when cleared to "0". This flag immediately becomes "0" when an interrupt is served. It is set by the EI instruction and cleared by the DI instruction.

[Half carry flag H]

After operation, this is set when there is a carry from bit 3 of ALU or there is no borrow from bit 4 of ALU. This bit can not be set or cleared except CLRV instruction with Overflow flag (V).

[Break flag B]

This flag is set by software BRK instruction to distinguish BRK from TCALL instruction with the same vector address.

[Direct page flag G]

This flag assigns RAM page for direct addressing mode. In the direct addressing mode, addressing area is from zero page 00_H to $0FF_H$ when this flag is "0". If it is set to "1", addressing area is 1 Page. It is set by SETG instruction and cleared by CLRG.

[Overflow flag V]

This flag 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.

[Negative flag N]

This flag is set to match the sign bit (bit 7) status of the result of a data or arithmetic operation. When the BIT instruction is executed, bit 7 of memory is copied to this flag.

8.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 20K/32K bytes program memory space only physically implemented. Accessing a location above $FFFF_H$ will cause a wrap-around to 0000_H.

Figure 8-5, shows a map of Program Memory. After reset, the CPU begins execution from reset vector which is stored in address $FFFE_H$ and $FFFF_H$ as shown in Figure 8-6.

As shown in Figure 8-5, each area is assigned a fixed location in Program Memory. Program Memory area contains the user program.

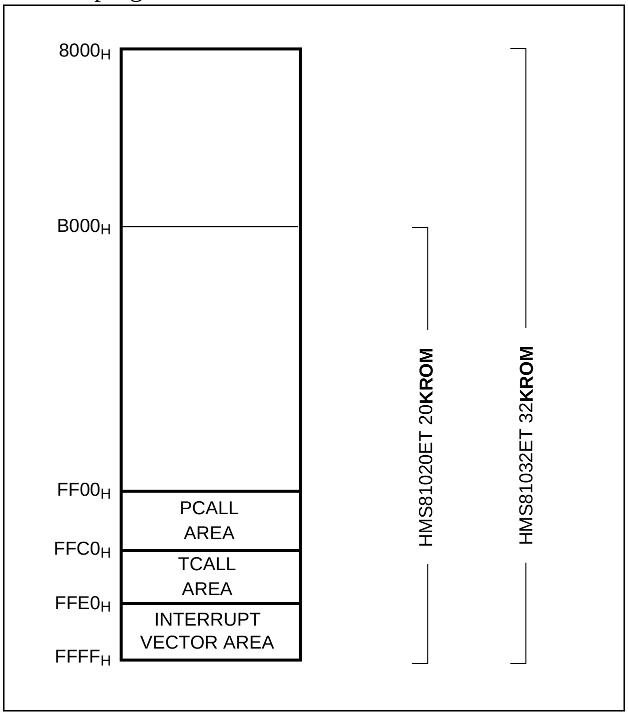


Figure 8-5 Program Memory Map

Page Call (PCALL) area contains subroutine program to reduce program byte length by using 2 bytes PCALL instead of 3 bytes CALL instruction. If it is frequently called, it is more useful to save program byte length.

Table Call (TCALL) causes the CPU to jump to each TCALL address, where it commences the execution of the service routine. The Table Call service area spaces 2-byte for every TCALL: $0FFC0_H$ for TCALL15, $0FFC2_H$ for TCALL14, etc., as shown in Figure 8-7.

Example: Usage of TCALL

```
LDA     #5
        TCALL  0FH           ;1BYTE INSTRUCTION
        :                    ;INSTEAD OF 2 BYTES
        :                    ;NORMAL CALL
;
;TABLE CALL ROUTINE
;
FUNC_A: LDA    LRG0
        RET
;
FUNC_B: LDA    LRG1      ②  ①
        RET
;
;TABLE CALL ADD. AREA
;
        ORG    0FFC0H        ;TCALL ADDRESS AREA
        DW     FUNC_A
        DW     FUNC_B
```

The interrupt causes the CPU to jump to specific location, where it commences the execution of the service routine. The External interrupt 0, for example, is assigned to location $0FFFA_H$. The interrupt service locations spaces 2-byte interval: $0FFF8_H$ and $0FFF9_H$ for External Interrupt 1, $0FFFA_H$ and $0FFFB_H$ for External Interrupt 0, etc.

Any area from $0FF00_H$ to $0FFFF_H$, if it is not going to be used, its service location is available as general purpose Program Memory.

Address	Vector Area Memory
$0FFDE_H$	S/W Interrupt Vector Area
E0	-
E2	-
E4	-
E6	Basic Interval Timer Interrupt Vector Area
E8	Watch Dog Timer Interrupt Vector Area
EA	-
EC	-
EE	Timer2 Interrupt Vector Area
F0	Timer1 Interrupt Vector Area
F2	Timer0 Interrupt Vector Area
F4	-
F6	External Interrupt 2 Vector Area
F8	External Interrupt 1 Vector Area
FA	Key Scan Interrupt Vector Area
FC	-
FE	RESET Vector Area

NOTE:
"-" means reserved area.

Figure 8-6 Interrupt Vector Area

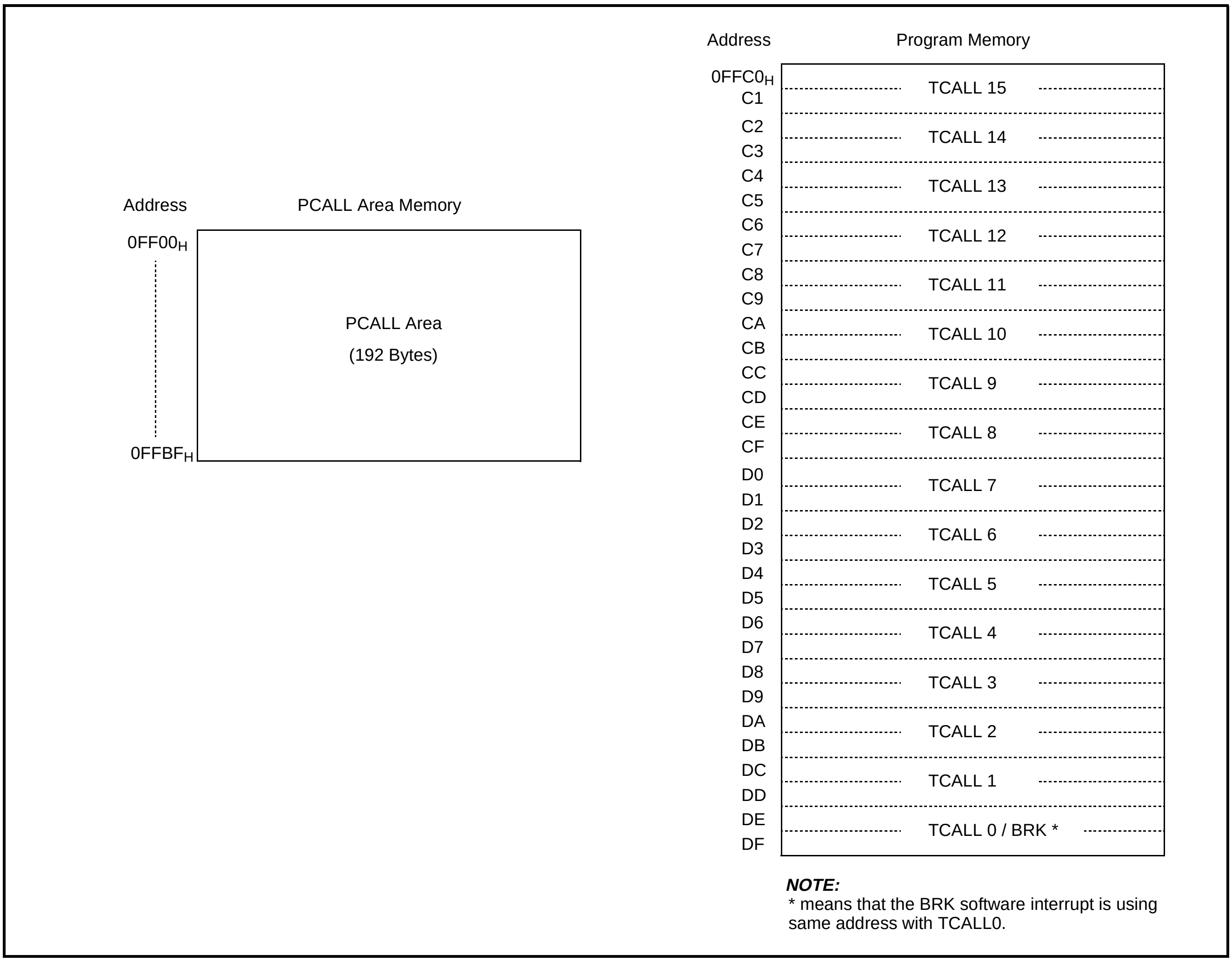
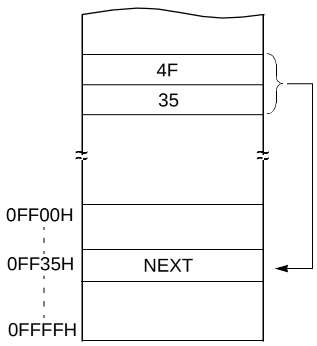


Figure 8-7 PCALL and TCALL Memory Area

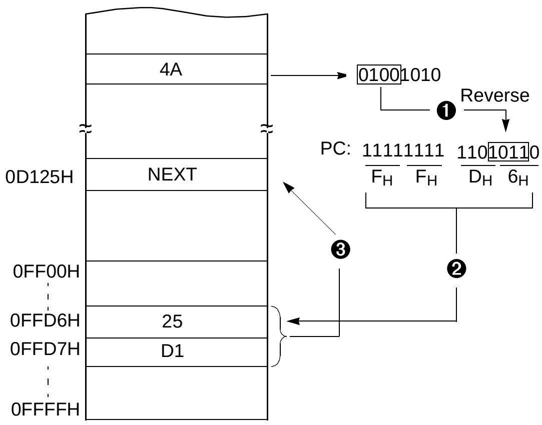
PCALL→ rel

```
4F35      PCALL  35H
```



TCALL→ n

```
4A        TCALL  4
```



Example: The usage software example of Vector address and the initialize part.

```
          ORG     0FFE0H

          DW      NOT_USED
          DW      NOT_USED
          DW      NOT_USED
          DW      BIT_INT                 ; BIT
          DW      WDT_INT                 ; Watch Dog Timer
          DW      NOT_USED
          DW      NOT_USED
          DW      TMR2_INT                ; Timer-2
          DW      TMR1_INT                ; Timer-1
          DW      TMR0_INT                ; Timer-0
          DW      NOT_USED                ;
          DW      INT2                    ; Int.2
          DW      INT1                    ; Int.1
          DW      KEY_INT                 ; Key Scan
          DW      NOT_USED                ;
          DW      RESET                   ; Reset


          ORG     08000H                  ;HMS81032E Program start address

;*********************************************
;              MAIN     PROGRAM            *
;*********************************************
;
RESET:    NOP
          CLRG
          DI                              ;Disable All Interrupts
          LDX     #0
RAM_CLR:  LDA     #0                      ;RAM Clear(!0000H->!00BFH)
          STA     {X}+
          CMPX    #0C0H
          BNE     RAM_CLR                 ;

          LDX     #0FFH                   ;Stack Pointer Initialize
          TXSP

          LDM     R0, #0                  ;Normal Port 0
          LDM     R0DD,#1000_0010B        ;Normal Port Direction
          LDM     P0PC,#1000_0010B        ;Pull Up Selection Set
          LDM     PMR1,#0000_0010B        ;R1 port / int
          :
          :
          LDM     CKCTLR,#0011_1101B      ;WDT ON, 16mS Time delay after stop mode release
          :
          :
```

8.3 Data Memory

Figure 8-8 shows the internal Data Memory space available. Data Memory is divided into 3 groups, a user RAM, control registers, Stack.

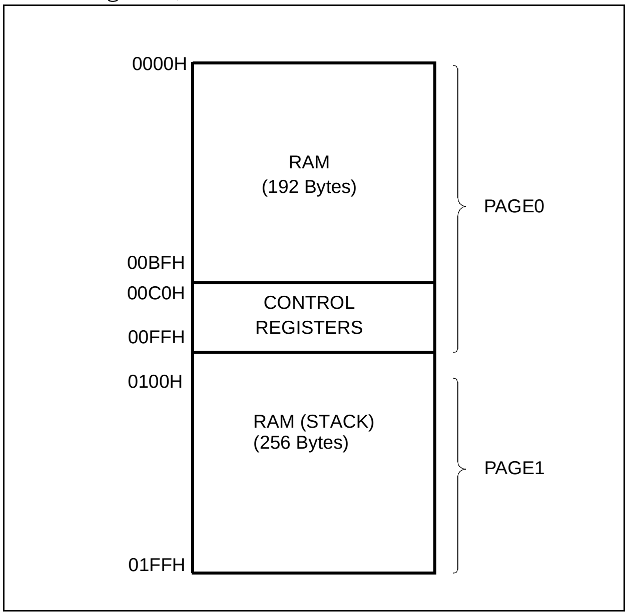


Figure 8-8 Data Memory Map

User Memory

The HMS81020ET/HMS81032ET has 448 × 8 bits for the user memory (RAM).

Control Registers

The control registers are used by the CPU and Peripheral function blocks for controlling the desired operation of the device. Therefore these registers contain control and status bits for the interrupt system, the timer/ counters, analog to digital converters and I/O ports. The control registers are in address range of $0C0_H$ to $0FF_H$.

Note that unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

More detailed informations of each register are explained in each peripheral section.

Note: *Write only registers can not be accessed by bit manipulation instruction. Do not use read-modify-write instruction. Use byte manipulation instruction.*

Example; To write at CKCTLR

```
        LDM    CLCTLR,#09H ;Divide ratio ÷16
```

Stack Area

The stack provides the area where the return address is saved before a jump is performed during the processing routine at the execution of a subroutine call instruction or the acceptance of an interrupt.

When returning from the processing routine, executing the subroutine return instruction [RET] restores the contents of the program counter from the stack; executing the interrupt return instruction [RETI] restores the contents of the program counter and flags.

The save/restore locations in the stack are determined by the stack pointed (SP). The SP is automatically decreased after the saving, and increased before the restoring. This means the value of the SP indicates the stack location number for the next save. Refer to Figure 8-3 on page 17.

8.4 List for Control Registers

Address	Function Register	Symbol	Read Write	RESET Value
00C0h	PORT R0 DATA REG.	R0	R/W	undefined
00C1h	PORT R0 DATA DIRECTION REG.	R0DD	W	00000000b
00C2h	PORT R1 DATA REG.	R1	R/W	undefined
00C3h	PORT R1 DATA DIRECTION REG.	R1DD	W	00000000b
00C4h	PORT R2 DATA REG.	R2	R/W	undefined
00C5h	PORT R2 DATA DIRECTION REG.	R2DD	W	- - -00000b
00C6h	Reserved			
00C7h	CLOCK CONTROL REG.	CKCTLR	W	- -110111b
	BASIC INTERVAL REG.	BITR	R	undefined
00C8h	WATCH DOG TIMER REG.	WDTR	W	-0001111b
00C9h	PORT R1 MODE REG.	PMR1	W	00000000b
00CAh	INT. MODE REG.	IMOD	R/W	- -000000b
00CBh	EXT. INT. EDGE SELECTION	IEDS	W	- -0000- -b
00CCh	INT. ENABLE REG. LOW	IENL	R/W	-00- - - - -b
00CDh	INT. REQUEST FLAG REG. LOW	IRQL	R/W	-00- - - - -b
00CEh	INT. ENABLE REG. HIGH	IENH	R/W	000-000-b
00CFh	INT. REQUEST FLAG REG. HIGH	IRQH	R/W	000-000-b
00D0h	TIMER0 (16bit) MODE REG.	TM0	R/W	00000000b
00D1h	TIMER1 (8bit) MODE REG.	TM1	R/W	00000000b
00D2h	TIMER2 (8bit) MODE REG.	TM2	R/W	00000000b
00D3h	TIMER0 HIGH-MSB DATA REG.	T0HMD	W	undefined
00D4h	TIMER0 HIGH-LSB DATA REG.	T0HLD	W	undefined
00D5h	TIMER0 LOW-MSB DATA REG.	T0LMD	W	undefined
	TIMER0 HIGH-MSB COUNT REG.	T0HMC	R	undefined
00D6h	TIMER0 LOW-LSB DATA REG.	T0LLD	W	undefined
	TIMER0 LOW-LSB COUNT REG.	T0LLC	R	undefined
00D7h	TIMER1 HIGH DATA REG.	T1HD	W	undefined
00D8h	TIMER1 LOW DATA REG.	T1LD	W	undefined
	TIMER1 LOW COUNT REG.		R	undefined
00D9h	TIMER2 DATA REG.	T2DR	W	undefined
	TIMER2 COUNT REG.		R	undefined
00DAh	TIMER0 / TIMER1 MODE REG.	TM01	R/W	00000000b
00DBh	Reserved			
00DCh	STANDBY MODE RELEASE REG0	SMPR0	W	00000000b
00DDh	STANDBY MODE RELEASE REG0	SMPR1	W	00000000b
00DEh	PORT R1 OPEN DRAIN ASSIGN REG.	R1ODC	W	00000000b

00DFh	PORT R2 OPEN DRAIN ASSIGN REG.	R2ODC	W	- - -00000b
00E0h	Reserved			
00E1h	Reserved			
00E2h	Reserved			
00E3h	Reserved			
00E4h	PORT R0 OPEN DRAIN ASSIGN REG.	R0ODC	W	00000000b
00E5h	Reserved			
00E6h	Reserved			
00E7h	Reserved			
00E8h	Reserved			
00E9h	Reserved			
00EAh	Reserved			
00EBh	Reserved			
00ECh	Reserved			
00EDh	Reserved			
00EEh	Reserved			
00EFh	LOW VOLTAGE INDICATION REG.	LVIR	R	- - - - - -00b
00F0h	SLEEP MODE REG.	SLPM	W	- - - - - - -0b
00F1h	LVD CONTROL REG.	LVDC	W	- - - - 0- - -b
00F2h	Reserved			
00F3h	Reserved			
00F4h	Reserved			
00F5h	Reserved			
00F6h	STANDBY RELEASE LEVEL CONT. REG. 0	SRLC0	W	00000000b
00F7h	STANDBY RELEASE LEVEL CONT. REG. 1	SRLC1	W	00000000b
00F8h	PORT R0 PULL-UP REG. CONT. REG.	R0PC	W	00000000b
00F9h	PORT R1 PULL-UP REG. CONT. REG.	R1PC	W	00000000b
00FAh	PORT R2 PULL-UP REG. CONT. REG.	R2PC	W	- - - 00000b
00FBh	Reserved			
00FCh	Reserved			
00FDh	Reserved			
00FEh	Reserved			
00FFh	Reserved			

W	Registers are controlled by byte manipulation instruction such as LDM etc., do not use bit manipulation instruction such as SET1, CLR1 etc. If bit manipulation instruction is used on these registers, content of other seven bits are may varied to unwanted value.
R/W	Registers are controlled by both bit and byte manipulation instruction.

- : this bit location is reserved.

LVIR,LVDC are newly added in HMS81032ET, but not supported in MASK and EVA version

8.5 Addressing Mode

The HMS81020ET/HMS81032ET uses six addressing modes;

- **Register addressing**
- **Immediate addressing**
- **Direct page addressing**
- **Absolute addressing**
- **Indexed addressing**
- **Register-indirect addressing**

(1) Register Addressing

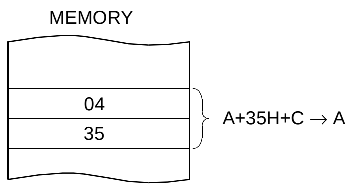
Register addressing accesses the A, X, Y, C and PSW.

(2) Immediate Addressing → #imm

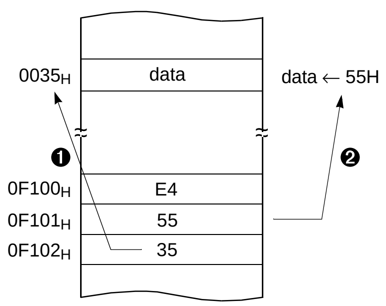
In this mode, second byte (operand) is accessed as a data immediately.

Example:

```
0435     ADC    #35H
```




```
E45535   LDM    35H,#55H
```

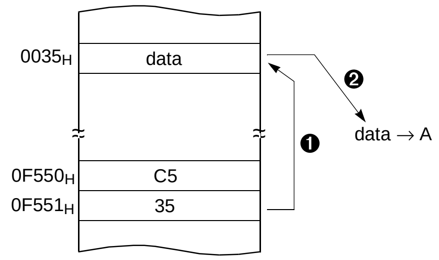


(3) Direct Page Addressing → dp

In this mode, a address is specified within direct page.

Example;

```
C535     LDA    35H          ;A ←RAM[35H]
```



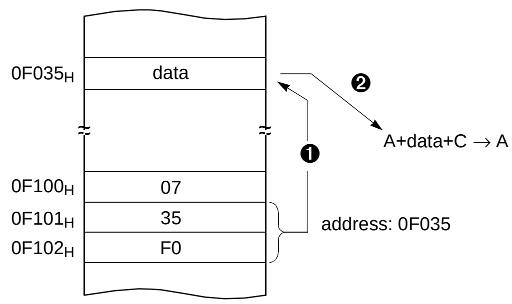
(4) Absolute Addressing → !abs

Absolute addressing sets corresponding memory data to Data, i.e. second byte(Operand I) of command becomes lower level address and third byte (Operand II) becomes upper level address.
With 3 bytes command, it is possible to access to whole memory area.

ADC, AND, CMP, CMPX, CMPY, EOR, LDA, LDX, LDY, OR, SBC, STA, STX, STY

Example;

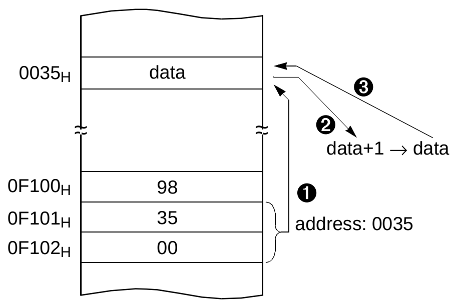
```
0735F0   ADC    !0F035H     ;A ←ROM[0F035H]
```



The operation within data memory (RAM)
ASL, BIT, DEC, INC, LSR, ROL, ROR

Example; Addressing accesses the address 0135_H.

```
983500   INC    !0035H      ;A ←RAM[035H]
```



(5) Indexed Addressing

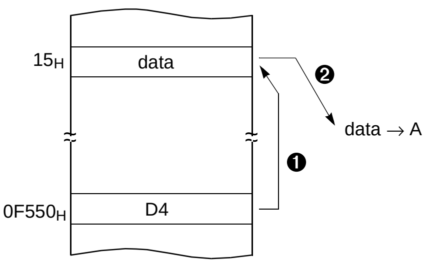
X indexed direct page (no offset) → {X}

In this mode, a address is specified by the X register.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA, XMA

Example; $X=15_H$

```
D4       LDA    {X}         ;ACC←RAM[X].
```



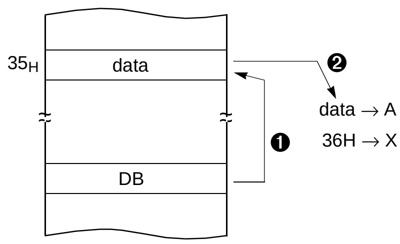
X indexed direct page, auto increment→ {X}+

In this mode, a address is specified within direct page by the X register and the content of X is increased by 1.

LDA, STA

Example; $X=35_H$

```
DB       LDA    {X}+
```



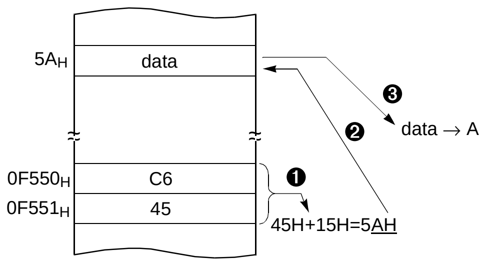
X indexed direct page (8 bit offset) → dp+X

This address value is the second byte (Operand) of command plus the data of X-register. And it assigns the memory in Direct page.

ADC, AND, CMP, EOR, LDA, LDY, OR, SBC, STA STY, XMA, ASL, DEC, INC, LSR, ROL, ROR

Example; $X=015_H$

```
C645     LDA    45H+X
```



Y indexed direct page (8 bit offset) → dp+Y

This address value is the second byte (Operand) of command plus the data of Y-register, which assigns Memory in Direct page.

