GMS81C50 Series

CMOS SINGLE-CHIP 8-BIT MICROCONTROLLER
WITH UR (Universal Remocon) & KEYBOARD

1. OVERVIEW

1.1 Description

The GMS81C50 Series is an advanced CMOS 8-bit microcontroller with 16K/24K/32K bytes of ROM. The device is one of GMS800 family. The MagnaChip Semicon GMS81C50 Series is a powerful microcontroller which provides a highly flexible and cost effective solution to many UR & Keyboard applications. The GMS81C50 Series provides the following standard features: 16K/24K/32K bytes of ROM, 448 bytes of RAM, 8-bit timer/counter, on-chip oscillator and clock circuitry. In addition, the GMS81C50 Series supports power saving modes to reduce power consumption.

1.2 Features

Device Name	ROM Size	RAM Size	Package
GMS81C5016	16K Bytes	448 Bytes (included 256 bytes stack memory)	28 SOP 28 Skinny DIP 40 PDIP 44 PLCC 44 QFP
GMS81C5024	24K Bytes		
GMS81C5032	32K Bytes		

- **Instruction Cycle Time:**
  - **1us at 4MHz**
- **Programmable I/O pins**

	28 PIN	40 PIN	44 PIN
INPUT	3	3	3
OUTPUT	2	2	2
I/O	21	33	33

- **Operating Voltage**
  - **2.2 ~ 4.0 V @ 4MHz**
- **Timer**
  - **Timer / Counter ..... 16 Bit * 1 ch**
  - **..... 8 Bit * 2 ch**
  - **Basic Interval Timer ..... 8 Bit * 1 ch**
  - **Watch Dog Timer ..... 6-bit * 1ch**
- **8 Interrupt sources**
  - *** Nested Interrupt control is available.**
  - **External input: 2**
  - **Keyscan input**
  - **Basic Interval Timer**
  - **Watchdog timer**
  - **Timer : 3**
- **Power On Reset**
- **Power Saving Operation Modes**
  - **STOP**
  - **SLEEP**
- **Low Voltage Detection Circuit**
- **Watch Dog Timer Auto Start (During 1 second after Power on Reset)**

Package
- **28SOP**
- **40PDIP**
- **44PLCC, QFP**
- **Avalilable Pb free package**

- Pb free package:

The "P" Suffix will be added at the original part number. For example,GMS81C5032(Normal package), GMS81C5032 P(Pb free package)

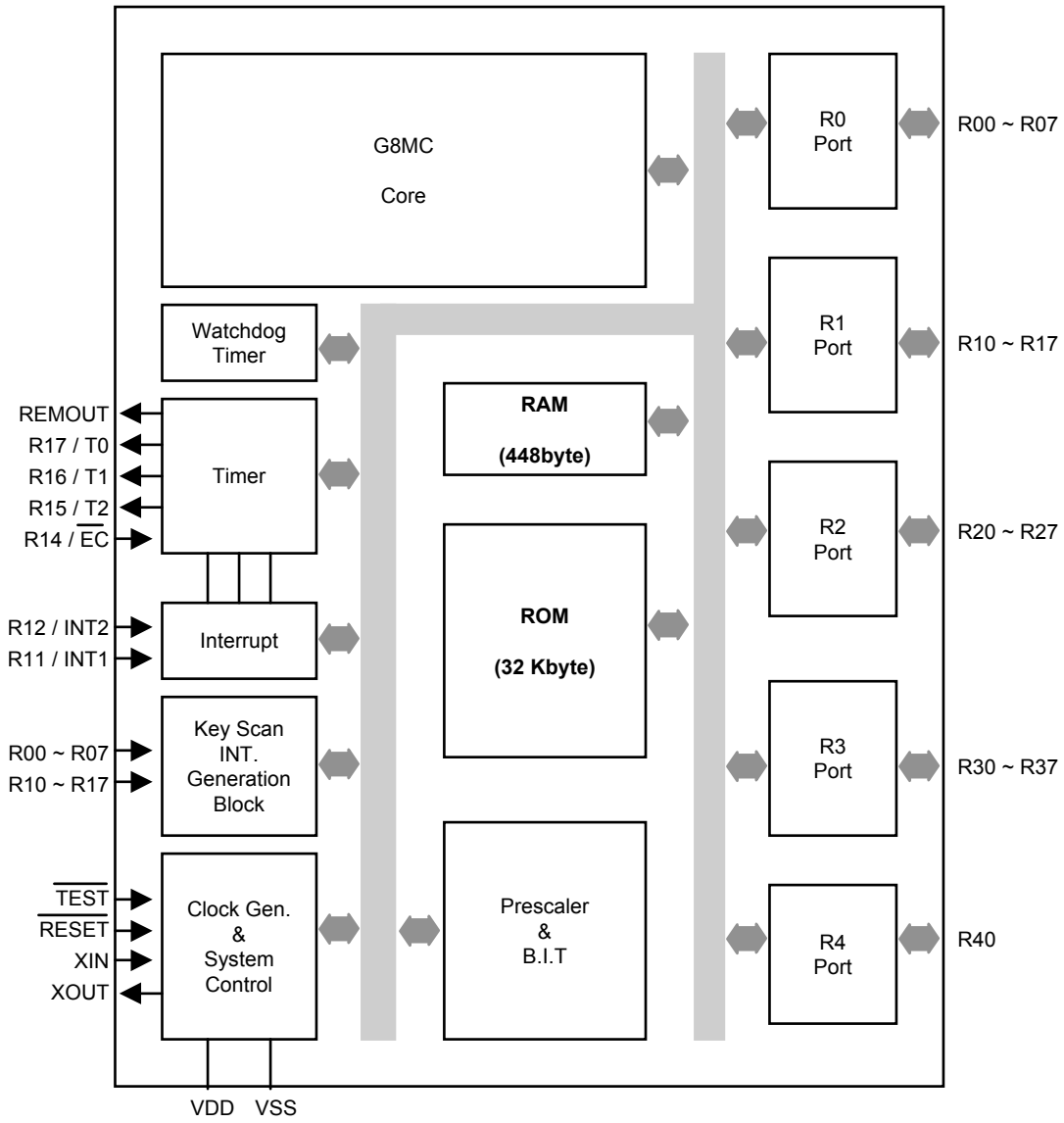
1.3 Development Tools

The GMS81C50 Series is supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr™.

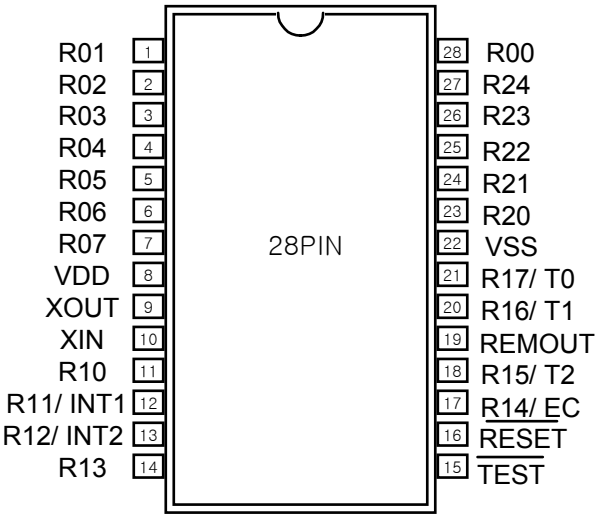
In Circuit Emulators	CHOICE-Dr. (with EVA81C)

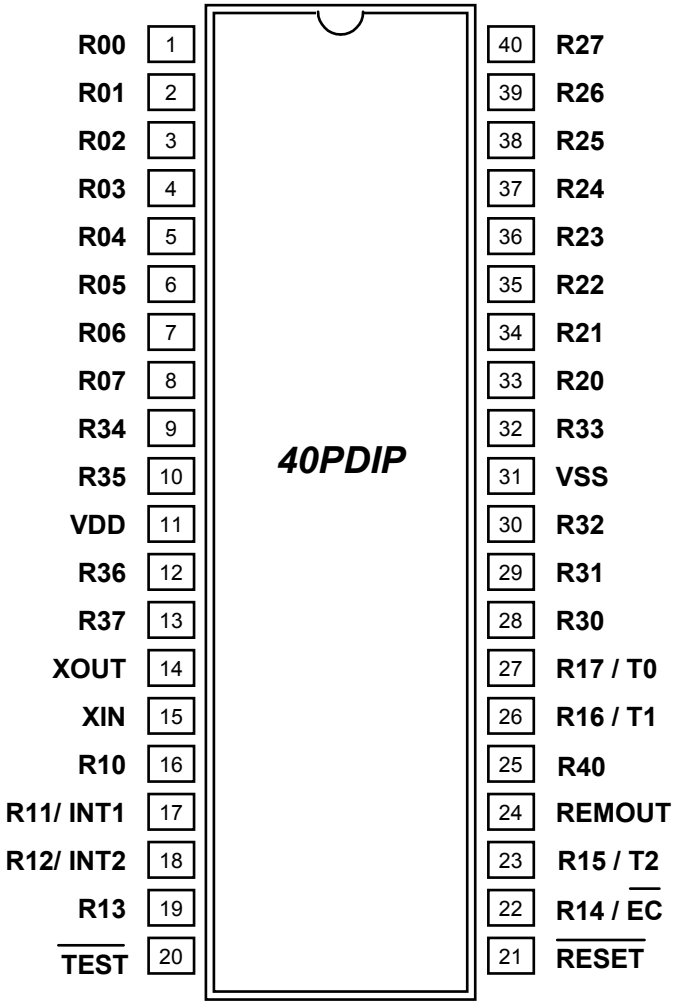
LCD Simulator	Under development
Assembler	MagnaChip Macro Assembler

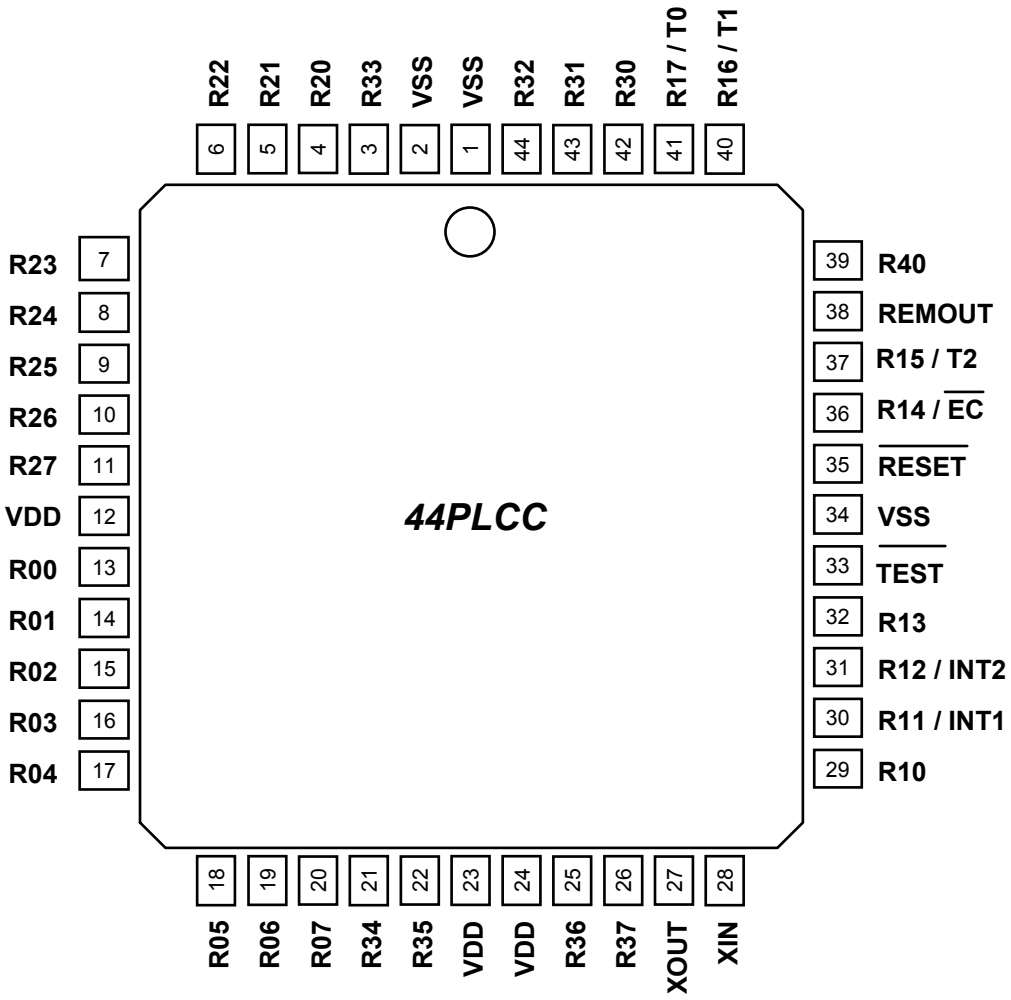
1.4 BLOCK DIAGRAM

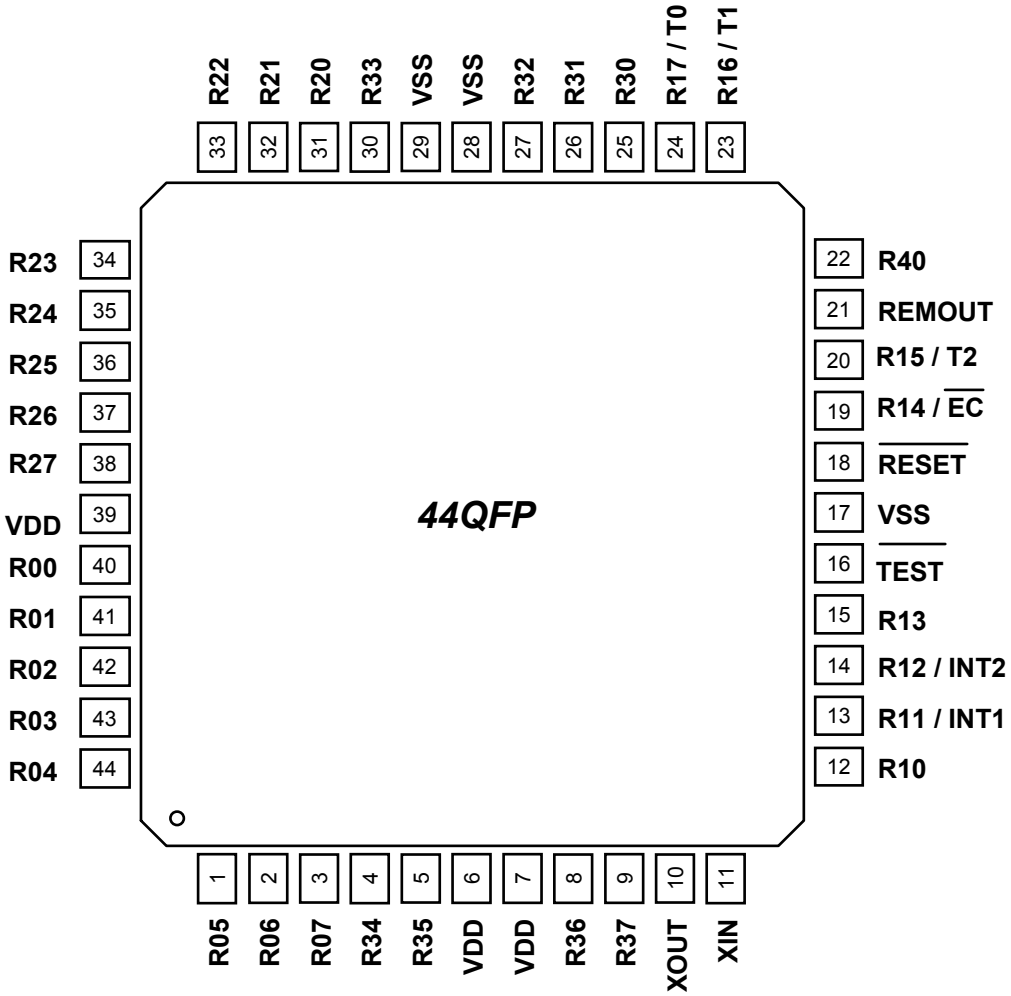


2. PIN ASSIGNMENT (Top View)



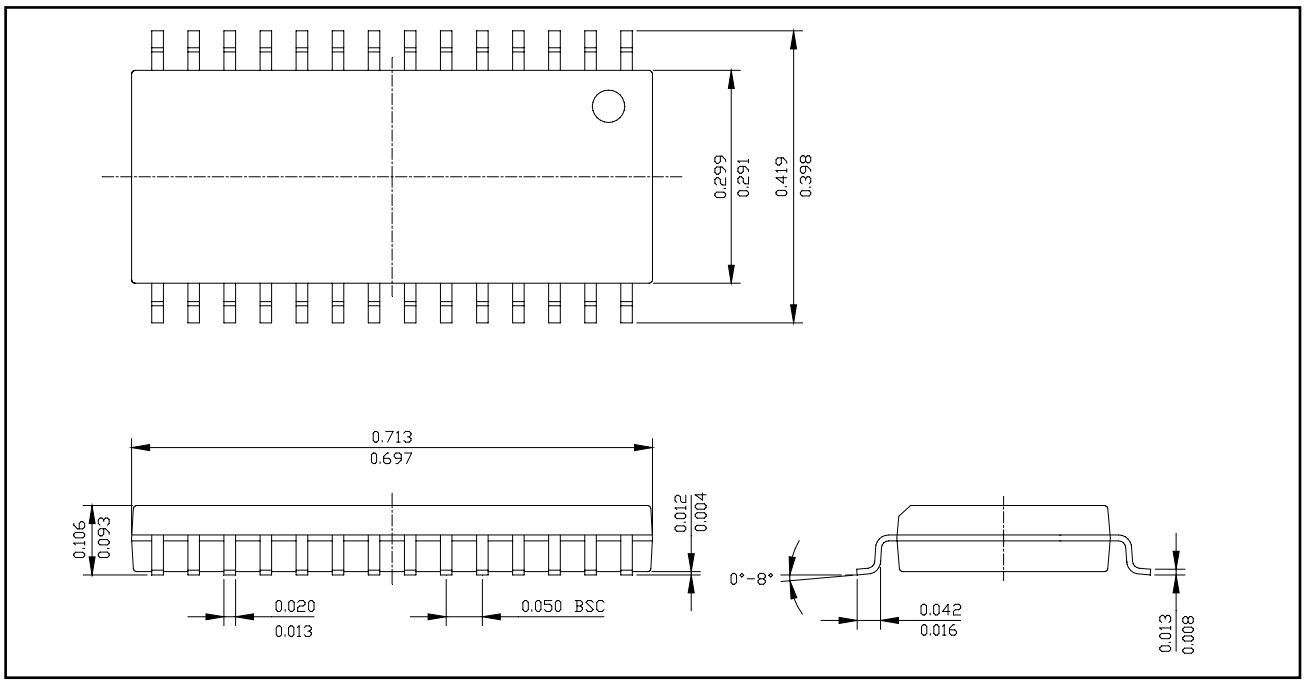



R22 6
R21 5
R20 4
R33 3
VSS 2
VSS 1
R32 44
R31 43
R30 42
R17 / T0 41
R16 / T1 40
R23 7
R24 8
R25 9
R26 10
R27 11
VDD 12
R00 13
R01 14
R02 15
R03 16
R04 17
44PLCC
39 R40
38 REMOUT
37 R15 / T2
36 R14 / EC
35 RESET
34 VSS
33 TEST
32 R13
31 R12 / INT2
30 R11 / INT1
29 R10
18 R05
19 R06
20 R07
21 R34
22 R35
23 VDD
24 VDD
25 R36
26 R37
27 XOUT
28 XIN