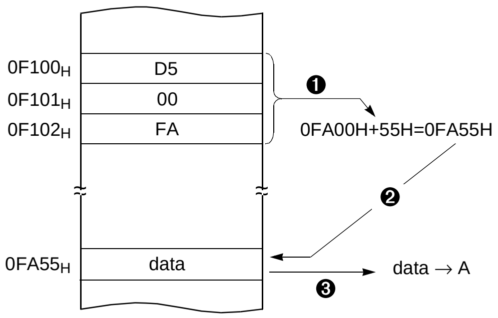
This is same with above (2). Use Y register instead of X.

Y indexed absolute →!abs+Y

Sets the value of 16-bit absolute address plus Y-register data as Memory. This addressing mode can specify memory in whole area.

Example; Y=55_H

```
D500FA   LDA    !0FA00H+Y
```



(6) Indirect Addressing

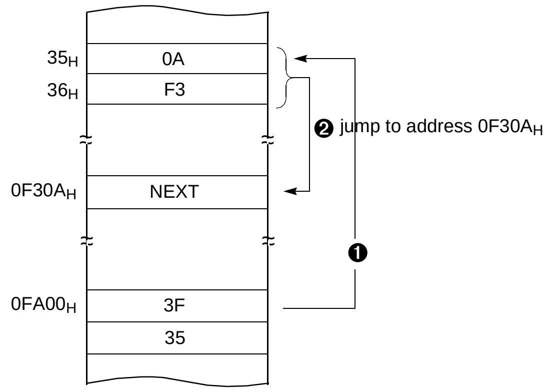
Direct page indirect → [dp]

Assigns data address to use for accomplishing command which sets memory data(or pair memory) by Operand. Also index can be used with Index register X,Y.

JMP, CALL

Example;

```
3F35     JMP    [35H]
```



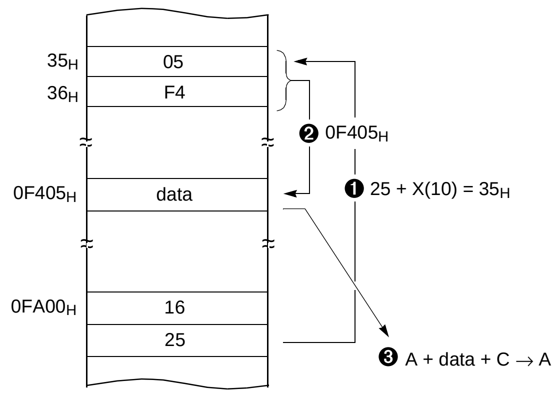
X indexed indirect → [dp+X]

Processes memory data as Data, assigned by 16-bit pair memory which is determined by pair data [dp+X+1][dp+X] Operand plus X-register data in Direct page.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA

Example; X=10_H

```
1625     ADC    [25H+X]
```



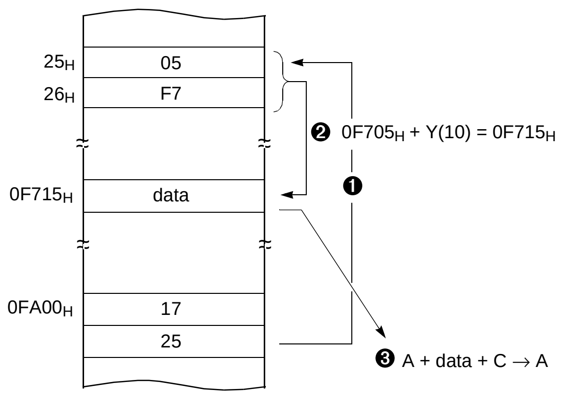
Y indexed indirect → [dp]+Y

Processes memory data as Data, assigned by the data [dp+1][dp] of 16-bit pair memory paired by Operand in Direct page plus Y-register data.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA

Example; Y=10_H

```
1725      ADC    [25H]+Y
```



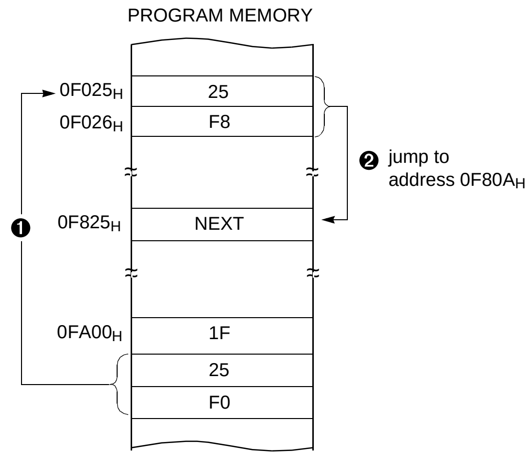
Absolute indirect → [!abs]

The program jumps to address specified by 16-bit absolute address.

JMP

Example;

```
1F25F0   JMP    [!0F025H]
```



9. I/O PORTS

The HMS81020ET/HMS81032ET has 21 I/O ports which are PORT0(8 I/O), PORT1 (8 I/O), PORT2 (5 I/O). Pull-up resistor of each port can be selectable by program. Each port contains data direction register which controls I/O and data register which stores port data.

9.1 R0 Ports

R0 is an 8-bit CMOS bidirectional I/O port (address $0C0_H$). Each I/O pin can independently used as an input or an output through the R0DD register (address $0C1_H$).

R0 has internal pull-ups that is independently connected or disconnected by R0PC. The control registers for R0 are shown below.

R0 Data Register (R/W) — ADDRESS: $0C0_H$, RESET VALUE: Undefined

R0	R07	R06	R05	R04	R03	R02	R01	R00

R0 Direction Register (W) — ADDRESS: $0C1_H$, RESET VALUE: 00_H

R0DD								

Port Direction
0: Input
1: Output

R0 Pull-up Control Register (W) — ADDRESS:$0F8_H$, RESET VALUE: 00_H

R0PC								

Pull-up select
1: Without pull-up
0: With pull-up

R0 Open drain Assign Register (W) — ADDRESS:$0E4_H$, RESET VALUE: 00_H

R0ODC								

Open drain select
0: Push-pull
1: Open drain

(1) R0 I/O Data Direction Register (R0DD)

R0 I/O Data Direction Register (R0DD) is an 8-bit register, and can assign input state or output state to each bit. If R0DD is "1", port R0 is in the output state, and if "0", it is in the input state. R0DD is write-only register. Since R0DD is initialized as "00h" in reset state, the whole port R0 becomes input state.

(2) R0 Data Register (R0)

R0 data register (R0) is an 8-bit register to store data of port R0. When set as the output state by R0DD, and data is written in R0, data is output into R0 pin. When set as the input state, input state of pin is read. The initial value of R0 is unknown in reset state.

(3) R0 Open drain Assign Register (R0ODC)

R0 Open Drain Assign Register (R0ODC) is an 8-bit register, and can assign R0 port as open drain output port each bit, if corresponding port is selected as output. If R0ODC is selected as "1", port R0 is open drain output, and if selected as, "0" it is push-pull output. R0ODC is write-only register and initialized as "00h" in reset state.

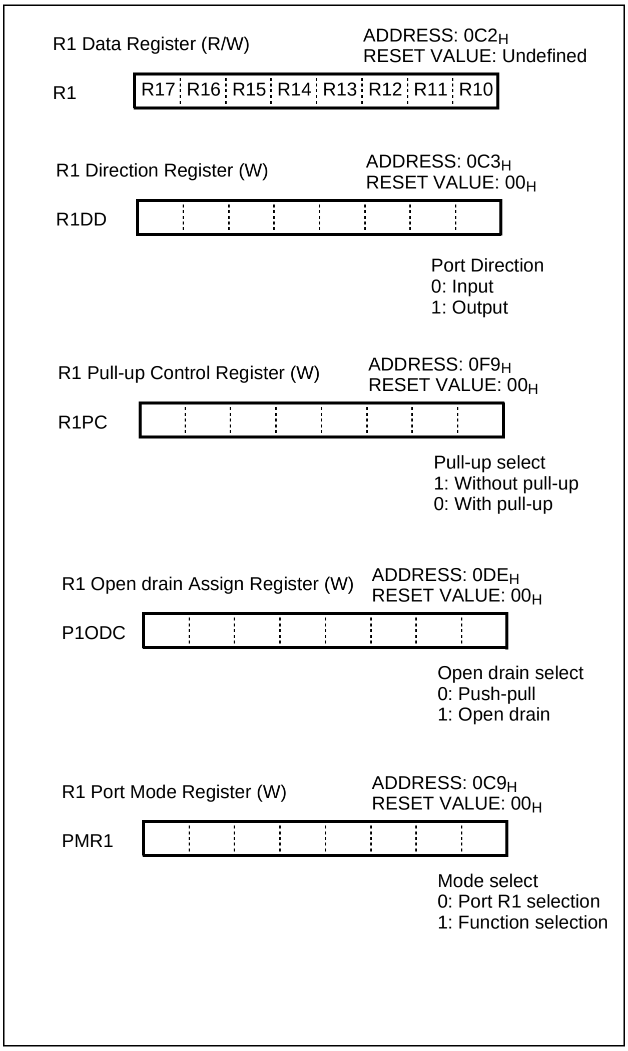
(4) R0 Pull-up Control Register (R0PC)

R0 Pull-up Control Register (R0PC) is an 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R0PC is selected as "1", pull-up is disabled and if selected as "0", it is enabled. R0PC is write-only register and initialized as "00h" in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

9.2 R1 Ports

R1 is an 8-bit CMOS bidirectional I/O port (address $0C2_H$). Each I/O pin can independently used as an input or an output through the R1DD register (address $0C3_H$).

R1 has internal pull-ups that is independently connected or disconnected by register R1PC. The control registers for R1 are shown at next page.



(1) R1 I/O Data Direction Register (R1DD)

R1 I/O Data Direction Register (R1DD) is an 8-bit register, and can assign input state or output state to each bit. If R1DD is "1", port R1 is in the output state, and if "0", it is in the input state. R1DD is write-only register. Since R1DD is initialized as "00h" in reset state, the whole port R1 becomes input state.

(2) R1 Data Register (R1)

R1 data register (R1) is an 8-bit register to store data of port R1. When set as the output state by R1DD, and data is written in R1, data is output into R1 pin. When set as the input state, input state of pin is read. The initial value of R1 is unknown in reset state.

(3) R1 Open drain Assign Register (R1ODC)

R1 Open Drain Assign Register (R1ODC) is an 8bit register, and can assign R1 port as open drain output port each bit, if corresponding port is selected as output. If R1ODC is selected as "1", port R1 is open drain output, and if selected as "0", it is push-pull output. R1ODC is write-only register and initialized as "00h" in reset state.

(4) R1 Port Mode Register (PMR1)

R1 Port Mode Register (PMR1) is an 8-bit register, and can assign the selection mode for each bit. When set as "0", corresponding bit of PMR1 acts as port R1 selection mode, and when set as "1", it becomes function selection mode.

PMR1 is write-only register and initialized as "00h" in reset state. Therefore, becomes Port selection mode. Port R1 can be I/O port by manipulating each R1DD bit, if corresponding PMR1 bit is selected as "0".

Bit Name	PMR1	Selection Mode	Remarks
T0S	0	R17 (I/O)	-
	1	T0 (O)	Timer0
T1S	0	R16 (I/O)	-
	1	T1 (O)	Timer1
T2S	0	R15 (I/O)	-
	1	T2 (O)	Timer2
ECS	0	R14 (I/O)	-
	1	$\overline{EC}$ (I)	Timer0 Event
INT2S	0	R12 (I/O)	
	1	INT2 (I)	Timer0 Input Capture
INT1S	0	R11 (I/O)	
	1	INT1 (I)	

Table 9-1 Selection mode of PMR1

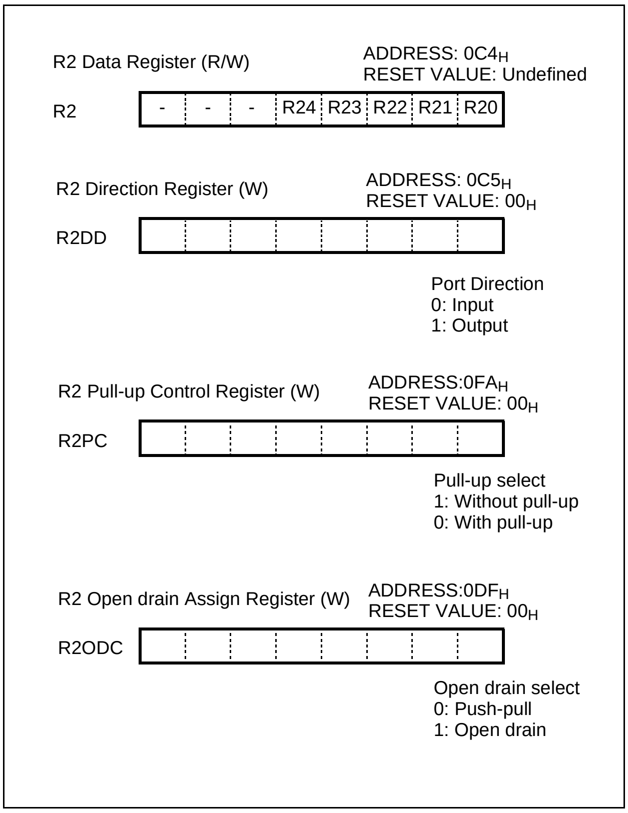
(5) R1 Pull-up Control Register (R1PC)

R1 Pull-up Control Register (R1PC) is an 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R1PC is selected as "1", pull-up is disabled and if selected as "0", it is enabled. R1PC is write-only register and initialized as "00h" in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

9.3 R2 Port

R2 is an 5-bit CMOS bidirectional I/O port (address $0C4_H$). Each I/O pin can independently used as an input or an output through the R2DD register (address $0C5_H$).

R2 has internal pull-ups that is independently connected or disconnected by R2PC (address $0FA_H$). The control registers for R2 are shown as below.



(1) R2 I/O Data Direction Register (R2DD)

R2 I/O Data Direction Register (R2DD) is an 8-bit register, and can assign input state or output state to each bit. If R2DD is “1”, port R2 is in the output state, and if “0”, it is in the input state. R2DD is write-only register. Since R2DD is initialized as “00h” in reset state, the whole port R2 becomes input state.

(2) R2 Data Register (R2)

R2 data register (R2) is an 8-bit register to store data of port R2. When set as the output state by R2DD, and data is written in R2, data is outputted into R2 pin. When set as the input state, input state of pin is read. The initial value of R2 is unknown in reset state.

(3) R2 Open drain Assign Register (R2ODC)

R2 Open Drain Assign Register (R2ODC) is an 8bit register, and can assign R2 port as open drain output port each bit, if corresponding port is selected as output. If R2ODC is selected as “1”, port R2 is open drain output, and if selected as “0”, it is push-pull output. R2ODC is write-only register and initialized as “00h” in reset state.

(4) R2 Pull-up Control Register (R2PC)

R2 Pull-up Control Register (R2PC) is an 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R2PC is selected as “1”, pull-up ia disabled and if selected as ”0”, it is enabled. R2PC is write-only register and initialized as “00h” in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

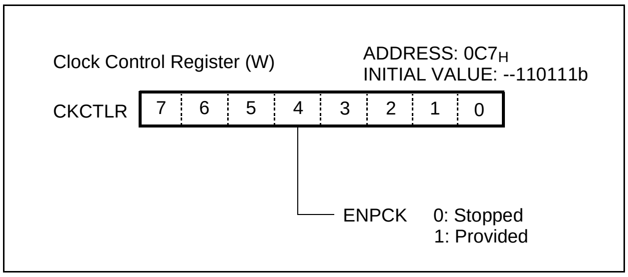
10. CLOCK GENERATOR

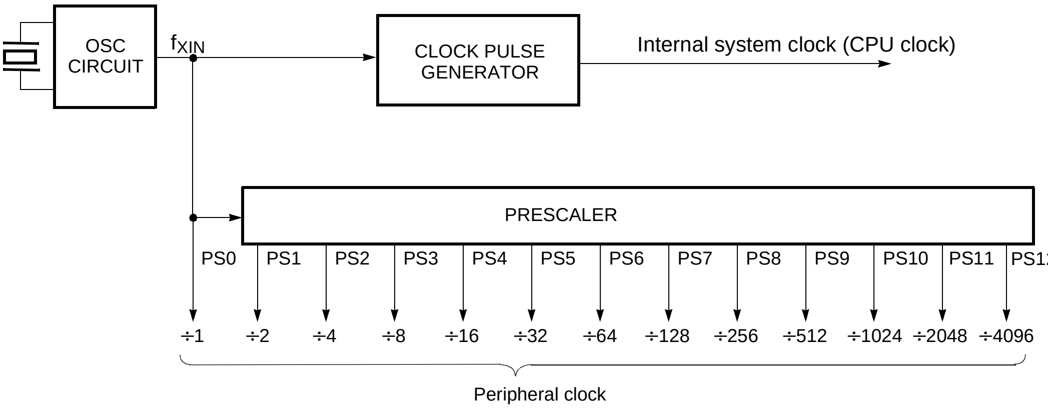
The Clock generating circuit consists of Clock Pulse Generator (C.P.G), Prescaler, Basic Interval Timer (BIT) and Watch Dog Timer. The clock applied to the Xin pin divided by two is used as the internal system clock.

Prescaler consist of 12-bit binary counter. The clock supplied from oscillation circuit is input to prescaler(f_{XIN})

The divided output from each bit of prescaler is provided to peripheral hardware

Clock to peripheral hardware can be stopped by bit4 (ENPCK) of CKCTLR Register. ENPCK is set to “1” in reset state.





f_{XIN}=4MHz	PS0	PS1	PS2	PS3	PS4	PS5	PS6	PS7	PS8	PS9	PS10	PS11	PS12
Frequency (Hz)	4M	2M	1M	500K	250K	125K	62.5K	31.25K	15.63K	7.183K	3.906K	1.953K	0.976K
period (S)	250n	500n	1u	2u	4u	8u	16u	32u	64u	128u	256u	512u	1024u

Figure 10-1 Block diagram of Clock Generator

10.1 Oscillation Circuit

Oscillation circuit is designed to be used either with a ceramic resonator or crystal oscillator. Figure 10-2 shows circuit diagrams using a crystal (or ceramic) oscillator. As shown in the diagram, oscillation circuits can be constructed by connecting a oscillator between Xout and Xin. Clock from oscillation circuit makes CPU clock via clock pulse generator, and then enters prescaler to make peripheral hardware clock. Alternately, the oscillator may be driven from an external source as Figure 10-3 . In the STOP mode, oscillation stop, Xout state goes to “Low”, Xin state goes to “LOW”, and built-in feed back resistor is disabled.

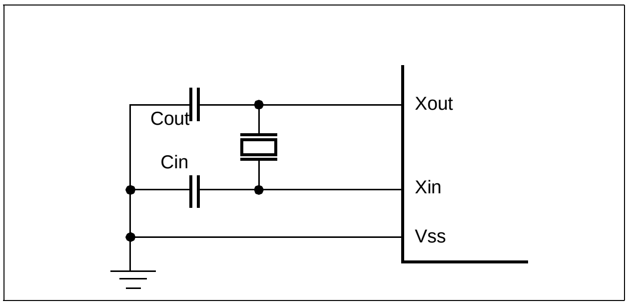


Figure 10-2 External Crystal(Ceramic) oscillator circuit

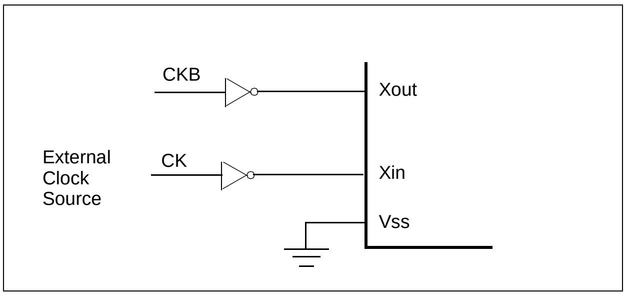


Figure 10-3 External clock input circuit

Oscillation circuit is designed to be used either with a ceramic resonator or crystal oscillator. Since each crystal and ceramic resonator have their own characteristics, the user should consult the crystal manufacturer for appropriate values of external components. In addition, see Figure 10-4 for the layout of the crystal.

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

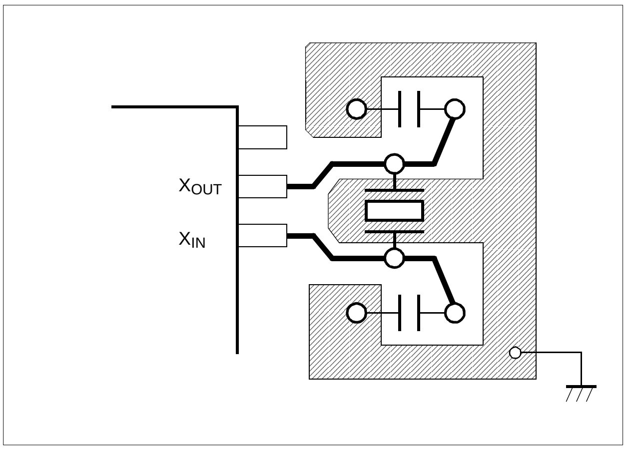


Figure 10-4 Recommend Layout of Oscillator PCB circuit

Frequency	Resonator Maker	Part Name	Load Capacitor	Operating Voltage(V)
2.00MHz	CQ	ZTA2.00M	Cin=Cout=33pF	2.0~4.0
	MURATA	CSTCC2M00G56-R0	Cin=Cout=open	2.0~4.0
	MURATA	CSTCC2.00MG0H6	Cin=Cout=open	2.0~4.0
4.00MHz	CQ	ZTA4.00M	Cin=Cout=33pF	2.0~4.0
	MURATA	CSTS0400MG03	Cin=Cout=open	2.0~4.0
	MURATA	CSTLS4M00G53-B0	Cin=Cout=open	2.0~4.0
	MURATA	CSTCR4M00G53-R0	Cin=Cout=open	2.0~4.0
	TDK	FCR4.0MC5	Cin=Cout=open	2.0~4.0

Table 10-1 recommendation resonator

11. BASIC INTERVAL TIMER

The HMS81020ET/HMS81032ET has one 8-bit Basic Interval Timer that is free-run and can not stop. Block diagram is shown in Figure 11-1.

The Basic Interval Timer generates the time base for Standby release time, watchdog timer counting, and etc. It also provides a Basic interval timer interrupt (IFBIT). As the count overflow from FF_H to 00_H, this overflow causes the interrupt to be generated.

-8bit binary up-counter

-Use the bit output of prescaler as input to secure the oscillation stabilization time after power-on

-Secures the oscillation stabilization time in standby mode (stop mode) release

-Contents of BIT can be read

-Provides the clock for watch dog timer

The Basic Interval Timer is controlled by the clock control register (CKCTLR) shown in Figure 11-2. If bit3(BTCL) of CKCTLR is set to “1”, BIT is cleared, and then, after one machine cycle, BTCL becomes “0”, and BIT starts counting. BTCL is set to ``0`` in reset state.

The input clock of BIT can be selected from the prescaler within a range of 2us to 256us by clock input selection bits (BTS2~BTS0). (at f_{XIN} = 4MHz). In reset state, or power on reset, BTS2=“1”, BTS1= “1”, BTS0= “1” to secure the longest oscillation stabilization time. BIT can generate the wide range of basic interval time interrupt request (IFBIT) by selecting prescaler output.

By reading of the Basic Interval Timer Register (BITR), we can read counter value of BIT Because BIT can be cleared or read, the spending time up to maximum 65.5ms can be available. BIT is read-only register. If BIT register is written, then CKCTLR register with same address is written.