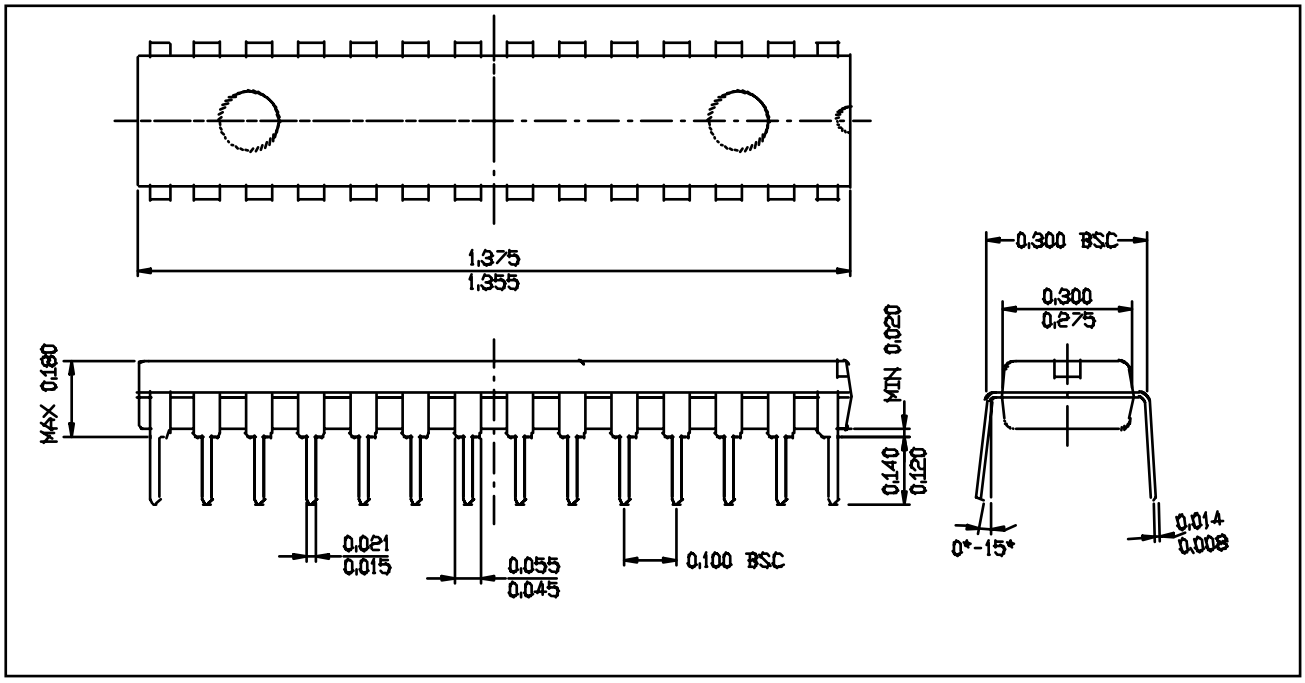

44QFP
1 R05
2 R06
3 R07
4 R34
5 R35
6 VDD
7 VDD
8 R36
9 R37
10 XOUT
11 XIN
12 R10
13 R11 / INT1
14 R12 / INT2
15 R13
16 TEST
17 VSS
18 RESET
19 R14 / EC
20 R15 / T2
21 REMOUT
22 R40
23 R16 / T1
24 R17 / T0
25 R30
26 R31
27 R32
28 VSS
29 VSS
30 R33
31 R20
32 R21
33 R22
34 R23
35 R24
36 R25
37 R26
38 R27
39 VDD
40 R00
41 R01
42 R02
43 R03
44 R04

3. PACKAGE DIMENSION

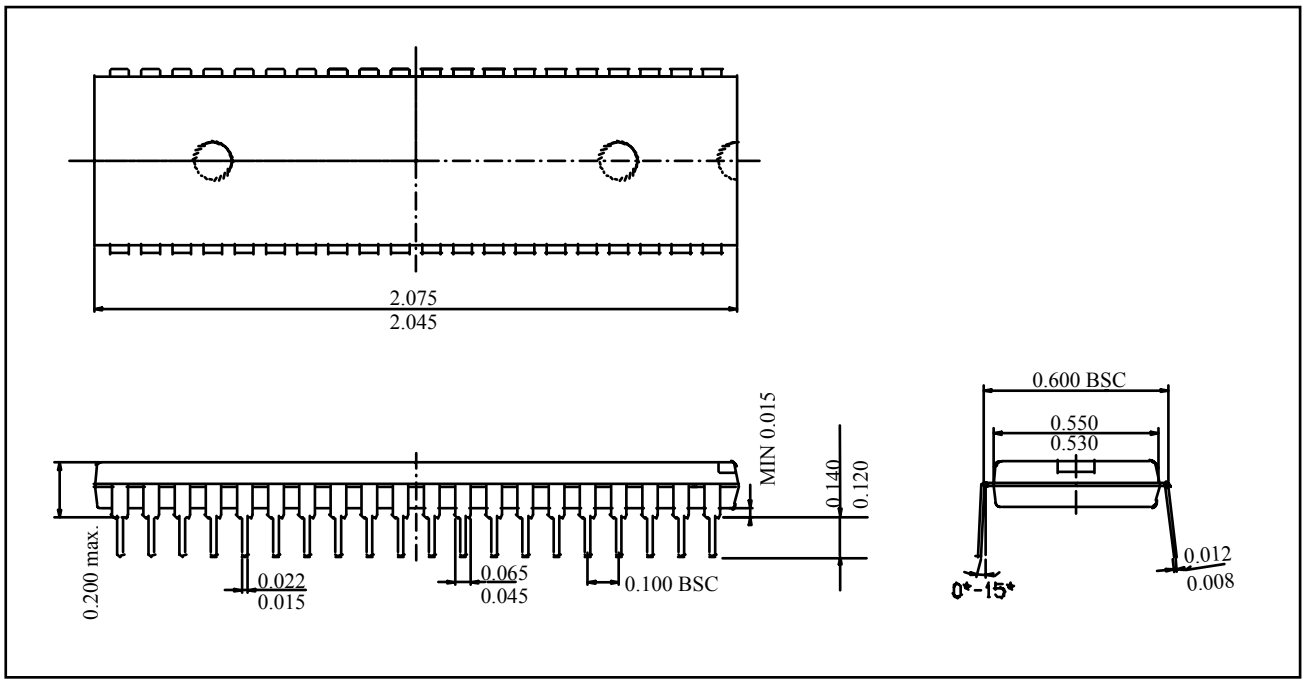
3.1 28 SOP Pin Dimension (Dimension in Inch)



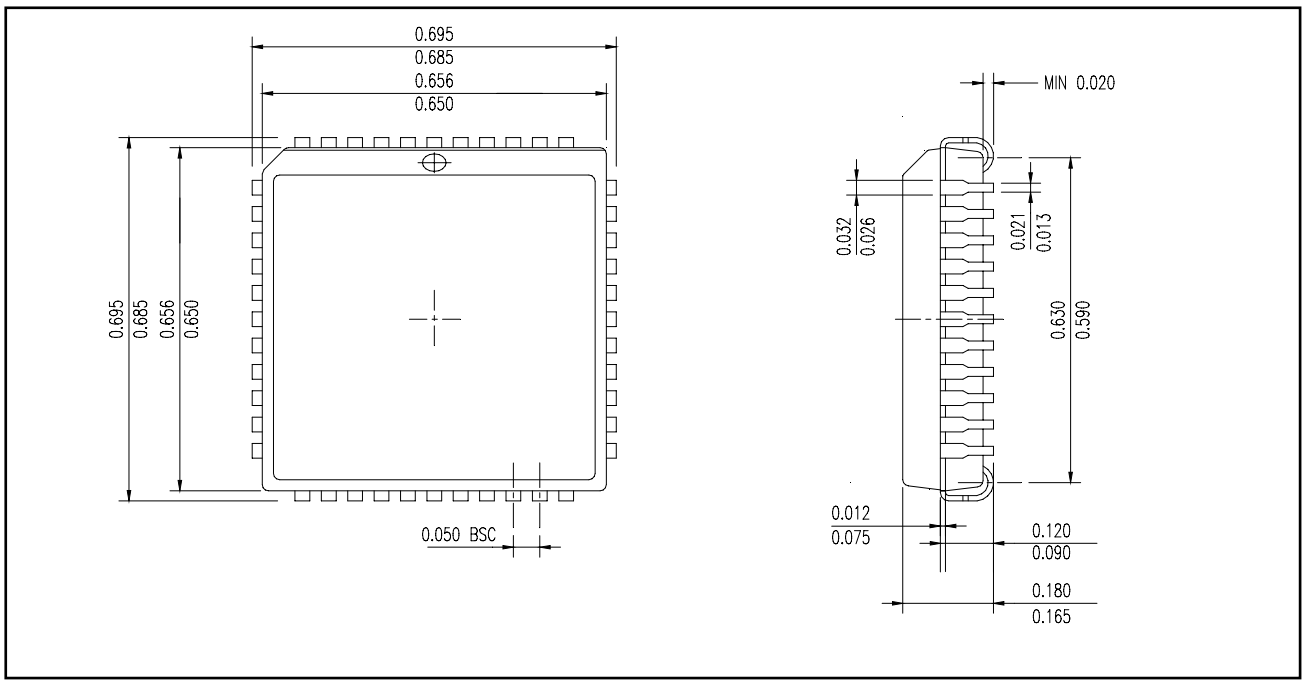
3.2 28 Skinny DIP Pin Dimension (Dimension in Inch)



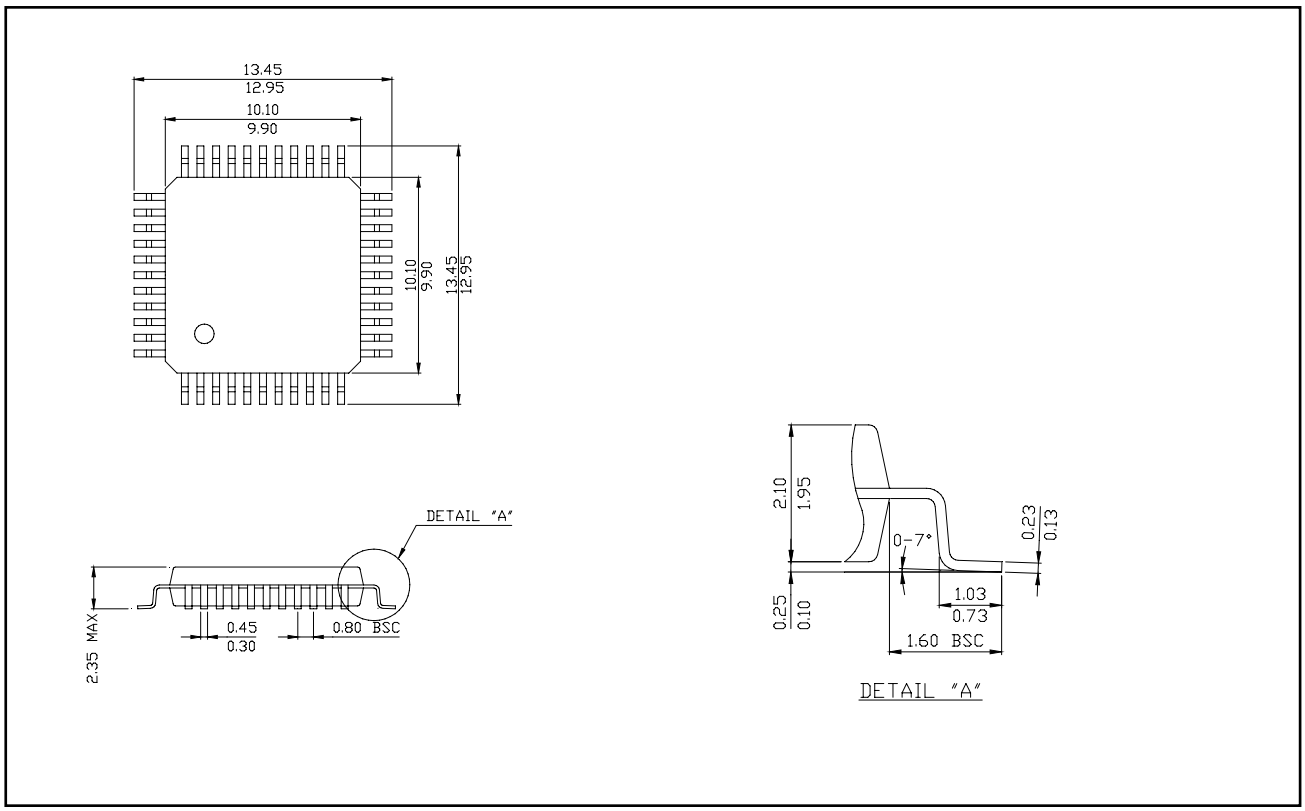
3.3 40 PDIP Pin Dimension (Dimension in Inch)



3.4 44 PLCC Pin Dimension (Dimension in mm)



3.5 44 QFP Pin Dimension (Dimension in mm)



4. 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 pin 1 or 0 written to the Port Direction Register can be used as output or input.

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	/EC (Event Counter input)
R15	T2 (Timer / Counter input 2)
R16	T1 (Timer / Counter input 1)
R17	T0 (Timer / Counter input 0)

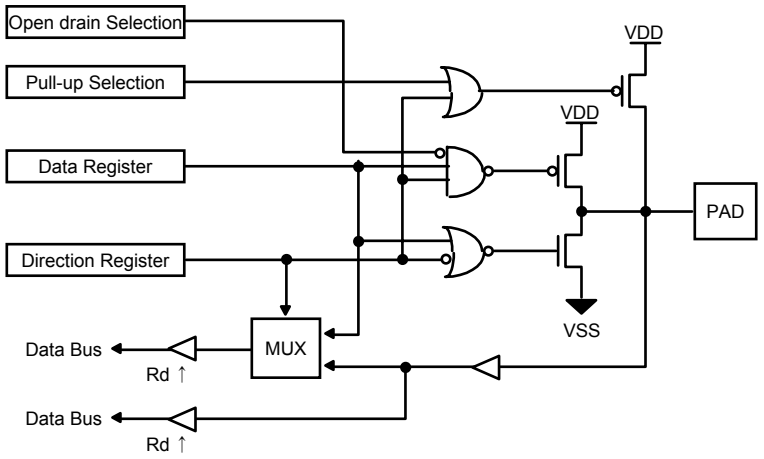
R20~R22, **R30~R37** : R2 & R3 is a 8-bit CMOS bidirectional I/O port. Each pin 1 or 0 written to its Port Direction Register can be used as output or input.

R40 : R40 is 1-bit CMOS bidirectional I/O port. This pin 1 or 0 written to the its Port Direction Register can be used as output or input.

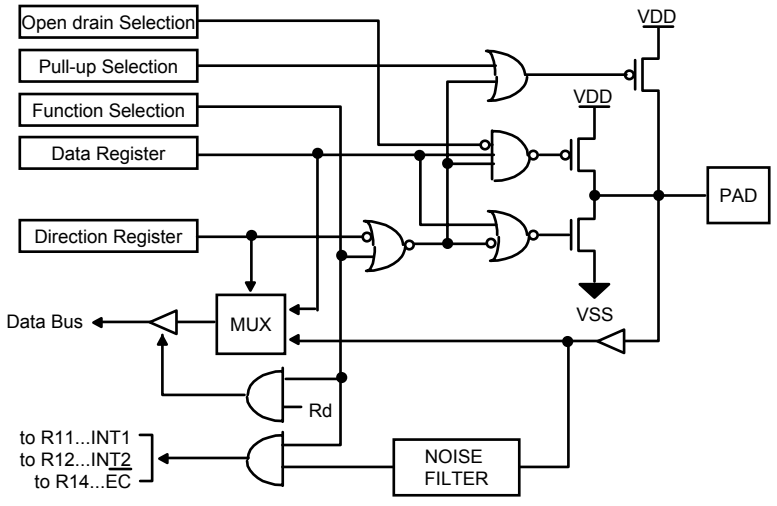
PIN NAME	INPUT/ OUTPUT	Pin Numbers				Function	@ RESET	@ STOP
		28Pin	40PDIP	44PLCC	44QFP			
R00	I/O	28	1	13	41	- Each bit of the port can be individually configured as an input or an output by user software - Push-pull output - CMOS input with pull-up resistor (can be selectable by user software) - Can be programmable as Key Scan Input or Open drain output - Pull-ups are automatically disabled at output mode	INPUT	State of before STOP
R01	I/O	1	2	14	42			
R02	I/O	2	3	15	43			
R03	I/O	3	4	16	44			
R04	I/O	4	5	17	1			
R05	I/O	5	6	18	2			
R06	I/O	6	7	19	3			
R07	I/O	7	8	20	4			
R10	I/O	11	16	29	12			
R11/INT1	I/O	12	17	30	13			
R12/INT2	I/O	13	18	31	14			
R13	I/O	14	19	32	15			
R14/EC	I/O	17	22	36	19			
R15/T2	I/O	18	23	37	20			
R16/T1	I/O	20	26	40	23			
R17/T0	I/O	21	27	41	24			
R20	I/O	23	33	4	31	- Each bit of the port can be individually configured as an input or an output by user software - CMOS input with pull-up resistor (can be selectable by user software) - Push-pull output - Can be programmable as Open drain output - Direct Driving of LED(N-TR) - Pull-ups are disabled at output mode		
R21	I/O	24	34	5	32			
R22	I/O	25	35	6	33			
R23	I/O	26	36	7	34			
R24	I/O	27	37	8	35			
R25	I/O	-	38	9	36			
R26	I/O	-	39	10	37			
R27	I/O	-	40	11	38			
R30	I/O	-	28	42	25			
R31	I/O	-	29	43	26			
R32	I/O	-	30	44	27			
R33	I/O	-	32	3	30			
R34	I/O	-	9	21	4			
R35	I/O	-	10	22	5			
R36	I/O	-	12	25	8			
R37	I/O	-	13	26	9			
R40	I/O	-	25	39	22			
XIN	I	10	15	28	11	- Oscillator Input		Low
XOUT	O	9	14	27	10	- Oscillator Output		High
REMOUT	O	19	24	38	21	- High Current Output	`L` output	`L` Output
$\overline{\text{RESET}}$	I	16	21	35	18	- Includes pull-up resistor	`L` level	state of before STOP
$\overline{\text{TEST}}$	I	15	20	33	16	- Includes pull-up resistor		
VDD	P	8	11	12,23,24	6,7,39	- Positive power supply		
VSS	P	22	31	1,2.34	17,28,29	- Ground		

5. PORT STRUCTURES

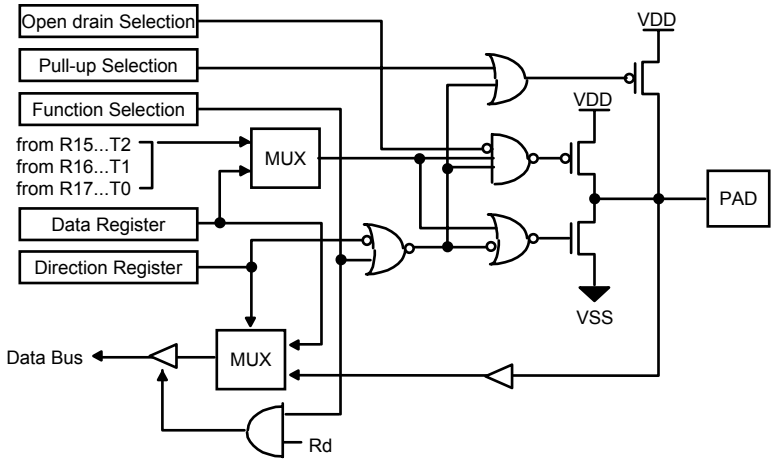
5.1 R0 Ports

Pin Name	Circuit Type	@ RESET
R00 ~ R07		Hi - Z or High-Input (with pullup)

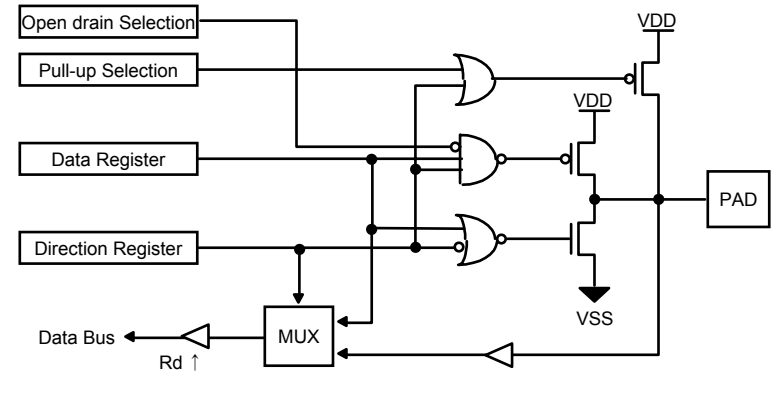
5.2 R1 Ports (R10, R11, R12, R13, R14)

Pin Name	Circuit Type	@ RESET
R10 R11 / INT1 R12 / INT2 R13 ___ R14 / EC		Hi - Z or High-Input (with pullup)

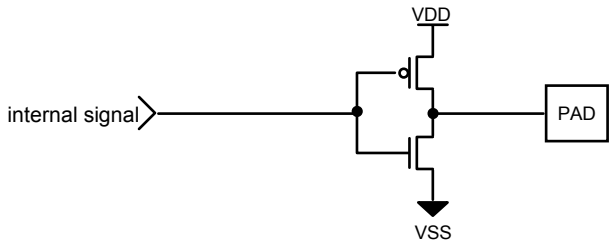
5.3 R1 Ports (R15, R16, R17)

Pin Name	Circuit Type	@ RESET
R15 / T2 R16 / T1 R17 / T0		Hi - Z or High-Input (with pullup)

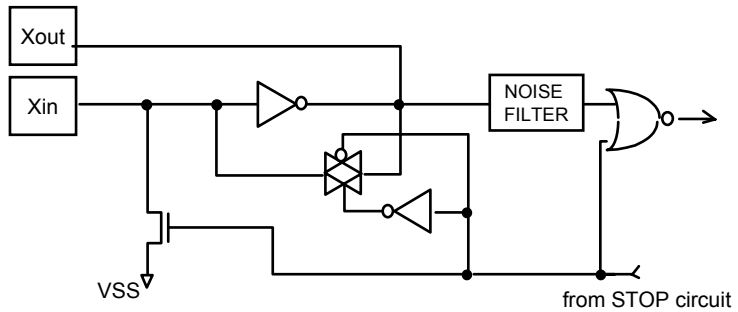
5.4 R2, R3, R4 Ports

Pin Name	Circuit Type	@ RESET
R20 ~ R27 R30 ~ R37 R40		Hi - Z or High-Input (with pullup)

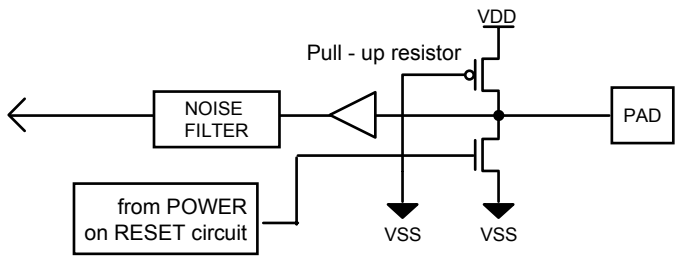
5.5 REMOUT Port

Pin Name	Circuit Type	@ RESET
REMOUT		Low level

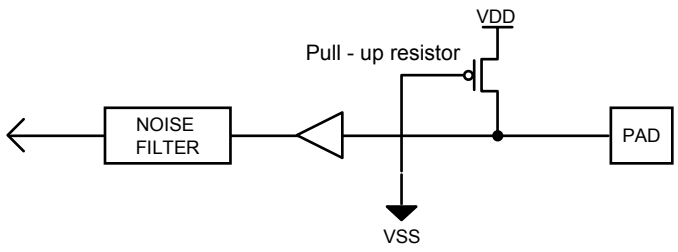
5.6 Xin, Xout Ports

Pin Name	Circuit Type	@ RESET
Xin Xout		oscillation

5.7 $\overline{\text{RESET}}$ Port

Pin Name	Circuit Type	@ RESET
$\overline{\text{RESET}}$		Low level

5.8 $\overline{\text{TEST}}$ Port

Pin Name	Circuit Type	@ RESET
$\overline{\text{TEST}}$		High level

6. ELECTRICAL CHARACTERISTICS

6.1 Absolute Maximum Ratings (Ta=25℃)

Parameter	Symbol	Rating	Unit
Supply Voltage	VDD	-0.3 ~ +7.0	V
Input Voltage	VI	-0.3 ~ VDD + 0.3	V
Output Voltage	VO	-0.3 ~ VDD + 0.3	V
Operating Temperature	Topr	0 ~ 70	℃
Storage Temperature	Tstg	-65 ~ 150	℃
Power Dissipation	PD	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 the functional operation of the device at any other condition 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*

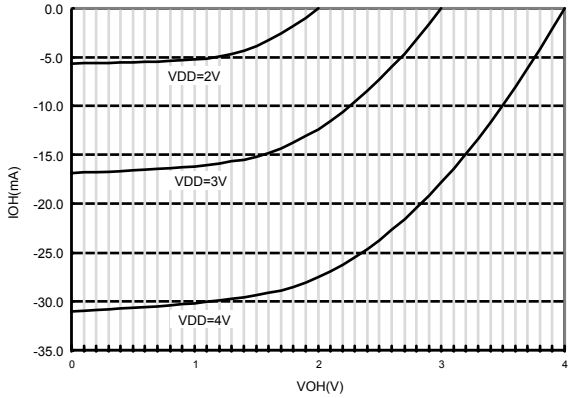
6.2 Recommended Operating Ranges

Parameter	Symbol	Condition	min.	typ.	max.	Unit
Supply Voltage	VDD	fXin = 4MHz	2.2		4.0	V
Oscillation Frequency	fXin		1.0		4.0	MHz
Operating Temperature	Topr		0		70	℃

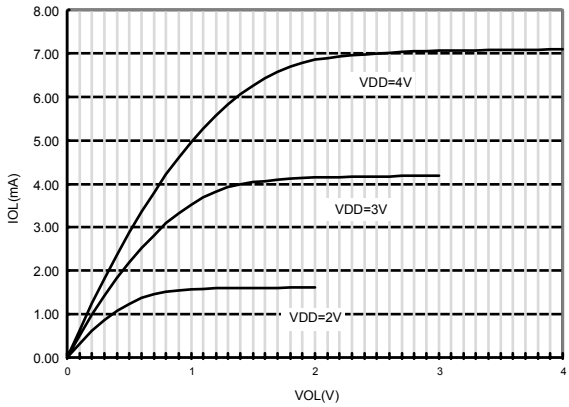
6.3 DC Characteristics (VDD=2.2~4.0, Vss=0, Ta=0~70℃)

Parameter	Symbol	Condition			Specification			Unit
					min	typ	max	
high level input voltage	V_{IH1}	R11, R12, R14, $\overline{RESET}$			0.8V_{DD}		V_{DD}	V
	V_{IH2}	R0, R1(Except R11,R12,R14) , R2 R3 , R4			0.7V_{DD}		V_{DD}	V
low level input voltage	V_{IL1}	R11, R12, R14, $\overline{RESET}$			0		0.2V_{DD}	V
	V_{IL2}	R0, R1(Except R11,R12,R14) , R2 R3 , R4			0		0.3V_{DD}	V
high level input leakage current	I_{IH}	R0 ~ R4 , $\overline{RESET}$	V_{IH}= V_{DD}				1	uA
low level input leakage current	I_{IL}	R0 ~ R4 ,$\overline{RESET}$ (without pull-up)	V_{IL}= 0V				-1	uA
high level output voltage	V_{OH1}	R0	I_{OH}= -0.5mA		V_{DD}-0.4			V
	V_{OH2}	R1(ExceptR17),R2 R3 , R4	I_{OH}= -1mA		V_{DD}-0.4			V
	V_{OH3}	OSC	I_{OH}= -200uA		V_{DD}-0.9			V
low level output voltage	V_{OL1}	R0	I_{OL}= 1mA				0.4	V
	V_{OL2}	R1, R2, R3, R4	I_{OL}= 5mA				0.8	V
	V_{OL3}	OSC	I_{OL}= 200uA				0.8	V
high level output leakage current	I_{OHL}	R0 ~ R4	V_{OH}= V_{DD}				1	uA
low level output leakage current	I_{OLL}	R0 ~ R4	V_{OL}= 0V				-1	uA
high level output current	I_{OH}	REMOUT , R17	V_{OH}= 2V		-30	-12	-5	mA
low level output current	I_{OL}	REMOUT	V_{OL}= 1V		0.5	-	3	mA
input pull-up current	I_{P1}	$\overline{RESET}$	V_{DD}= 3V		15	30	60	uA
	I_{P2}	R0 ~ R4	V_{DD}= 3V		15	30	60	uA
POWER SUPPLY CURRENT	I_{DD}	operating current	f_{XIN}= 4MHz	V_{DD}= 4V		4	10	mA
				V_{DD}= 2.2V		2.4	6	mA
	I_{SLEEP}	sleep mode current	f_{XIN}= 4MHz	V_{DD}= 4V	---	2	3	mA
				V_{DD}= 2.2V	---	1	2	mA
	I_{STOP}	stop mode current	oscillator stop	V_{DD}= 4V	---	3	10	uA
				V_{DD}= 2V	---	2	8	uA
RAM retention supply voltage	VRET				0.7			V

6.4 REMOUT Port Ioh Characteristics Graph



6.5 REMOUT Port Iol Characteristics Graph



6.6 AC Characteristics (VDD=2.2~4.0V, Vss=0V, Ta=0~70℃)

No.	Parameter	Symbol	Pin	Specification			Unit
				min.	typ.	max.	
1	External clock input cycle time	tcp	Xin	250	500	1000	ns
2	System clock cycle time	tsys		500	1000	2000	ns
3	External clock pulse width High	tcpH	Xin	40			ns
4	External clock pulse width Low	tcpL	Xin	40			ns
5	External clock rising time	trcp	Xin			40	ns
6	External clock falling time	tfcp	Xin			40	ns
7	interrupt pulse width High	tIH	INT1~ INT2	2			tsys
8	Interrupt pulse width Low	tIL	INT1~ INT2	2			tsys
9	Reset input pulse width low	tRSTL	$\overline{\text{RESET}}$	8			tsys
10	Event counter input pulse width high	tECH	$\overline{\text{EC}}$	2			tsys
11	Event counter input pulse width low	tECL	$\overline{\text{EC}}$	2			tsys
12	Event counter input pulse rising time	trEC	$\overline{\text{EC}}$			40	ns
13	Event counter input pulse falling time	tfEC	$\overline{\text{EC}}$			40	ns

(Continued)

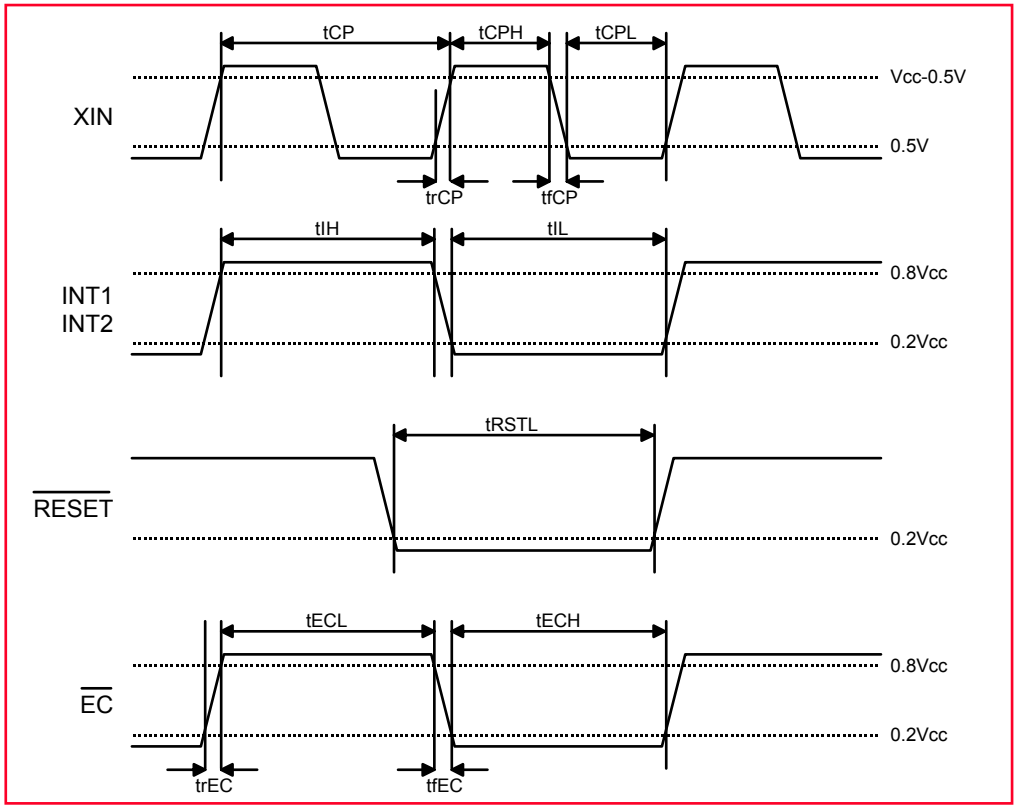


Figure 6-1 Clock, Interrupt, $\overline{\text{RESET}}$, $\overline{\text{EC}}$ Input Timing

7. MEMORY ORGANIZATION

The GMS81C50 Series has separate address spaces for Program memory, Data Memory and Display 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 up to 448 bytes including the stack area.

7.1 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.

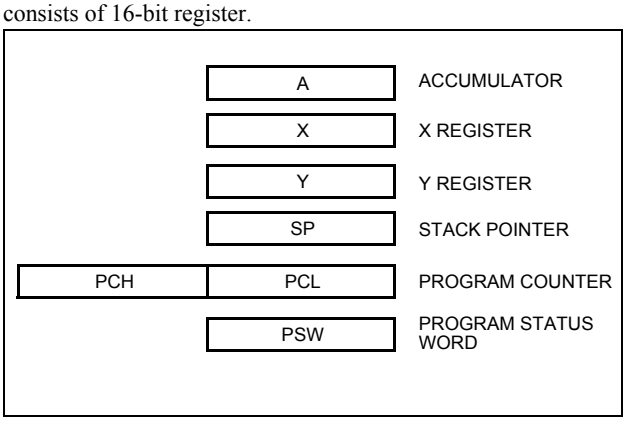


Figure 7-1 Configuration of Registers

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

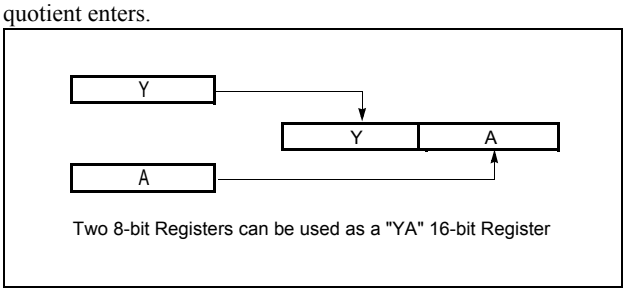


Figure 7-2 Configuration of YA 16-bit Register

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, it operates as register.

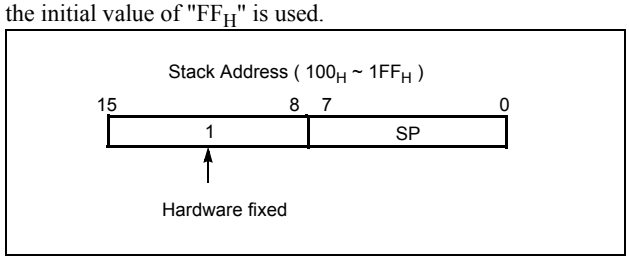
* Y Register : In the case of 16-bit operation instruction, it operates as the upper 8-bit of YA. (16-bit accumulator). In the case of multiplication instruction, it operates as a multiplicand register. After multiplication operation, the upper 8 bits of the result enter. In the case of division instruction, it operates as the upper 8-bit of dividend. After division operation, remains enter. 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-decreased when a subroutine call or a push instruction is executed, or when an interrupt is accepted. The SP is pre-increased 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 where 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
```

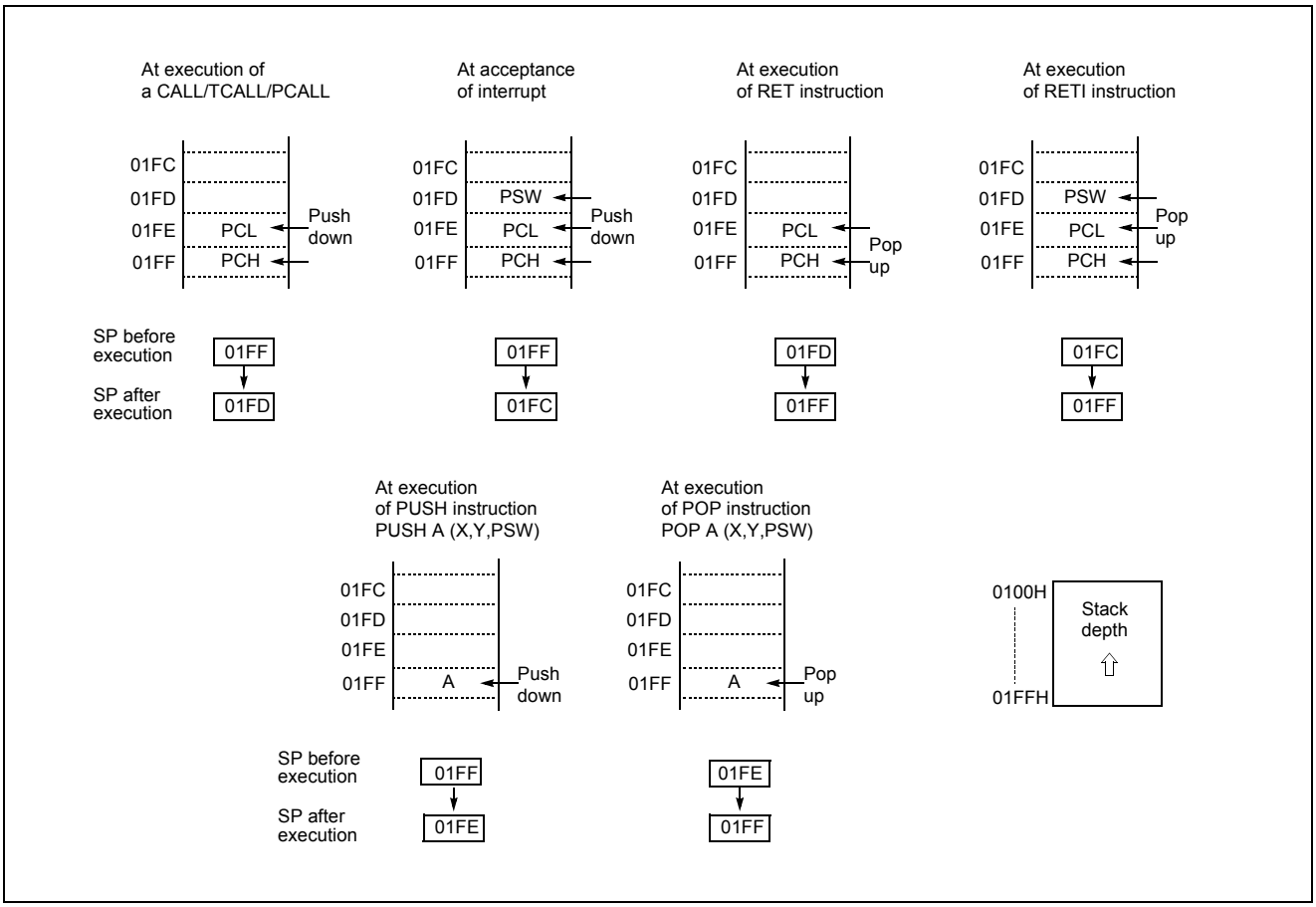


Figure 7-3 Stack Operation

Program Counter:

The Program Counter is 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 7-4. It contains the Negative flag, the Overflow flag, the Break flag, the Half Carry flag (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.

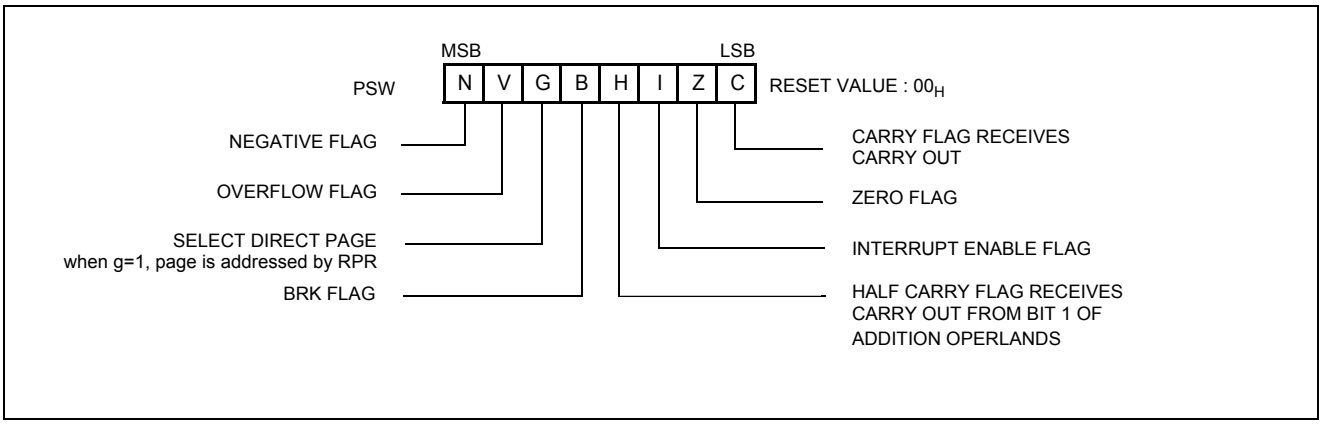


Figure 7-4 PSW (Program Status Word) Register

[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.

7.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 16K/24K/32K bytes program memory space only physically implemented. Accessing a location above $FFFF_H$ will cause a wrap-around to 0000_H.

Figure 7-5, 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 7-6.

As shown in Figure 7-5, each area is assigned to a fixed location in Program Memory. Program Memory area contains the user program.

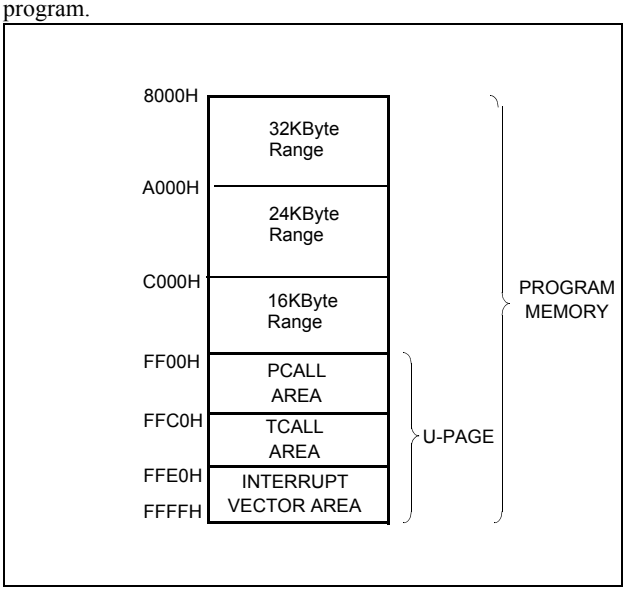


Figure 7-5 Program Memory Map

Page Call (PCALL) area contains subroutine program to reduce program byte length by using 2 bytes PCALL instead of 3 bytes CALL instruction. When 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 7-7.

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.

If any area from $0FF00_H$ to $0FFFF_H$ is not 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 7-6 Interrupt Vector Area

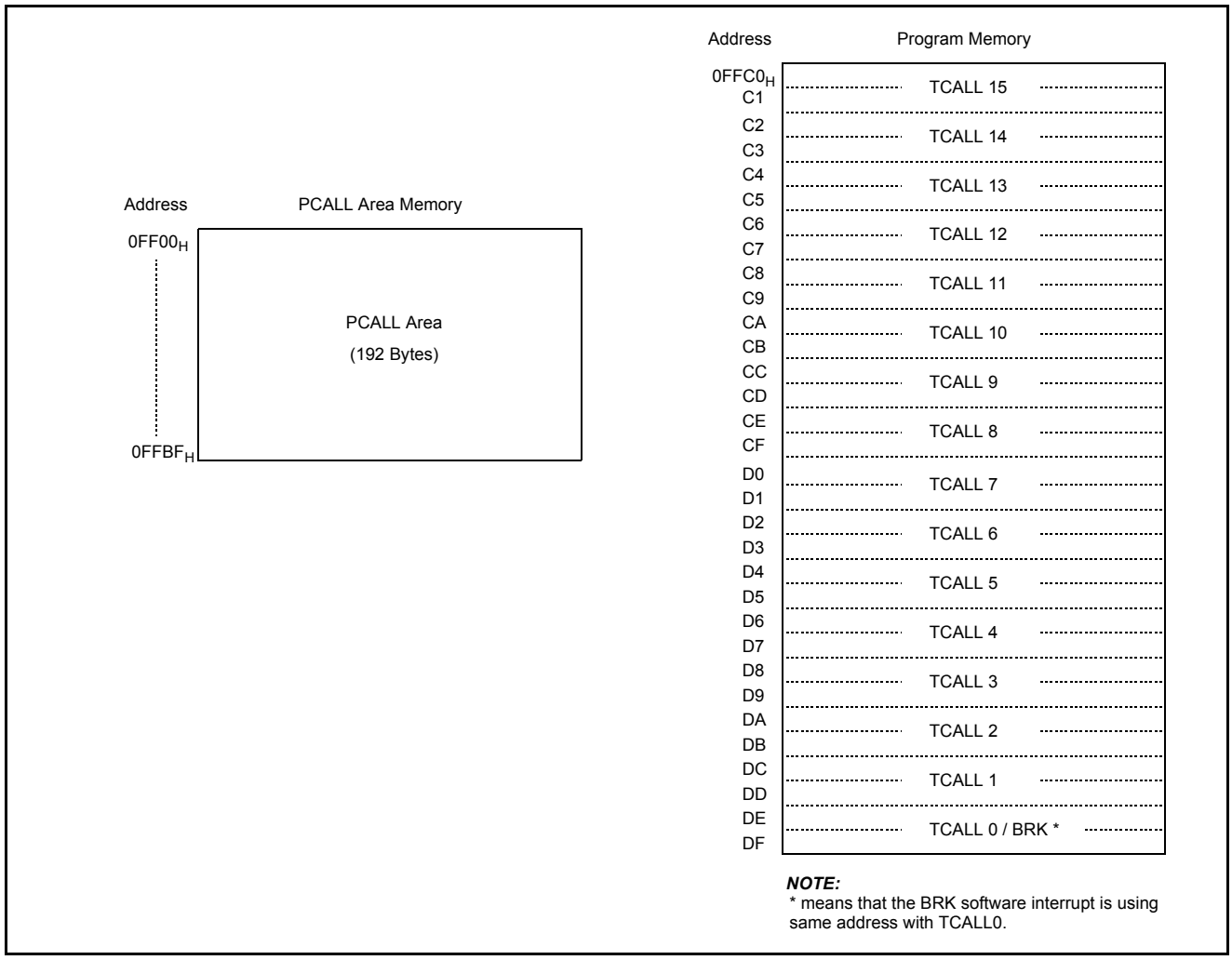
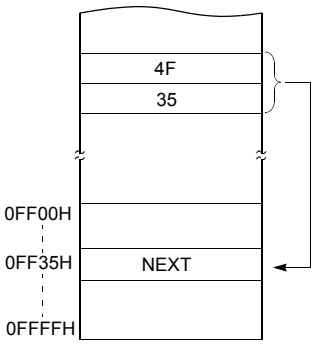


Figure 7-7 PCALL and TCALL Memory Area

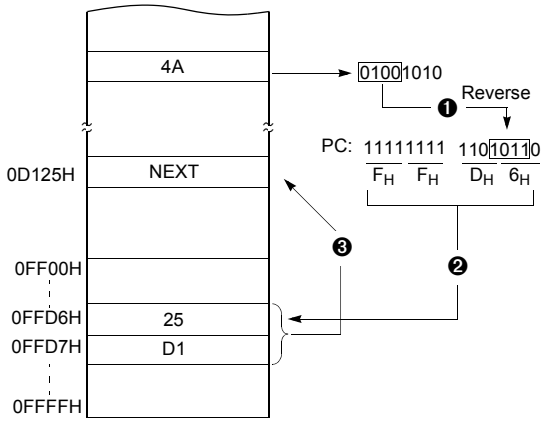
PCALL→ rel