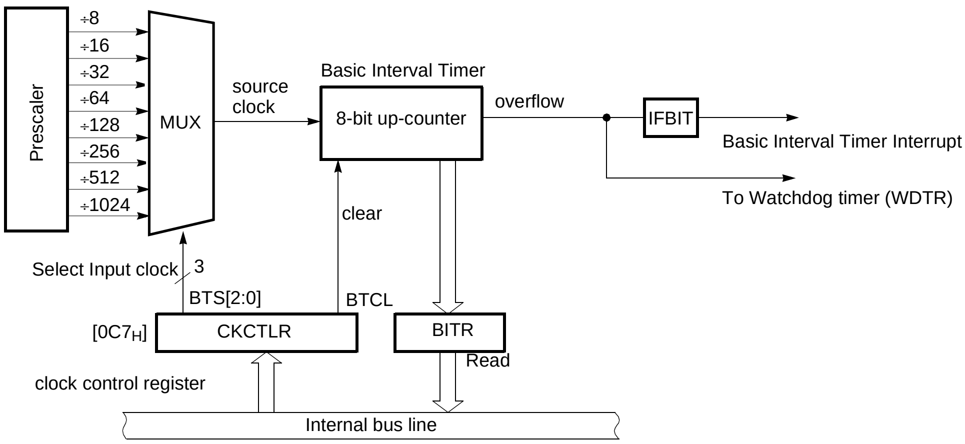


Figure 11-1 Block diagram of Basic Interval Timer

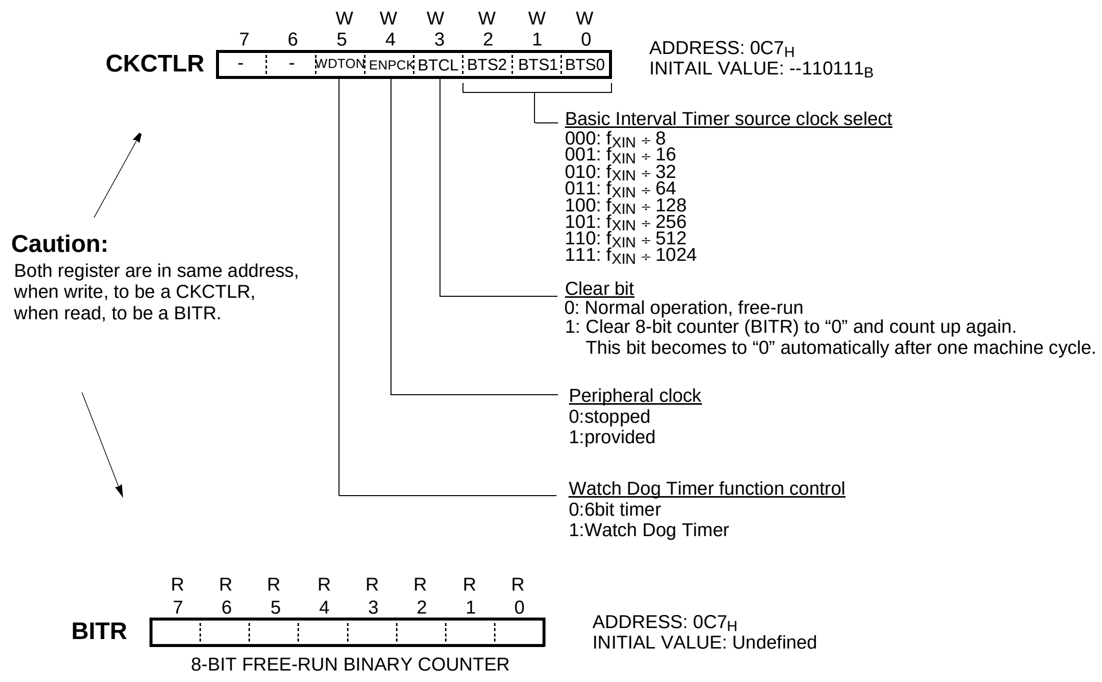


Figure 11-2 CKCTLR AND BITR

BTS[2:0]	CPU Source clock	BIT Input clock@4MHz(us)	Standby release time(ms)
000	÷ 8	2	0.512
001	÷16	4	1.024
010	÷32	8	2.048
011	÷64	16	4.096
100	÷128	32	8.192
101	÷256	64	16.384
110	÷512	128	32.768
111	÷1024	256	65.536

Table 11-1 Standby release time according to BTS

12. WATCH DOG TIMER

Watch Dog Timer (WDT) consists of 6-bit binary counter, 6-bit comparator, and Watch Dog Timer Register (WDTR).Watch Dog Timer can be used 6-bit general Timer or specific Watch dog timer by setting bit5 (WDTON) of Clock Control Register (CKCTLR).By assigning bit6(WDTCL) of WDTR, 6-bit counter can be cleared.

WDT Interrupt (IFWDT) interval is determined by the interrupt IFBIT interval of Basic Interval Timer and the value of WDT Register.

-Interval of IFWDT = (IFBIT interval) $\times$ (WDTR value)

As IFBIT (Basic Interval Timer Interrupt Request) is used for input clock of WDT, Input clock cycle is possible from 512 us to 65,536 us by BTS. (at f_{XIN} = 4MHz)

*At Hardware reset time, WDT starts automatically. Therefore the user must select the CKCTLR and WDTR before WDT overflow.

-Reset WDTR value = $0F_h$=15

-Interval of WDT = 65,536 $\times$ 15 = 983,040 us

(about 1second)

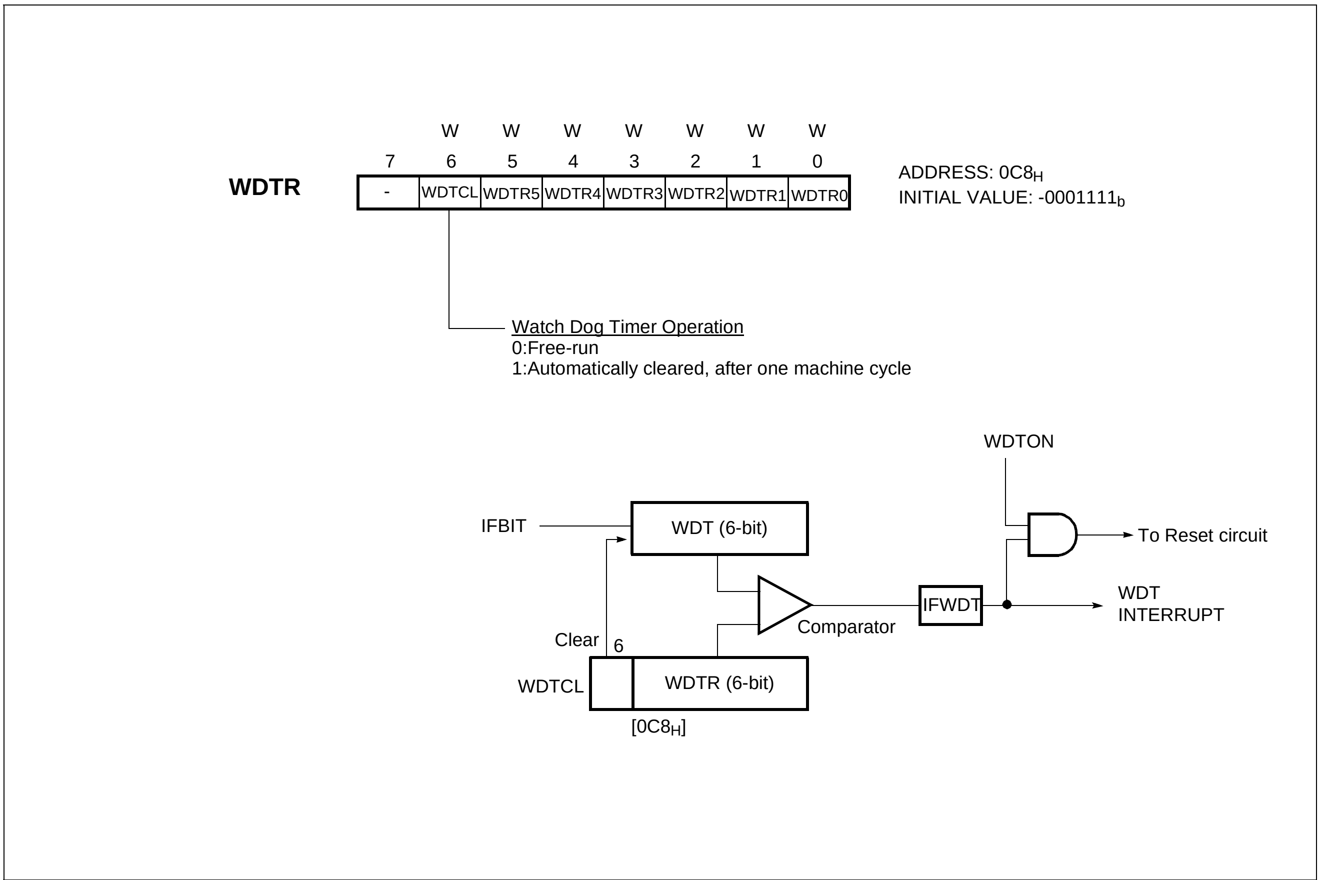


Figure 12-1 Block diagram of Watch Dog Timer

13. Timer0, Timer1, Timer2

(1) Timer Operation Mode

Timer consists of 16bit binary counter Timer0 (T0), 8bit binary Timer1 (T1), Timer2 (T2), Timer Data Register, Timer Mode Register (TM01, TM0, TM1, TM2) and control circuit. Timer Data Register Consists of Timer0 High-MSB Data Register (T0HMD), Timer0 High-LSB Data Register (T0HLD), Timer0 Low-MSB Data Register (T0LMD), Timer0 Low-LSB Data Register (T0LLD), Timer1 High Data Register (T1HD), Timer1 Low Data Register (T1LD), Timer2 Data Register (T2DR). Any of the PS0 ~ PS5, PS11 and external event input $\overline{EC}$ can be selected as clock source for T0. Any of the PS0 ~ PS3, PS7 ~ PS10 can be selected as clock T1. Any of the PS5 ~ PS12 can be selected as clock source for T2.

* Relevant Port Mode Register (PMR1: 00C9h) value should be assigned for event counter.

<table>
<tr><td>Timer0</td><td>- 16-bit Interval Timer
- 16-bit Event Counter
- 16-bit Input Capture
- 16-bit rectangular-wave output</td><td rowspan="2">- Single/Modulo-N Mode
- Timer Output Initial Value Setting
- Timer0~Timer1 combination Logic Output
- One Interrupt Generating Every 2nd Counter Overflow</td></tr>
<tr><td>Timer1</td><td>- 8-bit Interval Timer
- 8-bit rectangular-wave output</td></tr>
<tr><td>Timer2</td><td colspan="2">- 8-bit Interval Timer
- 8-bit rectangular-wave output
- Modulo-N Mode</td></tr>
</table>

Table 13-1 Timer Operation

16bit Timer 0 (f_{XIN}=4MHz)		8bit Timer 1 (f_{XIN}=4MHz)		8bit Timer 2 (f_{XIN}=4MHz)	
Resolution	**MAX. Count**	**Resolution**	**MAX. Count**	**Resolution**	**MAX. Count**
PS0 (0.25us)	16,384us	PS0 (0.25us)	64us	PS5 (8us)	2,048us
PS1 (0.5us)	32,768us	PS1 (0.5us)	128us	PS6 (16us)	4,096us
PS2 (1us)	65,536us	PS2 (1us)	256us	PS7 (32us)	8,192us
PS3 (2us)	131,072us	PS3 (2us)	512us	PS8 (64us)	16,384us
PS4 (4us)	262,144us	PS7 (32us)	8,192us	PS9 (128us)	32,768us
PS5 (8us)	524,288us	PS8 (64us)	16,384us	PS10 (256us)	65,536us
PS11 (512us)	33,554,432us	PS9 (128us)	32,768us	PS11 (512us)	131,072us
$\overline{EC}$	-	PS10 (256us)	65,536us	PS12 (1024us)	262,144us

Table 13-2 Function of Timer & Counter

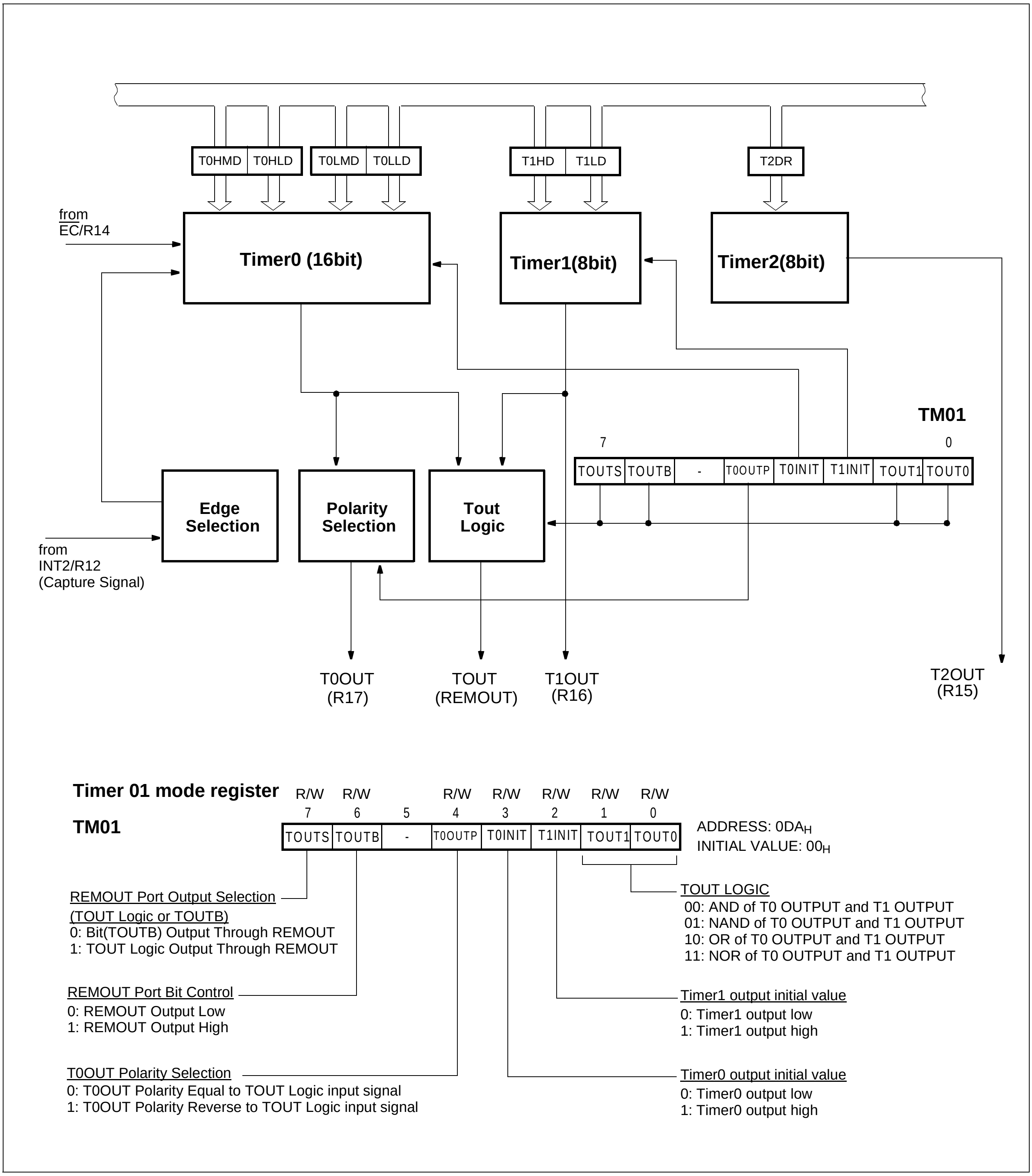


Figure 13-1 Block Diagram of Timer/Counter

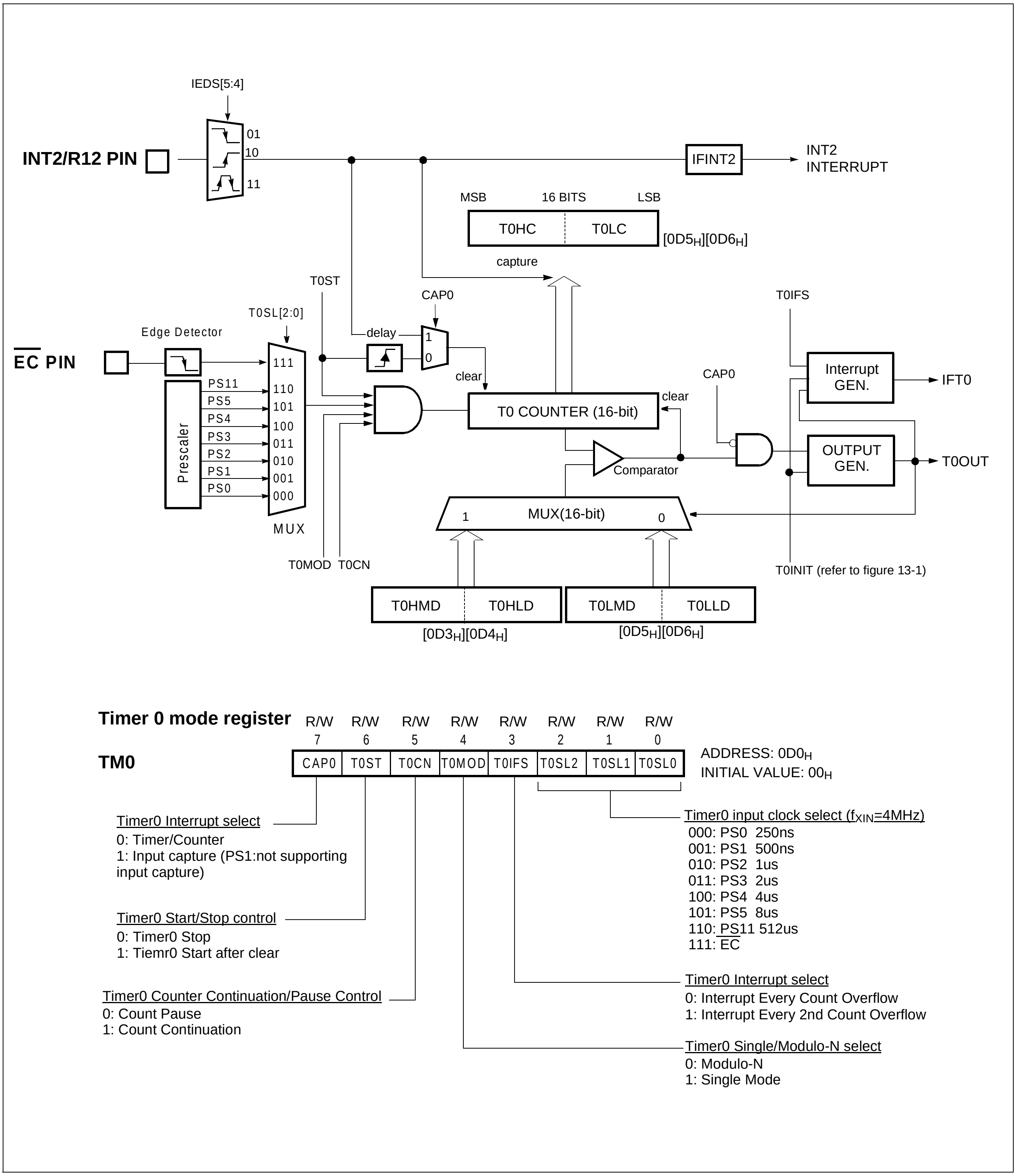
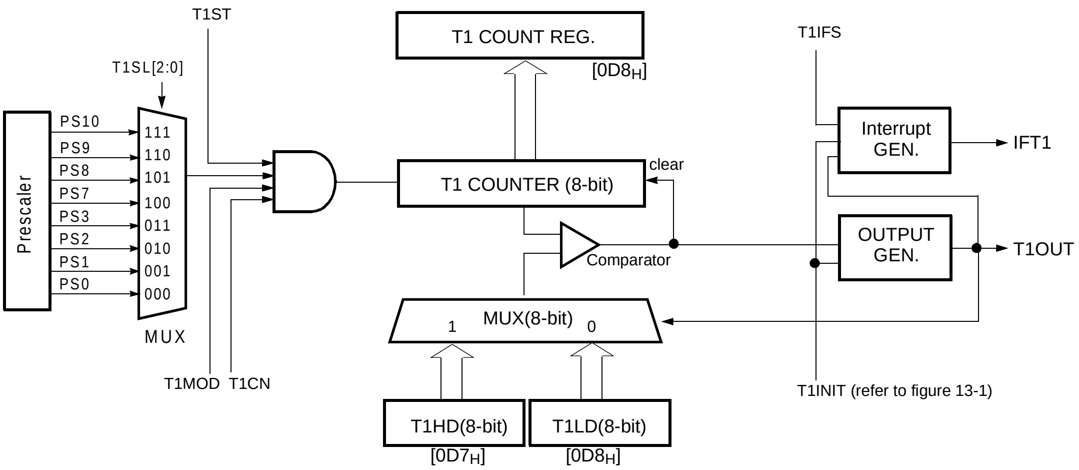


Figure 13-2 Block Diagram of Timer0



Timer 1 mode register

TM1

	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
TM1	T1ST	T1CN	T1MOD	T1IFS	-	T1SL2	T1SL1	T1SL0

ADDRESS: $0D1_H$
INITIAL VALUE: 00_H

Timer1 Start/Stop control
0: Timer1 Stop
1: Tiemr1 Start after clear

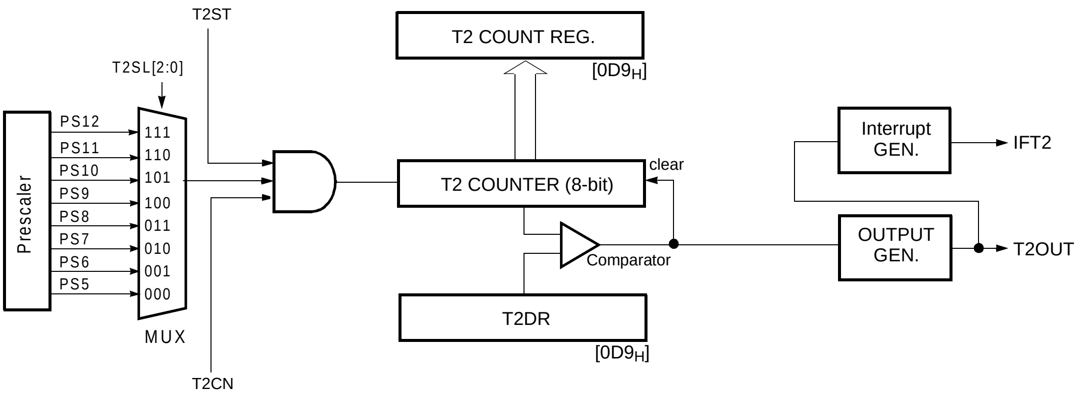
Timer1 Counter Continuation/Pause Control
0: Count Pause
1: Count Continuation

Timer1 Single/Modulo-N select
0: Modulo-N
1: Single Mode

Timer1 input clock select (f_{XIN}=4MHz)
000: PS0 250ns
001: PS1 500ns
010: PS2 1us
011: PS3 2us
100: PS7 32us
101: PS8 64us
110: PS9 128us
111: PS10 256us

Timer1 Interrupt select
0: Interrupt Every Count Overflow
1: Interrupt Every 2nd Count Overflow

Figure 13-3 Block Diagram of Timer1



Timer 2 mode register

TM2

	7	6	5	4	3	2	1	0
				R/W	R/W	R/W	R/W	R/W
TM2	-	-	-	T2ST	T2CN	T2SL2	T2SL1	T2SL0

ADDRESS: $0D2_H$
INITIAL VALUE: 00_H

Timer2 Start/Stop control
0: Timer2 Stop
1: Tiemr2 Start after clear

Timer2 Counter Continuation/Pause Control
0: Count Pause
1: Count Continuation

Timer2 input clock select (f_{XIN}=4MHz)
000: PS5 8us
001: PS6 16us
010: PS7 32us
011: PS8 64us
100: PS9 128us
101: PS10 256us
110: PS11 512us
111: PS12 1,024us

Figure 13-4 Block Diagram of Timer2

2) Timer0, Timer1

TIMER0 and TIMER1 have an up-counter. When value of the up-counter reaches the content of Timer Data Register (TDR), the up-counter is cleared to “00h”, and interrupt (IFT0, IFT1) is occurred at the next clock.

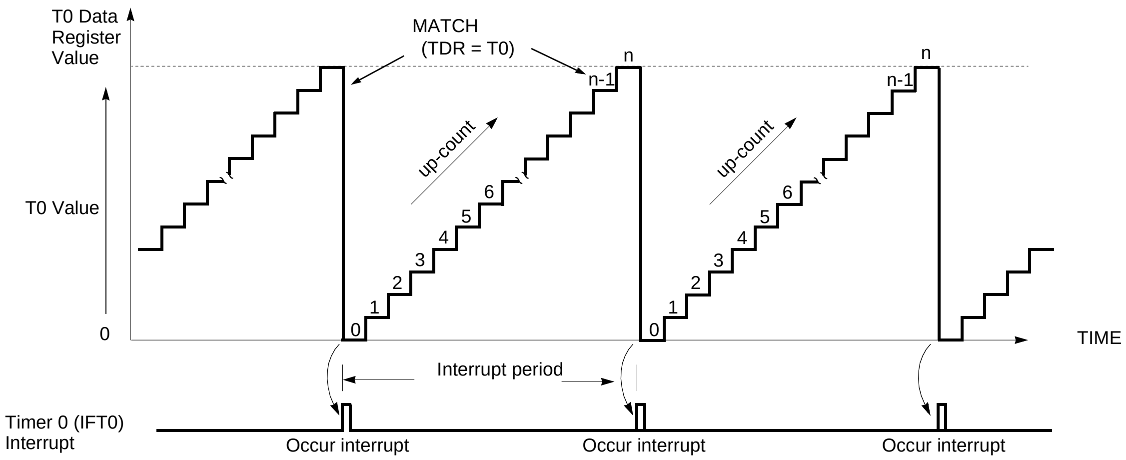


Figure 13-5 Operation of Timer0

For Timer0, the internal clock (PS) and the external clock $\overline{\text{(EC)}}$ can be selected as counter clock. But Timer1 and Timer2 use only internal clock. As internal clock is chosen, Timer0 can be used as internal-timer which period is determined by Timer Data Register (TDR). Chosen as external clock, Timer0 executes as event-counter. The counter execution of Timer0 and Timer1 is controlled by T0CN, T0ST, CAP0, T1CN, T1ST, of Timer Mode Register TM0 and TM1. T0CN, T1CN are used to stop and start Timer0 and Timer1 without clearing the counter. T0ST, T1ST is used to clear the counter. For clearing and starting the counter, T0ST or T1ST should be temporarily set to “0” and then set to “1”. T0CN, T1CN, T0ST and T1ST should be set “1”, when Timer counting-up. Controlling of CAP0 enables Timer0 as input capture. By programming of CAP0 to “1”, the period of signal from INT2 can be measured and then, event counter value for INT2 can be read. During counting-up, value of counter can be read. Timer execution is stopped by the reset signal (RESET=”L”)

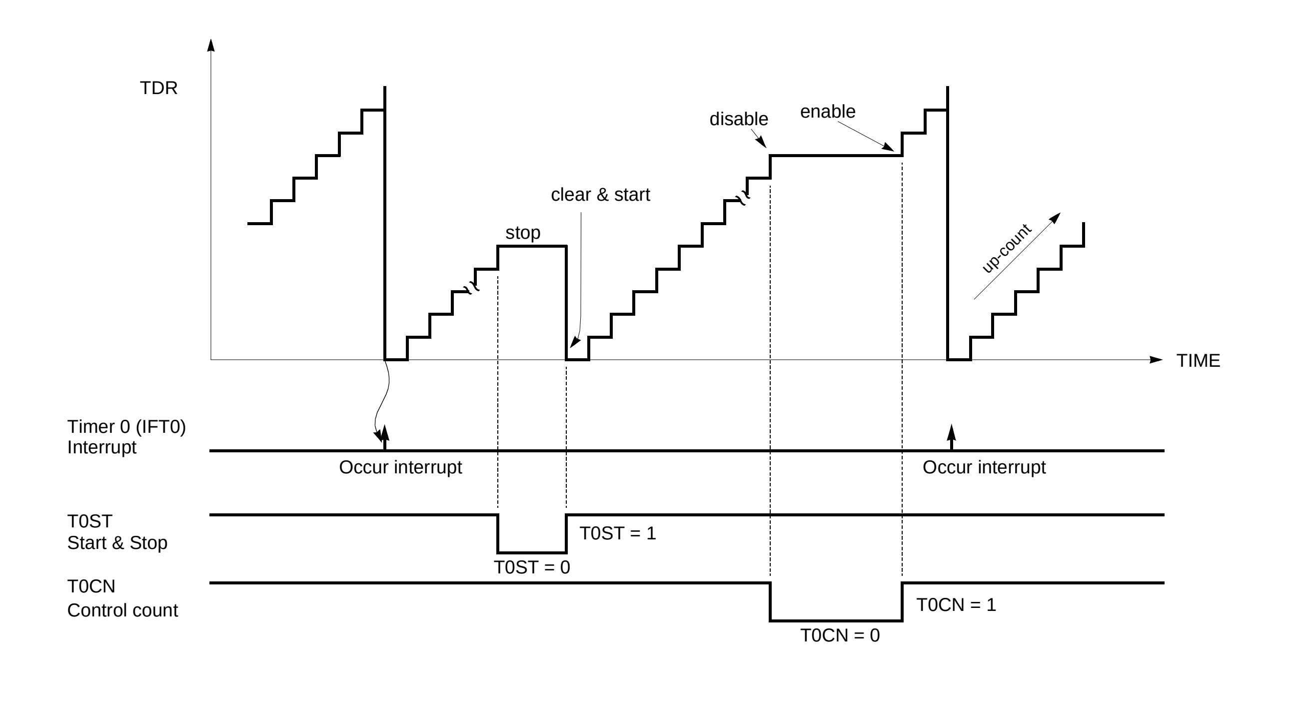


Figure 13-6 Start/Stop Operation of Timer0

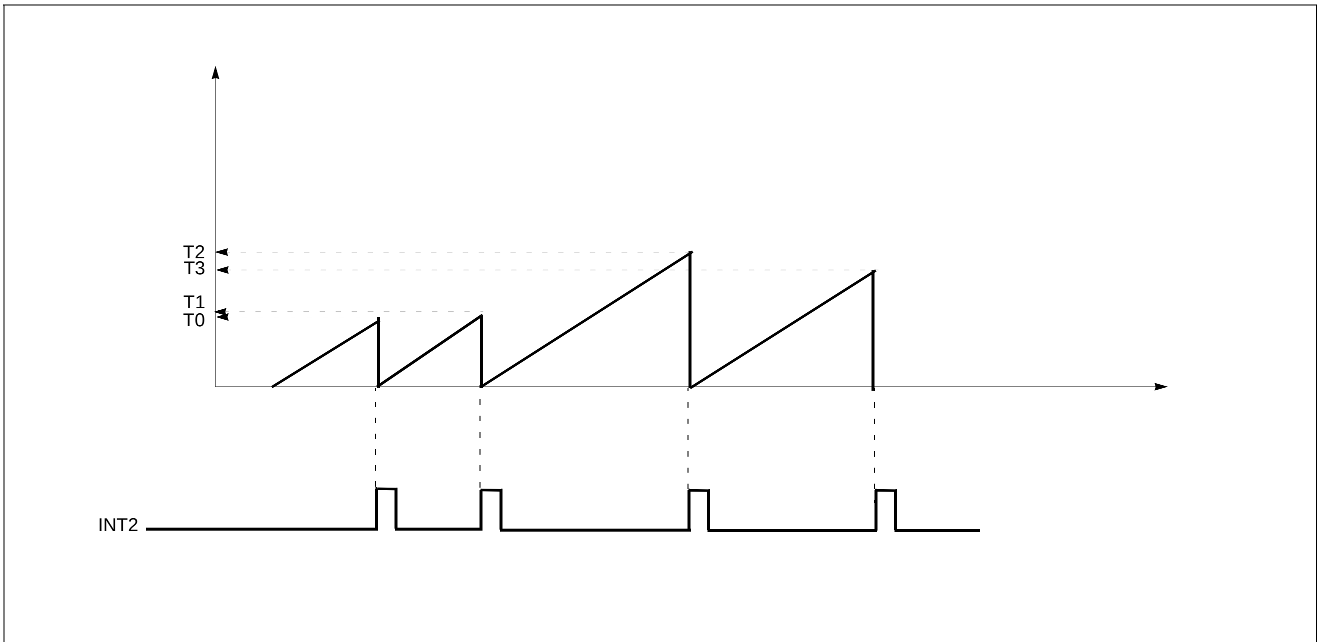


Figure 13-7 Input capture operation of Timer0

3) Single/Modulo-N Mode

Timer0 (Timer1) can select initial output level of Timer Output port by T0INIT, T1INIT of TM01. If initial level is “L”, Low-Data Register value of Timer Data Register is transferred to comparator and T0OUT (T1OUT) is to be “Low”. If initial level is “High”, High -Data Register is transferred and to be “High”. Single Mode can be set by Mode Select bit (T0MOD, T1MOD) of Timer Mode Register (TM0, TM1) to “1”. When used as Single Mode, Timer counts up and compares with value of Data Register. If the result is same, Time Out interrupt occurs and level of Timer Output port toggle, then counter stops as reset state. When used as Modulo-N Mode, T0MOD (T1MOD) should be set “0”. Counter counts up until the value of Data Register and occurs Time-out interrupt. The level of Timer Output port toggle and repeats process of counting the value which is selected in Data Register. During Modulo-N Mode, If interrupt select bit (T0IFS, T1IFS) of Mode Register is “0”, Interrupt occurs on every time-out. If it is “1”, Interrupt occurs every second time-out.Timer output is toggled whenever time-out happen.

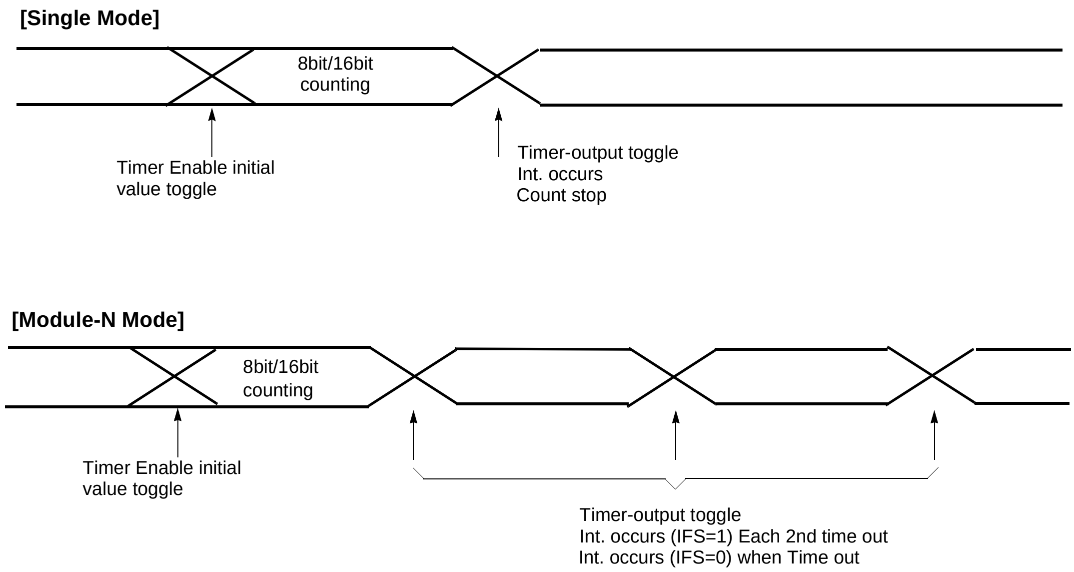


Figure 13-8 Operation Diagram for Single/Modulo-N Mode

(4) Timer 2

Timer2 operates as an up-counter. The contents of T2DR are compared with the contents of up-counter. If a match is found, Timer2 interrupt (IFT2) is generated and the up-counter is cleared to “00h”. Therefore, Timer2 executes as a interval timer. Interrupt period is determined by the count source clock for the Timer2 and contents of T2DR. When T2ST is set to “1”, count value of Timer 2 is cleared and starts counting-up. For clearing and starting the Timer2, T2ST have to set to “1” after set to “0”. In order to write a value directly into the T2DR, T2ST should be set to “0”. Counted value of Timer2 can be read at any time.

14. INTERRUPTS

The HMS81020ET/HMS81032ET interrupt circuits consist of Interrupt Mode Register (MOD), Interrupt enable register (IENH, IENL), Interrupt request flags of IRQH, IRQL, Priority circuit and Master enable flag ("I" flag of PSW). 8 interrupt sources are provided. The configuration of interrupt circuit is shown in Figure 14-1.

The HMS81020ET/HMS81032ET contains 8 interrupt sources; 3 externals and 5 internals. Nested interrupt services with priority control is also possible. Software interrupt is non-maskable interrupt, the others are all maskable interrupts.

- 8 interrupt source (2Ext, 3Timer, BIT, WDT and Key Scan)
- 8 interrupt vector
- Nested interrupt control is possible
- Programmable interrupt mode (Hardware and software interrupt accept mode)
- Read and write of interrupt request flag are possible.
- In interrupt accept, request flag is automatically cleared.

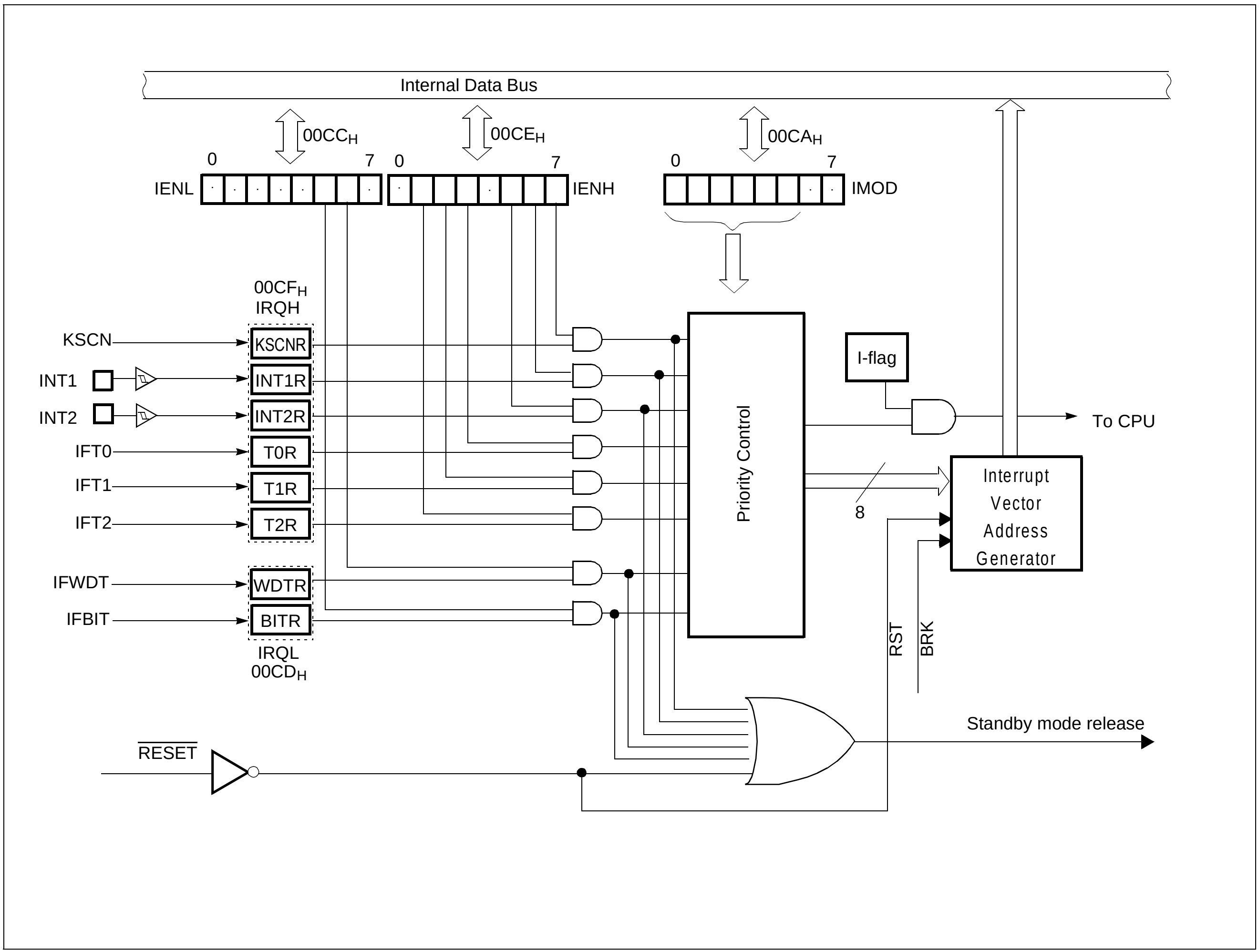


Figure 14-1 Block Diagram of Interrupt

14.1 Interrupt priority and sources

Each interrupt vector is independent and has its own priority. Software interrupt (BRK) is also available. Interrupt source classification is shown in Table 14-1.

14.2 Interrupt control register

I flag of PSW is a interrupt mask enable flag. When I flag = “0”, all interrupts become disable. When I flag = “1”, interrupts can be selectively enabled and disabled by contents of corresponding Interrupt Enable Register. When interrupt is occurred, interrupt request flag is set, and Interrupt request is detected at the edge of interrupt signal. The accepted interrupt request flag is automatically cleared during interrupt cycle process. The interrupt request flag maintains “1” until the interrupt is accepted or is cleared in program. In reset state, interrupt request flag register (IRQH, IRQL) is cleared to “0”. It is possible to read the state of interrupt register and to manipulate the contents of register and to generate interrupt. (Refer to software interrupt)

	Mask	Priority	Reset/Interrupt	Symbol	INT vector high address	INT vector low address
Hardware Interrupt	Non-maskable	-	Hardware Reset	$\overline{\text{RESET}}$	$FFFF_H$	$FFFE_H$
	maskable	1	Key Scan	KSCNR	$FFFB_H$	$FFFA_H$
		2	External Interrupt1	INT1R	$FFF9_H$	$FFF8_H$
		3	External Interrupt2	INT2R	$FFF7_H$	$FFF6_H$
		4	Timer0	T0R	$FFF3_H$	$FFF2_H$
		5	Timer1	T1R	$FFF1_H$	$FFF0_H$
		6	Timer2	T2R	$FFEF_H$	$FFEE_H$
		7	Watch Dog Timer	WDTR	$FFE9_H$	$FFE8_H$
		8	Basic Interval Timer	BITR	$FFE7_H$	$FFE6_H$
Software Interrupt	non-maskable	-	BRK Instruction	BRK	$FFDF_H$	$FFDE_H$

Table 14-1 Interrupt Source

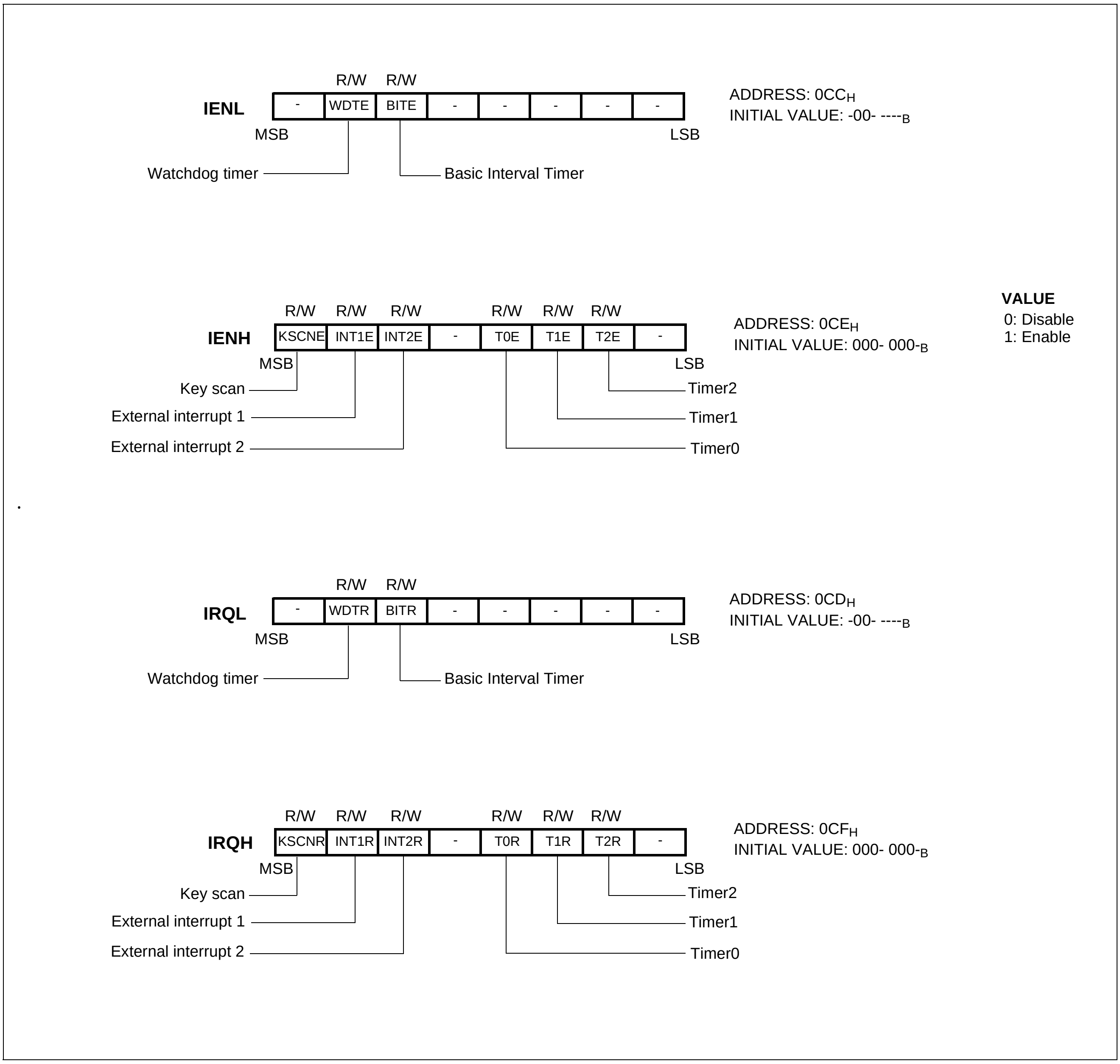


Figure 14-2 Interrupt Enable & Request Flag

14.3 Interrupt accept mode

The interrupt priority order is determined by bit (IM1, IM0) of IMOD register. The condition allow for accepting interrupt is set state of the interrupt mask enable flag and the interrupt enable bit must be “1”. In Reset state, these IP3 - IP0 registers become all “0”.

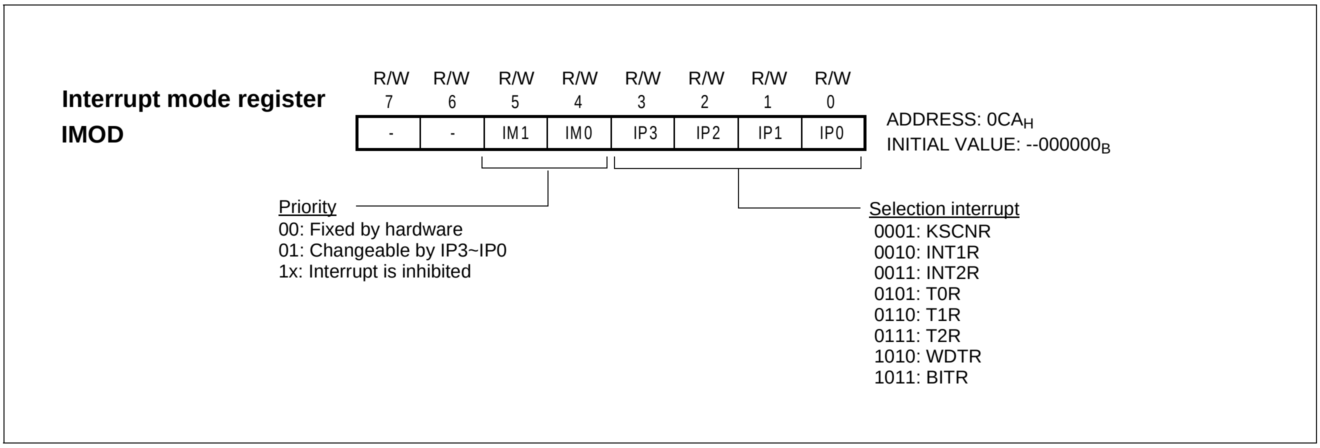


Figure 14-3 Interrupt Accept Mode & Selection by IP3~IP0

14.4 Interrupt Sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to “0” by a reset or an instruction. Interrupt acceptance sequence requires 8 f_{XIN} after the completion of the current instruction execution. The interrupt service task is terminated upon execution of an interrupt return instruction [RETI].

Interrupt acceptance

1. The interrupt master enable flag (I-flag) is cleared to “0” to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.
2. Interrupt request flag for the interrupt source accepted is cleared to “0”.
3. The contents of the program counter (return address) and the program status word are saved (pushed) onto the stack area. The stack pointer decreases 3 times.
4. The entry address of the interrupt service program is read from the vector table address and the entry address is loaded to the program counter.
5. The instruction stored at the entry address of the interrupt service program is executed.