```
4F35      PCALL  35H
```



TCALL→ n

```
4A        TCALL  4
```



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

```
          ORG0FFE0H

          DWNOT_USED
          DWNOT_USED
          DWNOT_USED
          DWBIT_INT; BIT
          DWWDT_INT; Watch Dog Timer
          DWNOT_USED
          DWNOT_USED
          DWTMR2_INT; Timer-2
          DWTMR1_INT; Timer-1
          DWTMR0_INT; Timer-0
          DWNOT_USED;
          DWINT2; Int.2
          DWINT1; Int.1
          DWKEY_INT; Key Scan
          DWNOT_USED;
          DWRESET; Reset


          ORG08000H

;*****************************************
;              MAIN     PROGRAM *
;*****************************************
;
RESET:    DI     ;Disable All Interrupts
          LDX#0
RAM_CLR: LDA#0  ;RAM Clear(!0000H->!00BFH)
          STA{X}+
          CMPX#0C0H
          BNERAM_CLR
;         ;
          LDX#03FH;Stack Pointer Initialize
          TXSP

          LDMR0, #0;Normal Port 0
          LDMR0DD,#1000_0010B;Normal Port Direction
          LDMPUR0,#1000_0010B;Pull Up Selection Set
          LDMPMR0,#0000_0001B;R0 port / int
          :
          :
          LDMPCOR,#1;Enable Peripheral clock
          :
          :
```

7.3 Data Memory

Figure 7-8 shows the internal Data Memory space available. Data Memory is divided into 3 groups, a user RAM, control registers, Stack.

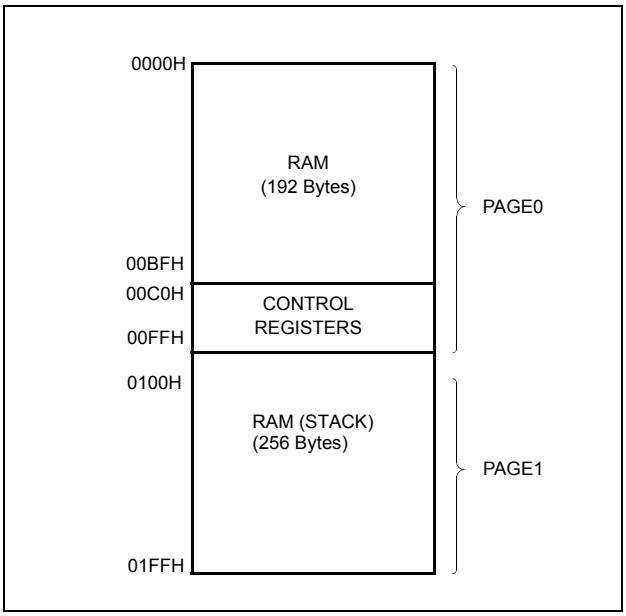


Figure 7-8 Data Memory Map

User Memory

The GMS81C50 Series 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 information of each register is 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 ÷8
```

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 7-3 on page 21.

Address	Function Register	Read Write	Symbol	RESET Value
00C0h	PORT R0 DATA REG.	R/W	R0	Undefined
00C1h	PORT R0 DATA DIRECTION REG.	W	R0DD	00000000b
00C2h	PORT R1 DATA REG.	R/W	R1	Undefined
00C3h	PORT R1 DATA DIRECTION REG.	W	R1DD	00000000b
00C4h	PORT R2 DATA REG.	R/W	R2	Undefined
00C5h	PORT R2 DATA DIRECTION REG.	W	R2DD	00000000b
00C6h	Reserved			
00C7h	CLOCK CONTROL REG.	W	CKCTLR	--110111b
	BASIC INTERVAL REG.	R	BTR	Undefined
00C8h	WATCH DOG TIMER REG.	W	WDTR	-0001111b

00C9h	PORT R1 MODE REG.	W	PMR1	00000000b
00CAh	INT. MODE REG.	R/W	IMOD	-0000000b
00CBh	EXT. INT. EDGE SELECTION	W	IEDS	00000000b
00CCh	INT. ENABLE REG. LOW	R/W	IENL	-00-----b
00CDh	INT. REQUEST FLAG REG. LOW	R/W	IRQL	-00-----b
00CEh	INT. ENABLE REG. HIGH	R/W	IENH	000-000-b
00CFh	INT. REQUEST FLAG REG. HIGH	R/W	IRQH	000-000-b
00D0h	TIMER0 (16bit) MODE REG.	R/W	TM0	00000000b
00D1h	TIMER1 (8bit) MODE REG.	R/W	TM1	00000000b
00D2h	TIMER2 (8bit) MODE REG.	R/W	TM2	00000000b
00D3h	TIMER0 HIGH-MSB DATA REG.	W	T0HMD	Undefined
00D4h	TIMER0 HIGH-LSB DATA REG.	W	T0HLD	Undefined
00D5h	TIMER0 LOW-MSB DATA REG.	W	T0LMD	Undefined
	TIMER0 HIGH-MSB COUNT REG.	R		Undefined
00D6h	TIMER0 LOW-LSB DATA REG.	W	T0LLD	Undefined
	TIMER0 LOW-LSB COUNT REG.	W		Undefined
00D7h	TIMER1 HIGH DATA REG.	W	T1HD	Undefined
00D8h	TIMER1 LOW DATA REG.	W	T1LD	Undefined
	TIMER1 LOW COUNT REG.	R		Undefined
00D9h	TIMER2 DATA REG.	W	T2DR	Undefined
	TIMER2 COUNT REG.	R		Undefined
00DAh	TIMER0 / TIMER1 MODE REG.	R/W	TM01	00000000b
00DBh	Reserved			
00DCh	STANDBY MODE RELEASE REG0	W	SMPR0	00000000b
00DDh	STANDBY MODE RELEASE REG0	W	SMPR1	00000000b
00DEh	PORT R1 OPEN DRAIN ASSIGN REG.	W	R1ODC	00000000b
00DFh	PORT R2 OPEN DRAIN ASSIGN REG.	W	R2ODC	00000000b
00E0h	PORT R3 OPEN DRAIN ASSIGN REG.	W	R3ODC	00000000b
00E1h	PORT R4 OPEN DRAIN ASSIGN REG.	W	R4ODC	-------0b
00E2h	Reserved			
00E3h	Reserved			
00E4h	PORT R0 OPEN DRAIN ASSIGN REG.	W	R0ODC	00000000b
00E5h	PORT R3 DATA REG.	R/W	R3	Undefined
00E6h	PORT R3 DATA DIRECTION REG.	W	R3DD	00000000b
00E7h	PORT R4 DATA REG.	R/W	R4	-------Xb
00E8h	PORT R4 DATA DIRECTION REG.	W	R4DD	-------0b
00E9h	Reserved			

00EAh	Reserved			
00EBh	Reserved			
00ECh	Reserved			
00EDh	Reserved			
00EEh	Reserved			
00EFh	LOW VOLTAGE INDICATION REG.	R	LVIR	------00b
00F0h	SLEEP MODE REG.	W	SLPM	-------0b
00F1h	Reserved			
00F2	Reserved			
00F3h	Reserved			
00F4h	Reserved			
00F5h	Reserved			
00F6h	STANDBY RELEASE LEVEL CONT. REG. 0	W	SRLC0	00000000b
00F7h	STANDBY RELEASE LEVEL CONT. REG. 1	W	SRLC1	00000000b
00F8h	PORT R0 PULL-UP REG. CONT. REG.	W	R0PC	00000000b
00F9h	PORT R1 PULL-UP REG. CONT. REG.	W	R1PC	00000000b
00FAh	PORT R2 PULL-UP REG. CONT. REG.	W	R2PC	00000000b
00FBh	PORT R3 PULL-UP REG. CONT. REG.	W	R3PC	00000000b
00FCh	PORT R4 PULL-UP REG. CONT. REG.	W	R4PC	-------0b
00FDh	Reserved			
00FEh	Reserved			
00FFh	Reserved			

7.4 Addressing Mode

The GMS81C50 Series uses six addressing modes;

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

(1) Register Addressing

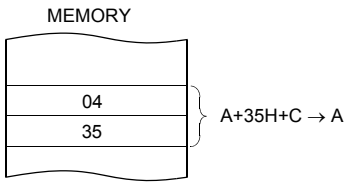
Register addressing accesses the A, X, Y, C and PSW.

(2) Immediate Addressing → #imm

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

Example:

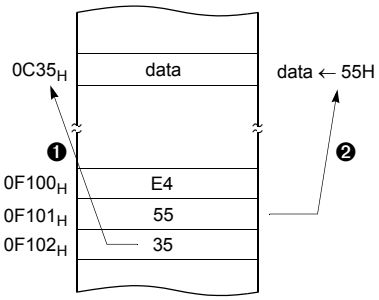
```
0435      ADC     #35H
```



When G-flag is 1, then RAM address is defined by 16-bit address which is composed of 8-bit RAM paging register (RPR) and 8-bit immediate data.

Example: G=1, RPR=0CH

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

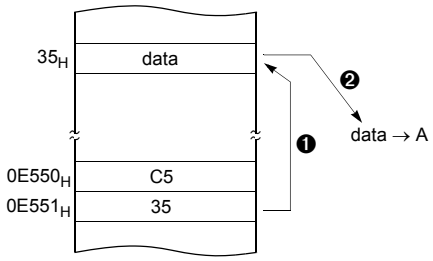


(3) Direct Page Addressing → dp

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

Example; G=0

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



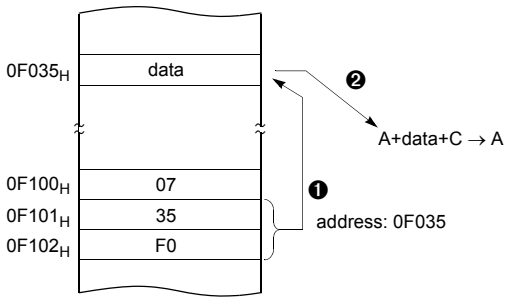
(4) Absolute Addressing → !abs

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 whole memory area.

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

Example;

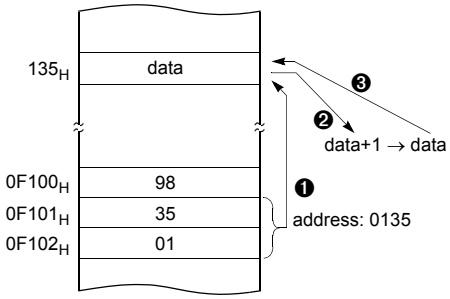
```
0735F0    ADC     !0F035H       ;A ←ROM[0F035H]
```



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

Example; Addressing accesses the address 0135_H regardless of G-flag and RPR.

```
983501   INC    !0135H        ;A ←ROM[135H]
```



(5) Indexed Addressing

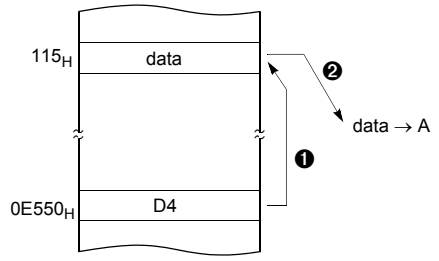
X indexed direct page (no offset) → {X}

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

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

Example; $X=15_H$, G=1, $RPR=01_H$

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



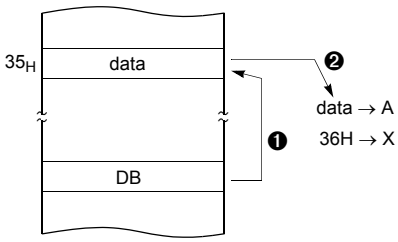
X indexed direct page, auto increment→ {X}+

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; G=0, $X=35_H$

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



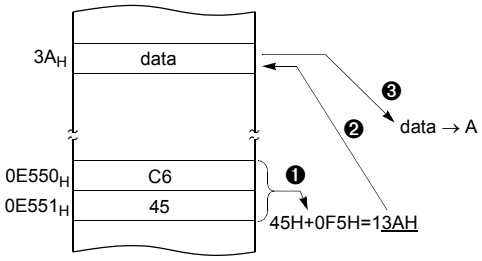
X indexed direct page (8 bit offset) → dp+X

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; G=0, $X=0F5_H$

```
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.

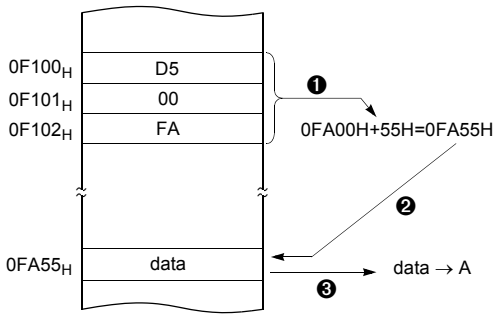
This is the same with above (2). Use Y register instead of X.

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

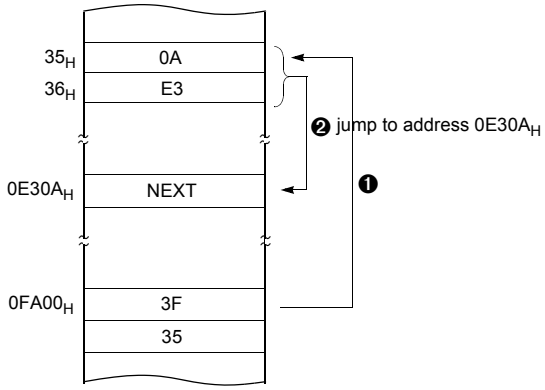
Direct page indirect → [dp]

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; G=0

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



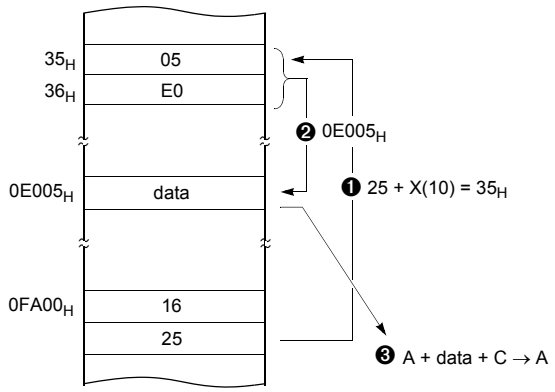
X indexed indirect → [dp+X]

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; G=0, $X=10_H$

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



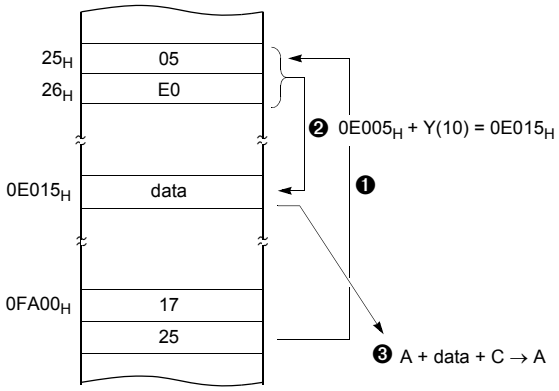
Y indexed indirect → [dp]+Y

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; G=0, Y=10_H

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



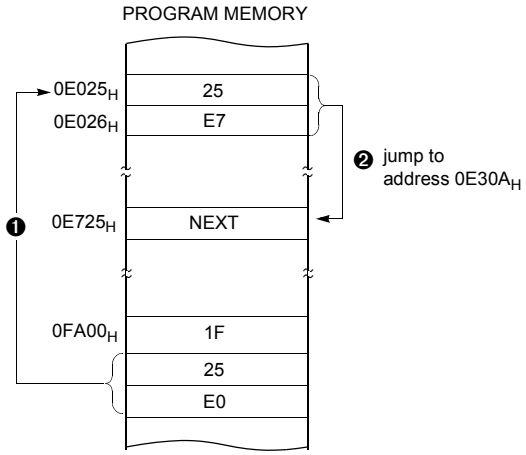
Absolute indirect → [!abs]

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

JMP

Example; G=0