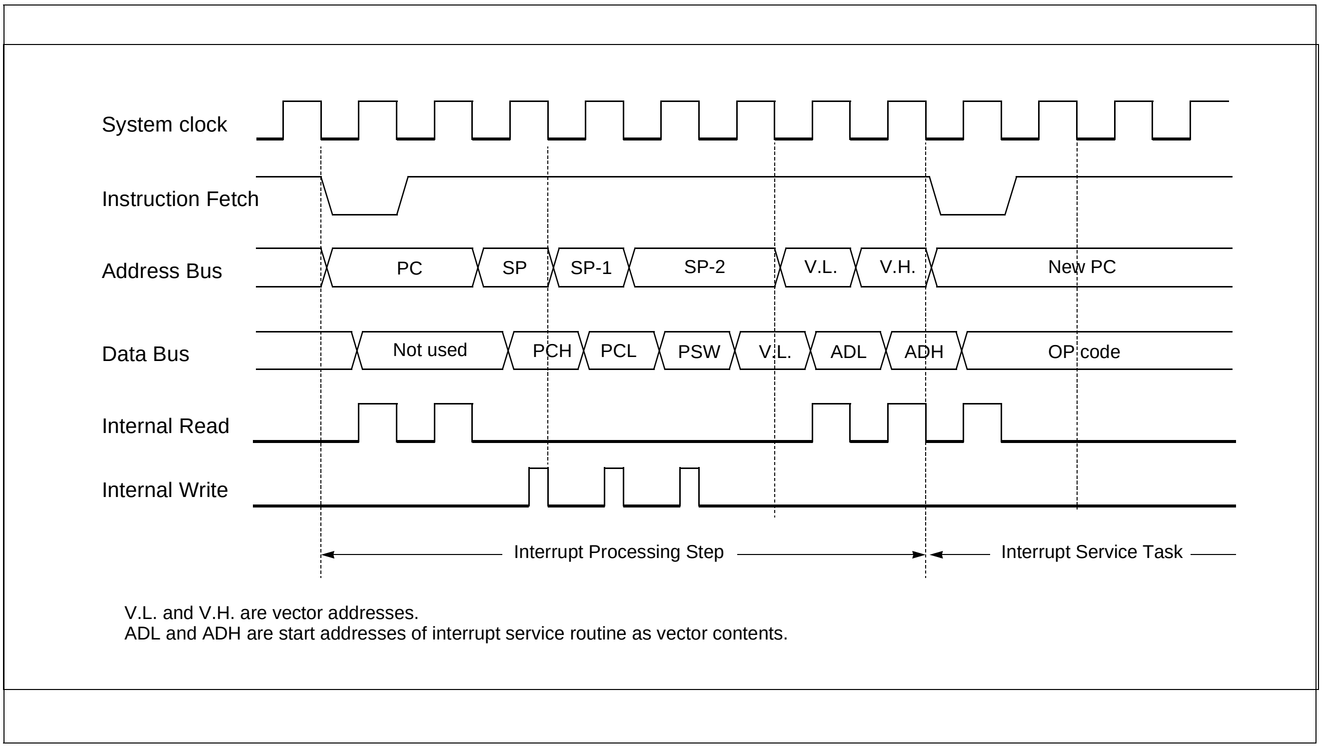
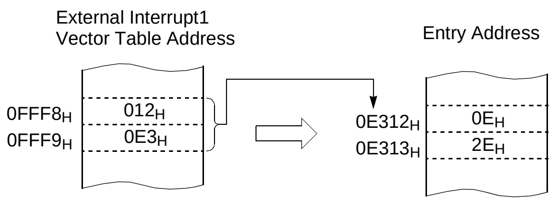


Figure 14-4 Timing chart of Interrupt Acceptance and Interrupt Return Instruction



Correspondence between vector table address for External Interrupt1 and the entry address of the interrupt service program.

A interrupt request is not accepted until the I-flag is set to “1” even if a requested interrupt has higher priority than that of the current interrupt being serviced.

When nested interrupt service is required, the I-flag should be set to “1” by “EI” instruction in the interrupt service program. In this case, acceptable interrupt sources are selectively enabled by the individual interrupt enable flags.

Saving/Restoring General-purpose Register

During interrupt acceptance processing, the program counter and the program status word are automatically saved on the stack, but accumulator and other registers are not saved itself. These registers are saved by the software if necessary. Also, when multiple interrupt services are nested, it is necessary to avoid using the same data memory area for saving registers.

The following method is used to save/restore the general-purpose registers.

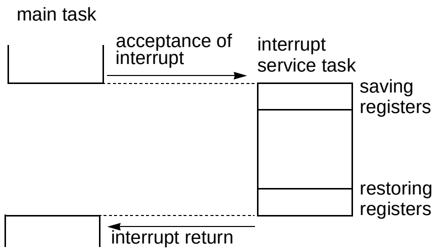
Example: Register save using push and pop instructions

```
INTxx:  PUSH    A       ;SAVE ACC.
        PUSH    X       ;SAVE X REG.
        PUSH    Y       ;SAVE Y REG.

        interrupt processing

        POP     Y       ;RESTORE Y REG.
        POP     X       ;RESTORE X REG.
        POP     A       ;RESTORE ACC.
        RETI            ;RETURN
```

General-purpose register save/restore using push and pop instructions;



14.5 BRK Interrupt

Software interrupt can be invoked by BRK instruction, which has the lowest priority order.

Interrupt vector address of BRK is shared with the vector of TCALL 0 (Refer to Program Memory Section). When BRK interrupt is generated, B-flag of PSW is set to distinguish BRK from TCALL 0.

Each processing step is determined by B-flag as shown in Figure 14-5

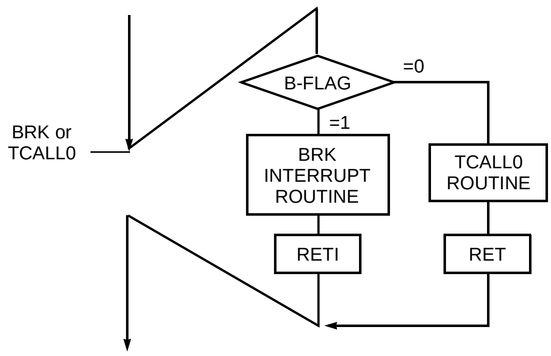


Figure 14-5 Execution of BRK/TCALL0

14.6 Multi Interrupt

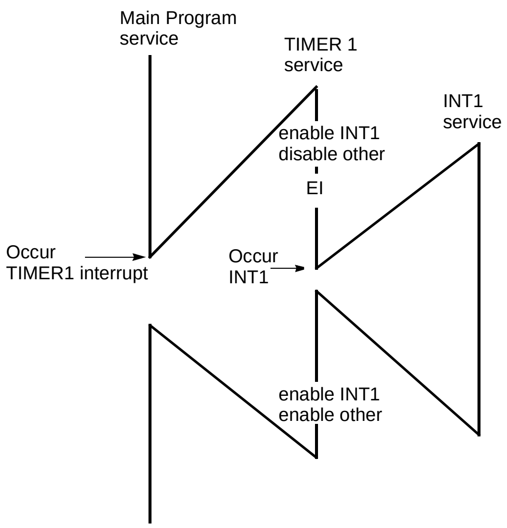
If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the interrupt are received at the same time simultaneously, an internal polling sequence determines by hardware which request is serviced.

However, multiple processing through software for special features is possible. Generally when an interrupt is accepted, the I-flag is cleared to disable any further interrupt. But as user sets I-flag in interrupt routine, some further interrupt can be serviced even if certain interrupt is in progress.

Example: During Timer1 interrupt is in progress, INT1 interrupt serviced without any suspend.

```
TIMER1: PUSH   A
        PUSH   X
        PUSH   Y
        LDM    IENH,#40H    ;Enable INT1 only
        LDM    IENL,#00H    ;Disable other
        EI                  ;Enable Interrupt
        :
        :


        :
        :
        LDM    IENH,#0FFH   ;Enable all interrupts
        LDM    IENL,#0FFH
        POP    Y
        POP    X
        POP    A
        RETI
```



In this example, the INT1 interrupt can be serviced without any pending, even TIMER1 is in progress.
Because of re-setting the interrupt enable registers IENH,IENL and master enable "EI" in the TIMER1 routine.

Figure 14-6 Execution of Multi Interrupt

14.7 External Interrupt

The external interrupt on INT1 and INT2 pins are edge triggered depending on the edge selection register IEDS (address $0D8_H$) as shown in Figure14-7.

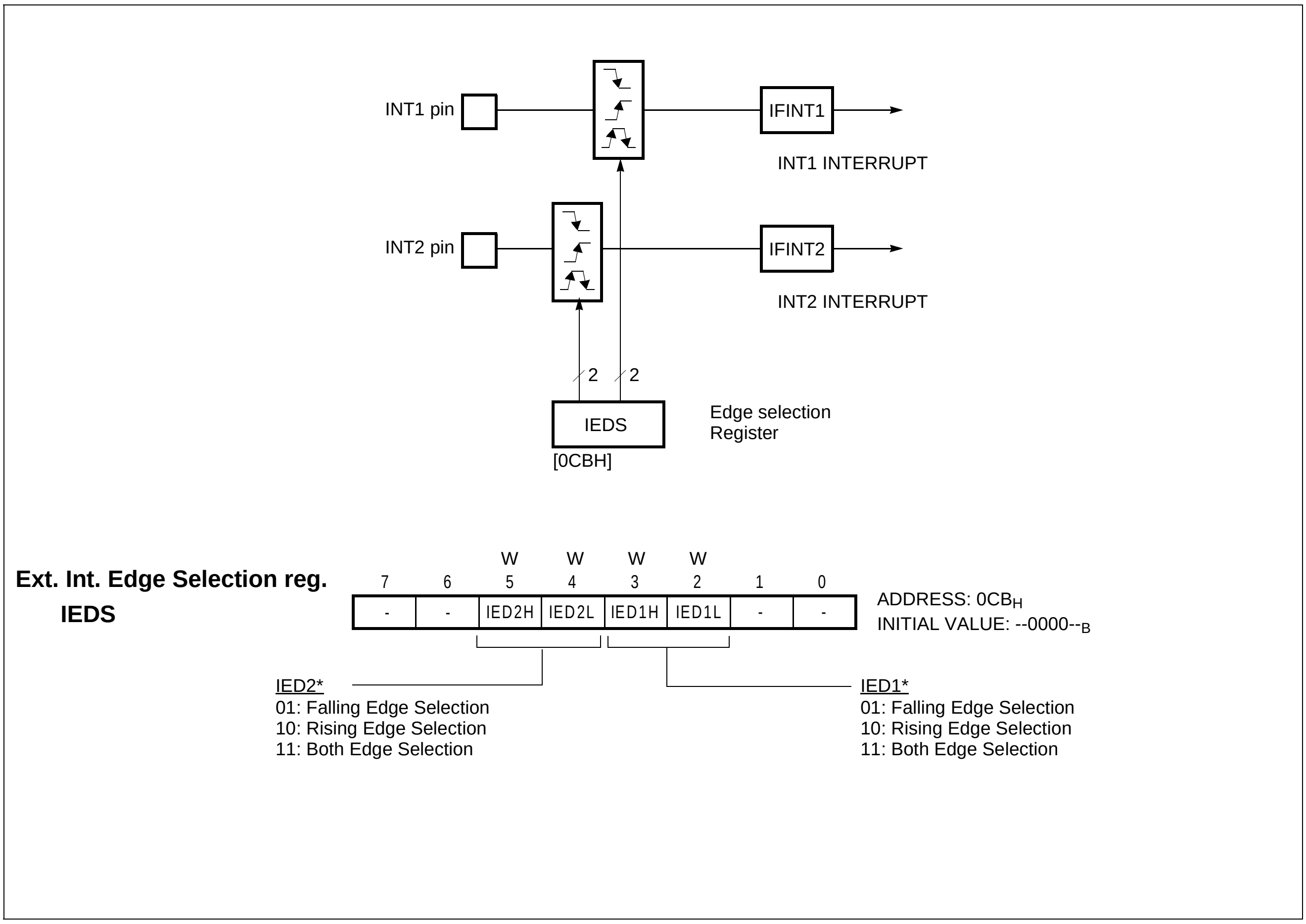


Figure 14-7 External Interrupt Block Diagram

Response Time

The INT1 ~ INT2 edge are latched into IFINT1 ~ IFINT2 at every machine cycle. The values are not actually polled by the circuitry until the next machine cycle. If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The DIV itself takes twelve cycles. Thus, a minimum of twelve complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine.

Figure 14-8 shows interrupt response timings.

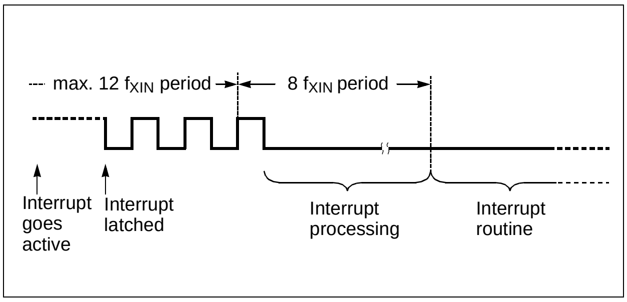
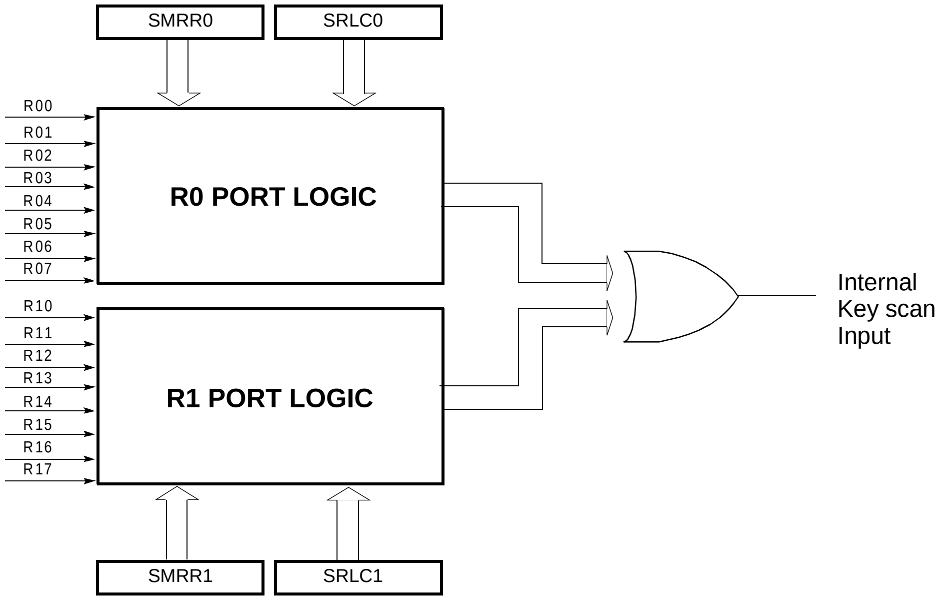


Figure 14-8 Interrupt Response Timing Diagram

14.8 Key Scan Input Processing

Key Scan Interrupt is generated by detecting low or high Input from each Input pin (R0, R1) is one of the sources which release standby (SLEEP, STOP) mode. Key Scan ports are all 16bit which are controlled by Standby Mode Release Register (SMRR0, SMRR1). Key Input is considered as Interrupt, therefore, KSCNE bit of IEHN should be set for correct interrupt executing, SLEEP mode and STOP mode, the rest of executing is the same as that of external Interrupt. Each SMRR Register bit is allowed for each port (for Bit= “0”, no Key Input, for Bit= “1”, Key Input available). At reset, SMRR becomes “00h”. So, there is no Key Input source.

Standby release level control register (SRLC) can select the key scan input level “L” or “H” for standby release by each bit pin (R0, R1). Standby release level control register (SRLC) is write-only register and initialized as “00h” in reset state.



SMRR0

W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
KR07	KR06	KR05	KR04	KR03	KR02	KR01	KR00

ADDRESS: $0DC_H$
INITIAL VALUE: 00_H

KR0[0:7]
1: Select
0: No Select

SMRR1

W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
KR17	KR16	KR15	KR14	KR13	KR12	KR11	KR10

ADDRESS: $0DD_H$
INITIAL VALUE: 00_H

KR1[0:7]
1: Select
0: No Select

SRLC0

W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
KLR07	KLR06	KLR05	KLR04	KLR03	KLR02	KLR01	KLR00

ADDRESS: $0F6_H$
INITIAL VALUE: 00_H

KLR[0:7]
1: High
0: Low

SRLC1

W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
KLR17	KLR16	KLR15	KLR14	KLR13	KLR12	KLR11	KLR10

ADDRESS: $0F7_H$
INITIAL VALUE: 00_H

KLR1[0:7]
1: High
0: Low

Figure 14-9 Block Diagram of Key Scan Block

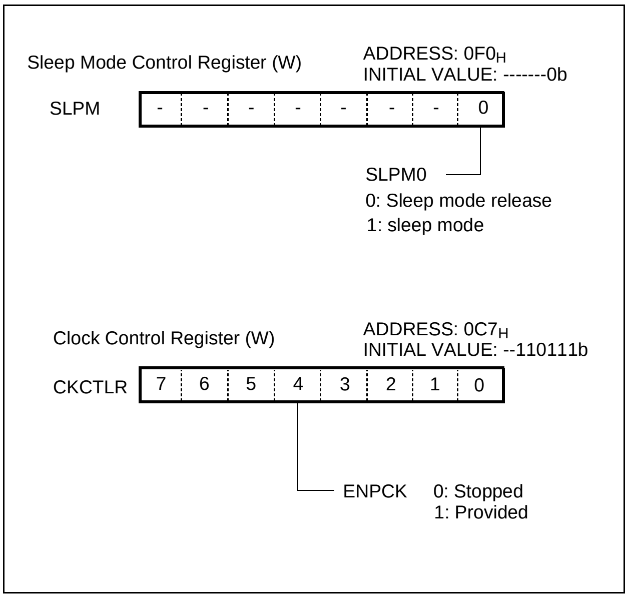
15. STANDBY FUNCTION

15.1 Sleep Mode

SLEEP mode can be entered by setting the bit of SLEEP mode register (SLPM). In the mode, CPU clock stops but oscillator keeps running. BIT and a part of peripheral hardware execute, but prescaler output which provide clock to peripherals can be stopped by program. (Except, PS10 can't stopped.) In SLEEP mode, more consuming power can be saved by not using other peripheral hardware except for BIT By setting ENPCK (peripheral clock control bit) of CKCTLR (clock control register) to "0", peripheral hardware halted, and SLEEP mode is entered. To release SLEEP mode by BITR (basic interval timer interrupt), bit10 of prescaler should be selected as BIT input clock before entering SLEEP mode. "NOP" instruction should be follows setting of SLEEP mode for rising precharge time of data bus line.

Example:

```
LDM    SLPM, #01H  ;setting of sleep
                    ;mode
NOP                ;NOP instruction
```



15.2 Stop Mode

STOP mode can be entered by STOP instruction during program. In STOP mode, oscillator is stopped to make all clocks stop, which leads to less power consumption. All registers and RAM data are preserved. "NOP" instruction should be follows STOP instruction for rising precharge time of data bus line.

Example:

```
STOP   ;STOP instruction
NOP    ;NOP instruction
```

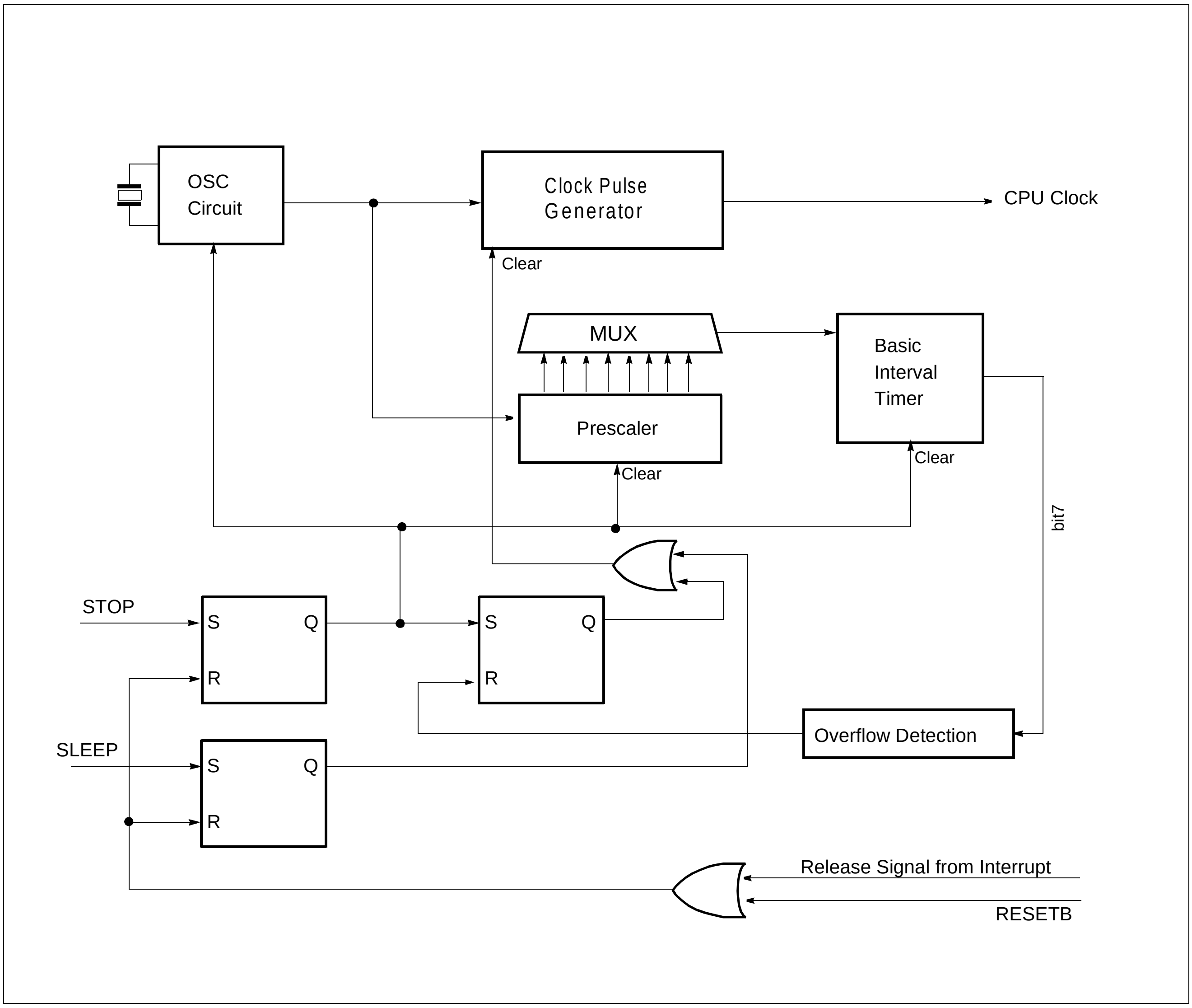


Figure 15-1 Block Diagram of Standby Circuit

15.3 Standby mode release

Release of STANDBY mode is executed by RESET input and Interrupt signal. Register value is defined when Reset. When there is a release signal of STOP mode (Interrupt, RESET input), the instruction execution starts after stabilization oscillation time is set by value of BTS2 ~ BTS0 and set ENPCK to “1”.

Release Signal	SLEEP	STOP
$\overline{\text{RESET}}$	O	O
KSCN(Key Input)	O	O
INT1,INT2	O	O
BIT, WDT	O	×

Table 15-1 Release Signal of Standby Mode

Release Factor	Release Method
RESETB	By RESETB Pin=Low level, Standby mode is released and system is initialized
KSCN(Key Input)	Standby mode is released by low input of selected pin by key scan Input(SMRR0,SMRR1). In case of interrupt mask enable flag= “0”, program executes just after standby instruction, if flag= “1” enters each interrupt service routine.
INT1,INT2	When external interrupt (INT1,INT2) enable flag is “1”, standby mode is released at the rising edge of each terminal. When standby mode is released at interrupt. Mask Enable flag= “0”, program executes from the next instruction of standby instruction. When “1”, enters each interrupt service routine.
Basic Interval Timer(IFBIT)	When BIT interrupt enable flag is “1”, SLEEP mode can be released by BIT overflow interrupt(IFBIT). When SLEEP mode is released by interrupt. Mask enable flag= “0”, program executes from the next instruction setting SLEEP mode. When “1”, enters each interrupt service routine.
Watch Dog Timer(IFWDT)	When WDT interrupt enable flag is “1”, SLEEP mode can be released by WDT overflow interrupt(IFWDT). When SLEEP mode is released by interrupt. Mask enable flag= “0”, program executes from the next instruction setting SLEEP mode. When “1”, enters each interrupt service routine.

Table 15-2 Standby mode release method

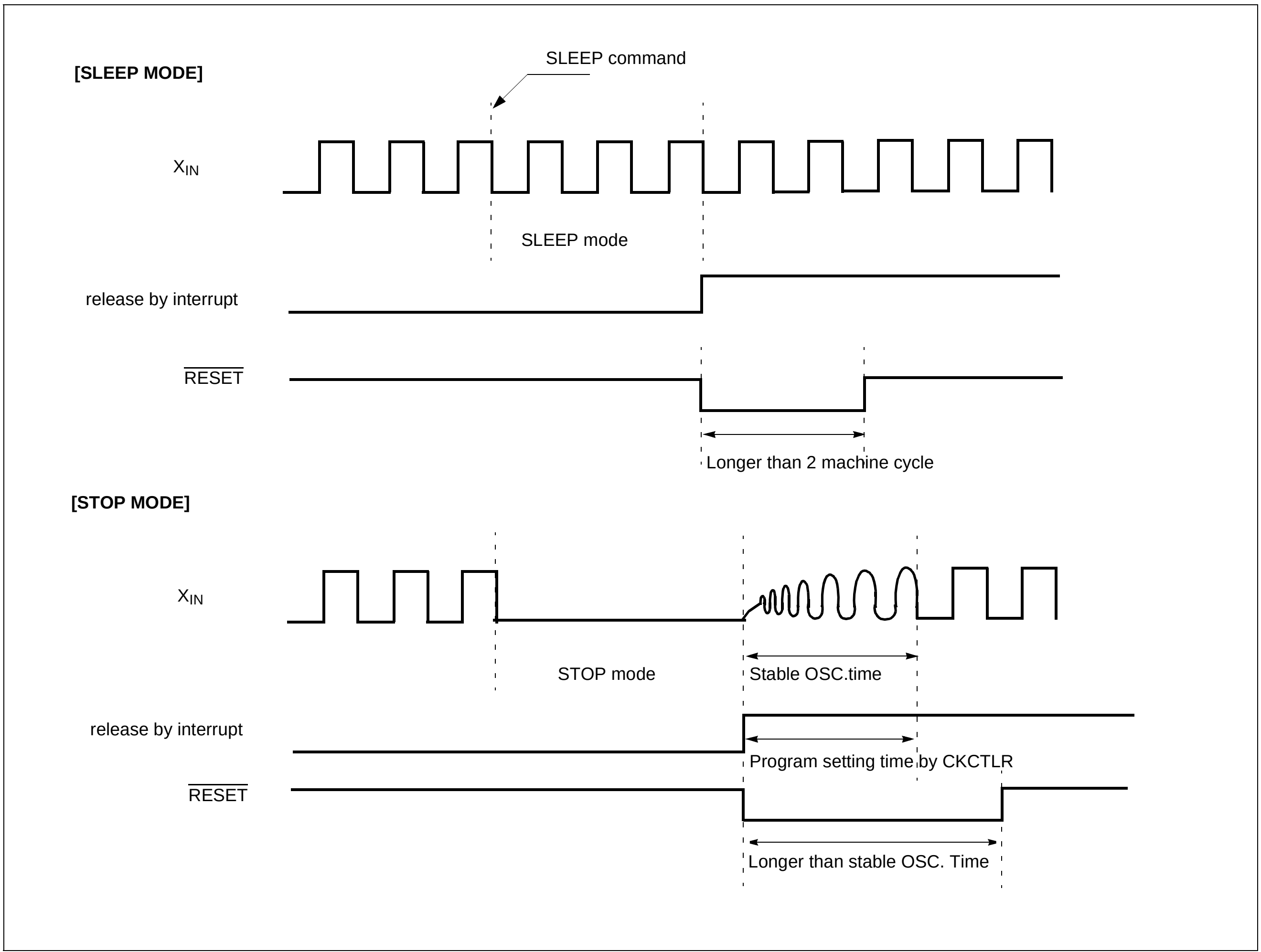


Figure 15-2 Block Diagram of Standby Circuit

15.4 Operation of standby mode release

After standby mode is released, the operation begins according to content of related interrupt register just before standby mode start (Figure 15-3).

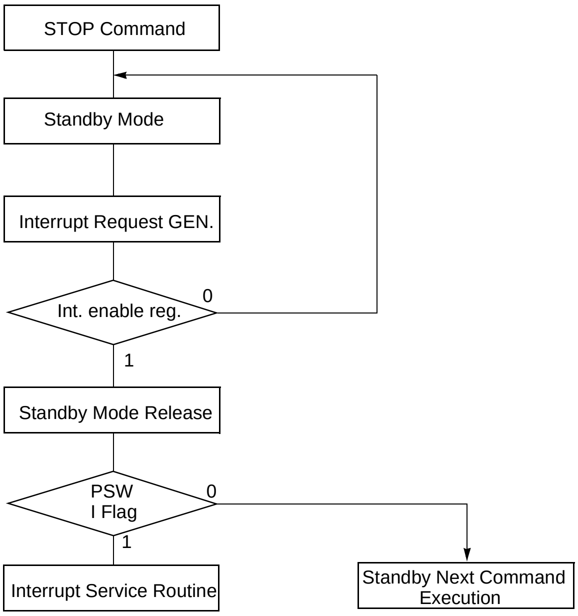


Figure 15-3 Standby Mode Release Flow

(1) Interrupt Enable Flag(I) of PSW = "0"

Release by only interrupt which interrupt enable flag = "1", and starts to execute from next to standby instruction (SLEEP or STOP).

(2) Interrupt Enable Flag(I) of PSW = "1"

Released by only interrupt which each interrupt enable flag = "1", and jump to the relevant interrupt service routine.

Note: *When STOP instruction is used, BIT should guarantee the stabilization oscillation time. Thus, just before entering STOP mode, clock of bit10 (PS10) of prescaler is selected or peripheral hardware clock control bit (ENPCK) to "1", Therefore the clock necessary for stabilization oscillation time should be input into BIT otherwise, standby mode is released by reset signal. In case of interrupt request flag and interrupt enable flag are both "1", standby mode is not entered.*

Internal circuit	SLEEP mode	STOP mode
Oscillator	Active	Stop
Internal CPU	Stop	Stop
Register	Retained	Retained
RAM	Retained	Retained
I/O port	Retained	Retained
Prescaler	Active	Stop
Basic Interval Timer	PS10 selected:Active Others: Stop	Stop
Watch-dog Timer	Active	Stop
Timer	Stop	Stop
Address Bus, Data Bus	Retained	Retained

Table 15-3 Operation State in Standby Mode

16. RESET FUNCTION

16.1 External RESET

The $\overline{\text{RESET}}$ pin should be held at low for at least 2machine cycles with the power supply voltage within the operating voltage range and must be connected 0.1uF capacitor for stable system initialization. The RESET pin contains a Schmitt trigger with an internal pull-up resistor.

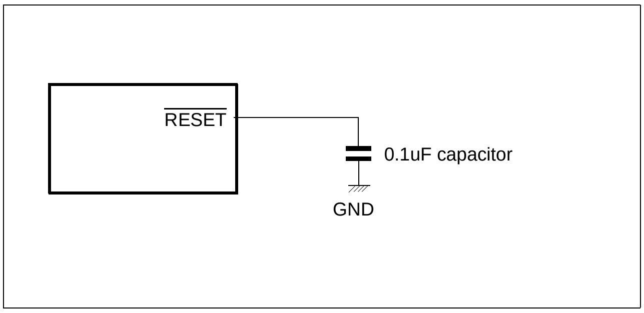


Figure 16-1 $\overline{\text{RESET}}$ Pin connection

16.2 Power on RESET

Power On Reset circuit automatically detects the rise of power voltage (the rising time should be within 50ms) the power voltage reaches a certain level, RESET terminal is maintained at “L” Level until a crystal ceramic oscillator oscillates stably. After power applies and starting of oscillation, this reset state is maintained for about oscillation cycle of 219 (about 65.5ms: at 4MHz). The execution of built-in Power On Reset circuit is as follows:

(1) Latch the pulse from Power On Detection Pulse Generator circuit, and reset Prescaler, BIT and BIT Overflow detection circuit.

(2) Once BIT Overflow detection circuit is reset. Then, Prescaler starts to count.

(3) Prescaler output is inputted into BIT and PS10 of Prescaler output is automatically selected. If overflow of BIT is detected, Overflow detection circuit is set.

4) Reset circuit generates maximum period of reset pulse from Prescaler and BIT

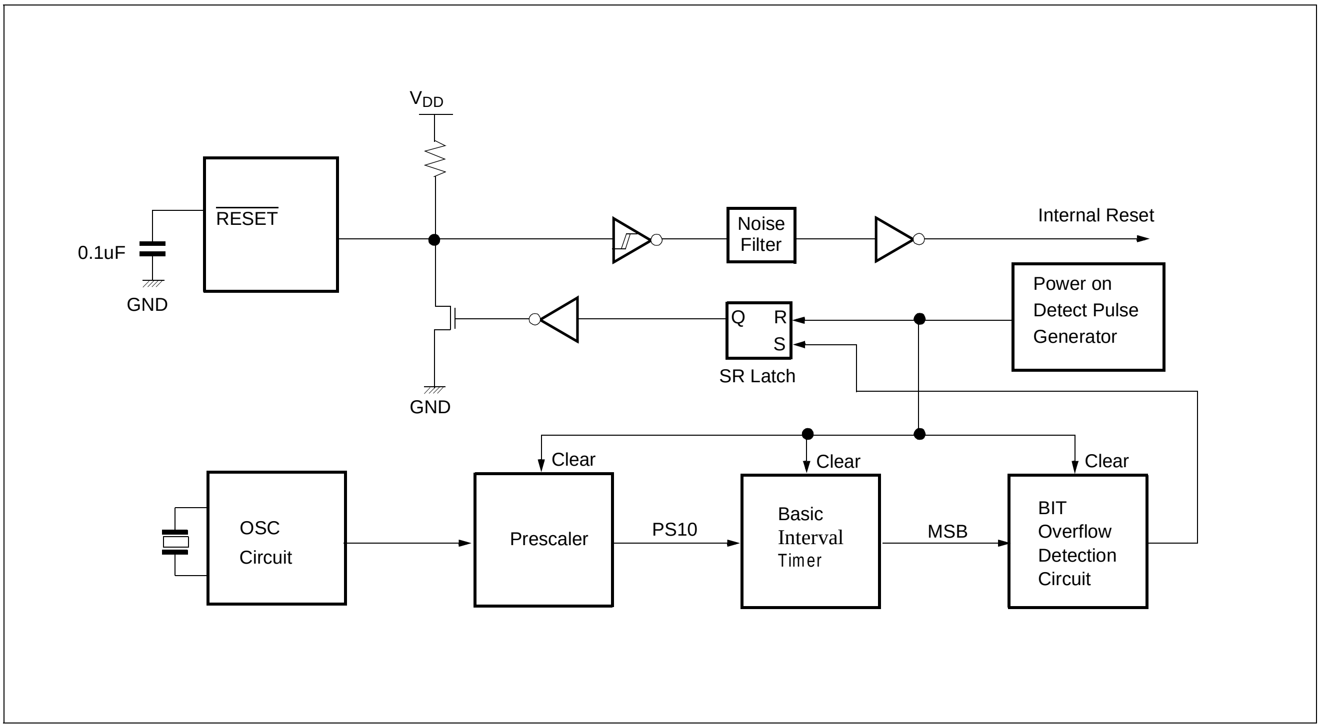


Figure 16-2 Block Diagram of Power On Reset Circuit

Note: *When Power On Reset, oscillator stabilization time doesn`t include OSC. Start time.*

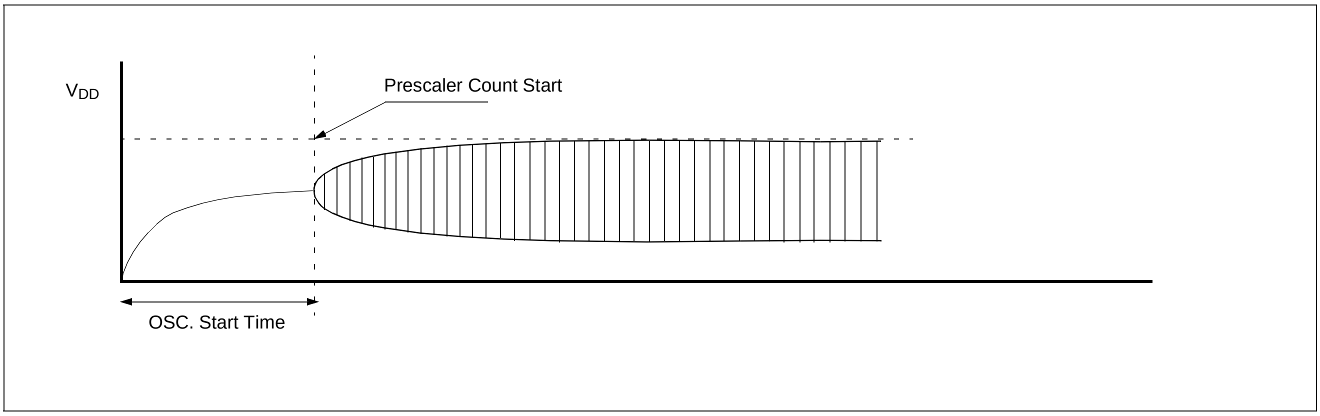
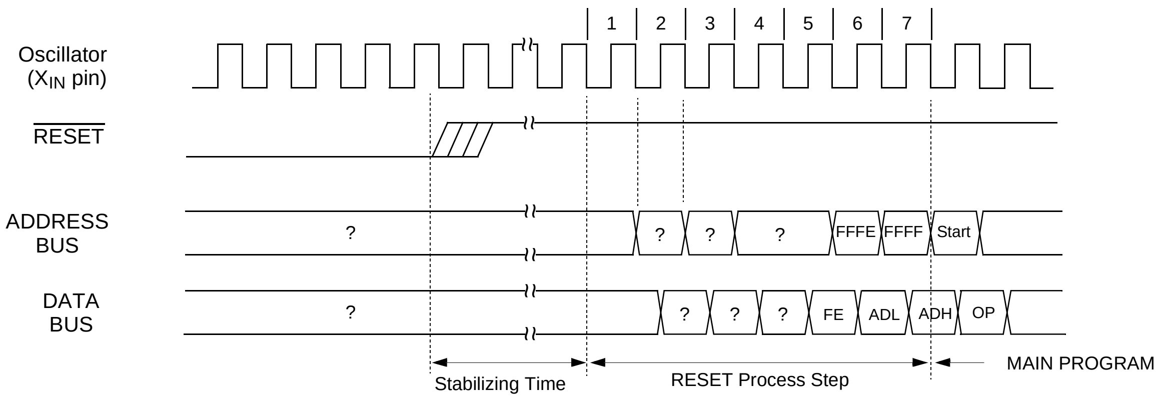


Figure 16-3 Oscillator stabilizing diagram



ADL and ADH are start addresses of interrupt service routine as vector contents.

Figure 16-4 Timing Diagram of Reset

16.3 Low voltage detection mode

(1) Low voltage detection condition

An on board voltage comparator checks that V_{DD} is at the required level to ensure correct operation of the device. If V_{DD} is below a certain level, Low voltage detector forces the device into low voltage detection mode.

(2) Low Voltage Detection Mode

There is no power consumption except stop current, stop mode release function is disabled. All I/O port is configured as input mode and Data memory is retained until voltage through external capacitor is worn out. In this mode, all port can be selected with Pull-up resistor by Mask option. If there is no information on the Mask option sheet, the default pull up option (all port connect to pull-up resistor) is selected.

(3) Release of Low Voltage Detection Mode

Reset signal result from new battery (normally 3V) wakes the low voltage detection mode and come into normal reset state. It depends on user whether to execute RAM clear routine or not

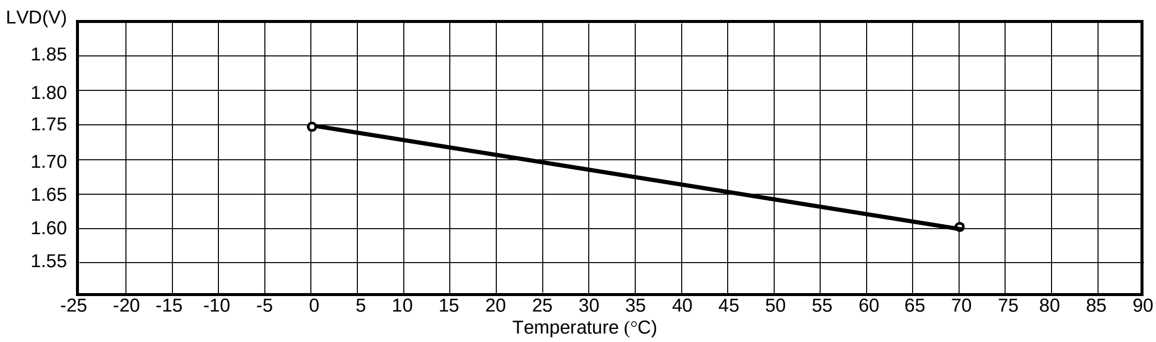


Figure 16-5 Low Voltage vs Temperature

(4)LVD Control register(LVDC)

LVDC

				W			
7	6	5	4	3	2	1	0
-	-	-	-	LD	-	-	-

ADDRESS: $0F1_H$
INITIAL VALUE: $----0---_B$

LD: LVD enable/disable bit when REMOUT outputs ‘High’
0: LVD enable
1: LVD disable at REMOUT=’H’

Figure 16-6 LVD control register(LVDC)

(5) SRAM BACK-UP after Low Voltage Detection.

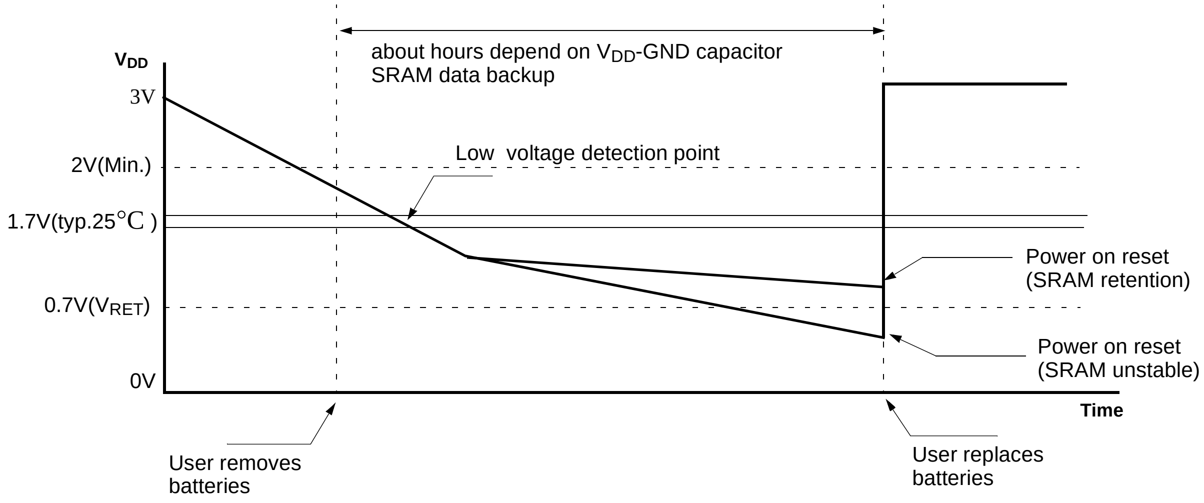


Figure 16-7 Low voltage detection and protection

Interrupt	disable
Stop release	disable
All I/O port	input mode
Remout port	low level
OSC	STOP
All I/O port pull-up resistor	on
SRAM data	retention until V_{RET}

Table 16-1 The operation after low voltage detection

(6) S/W flow chart example after Reset using SRAM Back-up

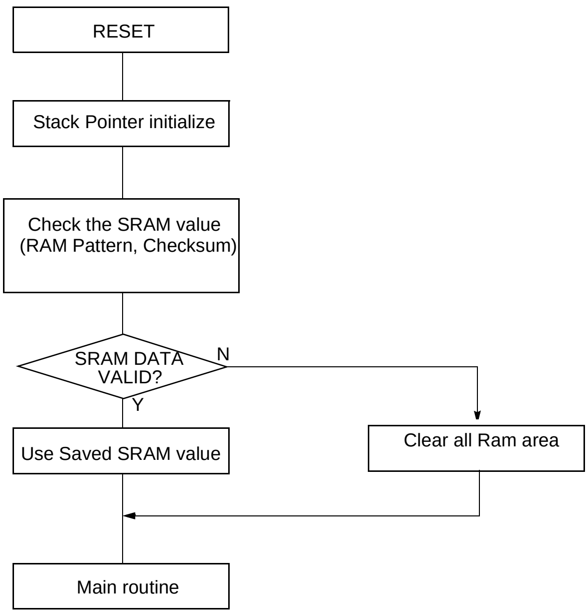


Figure 16-8 S/W flow chart example after Reset using SRAM Back-up

(7)Low voltage indication register(LVIR)

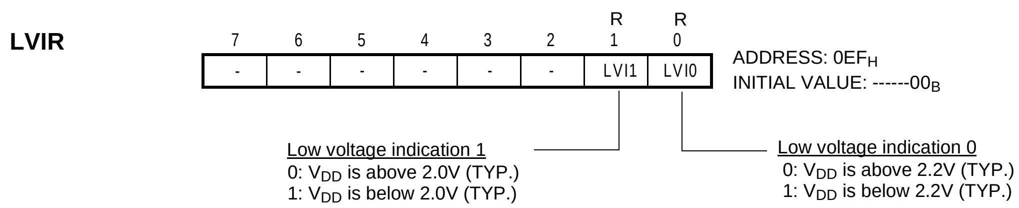
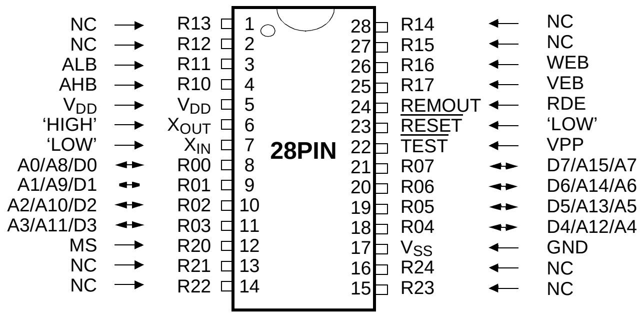


Figure 16-9 Low voltage indication registor(LVIR)

Low voltage indication register(LVIR) is read only register. It is useful to display the consumption of batteries. If V_{DD} power level is below a certain level which is higher than low voltage detection level, the bit of LVIR register could be set according to the V_{DD} level sequentially. The V_{DD} detection levels for indication are two, that is, Bit1 and Bit0 of LVIR registers. The detection levelof Bit0 is higher than Bit1.