```
1F25E0   JMP    [!0C025H]
```



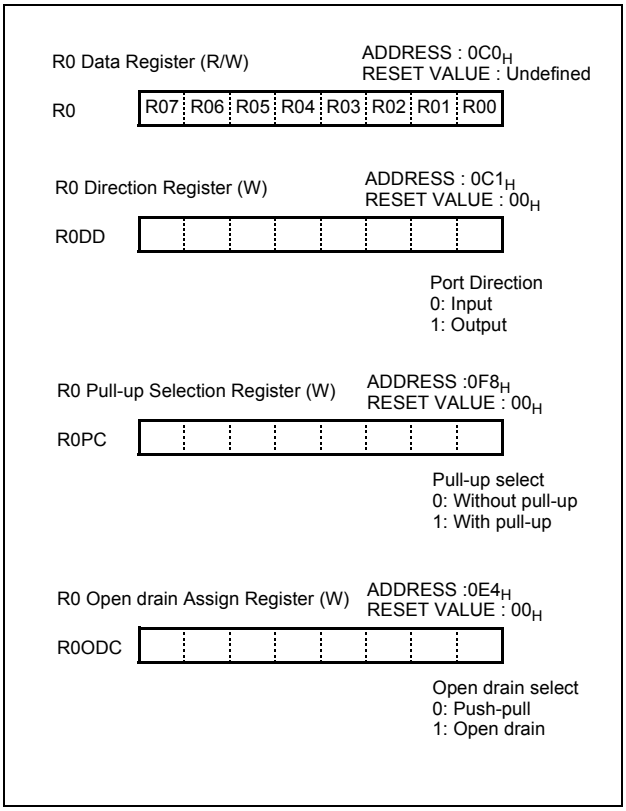
8. I/O PORTS

The GMS81C50 Series has 33 I/O ports which are PORT0(8 I/O), PORT1 (8 I/O), PORT2 (8 I/O), PORT3 (8 I/O), PORT4 (1 I/O). Pull-up resistor of each port is selectable by program. Each port contains data direction register which controls I/O and data register which stores port data.

8.1 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 are independently connected or disconnected by R0PC. The control registers for R0 are shown below.



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

R0 I/O Data Direction Register (R0DD) is 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 ``00 h`` in reset state, the whole port R0 becomes input state.

(2) R0 Data Register (R0)

R0 data register (R0) is 8-bit register to store data of port R0. When this is set as the output state by R0DD and data is written in R0, data is outputted into R0 pin. When this is 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 8bit register, and can assign R0 port as open drain output port for 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 ``00 h`` in reset state.

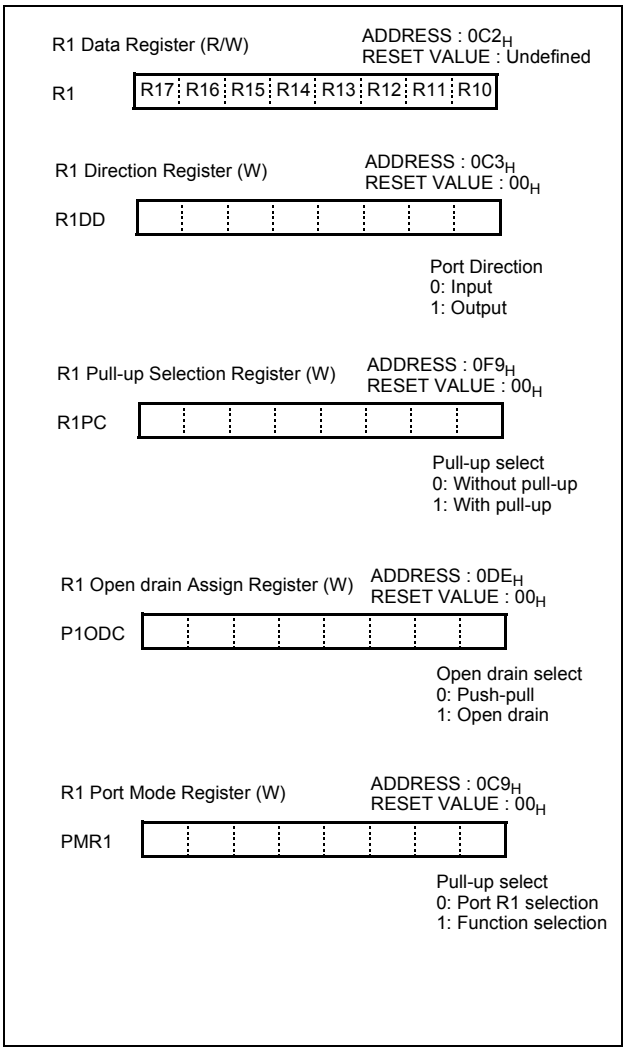
(4) R0 Pull-up Resistor Control Register (R0PC)

R0 pull-up resistor control register (R0PC) is 8-bit register and can control pull-up on or off for 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 ``00 h`` in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

8.2 R1 Ports

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

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



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

R1 I/O Data Direction Register (R1DD) is 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 ``00 h`` in reset state, the whole port R1 becomes input state.

(2) R1 Data Register (R1)

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

(3) R1 Mode Register (PMR1)

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

PMR1 is write-only register and initialized as ``00 h`` 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``.

Pin 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	/EC (I)	Timer0 Event
INT2S	0	R12 (I/O)	
	1	INT2 (I)	Timer0 Input Capture
INT1S	0	R11 (I/O)	
	1	INT1 (I)	

Table 8-1 Selection mode of PMR1

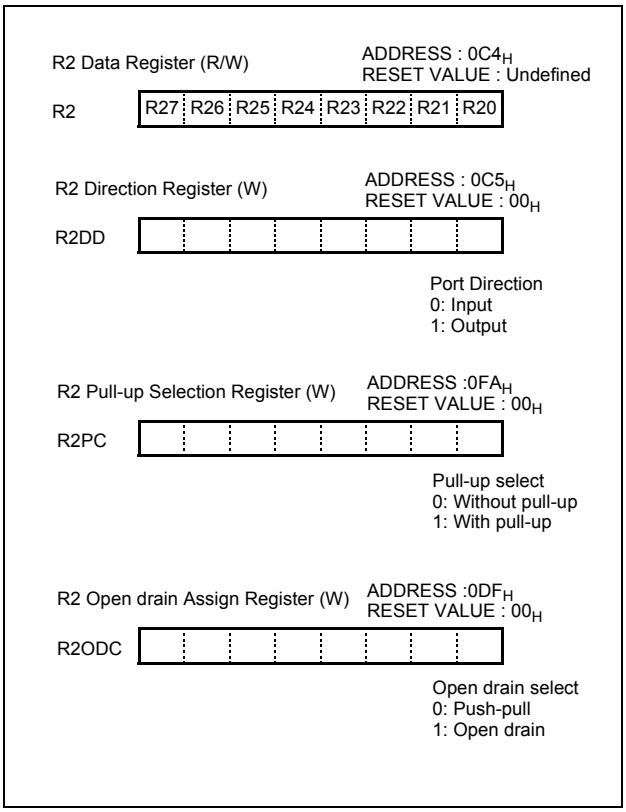
(4) R1 Pull-up Resistor Control Register (R1PC)

R1 pull-up resistor control register (R1PC) is 8-bit register and can control pull-up on or off at 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 ``00 h`` in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

8.3 R2 Port

R2 is an 8-bit CMOS bidirectional I/O port (address $0C4_H$). Each I/O pin can be independently used as an input or an output through the R2DD register (address $0C5_H$).

R2 has internal pull-ups that are 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 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 ``00 h`` in reset state, the whole port R2 becomes input state.

(2) R2 Data Register (R2)

R2 data register (R2) is 8-bit register to store data of port R2. When this is set as the output state by R2DD and data is written in R2, data is outputted into R2 pin. When this is 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 8-bit register, and can assign R2 port as open drain output port for 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 ``00 h`` in reset state.

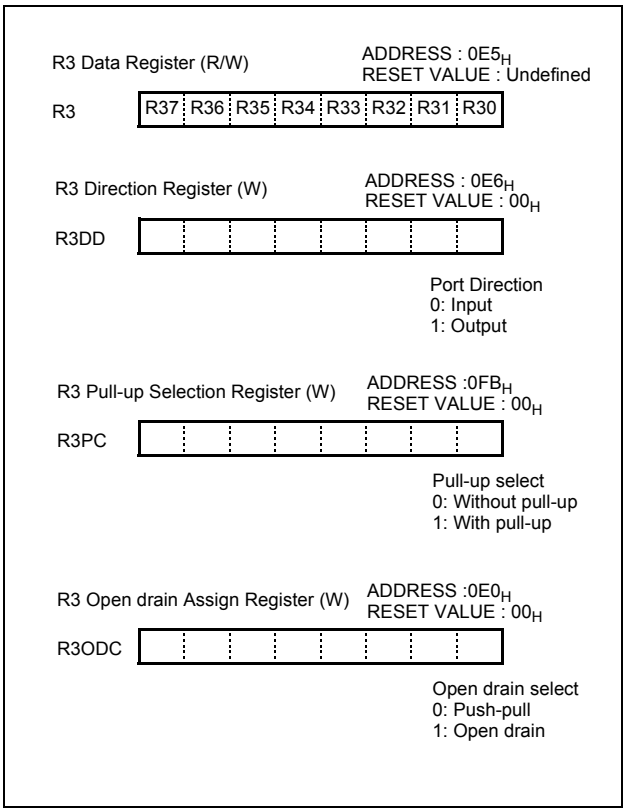
(4) R2 Pull-up Resistor Control Register (R2PC)

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

R3 Port

R3 is an 8-bit CMOS bi-directional I/O port (address $0E5_H$). Each I/O pin can be independently used as an input or an output through the R3DD register (address $0E6_H$).

R3 has internal pull-ups that are independently connected or disconnected by R3PC (address $0FB_H$). The control registers for R3 are shown as below.



(1) R3 I/O Data Direction Register (R3DD)

R3 I/O Data Direction Register (R3DD) is 8-bit register and can assign input state or output state to each bit. If R3DD is ``1``, port R3 is in the output state, and if ``0``, it is in the input state. R3DD is write-only register. Since R3DD is initialized as ``00 h`` in reset state, the whole port R3 becomes input state.

(2) R3 Data Register (R3)

R3 data register (R3) is 8-bit register to store data of port R3. When this is set as the output state by R3DD and data is written in R3, data is outputted into R3 pin. When this is set as the input state, input state of pin is read. The initial value of R3 is unknown in reset state.

(3) R3 Open drain Assign Register (R3ODC)

R3 Open Drain Assign Register (R3ODC) is 8-bit register, and can assign R3 port as open drain output port for each bit if corresponding port is selected as output. If R3ODC is selected as ``1``, port R3 is open drain output, and if selected as ``0``, it is push-pull output. R3ODC is write-only register and initialized as ``00 h`` in reset state.

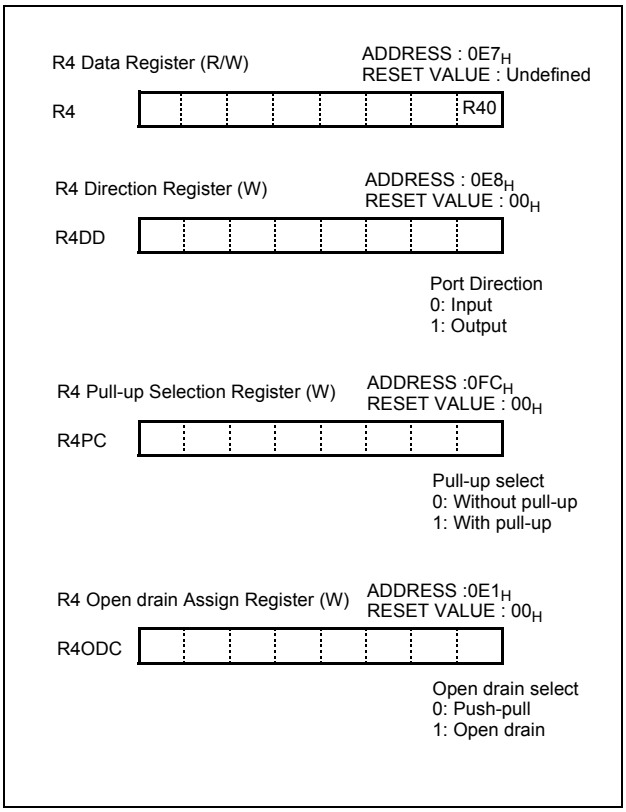
(4) R3 Pull-up Resistor Control Register (R3PC)

R3 pull-up resistor control register (R3PC) is 8-bit register and can control pull-up on or off for each bit if corresponding port is selected as input. If R3PC is selected as ``1``, pull-up is disabled and if selected as ``0``, it is enabled. R3PC is write-only register and initialized as ``00 h`` in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

R4 Port

R4 is an 1-bit CMOS bi-directional I/O port (address $0E7_H$). Each I/O pin can independently used as an input or an output through the R4DD register (address $0E8_H$).

R3 has internal pull-ups that is independently connected or disconnected by R4PC (address $0FC_H$). The control registers for R4 are shown as below.



(1) R4 I/O Data Direction Register (R4DD)