17. EPROM MODE

17.1 PIN ASSIGNMENT (Top View)



17.2 EPROM mode pin function

PIN NUMBER	PIN NAME		I/O TYPE	FUNCTION
	STANDARD MODE	EPROM MODE		
1	R13	-	-	-
2	R12	-	-	-
3	R11	ALB	INPUT	Address low input enable
4	R10	AHB	INPUT	Address high input enable
5	V_{DD}	V_{DD}	POWER	Positive power supply
6	X_{OUT}	HIGH	OUTPUT	Logic high
7	X_{IN}	LOW	INPUT	Logic low
8	R00	A0/A8/D0	I/O	Input mode - Address: 3~0 - Address: 11~8, Data:3~0 Output mode - Data: 3~0
9	R01	A1/A9/D1	I/O	
10	R02	A2/A10/D2	I/O	
11	R03	A3/A11/D3	I/O	
12	R20	MS	INPUT	-
13	R21	-	-	-
14	R22	-	-	-
15	R23	-	-	-
16	R24	-	-	-
17	V_{SS}	V_{SS}	POWER	Ground
18	R04	A4/A12/D4	I/O	Input mode - Address: 7~4 - Address: 15~12, Data:7~4 Output mode - Data: 7~4
19	R05	A5/A13/D5	I/O	
20	R06	A6/A14/D6	I/O	
21	R07	A7/A15/D7	I/O	
22	$\overline{TEST}$	V_{PP}	INPUT	Program voltage
23	$\overline{RESET}$	LOW	INPUT	Logic low
24	REMOUT	RDE	INPUT	-
25	R17	VEB	INPUT	Verify enable
26	R16	WEB	INPUT	Program enable
27	R15	-	-	-
28	R14	-	-	-

17.3 EPROM Mode setting

Setting mode

When setting mode	V_{PP}	RDE	MS	VEB	WEB	ALB	AHB
PROM write & verify	11.5V	H	H	H	H	H	H
PROM verify	11.5V	H	H	H	H	H	H
LOCK bit write	11.5V	H	L	H	H	H	H
LOCK bit read	11.5V	H	L	H	H	H	H

Table 17-1 Setting mode

After setting mode

After setting mode	V_{PP}	RDE	MS	VEB	WEB	ALB	AHB	Remark
PROM write & verify	11.5V	L	H	H	H	H	L	Address high latch
				H	H	L	H	Address low latch
				H	L	H	H	PROM write
				L	H	H	H	PROM verify
PROM verify	11.5V	L	H	H	H	H	L	Address high latch
				H	H	L	H	Address low latch
				L	H	H	H	PROM verify
LOCK bit write	11.5V	L	L	H	H	H	L	R0=6X: LOCK Bit write data
				H	L	H	H	program
LOCK bit read	11.5V	L	L	H	H	H	L	LOCK latch
				H	H	L	H	LOCK read (R04 port high, R05 port low)

Table 17-2 after setting mode

17.4 General view of EPROM operation

Programming DC characteristics

(Ta= 25±5°C)

Item	Symbol	Min.	Typ.	Max.	Unit	Test condition
Input 'H' voltage	V_{IH}	4.5	V_{DD}	V_{DD}+0.3	V	-
Input 'L' voltage	V_{IL}	-0.1	V_{SS}	0.4	V	-
Output 'H' voltage	V_{OH}	4.0	-	-	V	I_{OH} = -2.5mA
Output 'L' voltage	V_{OL}	-	-	0.4	V	I_{OL} = 1mA
V_{DD} current	I_{DD}	-	-	100	mA	-
V_{PP} current	I_{PP}	-	-	100	mA	WEB=V_{IL}
V_{DD} voltage	V_{DD}	4.5	5.0	5.5	V	-
V_{PP} voltage	V_{PP}	11.2	11.5	11.8	V	-

Table 17-3 Programming DC characteristics

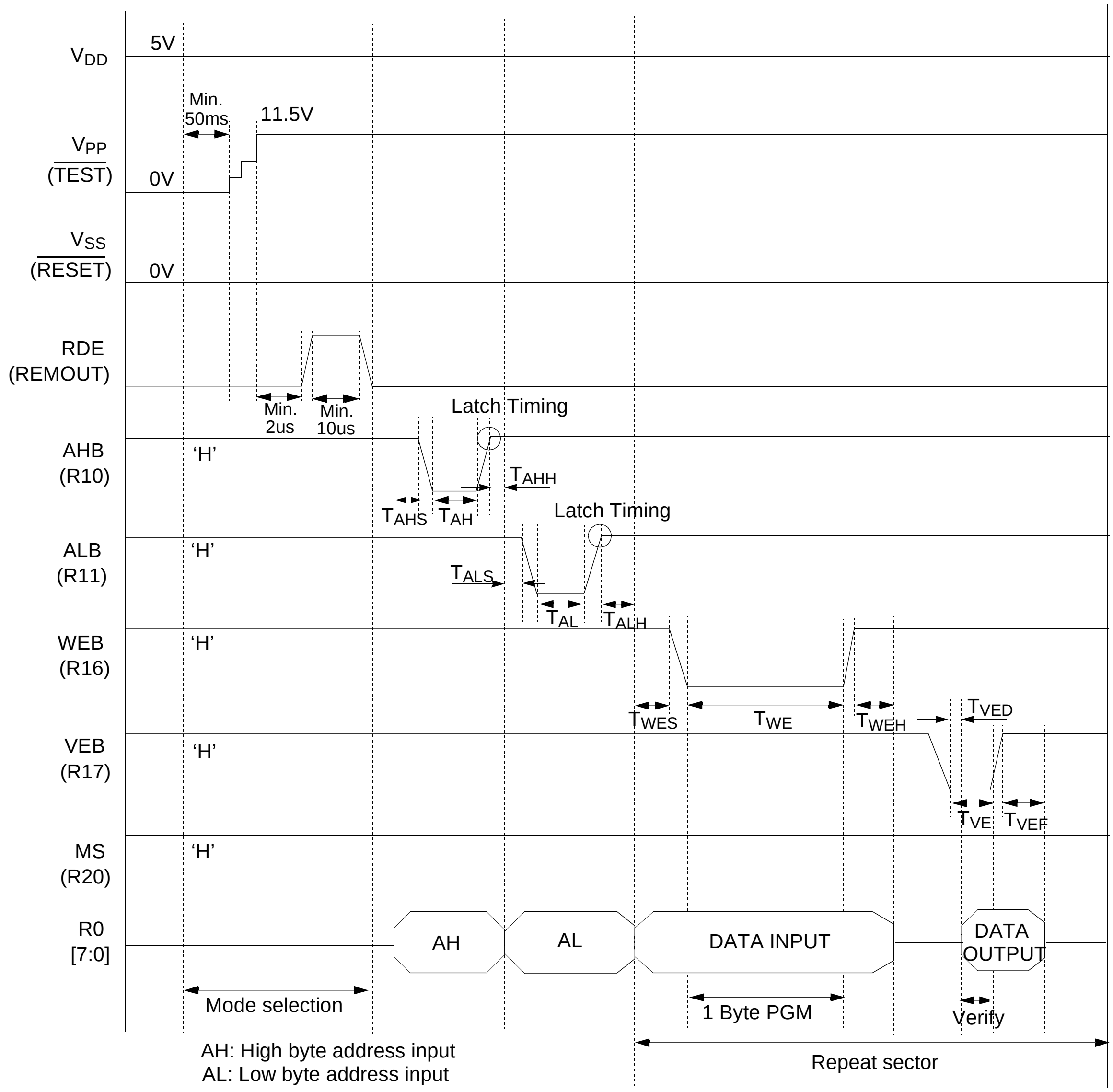
Programming AC characteristics

(V_{DD}=5.0V±0.5V, V_{PP} =11.5V±0.3V, V_{SS} =0V,Ta= 25±5°C)

Item	Symbol	Min.	Typ.	Max.	Unit
High address setup time	T_{AHS}	2	-	-	uS
High address pulse width	T_{AH}	2	-	-	uS
High address hold time	T_{AHH}	2	-	-	uS
Low address setup time	T_{ALS}	2	-	-	uS
Low address pulse width	T_{AL}	2	-	-	uS
Low address hold time	T_{ALH}	2	-	-	uS
Program setup time	T_{WES}	2	-	-	uS
Program pulse width	T_{WE}	180	200	220	uS
Program hold time	T_{WEH}	2	-	-	uS
Address delay time	T_{ADD}	-	-	170	nS
Data out delay time	T_{VED}	-	-	150	nS
Data out time	T_{VE}	2	-	-	uS
Data out floating time	T_{VEF}	-	-	130	nS

Table 17-4 Programming AC characteristics

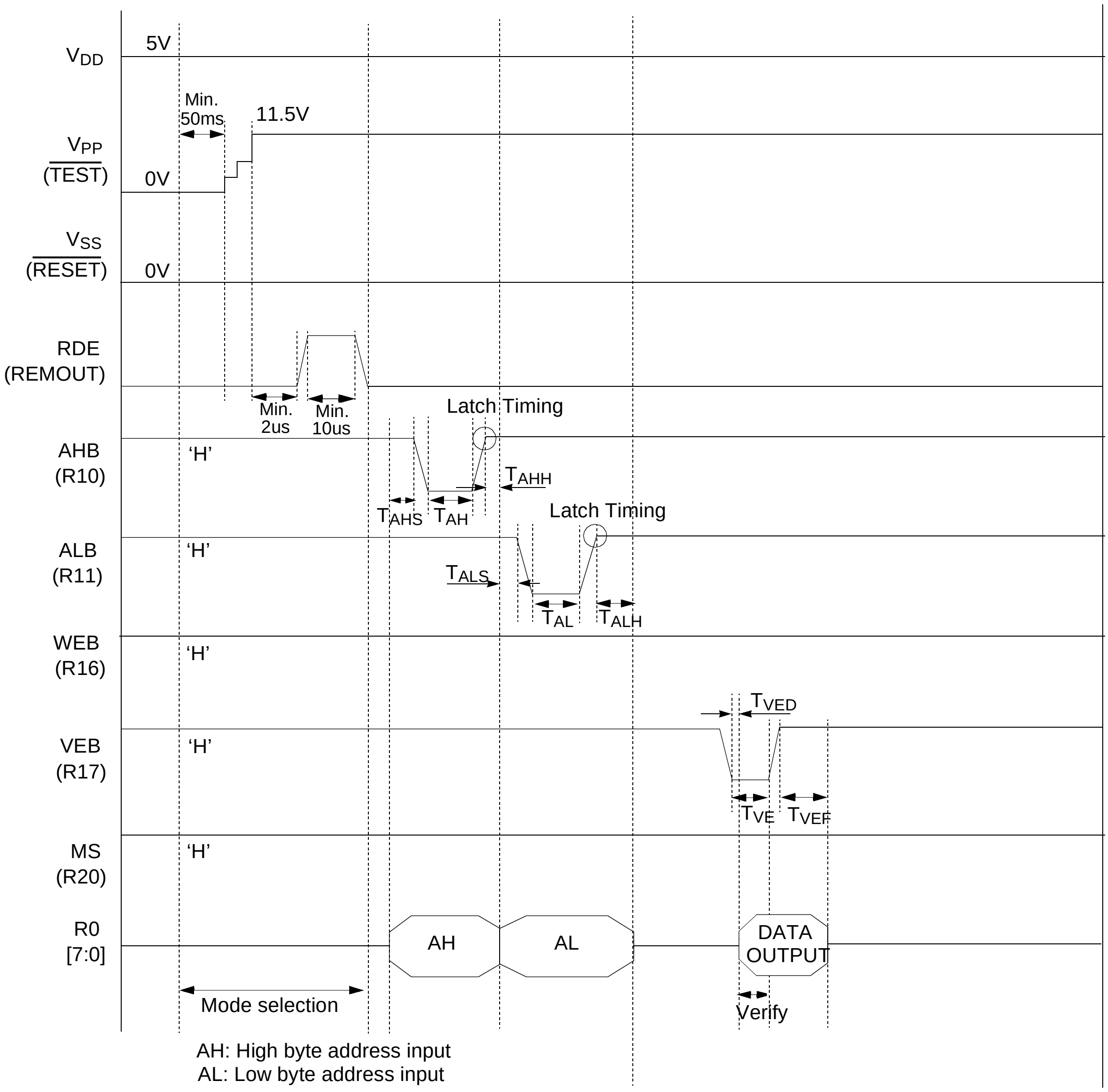
17.5 EPROM write & verify



1. AH and AL, inputs of R0[7:0] address are latched to IC at rising edge of R10 and R11.
2. DATA OUTPUT, output of R0[7:0] verify data is out during the period of R17 'Low'.
3. If writing is not completed in 200us of R16 'Low' the action of repeat sector is done more again.
   After confirming the writing success, it is repeated more 2 times. If the number of repeat action
   is more 20, EPROM write & verify mode is failure.
4. When V_{PP} is supplied, it is recommended to raise the voltage step by step for preventing voltage overshoot.

Figure 17-1 EPROM write & verify

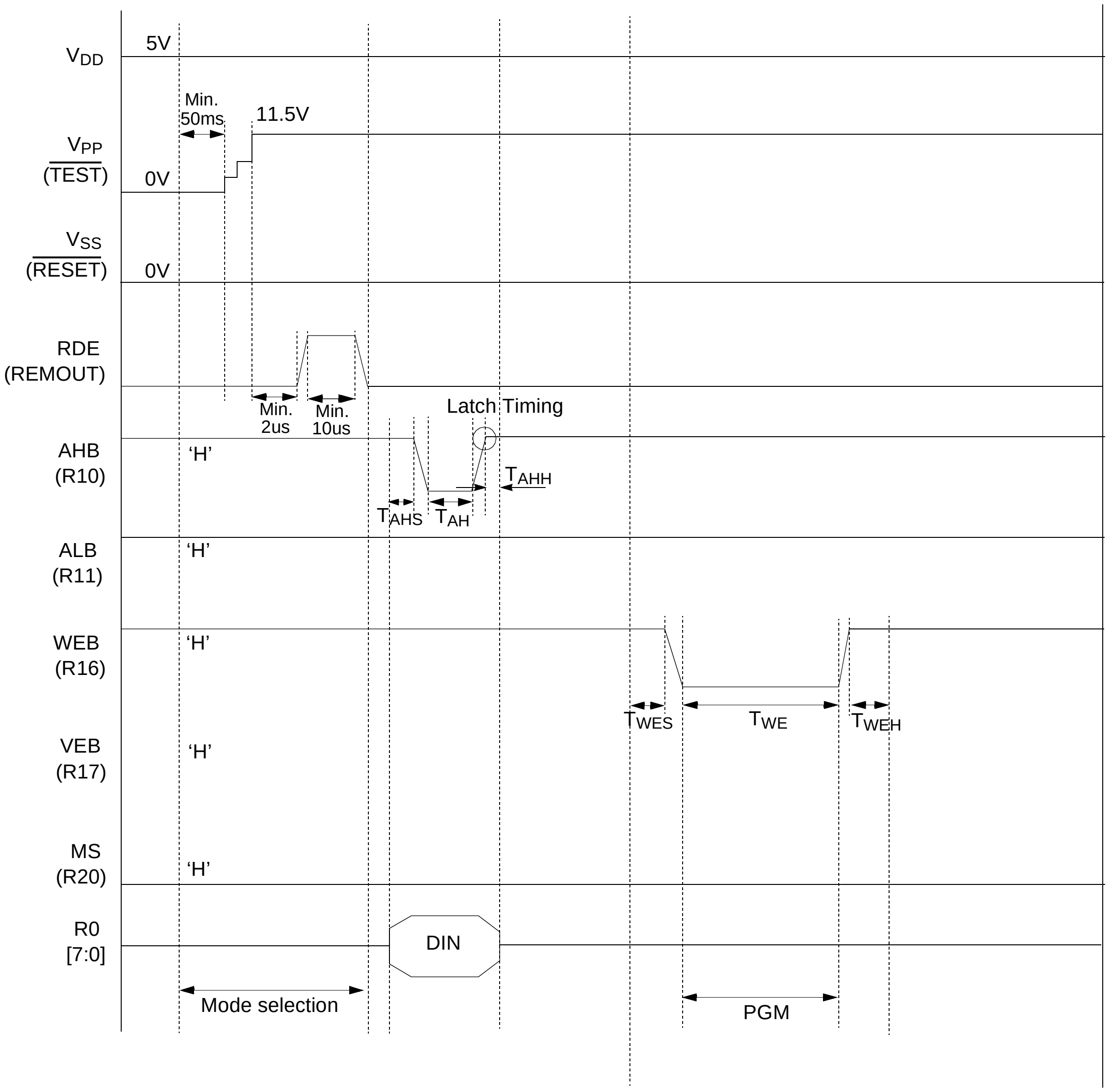
17.6 EPROM verify



1. AH and AL, inputs of R0[7:0] address are latched to IC at rising edge of R10 and R11.
2. DATA OUTPUT, output of R0[7:0] verify data is out during the period of R17 'Low'.

Figure 17-2 EPROM verify

17.7 LOCK bit write

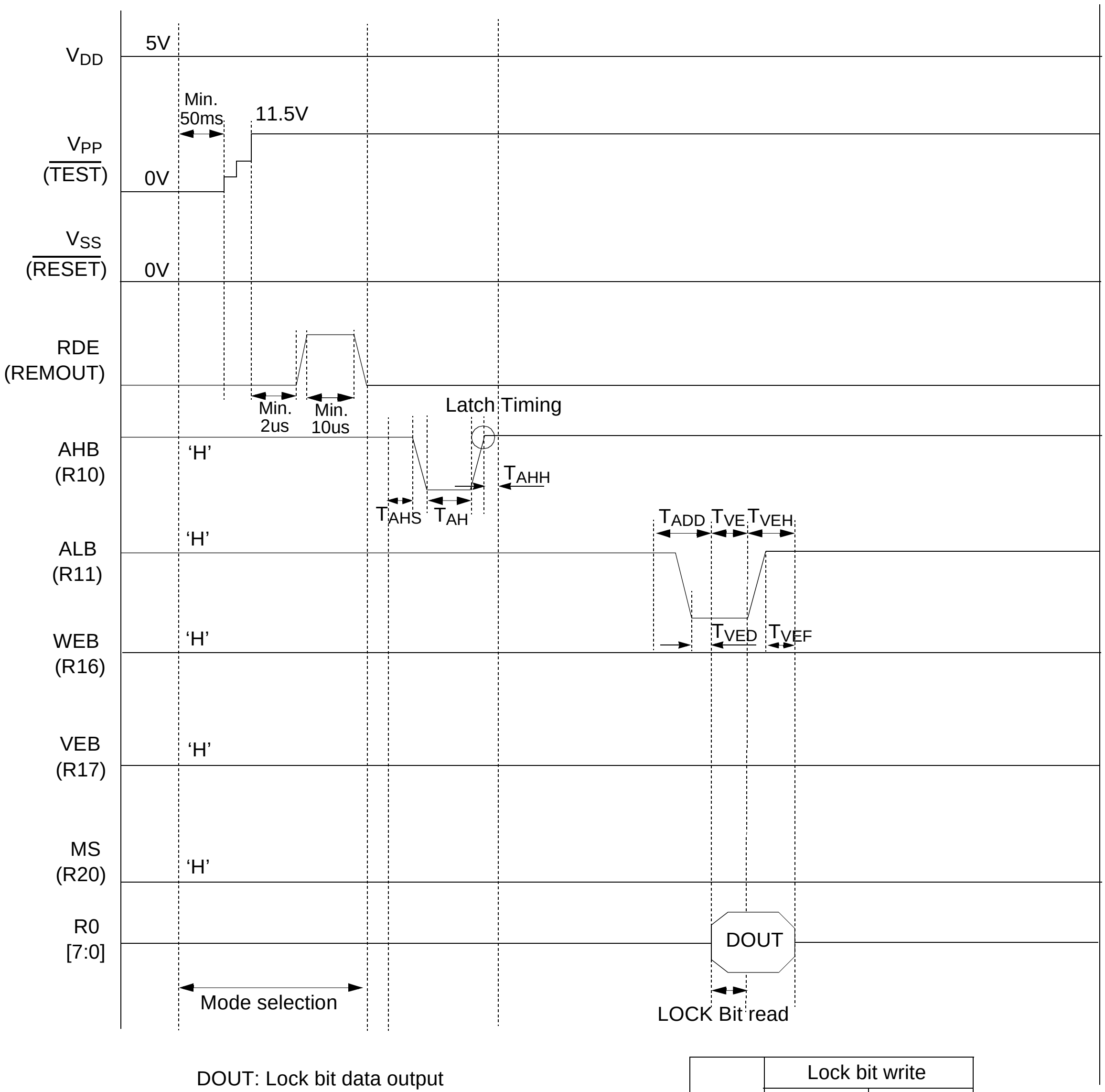


DIN=0110_XXXX: Lock bit write data

1. The steps of T_{WE} is repeated 3 times. The period of T_{WE} is 200us when R16 is 'Low'
2. When V_{PP} is supplied, it is recommended to raise the voltage step by step for preventing voltage overshoot.

Figure 17-3 LOCK bit write

17.8 LOCK bit read



	Lock bit write	
	Before	After
R04 R05	L H	H L

1. Lock data is latched at rising edge of R10.
2. DOUT, output of R0[5:4] Lock bit read data, is out during the period of R11 'Low'

Figure 17-4 LOCK bit read

17.9 Considerations on programming

Writing must be occurred in recommended time and voltage. Programming voltage is 11.5V on EPROM mode. If the voltage including the overshooting voltage is higher than the recommendation, device will be broken out not to be recoverable. When V_{PP} is supplied, it is recommended to raise the voltage step by step(0V→6V→10V→11.5V) for preventing voltage overshoot. It is essential to check the socket of EPROM writer and characteristics of socket adapter before using them. If the user touch the socket adapter or device on writing mode, writing error can be occurred by contact problem. Power capacitor and noise filtering capacitor should be located as near as possible from V_{PP} pin.

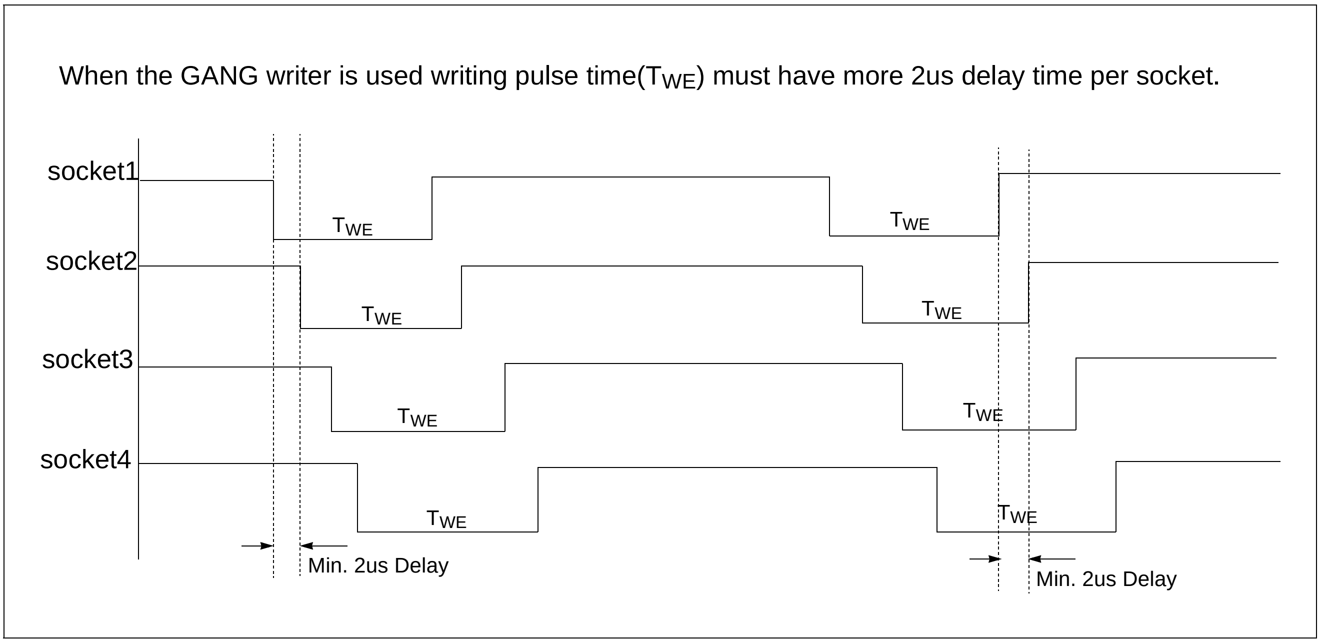
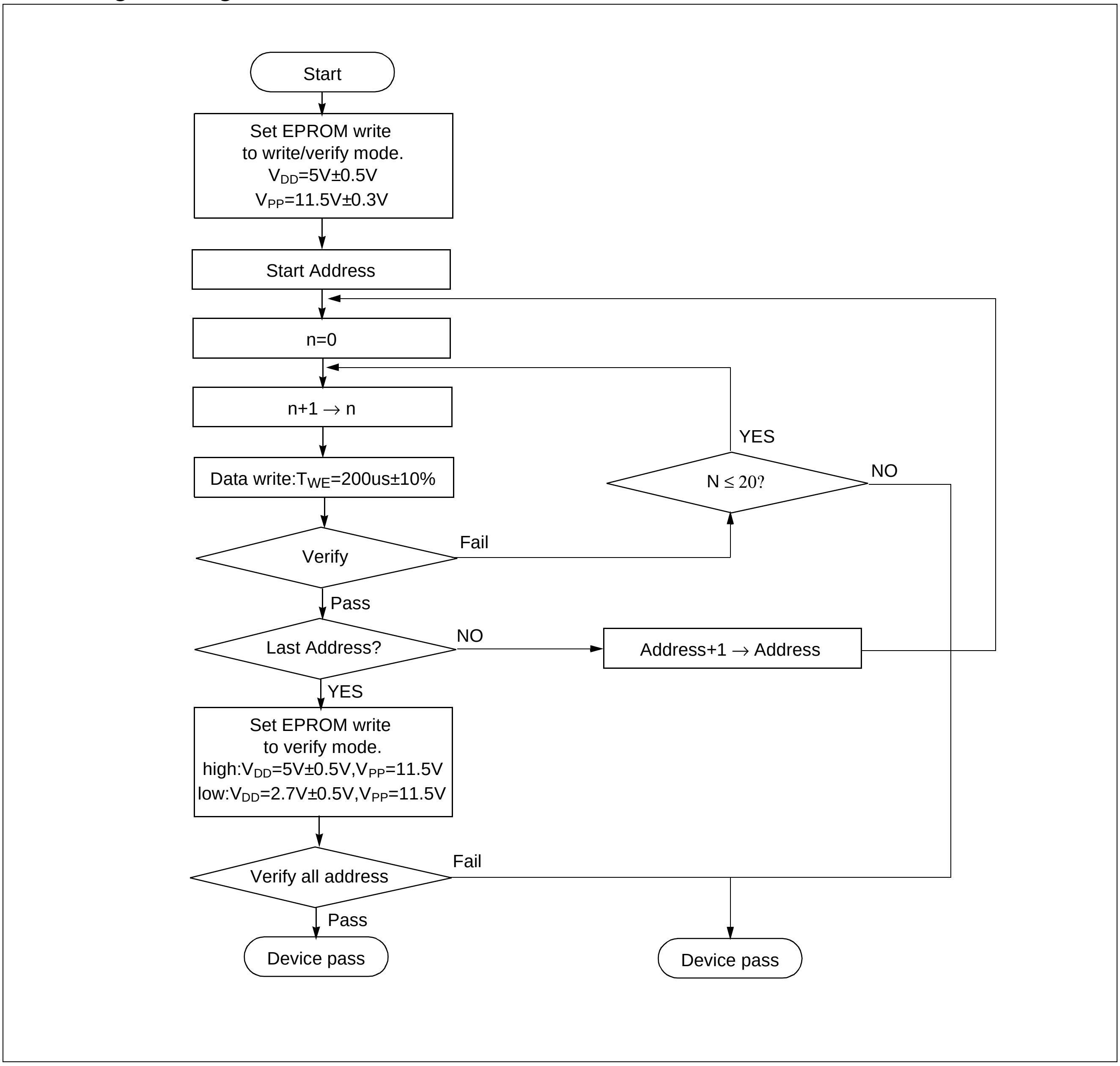
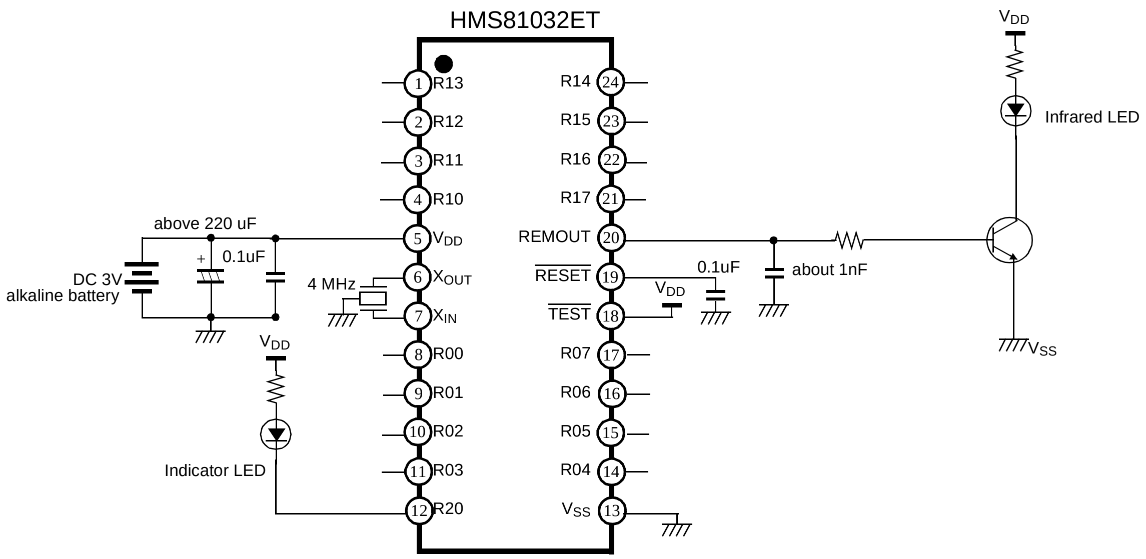


Figure 17-5 Writing pulse time

17.10 Programming flow chart



18. Recommended circuit diagram of HMS81032ET



1. Tr. and IR LED should be located as far as possible from the IC.
2. Power capacitor and noise filtering capacitor should be located as near as possible from V_{DD} pin.
3. Ceramic resonator should be located as near as possible from the IC.