R4 I/O Data Direction Register (R4DD) is 1-bit register and can assign input state or output state to each bit. If R4DD is ``1``, port R4 is in the output state, and if ``0``, it is in the input state. R4DD is write-only register. Since R4DD is initialized as ``00 h`` in reset state, the whole port R4 becomes input state.

(2) R4 Data Register (R4)

R4 data register (R4) is 1-bit register to store data of port R4. When this is set as the output state by R4DD, and data is written in R4, data is outputted into R4 pin. When this is set as the input state, input state of pin is read. The initial value of R4 is unknown in reset state.

(3) R4 Open drain Assign Register (R4ODC)

R4 Open Drain Assign Register (R4ODC) is 1-bit register and can assign R4 port as open drain output port for each bit, if corresponding port is selected as output. If R4ODC is selected as ``1``, port R4 is open drain output, and if selected as ``0``, it is push-pull output. R4ODC is write-only register and initialized as ``00 h`` in reset state.

(4) R4 Pull-up Resistor Control Register (R4PC)

R4 pull-up resistor control register (R4PC) is 1-bit register and can control pull-up on or off for each bit if corresponding port is selected as input. If R4PC is selected as ``1``, pull-up is disabled and if selected as ``0``, it is enabled. R4PC is write-only register and initialized as ``00 h`` in reset state. The pull-up is automatically disabled, if corresponding port is selected as output.

9. CLOCK GENERATOR

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

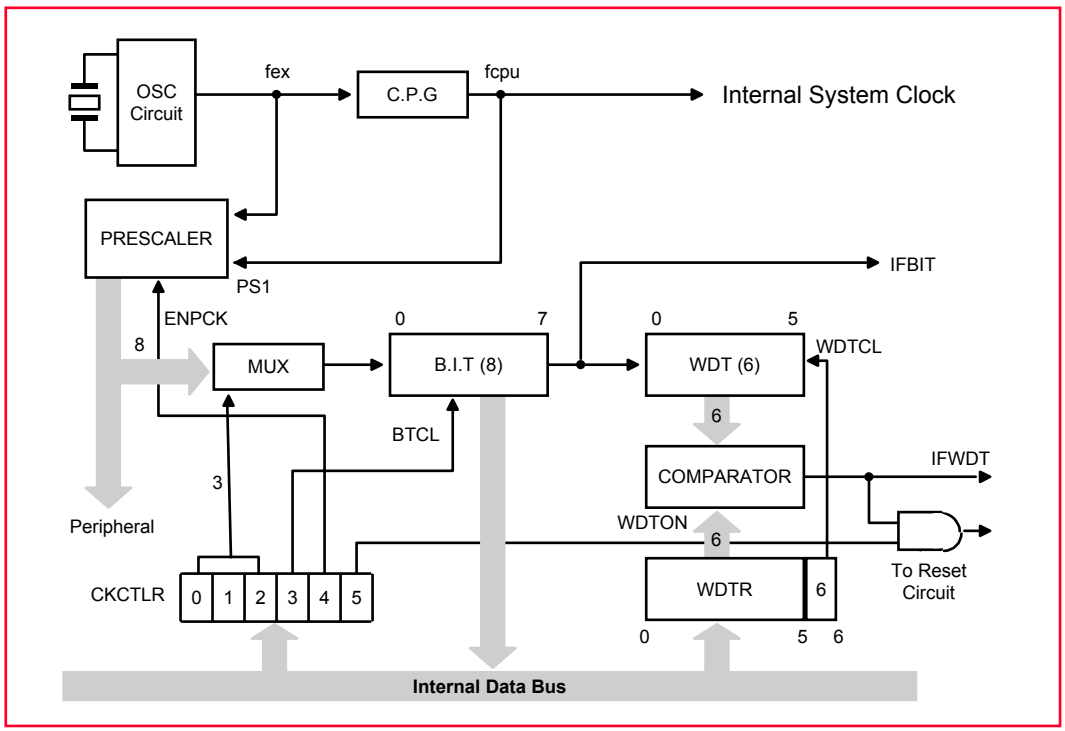


Figure 9-1 Block Diagram of Clock Generator

Prescaler consists of 12-bit binary counter. The clock supplied from oscillation circuit is input to prescaler (fex). The divided output from each bit of prescaler is provided to peripheral hardware.

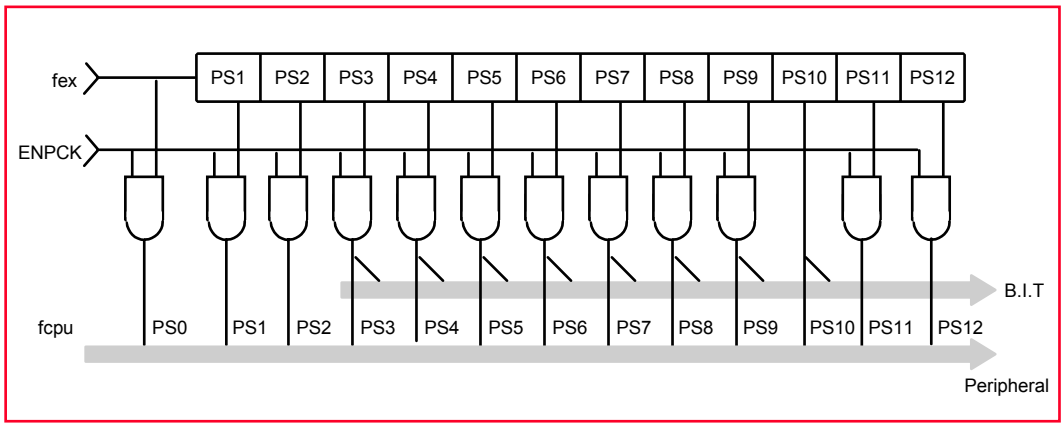


Figure 9-2 Block diagram of Prescaler

fex (MHz)	4 MHz		2 MHz	
	frequency	period	frequency	period
ps 0	4 MHz	250 ns	2 MHz	500 ns
ps 1	2 MHz	500 ns	1 MHz	1 us
ps 2	1 MHz	1 us	500 KHz	2 us
ps 3	500 KHz	2 us	250 KHz	4 us
ps 4	250 KHz	4 us	125 KHz	8 us
ps 5	125 KHz	8 us	62.5 KHz	16 us
ps 6	62.5 KHz	16 us	31.25 KHz	32 us
ps 7	31.25 KHz	32 us	15.63 KHz	64 us
ps 8	15.63 KHz	64 us	7.183 KHz	128 us
ps 9	7.183 KHz	128 us	3.906 KHz	256 us
ps 10	3.906 KHz	256 us	1.953 KHz	512 us
ps 11	1.953 KHz	512 us	0.976 KHz	1024 us
ps 12	0.976 KHz	1024 us	0.488 KHz	2048 us

Table 9-1 ps output period

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

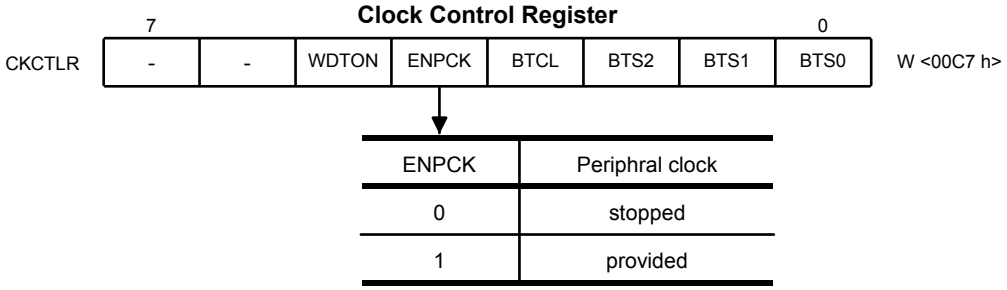


Figure 9-3 Clock Control Register

9.1 Operation Mode

The system clock controller starts or stops the main-frequency clock oscillator. Figure 10-2 shows the operating mode transition diagram.

Main-clock operating mode

This mode is fast-frequency operating mode.
The CPU and the peripheral hardwares operate on the high-frequency clock. At reset release, this mode is invoked.

STOP mode

In this mode, the system operations all stop, holding the internal states valid immediately before the stop at the low power consumption level.

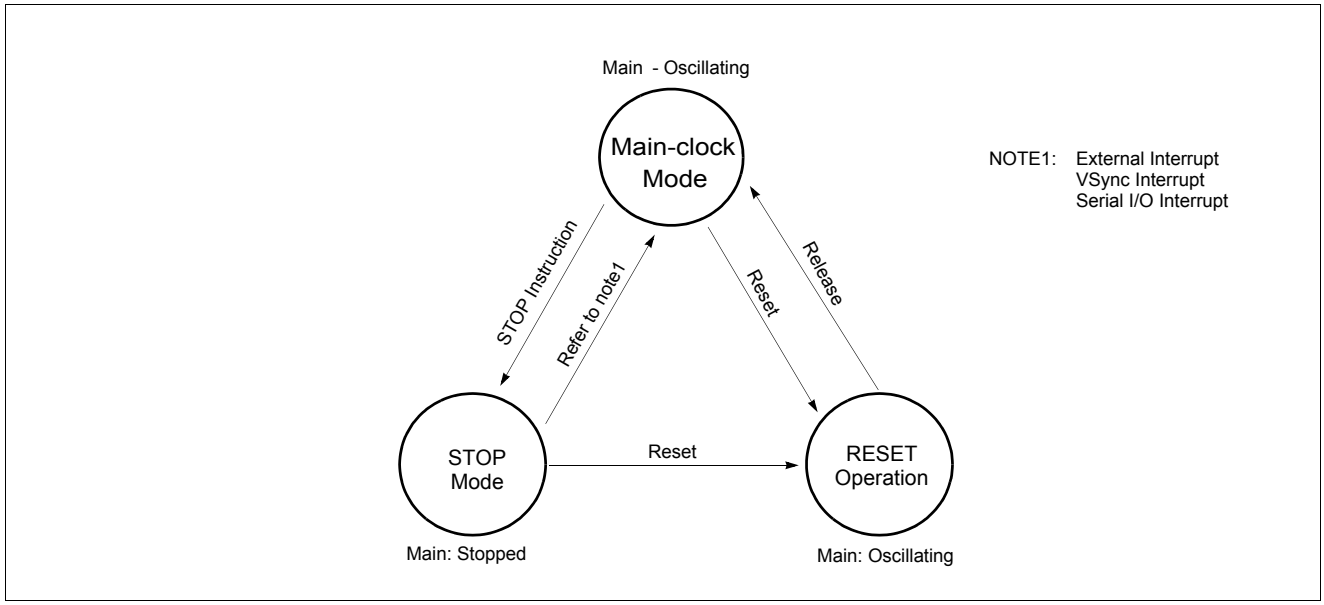


Figure 9-4 Operating Mode

10. TIMER

10.1 Basic Interval Timer

The GMS81C50 Series has one 8-bit Basic Interval Timer that is free-run and can not stop. Block diagram is shown in Figure 10-1.

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

-8-bit binary 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 B.I.T can be read

-Provides the clock for watch dog timer.

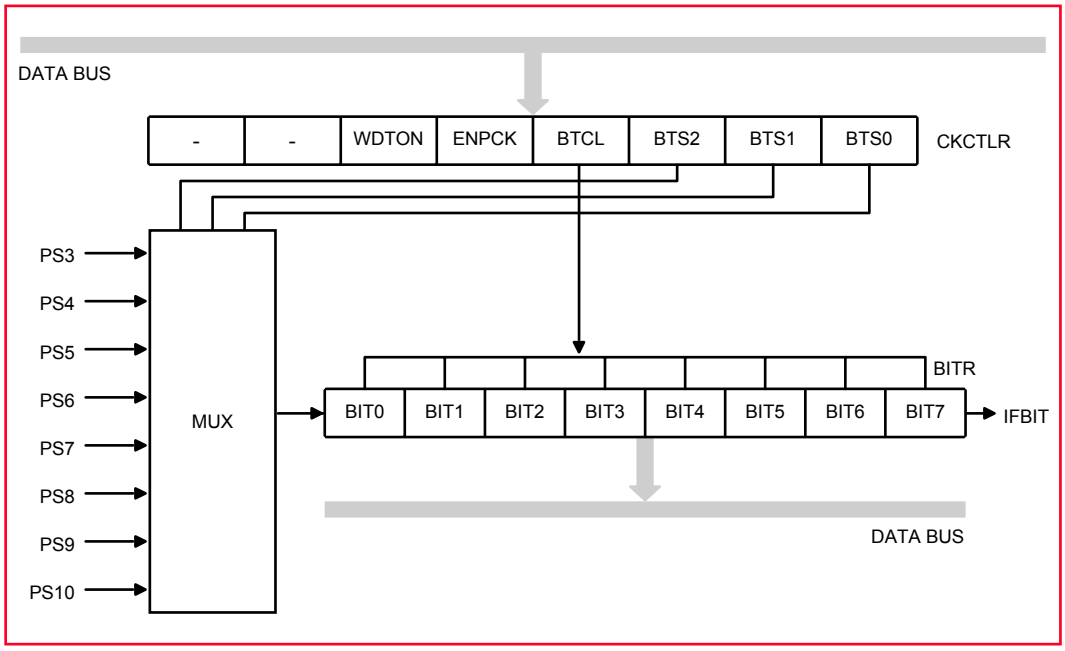


Figure 10-1 Block Diagram of Basic Interval Timer

(1) Control of B.I.T

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``, B.I.T is cleared, and then, after one machine cycle, BTCL becomes ``0``, and B.I.T starts counting. BTCL is set to ``0`` in reset state.

Clock Control Register

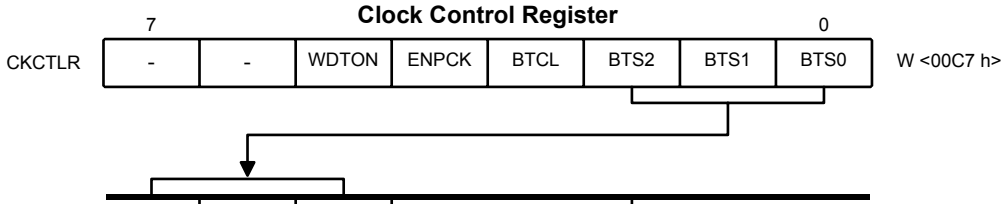
	7							0	
CKCTLR	-	-	WDTON	ENPCK	BTCL	BTS2	BTS1	BTS0	W <00C7 h>

ENPCK	Periphral clock
0	free-run
1	Automatically cleared, after one cycle

Figure 10-2 ENPCK mode of B.I.T

(2) Input clock selection of B.I.T

The input clock of B.I.T can be selected from the prescaler within a range of 2us to 256us by clock input selection bits (BTS2~BTS0). (at fex = 4MHz). In reset state, or power on reset, BTS2=``1``, BTS1=``1``, BTS0=``1`` to secure the longest oscillation stabilization time. B.I.T can generate the wide range of basic interval time interrupt request (IFBIT) by selecting prescaler output. Interrupt interval can be selected to kinds of interval time as shown in
Figure 10-3.

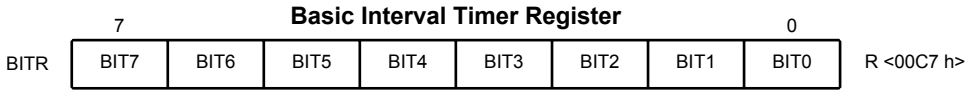


BTS2	BTS1	BTS0	B.I.T. Input clock	Standby release time
0	0	0	PS3 (2us)	512 us
0	0	1	PS4 (4us)	1,024 us
0	1	0	PS5 (8us)	2,048 us
0	1	1	PS6 (16us)	4,096 us
1	0	0	PS7 (32us)	8,192 us
1	0	1	PS8 (64us)	16,384 us
1	1	0	PS9 (128us)	32,768 us
1	1	1	PS10 (256us)	65,536 us

Figure 10-3 Basic Interval Timer Interrupt Time

(3) Reading Basic Interval Timer

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



10.2 Timer0, Timer1, Timer2

(1) Timer Operation Mode

Timer consists of 16-bit binary counter Timer0 (T0), 8-bit 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 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 : 00C9 h) value should be assigned for event counter,

Timer0	**- 16-bit Interval Timer** **- 16-bit Event Counter** **- 16-bit Input Capture** **- 16-bit Rectangular-wave Output**	**- Single/Modulo-N Mode** **- Timer Output Initial Value Setting** **- Timer0~Timer1 Combination Logic Output** **- One Interrupt Generating Every 2nd Counter Overflow**
Timer1	**- 8-bit Interval Timer** **- 8-bit Rectangular-wave Output**	
Timer2	**- 8-bit Interval Timer** **- 8-bit Rectangular-wave Output** **- Modulo-N Mode**	

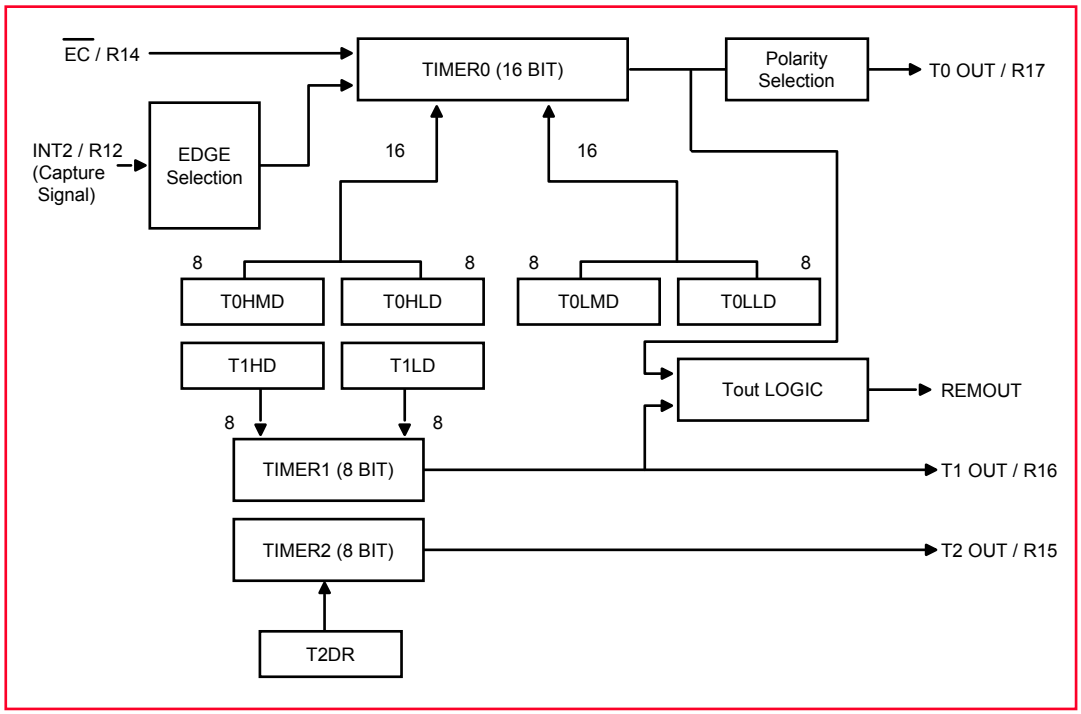


Figure 10-4 Timer / Counter Block diagram

(2) Function of Timer & Counter

fex = 4MHz

16bit Timer (T0)		8bit Timer (T1)		8bit Timer (T2)	
Resolution (CK)	Max. Count	Resolution (CK)	Max. Count	Resolution (CK)	Max. Count
PS0 (0.25 us)	16,384 us	PS0 (0.25 us)	64 us	PS5 (8 us)	2.048 us
PS1 (0. 5 us)	32,768 us	PS1 (0.5 us)	128 us	PS6 (16 us)	4,096 us
PS2 (1 us)	65,536 us	PS2 (1 us)	256 us	PS7 (32 us)	8,192 us
PS3 (2 us)	131,072 us	PS3 (2 us)	512us	PS8 (64 us)	16,384 us
PS4 (4 us)	262,144 us	PS7 (32 us)	8,192 us	PS9 (128 us)	32,768 us
PS5 (8 us)	524,288 us	PS8 (64 us)	16,384 us	PS10 (256 us)	65,536 us
PS11 (512 us)	33,554,432 us	PS9 (128 us)	32,768 us	PS11 (512 us)	131,072 us
EC	-	PS10 (256 us)	65,536 us	PS12 (1,024 us)	262,144 us

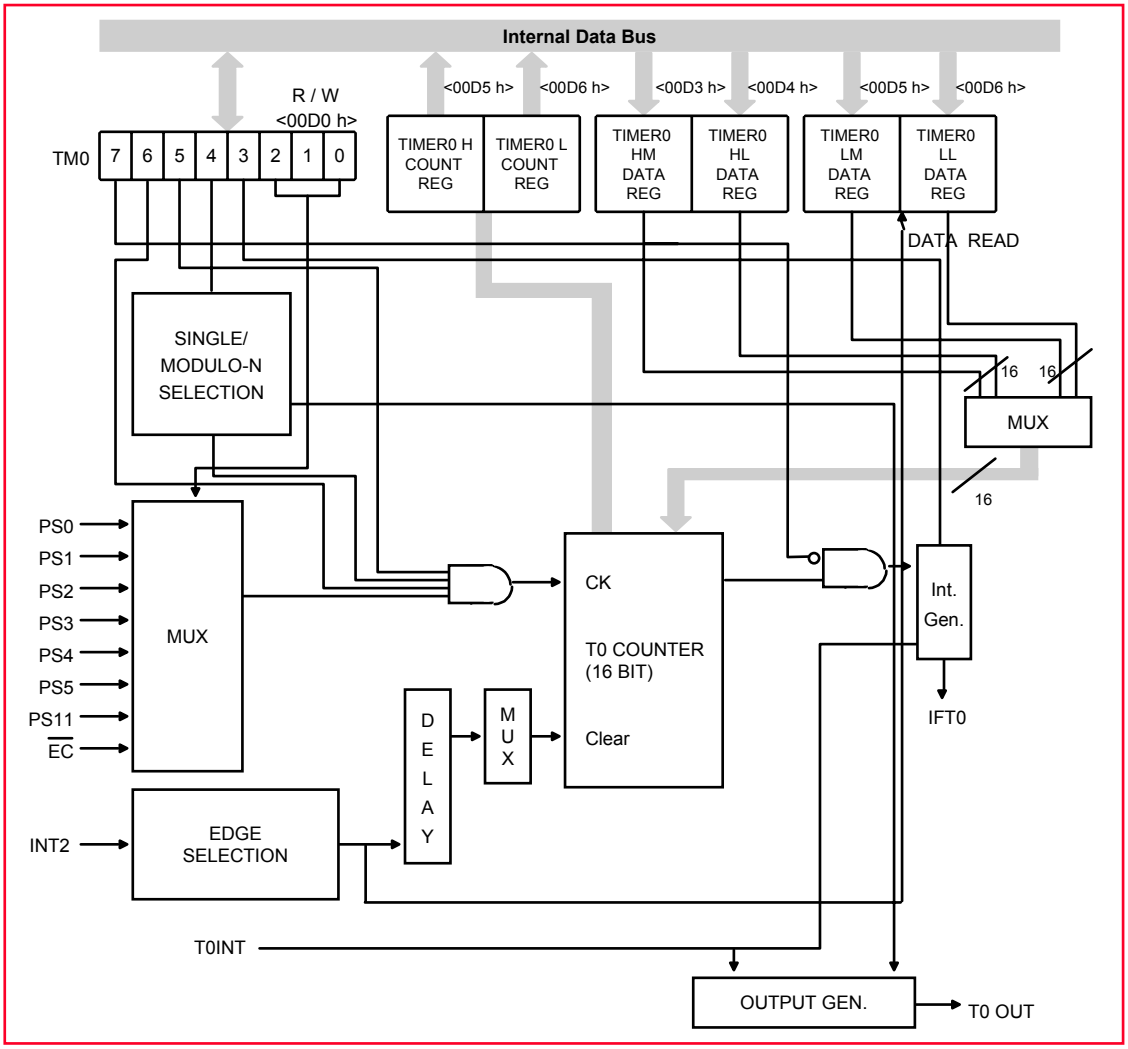


Figure 10-5 Block Diagram of Timer0

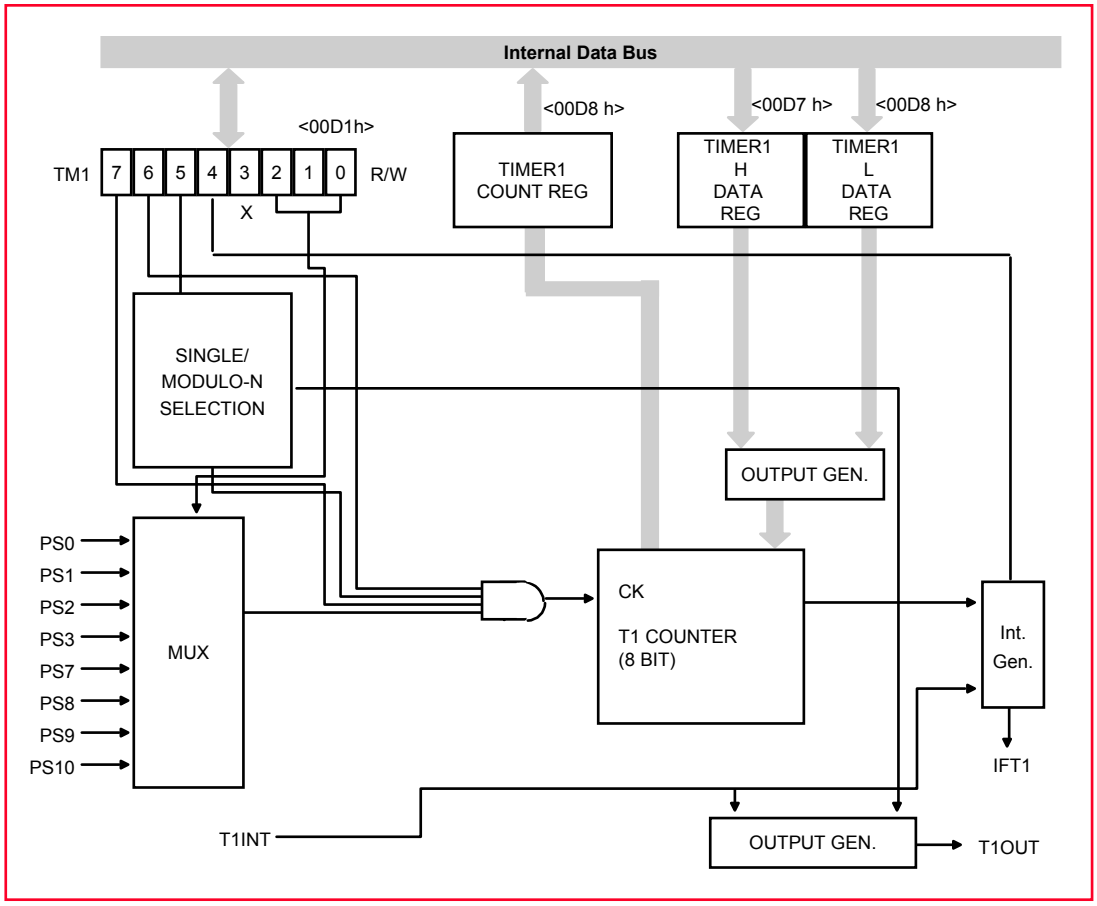


Figure 10-6 Block Diagram of Timer1

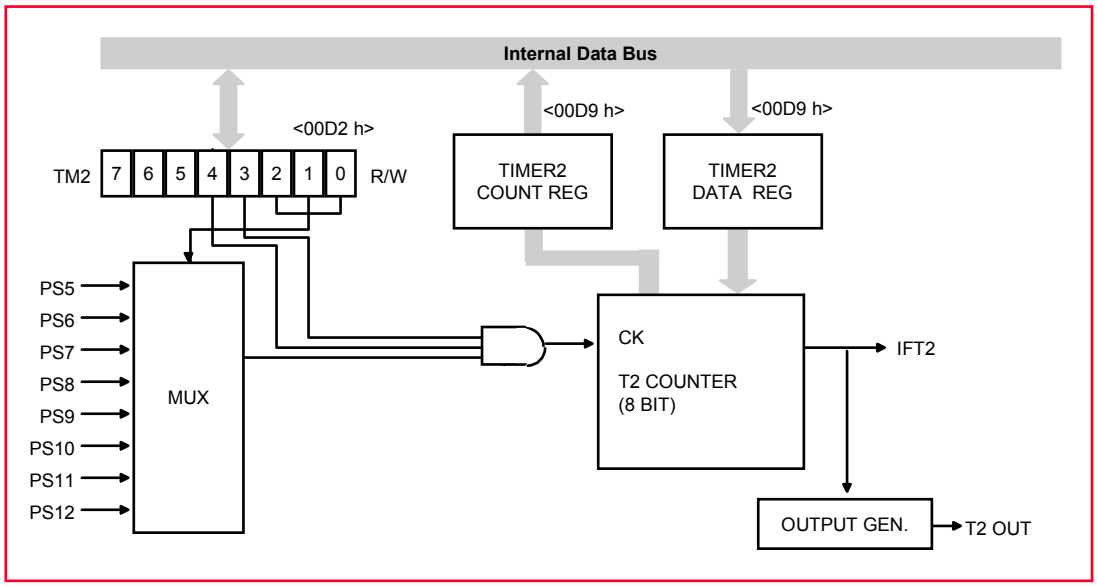


Figure 10-7 Block Diagram of Timer2

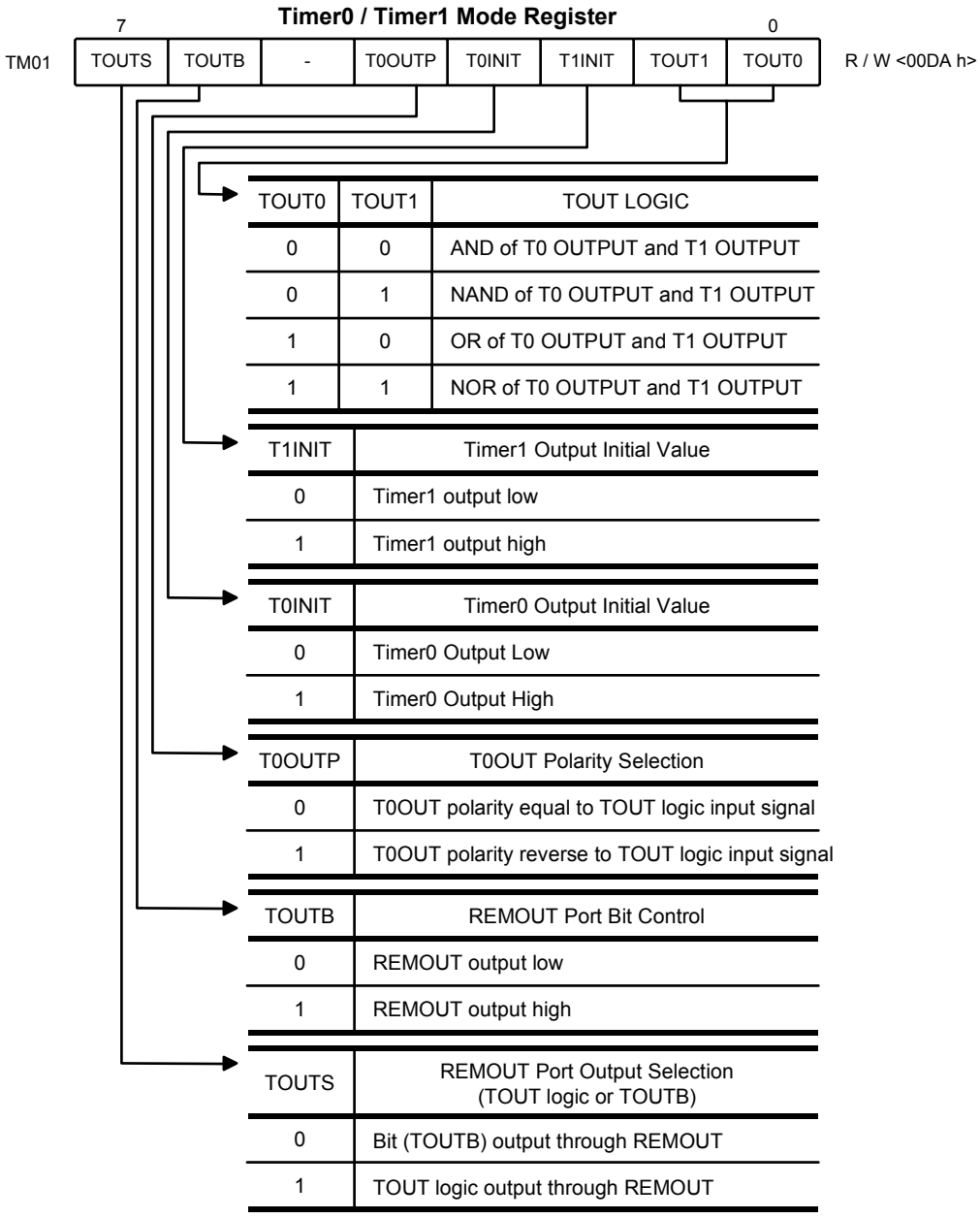


Figure 10-8 Timer0 / Timer1 Mode Register

Timer0 Mode Register

	7							0	
TM0	CAP0	T0ST	T0CN	T0MOD	T0IFS	T0SL2	T0SL1	T0SL0	R / W <00D0 h>

T0SL2	T0SL1	T0SL0	Input clock selection	Notes
0	0	0	PS0 (250ns)	
0	0	1	PS1 (500ns)	*
0	1	0	PS2 (1us)	
0	1	1	PS3 (2us)	
1	0	0	PS4 (4us)	
1	0	1	PS5 (8us)	
1	1	0	PS11 (512us)	Event
1	1	1	$\overline{EC}$	Counter

T0IFS	Timer0 Interrupt Selection
0	Interrupt every counter overflow
1	Interrupt every 2nd counter overflow

T0MOD	Timer0 Single/Modulo-N Selection
0	Modulo-N
1	Single

T0CN	Timer0 Counter Continuation/Pause Control
0	Count pause
1	Count contination

T0ST	Timer0 Start/Stop Control
0	Timer0 Stop
1	Timer Start after clear

CAP0	Timer0 Interrupt Selection
0	Timer/Counter
1	Input capture *

* PS1 : not supporting input capture.

Figure 10-9 Timer0 Mode Register

Timer1 Mode Register

	7							0	
TM1	T1ST	T1CN	T1MOD	T1IFS	-	T1SL2	T1SL1	T1SL0	R / W <00D1 h>

T1SL2	T1SL1	T1SL0	Input clock selection
0	0	0	PS0 (250ns)
0	0	1	PS1 (500ns)
0	1	0	PS2 (1us)
0	1	1	PS3 (2us)
1	0	0	PS7 (32us)
1	0	1	PS8 (64us)
1	1	0	PS9 (128us)
1	1	1	PS10 (256us)

T1IFS	Timer1 Interrupt Selection
0	Interrupt every counter overflow
1	Interrupt every 2nd counter overflow

T1MOD	Timer1 Single/Modulo-N Selection
0	Modulo-N
1	Single

T1CN	Timer1 Counter Continuation/Pause Control
0	Count pause
1	Count contination

T1ST	Timer1 Start/Stop Control
0	Timer1 Stop
1	Timer1 Start after clear

Figure 10-10 Timer1 Mode Register

Timer2 Mode Register

	7							0	
TM2	-	-	-	T2ST	T2CN	T2SL2	T2SL1	T2SL0	R / W <00D2 h>

T2SL2	T2SL1	T2SL0	Input clock selection
0	0	0	PS5 (8us)
0	0	1	PS6 (16us)
0	1	0	PS7 (32us)
0	1	1	PS8 (64us)
1	0	0	PS9 (128us)
1	0	1	PS10 (256us)
1	1	0	PS11 (512us)
1	1	1	PS12 (1024us)

T2CN	Timer2 Counter Continuation/Pause Control
0	Count pause
1	Count contination

T2ST	Timer2 Start/Stop Control
0	Timer2 Stop
1	Timer2 Start after clear

Figure 10-11 Timer2 Mode Register

External Interrupt Signal Edge Selection Register

	7							0	
IEDS	-	-	IED2H	IED2L	IED1H	IED1L	-	-	W <00CB h>

IED*H	IED*L	INT*
0	0	-
0	1	Falling Edge Selection
1	0	Rising Edge Selection
1	1	Both Edge Selection

Figure 10-12 External Interrupt Signal Edge Selection Register

(3) 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 ``00 h``, and interrupt (IFT0, IFT1) is occurred at the next clock.

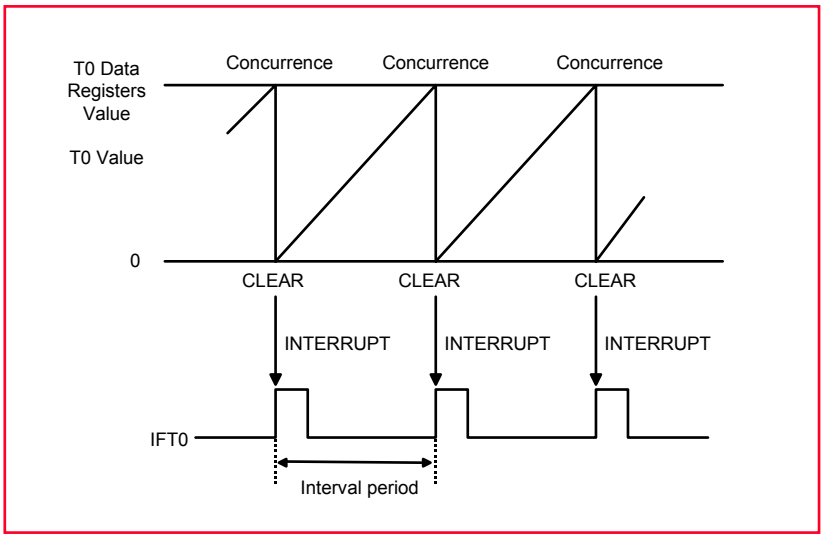


Figure 10-13 Operation of Timer0

For Timer0, the internal clock (PS) and the external clock (EC) can be selected as counter clock. But Timer1 and Timer2 use only internal clock as internal clock. 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 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``)

Note: *In the process of reading 16-bit Timer Data, first read the upper 8-bit data. Then read the lower 8-bit data, and read the upper 8-bit data again. If the former read upper 8-bit data are matched with the later read upper 8-bit data, read 16-bit data are correct. If not, be cautious in the selection of upper 8-bit data.*

(Example)

1) Upper 8-bit Read 0A 0A
2) Lower 8-bit Read FF 01
3) Upper 8-bit Read 0B 0B

==========

\- -

0AFF 0B01

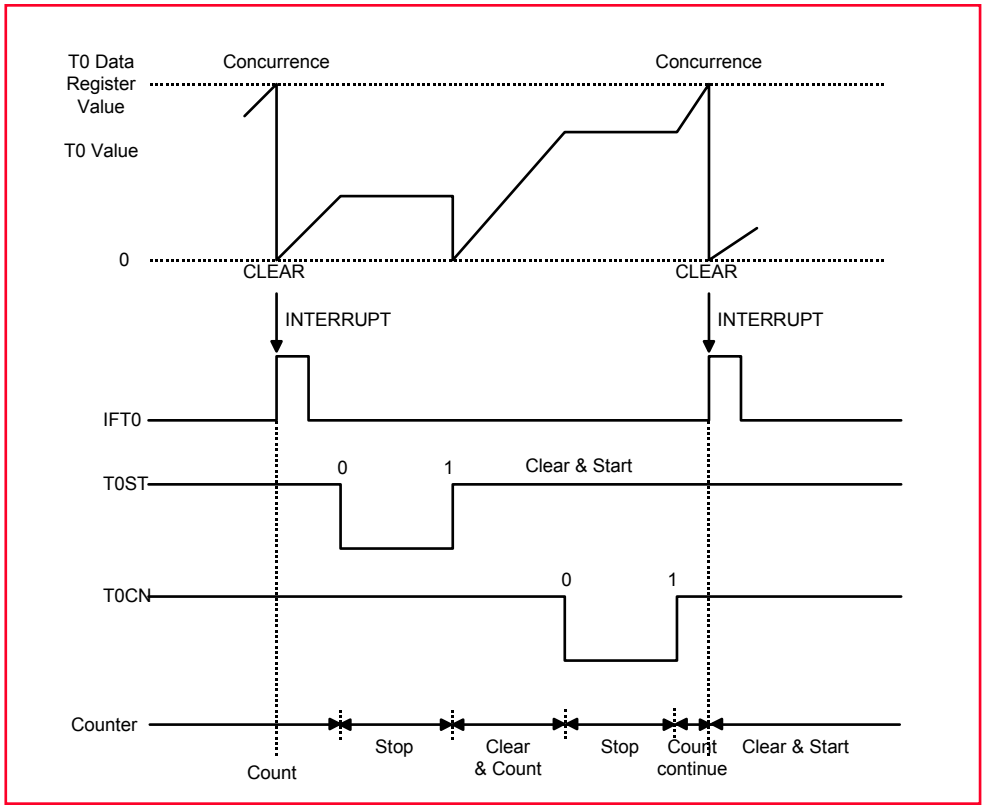


Figure 10-14 Start/Stop Operation of Timer0

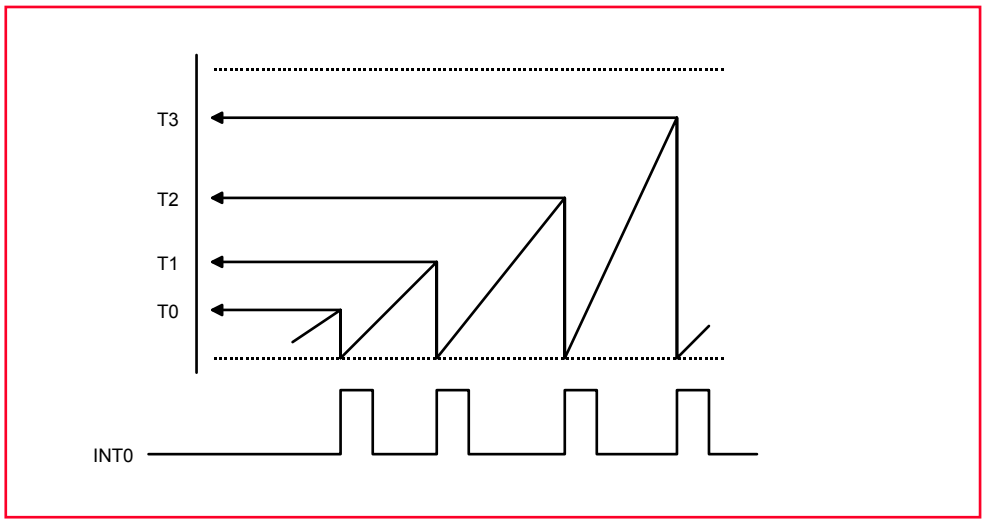


Figure 10-15 Input Capture Operation of Timer0

* Single/Modulo-N Mode

Timer0 (Timer1) can select initial (T0INIT, T1INIT of TM0, TM1) output level of Timer Output port. 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 iHigh? 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 during the value of Data Register and Time-out interrupt occurs. The level of Timer Output port toggles and repeats counting process of 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 on every second time-out.

Note: *(*note. Timer Output is toggled whenever time out happens)*

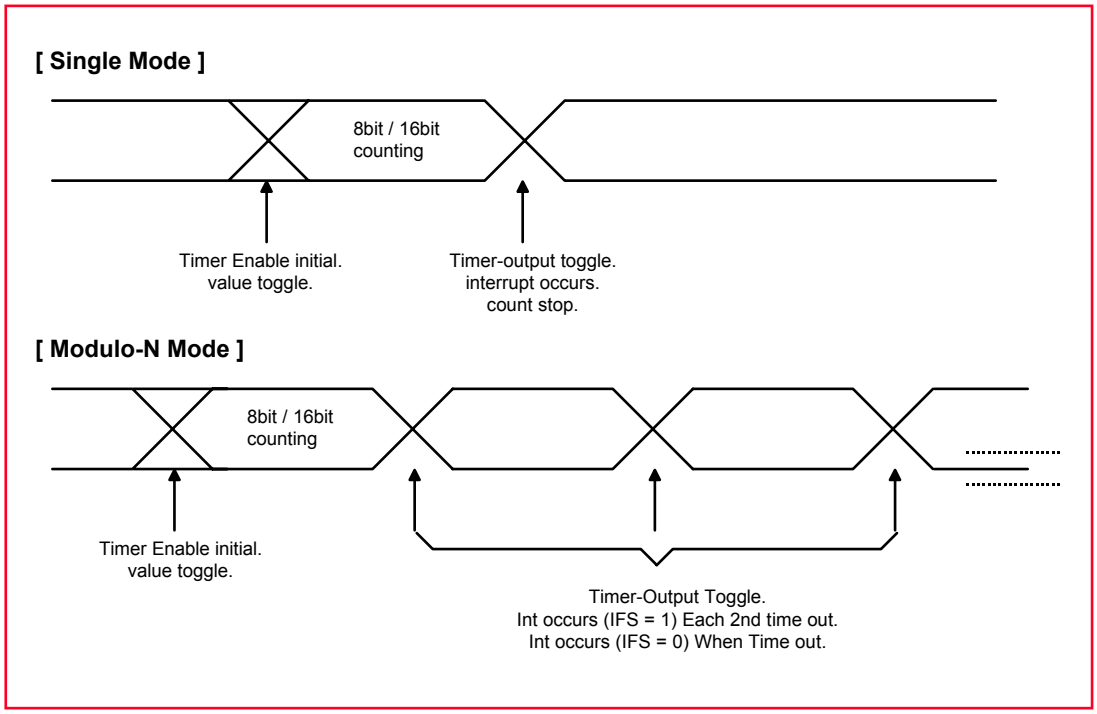


Figure 10-16 Operation Diagram for Single/Modulo-N Mode

(4) Timer 2

Timer2 operates as a 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 ``00 h``. Therefore, Timer2 executes as a interval timer. Interrupt period is determined by the count source clock for the Timer2 and content 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``. Count value of Timer2 can be read at any time.

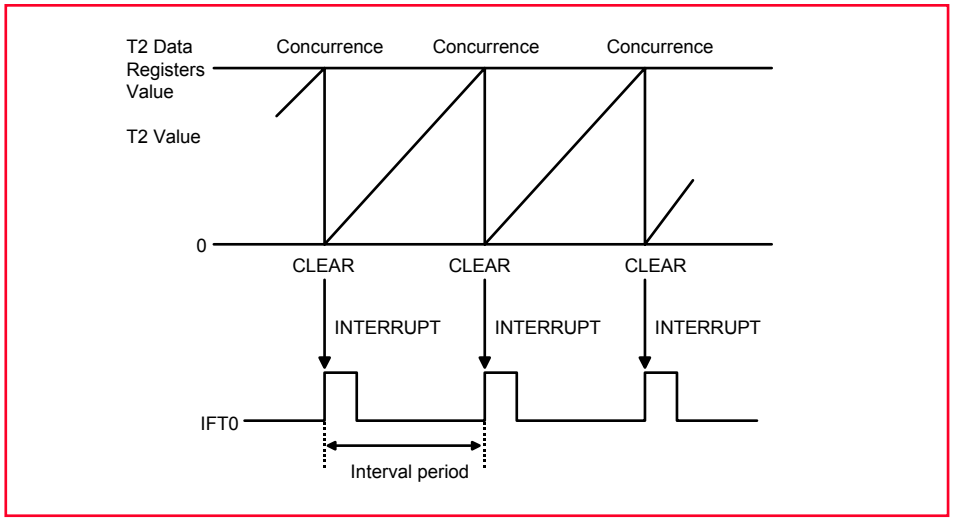


Figure 10-17 Operation of Timer2

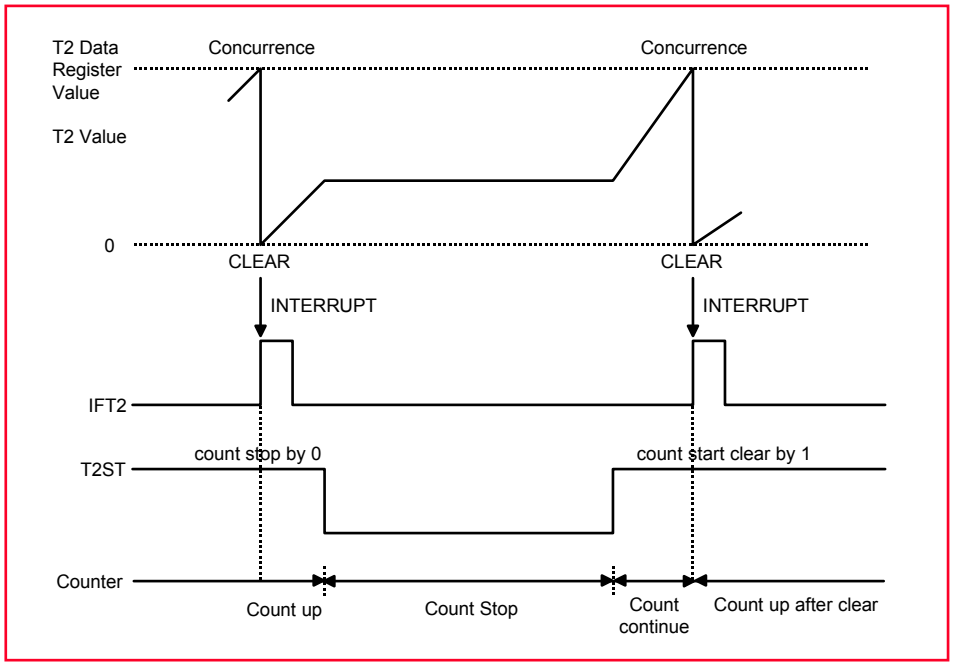


Figure 10-18 Start/Stop of Timer2

11. INTERRUPTS

The GMS81C50 Series 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 11-1.

The GMS81C50 Series contain 8 interrupt sources; 3 externals and 5 internals. Nested interrupt services with priority control are also possible. Software interrupt is non-maskable interrupt, the others are all maskable interrupts.

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

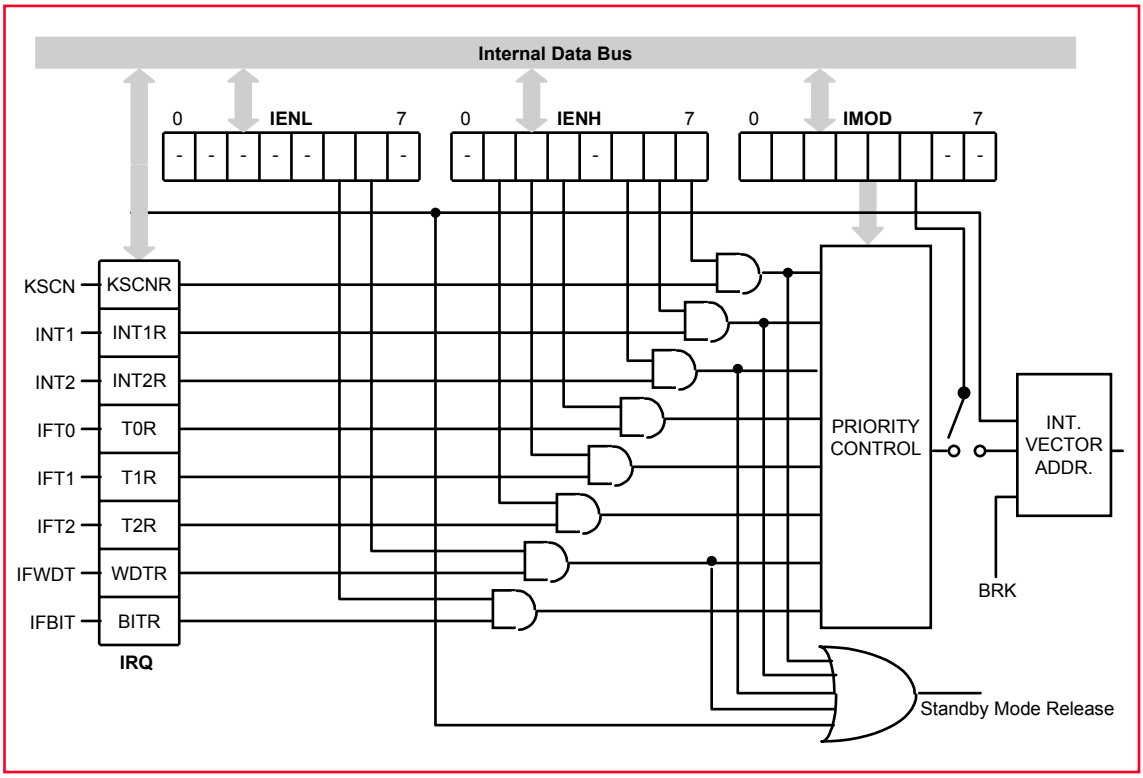


Figure 11-1 Block Diagram of Interrupt

11.1 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 11-1.

	Mask	Priority	Interrupt Source	INT Vector High	INT Vector Low
Hardware Interrupt	non-maskable	-	RST ($\overline{\text{RESET}}$ pin)	FFFF	FFFE
	maskable	0	KSCNR (Key Scan)	FFFB	FFFA
		1	INT1R (External Interrupt1)	FFF9	FFF8
		2	INT2R (External Interrupt2)	FFF7	FFF6