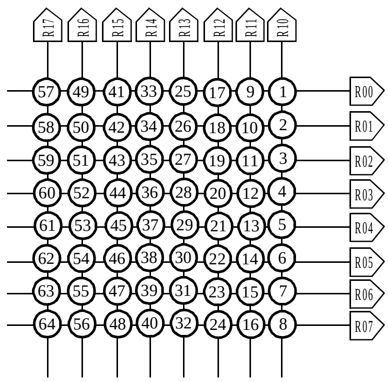


Figure 18-1 GENERAL CIRCUIT DIAGRAM OF HMS81032ET

APPENDIX

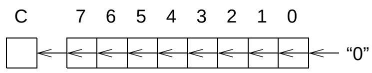
A. INSTRUCTION MAP

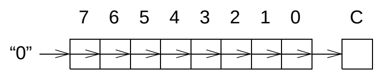
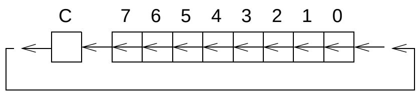
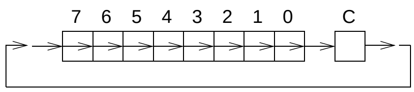
LOW HIGH	00000 00	00001 01	00010 02	00011 03	00100 04	00101 05	00110 06	00111 07	01000 08	01001 09	01010 0A	01011 0B	01100 0C	01101 0D	01110 0E	01111 0F
000	-	SET1 dp.bit	BBS A.bit,rel	BBS dp.bit,rel	ADC #imm	ADC dp	ADC dp+X	ADC !abs	ASL A	ASL dp	TCALL 0	SETA1 .bit	BIT dp	POP A	PUSH A	BRK
001	CLRC				SBC #imm	SBC dp	SBC dp+X	SBC !abs	ROL A	ROL dp	TCALL 2	CLRA1 .bit	COM dp	POP X	PUSH X	BRA rel
010	CLRG				CMP #imm	CMP dp	CMP dp+X	CMP !abs	LSR A	LSR dp	TCALL 4	NOT1 M.bit	TST dp	POP Y	PUSH Y	PCALL Upage
011	DI				OR #imm	OR dp	OR dp+X	OR !abs	ROR A	ROR dp	TCALL 6	OR1 OR1B	CMPX dp	POP PSW	PUSH PSW	RET
100	CLRV				AND #imm	AND dp	AND dp+X	AND !abs	INC A	INC dp	TCALL 8	AND1 AND1B	CMPY dp	CBNE dp+X	TXSP	INC X
101	SETC				EOR #imm	EOR dp	EOR dp+X	EOR !abs	DEC A	DEC dp	TCALL 10	EOR1 EOR1B	DBNE dp	XMA dp+X	TSPX	DEC X
110	SETG				LDA #imm	LDA dp	LDA dp+X	LDA !abs	TXA	LDY dp	TCALL 12	LDC LDCB	LDX dp	LDX dp+Y	XCN	DAS
111	EI				LDM dp,#imm	STA dp	STA dp+X	STA !abs	TAX	STY dp	TCALL 14	STC M.bit	STX dp	STX dp+Y	XAX	STOP

LOW HIGH	10000 10	10001 11	10010 12	10011 13	10100 14	10101 15	10110 16	10111 17	11000 18	11001 19	11010 1A	11011 1B	11100 1C	11101 1D	11110 1E	11111 1F
000	BPL rel	CLR1 dp.bit	BBC A.bit,rel	BBC dp.bit,rel	ADC {X}	ADC !abs+Y	ADC [dp+X]	ADC [dp]+Y	ASL !abs	ASL dp+X	TCALL 1	JMP !abs	BIT !abs	ADDW dp	LDX #imm	JMP [!abs]
001	BVC rel				SBC {X}	SBC !abs+Y	SBC [dp+X]	SBC [dp]+Y	ROL !abs	ROL dp+X	TCALL 3	CALL !abs	TEST !abs	SUBW dp	LDY #imm	JMP [dp]
010	BCC rel				CMP {X}	CMP !abs+Y	CMP [dp+X]	CMP [dp]+Y	LSR !abs	LSR dp+X	TCALL 5	MUL	TCLR1 !abs	CMPW dp	CMPX #imm	CALL [dp]
011	BNE rel				OR {X}	OR !abs+Y	OR [dp+X]	OR [dp]+Y	ROR !abs	ROR dp+X	TCALL 7	DBNE Y	CMPX !abs	LDYA dp	CMPY #imm	RETI
100	BMI rel				AND {X}	AND !abs+Y	AND [dp+X]	AND [dp]+Y	INC !abs	INC dp+X	TCALL 9	DIV	CMPY !abs	INCW dp	INC Y	TAY
101	BVS rel				EOR {X}	EOR !abs+Y	EOR [dp+X]	EOR [dp]+Y	DEC !abs	DEC dp+X	TCALL 11	XMA {X}	XMA dp	DECW dp	DEC Y	TYA
110	BCS rel				LDA {X}	LDA !abs+Y	LDA [dp+X]	LDA [dp]+Y	LDY !abs	LDY dp+X	TCALL 13	LDA {X}+	LDX !abs	STYA dp	XAY	DAA
111	BEQ rel				STA {X}	STA !abs+Y	STA [dp+X]	STA [dp]+Y	STY !abs	STY dp+X	TCALL 15	STA {X}+	STX !abs	CBNE dp	XYX	NOP

B. INSTRUCTION SET

1. ARITHMETIC/ LOGIC OPERATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	ADC #imm	04	2	2	Add with carry.	NV--H-ZC
2	ADC dp	05	2	3	$A \leftarrow (A) + (M) + C$	
3	ADC dp + X	06	2	4		
4	ADC !abs	07	3	4		
5	ADC !abs + Y	15	3	5		
6	ADC [dp + X]	16	2	6		
7	ADC [dp] + Y	17	2	6		
8	ADC { X }	14	1	3		
9	AND #imm	84	2	2	Logical AND	N-----Z-
10	AND dp	85	2	3	$A \leftarrow (A) \wedge (M)$	
11	AND dp + X	86	2	4		
12	AND !abs	87	3	4		
13	AND !abs + Y	95	3	5		
14	AND [dp + X]	96	2	6		
15	AND [dp] + Y	97	2	6		
16	AND { X }	94	1	3		
17	ASL A	08	1	2	Arithmetic shift left	N-----ZC
18	ASL dp	09	2	4		
19	ASL dp + X	19	2	5		
20	ASL !abs	18	3	5		
21	CMP #imm	44	2	2	Compare accumulator contents with memory contents	N-----ZC
22	CMP dp	45	2	3	$(A) - (M)$	
23	CMP dp + X	46	2	4		
24	CMP !abs	47	3	4		
25	CMP !abs + Y	55	3	5		
26	CMP [dp + X]	56	2	6		
27	CMP [dp] + Y	57	2	6		
28	CMP { X }	54	1	3		
29	CMPX #imm	5E	2	2	Compare X contents with memory contents	N-----ZC
30	CMPX dp	6C	2	3	$(X) - (M)$	
31	CMPX !abs	7C	3	4		
32	CMPY #imm	7E	2	2	Compare Y contents with memory contents	N-----ZC
33	CMPY dp	8C	2	3	$(Y) - (M)$	
34	CMPY !abs	9C	3	4		
35	COM dp	2C	2	4	1'S Complement : $(dp) \leftarrow \sim(dp)$	N-----Z-
36	DAA	DF	1	3	Decimal adjust for addition	N-----ZC
37	DAS	CF	1	3	Decimal adjust for subtraction	N-----ZC
38	DEC A	A8	1	2	Decrement	N-----Z-
39	DEC dp	A9	2	4	$M \leftarrow (M) - 1$	
40	DEC dp + X	B9	2	5		N-----Z-
41	DEC !abs	B8	3	5		
42	DEC X	AF	1	2		
43	DEC Y	BE	1	2		
44	DIV	9B	1	12	Divide : YA / X Q: A, R: Y	NV--H-Z-

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
45	EOR #imm	A4	2	2	Exclusive OR	
46	EOR dp	A5	2	3	A ← (A) ⊕ (M)	
47	EOR dp + X	A6	2	4		
48	EOR !abs	A7	3	4		N-----Z-
49	EOR !abs + Y	B5	3	5		
50	EOR [dp + X]	B6	2	6		
51	EOR [dp] + Y	B7	2	6		
52	EOR { X }	B4	1	3		
53	INC A	88	1	2	Increment	N-----Z-
54	INC dp	89	2	4	M ← (M) + 1	
55	INC dp + X	99	2	5		N-----Z-
56	INC !abs	98	3	5		
57	INC X	8F	1	2		
58	INC Y	9E	1	2		
59	LSR A	48	1	2	Logical shift right	
60	LSR dp	49	2	4		N-----ZC
61	LSR dp + X	59	2	5		
62	LSR !abs	58	3	5		
63	MUL	5B	1	9	Multiply : YA ← Y × A	N-----Z-
64	OR #imm	64	2	2	Logical OR	
65	OR dp	65	2	3	A ← (A) ∨ (M)	
66	OR dp + X	66	2	4		
67	OR !abs	67	3	4		N-----Z-
68	OR !abs + Y	75	3	5		
69	OR [dp + X]	76	2	6		
70	OR [dp] + Y	77	2	6		
71	OR { X }	74	1	3		
72	ROL A	28	1	2	Rotate left through carry	
73	ROL dp	29	2	4		N-----ZC
74	ROL dp + X	39	2	5		
75	ROL !abs	38	3	5		
76	ROR A	68	1	2	Rotate right through carry	
77	ROR dp	69	2	4		N-----ZC
78	ROR dp + X	79	2	5		
79	ROR !abs	78	3	5		
80	SBC #imm	24	2	2	Subtract with carry	
81	SBC dp	25	2	3	A ← (A) - (M) - ~(C)	
82	SBC dp + X	26	2	4		
83	SBC !abs	27	3	4		NV--HZC
84	SBC !abs + Y	35	3	5		
85	SBC [dp + X]	36	2	6		
86	SBC [dp] + Y	37	2	6		
87	SBC { X }	34	1	3		
88	TST dp	4C	2	3	Test memory contents for negative or zero (dp) - 00_H	N-----Z-
89	XCN	CE	1	5	Exchange nibbles within the accumulator A_7~A_4 ↔ A_3~A_0	N-----Z-

2. REGISTER / MEMORY OPERATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	LDA #imm	C4	2	2	Load accumulator A ← (M)	N-----Z-
2	LDA dp	C5	2	3		
3	LDA dp + X	C6	2	4		
4	LDA !abs	C7	3	4		
5	LDA !abs + Y	D5	3	5		
6	LDA [dp + X]	D6	2	6		
7	LDA [dp] + Y	D7	2	6		
8	LDA { X }	D4	1	3		
9	LDA { X }+	DB	1	4	X- register auto-increment : A ← (M) , X ← X + 1	
10	LDM dp,#imm	E4	3	5	Load memory with immediate data : (M) ← imm	--------
11	LDX #imm	1E	2	2	Load X-register X ← (M)	N-----Z-
12	LDX dp	CC	2	3		
13	LDX dp + Y	CD	2	4		
14	LDX !abs	DC	3	4		
15	LDY #imm	3E	2	2	Load Y-register Y ← (M)	N-----Z-
16	LDY dp	C9	2	3		
17	LDY dp + X	D9	2	4		
18	LDY !abs	D8	3	4		
19	STA dp	E5	2	4	Store accumulator contents in memory (M) ← A	--------
20	STA dp + X	E6	2	5		
21	STA !abs	E7	3	5		
22	STA !abs + Y	F5	3	6		
23	STA [dp + X]	F6	2	7		
24	STA [dp] + Y	F7	2	7		
25	STA { X }	F4	1	4		
26	STA { X }+	FB	1	4	X- register auto-increment : (M) ← A, X ← X + 1	
27	STX dp	EC	2	4	Store X-register contents in memory (M) ← X	--------
28	STX dp + Y	ED	2	5		
29	STX !abs	FC	3	5		
30	STY dp	E9	2	4	Store Y-register contents in memory (M) ← Y	--------
31	STY dp + X	F9	2	5		
32	STY !abs	F8	3	5		
33	TAX	E8	1	2	Transfer accumulator contents to X-register : X ← A	N-----Z-
34	TAY	9F	1	2	Transfer accumulator contents to Y-register : Y ← A	N-----Z-
35	TSPX	AE	1	2	Transfer stack-pointer contents to X-register : X ← sp	N-----Z-
36	TXA	C8	1	2	Transfer X-register contents to accumulator: A ← X	N-----Z-
37	TXSP	8E	1	2	Transfer X-register contents to stack-pointer: sp ← X	N-----Z-
38	TYA	BF	1	2	Transfer Y-register contents to accumulator: A ← Y	N-----Z-
39	XAX	EE	1	4	Exchange X-register contents with accumulator :X ↔ A	--------
40	XAY	DE	1	4	Exchange Y-register contents with accumulator :Y ↔ A	--------
41	XMA dp	BC	2	5	Exchange memory contents with accumulator (M) ↔ A	N-----Z-
42	XMA dp+X	AD	2	6		
43	XMA {X}	BB	1	5		
44	XYX	FE	1	4	Exchange X-register contents with Y-register : X ↔ Y	--------

3. 16-BIT OPERATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	ADDW dp	1D	2	5	16-Bits add without carry YA ← (YA) + (dp +1) (dp)	NV--H-ZC
2	CMPW dp	5D	2	4	Compare YA contents with memory pair contents : (YA) – (dp+1)(dp)	N-----ZC
3	DECW dp	BD	2	6	Decrement memory pair (dp+1)(dp) ← (dp+1) (dp) - 1	N-----Z-
4	INCW dp	9D	2	6	Increment memory pair (dp+1) (dp) ← (dp+1) (dp) + 1	N-----Z-
5	LDYA dp	7D	2	5	Load YA YA ← (dp +1) (dp)	N-----Z-
6	STYA dp	DD	2	5	Store YA (dp +1) (dp) ← YA	--------
7	SUBW dp	3D	2	5	16-Bits substact without carry YA ← (YA) - (dp +1) (dp)	NV--H-ZC

4. BIT MANIPULATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	AND1 M.bit	8B	3	4	Bit AND C-flag : C ← (C) ∧ (M .bit)	-------C
2	AND1B M.bit	8B	3	4	Bit AND C-flag and NOT : C ← (C) ∧ ~(M .bit)	-------C
3	BIT dp	0C	2	4	Bit test A with memory :	MM----Z-
4	BIT !abs	1C	3	5	Z ← (A) ∧ (M) , N ← (M_7) , V ← (M_6)	
5	CLR1 dp.bit	y1	2	4	Clear bit : (M.bit) ← "0"	--------
6	CLRA1 A.bit	2B	2	2	Clear A bit : (A.bit)← "0"	--------
7	CLRC	20	1	2	Clear C-flag : C ← "0"	-------0
8	CLRG	40	1	2	Clear G-flag : G ← "0"	--0-----
9	CLRV	80	1	2	Clear V-flag : V ← "0"	-0--0---
10	EOR1 M.bit	AB	3	5	Bit exclusive-OR C-flag : C ← (C) ⊕ (M .bit)	-------C
11	EOR1B M.bit	AB	3	5	Bit exclusive-OR C-flag and NOT : C ← (C) ⊕ ~(M .bit)	-------C
12	LDC M.bit	CB	3	4	Load C-flag : C ← (M .bit)	-------C
13	LDCB M.bit	CB	3	4	Load C-flag with NOT : C ← ~(M .bit)	-------C
14	NOT1 M.bit	4B	3	5	Bit complement : (M .bit) ← ~(M .bit)	--------
15	OR1 M.bit	6B	3	5	Bit OR C-flag : C ← (C) ∨ (M .bit)	-------C
16	OR1B M.bit	6B	3	5	Bit OR C-flag and NOT : C ← (C) ∨ ~(M .bit)	-------C
17	SET1 dp.bit	x1	2	4	Set bit : (M.bit) ← "1"	--------
18	SETA1 A.bit	0B	2	2	Set A bit : (A.bit) ← "1"	--------
19	SETC	A0	1	2	Set C-flag : C ← "1"	-------1
20	SETG	C0	1	2	Set G-flag : G ← "1"	--1-----
21	STC M.bit	EB	3	6	Store C-flag : (M .bit) ← C	--------
22	TCLR1 !abs	5C	3	6	Test and clear bits with A : A - (M) , (M) ← (M) ∧ ~(A)	N-----Z-
23	TSET1 !abs	3C	3	6	Test and set bits with A : A - (M) , (M) ← (M) ∨ (A)	N-----Z-

5. BRANCH / JUMP OPERATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	BBC A.bit,rel	y2	2	4/6	Branch if bit clear :	--------
2	BBC dp.bit,rel	y3	3	5/7	if (bit) = 0 , then pc ← (pc) + rel	
3	BBS A.bit,rel	x2	2	4/6	Branch if bit set :	--------
4	BBS dp.bit,rel	x3	3	5/7	if (bit) = 1 , then pc ← (pc) + rel	
5	BCC rel	50	2	2/4	Branch if carry bit clear if (C) = 0 , then pc ← (pc) + rel	--------
6	BCS rel	D0	2	2/4	Branch if carry bit set if (C) = 1 , then pc ← (pc) + rel	--------
7	BEQ rel	F0	2	2/4	Branch if equal if (Z) = 1 , then pc ← (pc) + rel	--------
8	BMI rel	90	2	2/4	Branch if minus if (N) = 1 , then pc ← (pc) + rel	--------
9	BNE rel	70	2	2/4	Branch if not equal if (Z) = 0 , then pc ← (pc) + rel	--------
10	BPL rel	10	2	2/4	Branch if minus if (N) = 0 , then pc ← (pc) + rel	--------
11	BRA rel	2F	2	4	Branch always pc ← (pc) + rel	--------
12	BVC rel	30	2	2/4	Branch if overflow bit clear if (V) = 0 , then pc ← (pc) + rel	--------
13	BVS rel	B0	2	2/4	Branch if overflow bit set if (V) = 1 , then pc ← (pc) + rel	--------
14	CALL !abs	3B	3	8	Subroutine call	
15	CALL [dp]	5F	2	8	M(sp)←(pc_H), sp←sp - 1, M(sp)← (pc_L), sp ←sp - 1, if !abs, pc← abs ; if [dp], pc_L← (dp), pc_H← (dp+1) .	--------
16	CBNE dp,rel	FD	3	5/7	Compare and branch if not equal :	--------
17	CBNE dp+X,rel	8D	3	6/8	if (A) ≠ (M) , then pc ← (pc) + rel.	
18	DBNE dp,rel	AC	3	5/7	Decrement and branch if not equal :	--------
19	DBNE Y,rel	7B	2	4/6	if (M) ≠ 0 , then pc ← (pc) + rel.	
20	JMP !abs	1B	3	3	Unconditional jump	
21	JMP [!abs]	1F	3	5	pc ← jump address	--------
22	JMP [dp]	3F	2	4		
23	PCALL upage	4F	2	6	U-page call M(sp) ←(pc_H), sp ←sp - 1, M(sp) ← (pc_L), sp ← sp - 1, pc_L ← (upage), pc_H ← "$0FF_H$" .	--------
24	TCALL n	nA	1	8	Table call : (sp) ←(pc_H), sp ← sp - 1, M(sp) ← (pc_L),sp ← sp - 1, pc_L ← (Table vector L), pc_H ← (Table vector H)	--------

6. CONTROL OPERATION & etc.

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	BRK	0F	1	8	Software interrupt : B ← "1", M(sp) ← (pc_H), sp ←sp-1, M(s) ← (pc_L), sp ← sp - 1, M(sp) ← (PSW), sp ← sp -1, pc_L ← ($0FFDE_H$) , pc_H ← ($0FFDF_H$) .	---1-0--
2	DI	60	1	3	Disable interrupts : I ← "0"	-----0--
3	EI	E0	1	3	Enable interrupts : I ← "1"	-----1--
4	NOP	FF	1	2	No operation	--------
5	POP A	0D	1	4	sp ← sp + 1, A ← M(sp)	--------
6	POP X	2D	1	4	sp ← sp + 1, X ← M(sp)	
7	POP Y	4D	1	4	sp ← sp + 1, Y ← M(sp)	
8	POP PSW	6D	1	4	sp ← sp + 1, PSW ← M(sp)	restored
9	PUSH A	0E	1	4	M(sp) ← A , sp ← sp - 1	--------
10	PUSH X	2E	1	4	M(sp) ← X , sp ← sp - 1	
11	PUSH Y	4E	1	4	M(sp) ← Y , sp ← sp - 1	
12	PUSH PSW	6E	1	4	M(sp) ← PSW , sp ← sp - 1	
13	RET	6F	1	5	Return from subroutine sp ← sp +1, pc_L ← M(sp), sp ← sp +1, pc_H ← M(sp)	--------
14	RETI	7F	1	6	Return from interrupt sp ← sp +1, PSW ← M(sp), sp ← sp + 1, pc_L ← M(sp), sp ← sp + 1, pc_H ← M(sp)	restored
15	STOP	EF	1	3	Stop mode (halt CPU, stop oscillator)	--------