		3	T0R (Timer0)	FFF3	FFF2
		4	T1R (Timer1)	FFF1	FFF0
		5	T2R (Timer2)	FFEF	FFEE
		6	WDTR (Watctdog Timer)	FFE9	FFE8
		7	BITR (Basic Interval Timer)	FFE7	FFE6
	-	-	BRK instruction	FFDF	FFDE

Table 11-1 Interrupt Priority & Source

11.2 Interrupt Control Register

I flag of PSW is an interrupt mask enable flag. When I flag = ``0``, all interrupts become disabled. When I flag = ``1``, interrupts can be selectively enabled and disabled by contents of corresponding Interrupt Enable Register. When an interrupt is occurred, an interrupt request flag is set, and the 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).

IENL	**-**	**WDTR**	**BITE**	**-**	**-**	**-**	**-**	**-**	R/W <00CCh>
IENH	**KSCNE**	**INT1E**	**INT2E**	**-**	**T0E**	**T1E**	**T2E**	**-**	R/W <00CEh>
IRQL	**-**	**WDTR**	**BITE**	**-**	**-**	**-**	**-**	**-**	R/W <00CDh>
IRQH	**KSCNE**	**INT1R**	**INT2R**	**-**	**T0R**	**T1R**	**T2R**	**-**	R/W <00CFh>

IENL : INTERRUPT ENABLE REGISTER LOW

IENH : INTERRUPT ENABLE REGISTER HIGH

IRQL : INTERRUPT REQUEST REGISTER LOW

IRQH : INTERRUPT REQUEST REGISTER HIGH

11.3 Interrupt Accept Mode

The interrupt priority order is determined by bit (IM1, IM0) of IMOD register.

Interrupt Mode Register

IMOD	7 -	-	IM1	IM0	IP3	IP2	IP1	0 IP0	R/W <00CA h>

Assigning by interrupt accept mode bit

IM1	IM0	Priority
0	0	fixed by hardware
0	1	changeable by IP3~ IP0
1	*	Interrupt is inhited

(1) Selection of Interrupt by IP3-IP0

The condition allowance for accepting interrupt is set the 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``.

IP3	IP2	IP1	IP0	Selection Interrupt
0	0	0	1	KSCNR (Key Scan)
0	0	1	0	INT1R (External interrupt 1)
0	0	1	1	INT2R (External interrupt 2)
0	1	0	0	Reserved
0	1	0	1	T0R (Timer 0)
0	1	1	0	T1R (Timer 1)
0	1	1	1	T2R (Timer 2)
1	0	0	0	Reserved
1	0	0	1	Reserved
1	0	1	0	WDTR (Watch Dog Timer)
1	0	1	1	BITR (Basic Interval Timer)
1	1	0	0	Reserved

Table 11-1 Interrupt Selection by IP3 - IP0

(2) Interrupt Timing

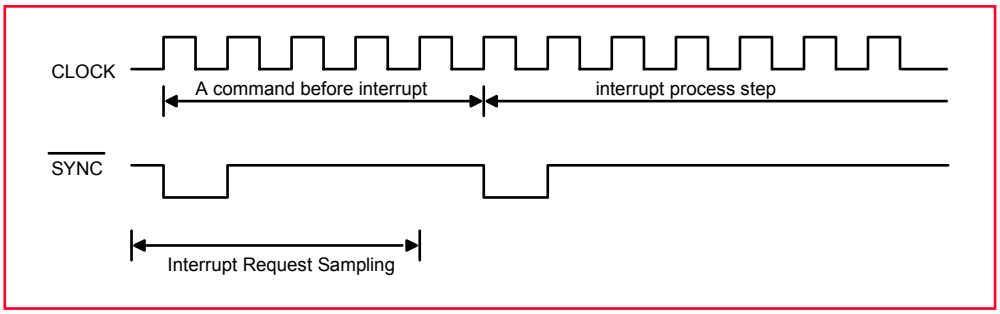


Figure 11-2 Interrupt Enable Accept Timing

*Interrupt Request sampling time

-Maximum 12 machine cycles (When execute DIV instruction)

-Minimum 0 machine cycle

*Interrupt preprocess step is 8 machine cycles

*Interrupt overhead

-Maximum 1 + 12 + 8 = 21 machine cycles

-Minimum 1 + 0 + 8 = 9 machine cycles

(3) The Valid Timing after the Execution of Interrupt Control Instructions

I flag is valid just after the execution of EI/DI on the contrary. Interrupt Enable register is valid one instruction after it is controlled.

11.4 Interrupt Processing Sequence

When an interrupt is accepted, the on-going process stops and the interrupt service routine is executed. After the interrupt service routine is completed it is necessary to restore everything to the state before the interrupt occurs. As soon as an interrupt is accepted, the content of the program counter and PSW are saved in the stack area. At the same time, the content of the vector address corresponding to the accepted interrupt, which is in the interrupt vector table, enters into the program counter and interrupt service is executed. In order to execute the interrupt service routine, it is necessary to write the jump addresses in the vector table (FFE0 h ~ FFFF h) corresponding to each interrupt

* Interrupt Processing Step

1) Store upper byte of Program Counter, SP <= SP

2) Store lower byte of Program Counter, SP <= SP - 1

3) Store Program Status Word, SP <= SP - 2

4) After reset of I-flag, clear accepted Interrupt Request Flag. (Set B-flag for BRK Instruction)

5) Call Interrupt service routine

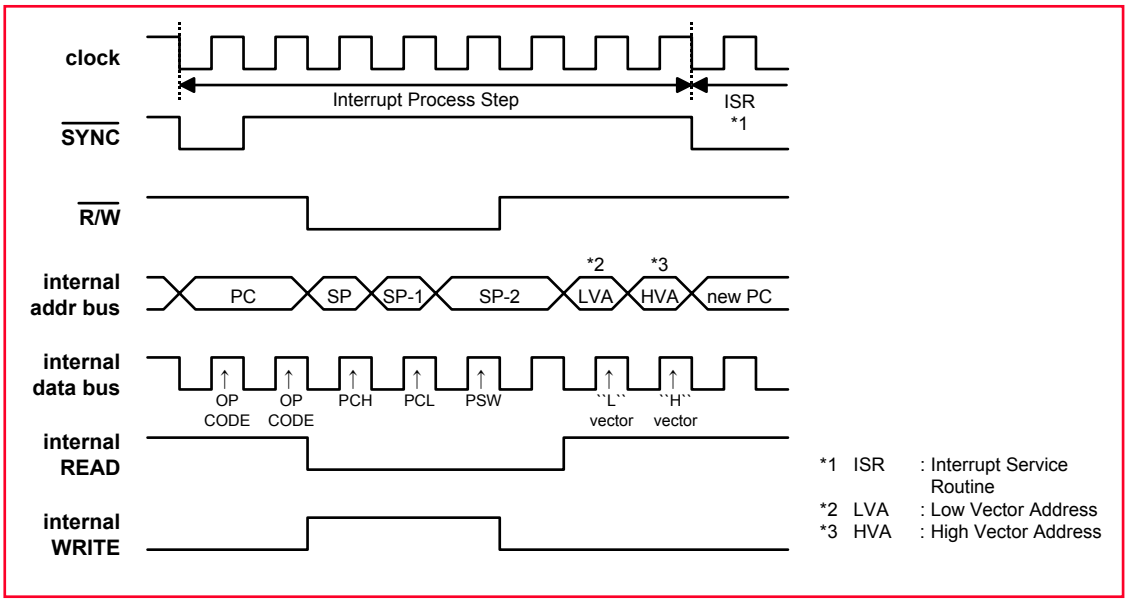
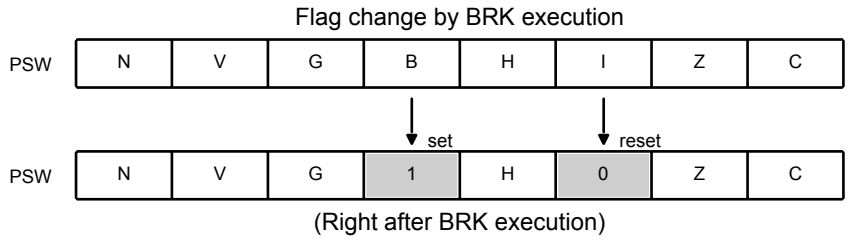


Figure 11-3 Interrupt Processing Step Timing

11.5 SOFTWARE INTERRUPT (Interrupt by Break (BRK) Instruction)

Software interrupt is available just by writing ``Break(BRK)`` instruction. The values of PC and PSW are stacked by BRK instruction and then B flag of PSW is set and I flag is reset.



Interrupt vector of BRK instruction is shared with vector of Table Call (TCALL0). When both instructions of BRK and TCALL0 are used, as shown in Figure 11-4. Each processing routine is judged by contents of B flag. There is no instruction to reset directly B flag.

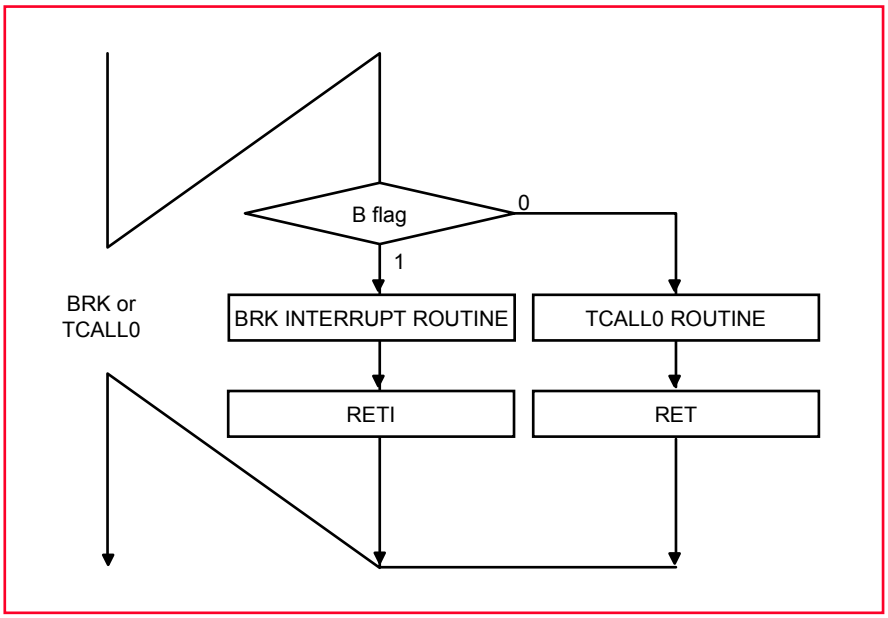


Figure 11-4 Execution of BRK or TCALL0

11.6 MULTIPLE INTERRUPT

If there is an interrupt, Interrupt Mask Enable Flag is automatically cleared before the Interrupt Service Routine enters. After then, no interrupt is accepted. If EI instruction is executed, interrupt mask enable bit becomes ``1``, and each enable bit can accept interrupt request. When two or more interrupts are generated simultaneously, the highest priority interrupt set by Interrupt Mode Register is accepted.

11.7 Key Scan Input Processing

(1) Standby Mode Release Register (SMRR)

Key Scan Interrupt 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 16-bit 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 execution, the

rest of execution, SLEEP mode and STOP mode, 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 ``00 h``. So, there is no Key Input source.

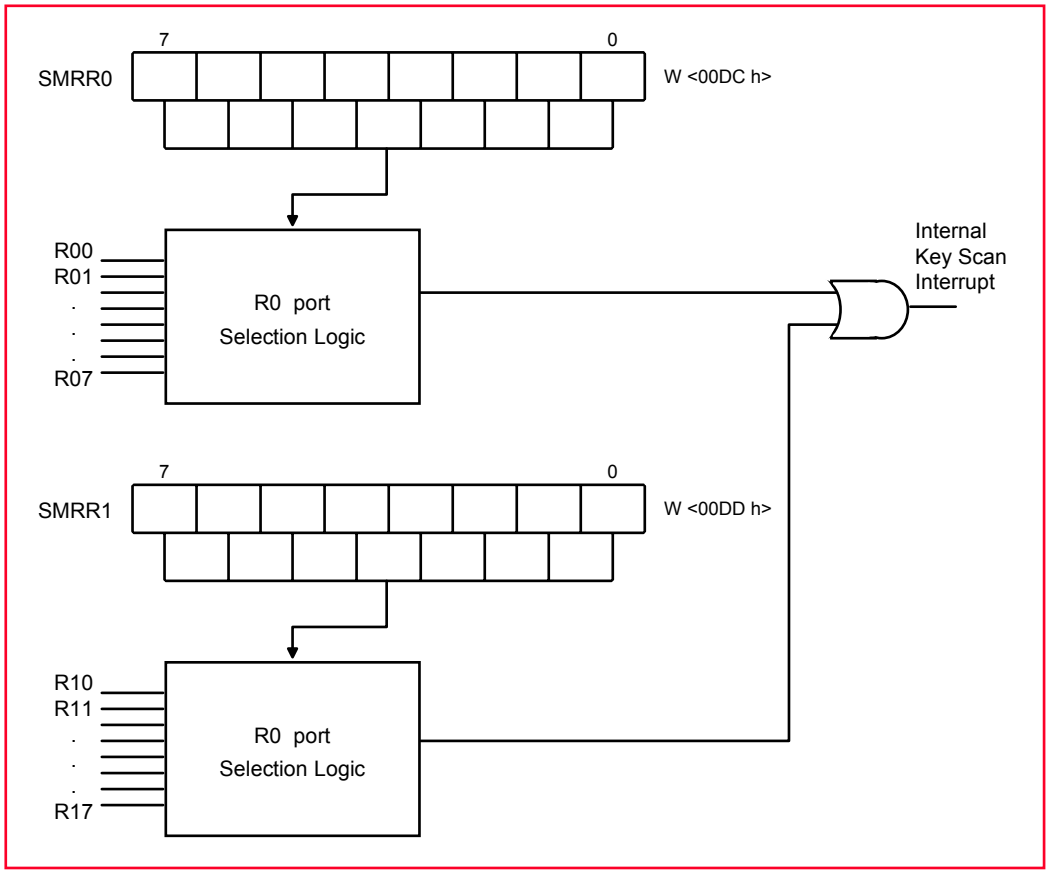


Figure 11-5 Key Scan Block

SMRR0 Register

	7							0	
SMRR0	KR07	KR06	KR05	KR04	KR03	KR02	KR01	KR00	W <00DC h>

SMRR1 Register

	7							0	
SMRR1	KR17	KR16	KR15	KR14	KR13	KR12	KR11	KR10	W <00DD h>

SMRR0		SMRR1		Key Input Selection
KR07	0	KR17	0	no select
	1		1	select
KR06	0	KR16	0	no select
	1		1	select
KR05	0	KR15	0	no select
	1		1	select
KR04	0	KR14	0	no select
	1		1	select
KR03	0	KR13	0	no select
	1		1	select
KR02	0	KR12	0	no select
	1		1	select
KR01	0	KR11	0	no select
	1		1	select
KR00	0	KR10	0	no select
	1		1	select

(2) Standby Release Level Control Register (SRLC)

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 ``00 h`` in reset state.

SRLC0 Register

	7							0	
SRLC0	KLR07	KLR06	KLR05	KLR04	KLR03	KLR02	KLR01	KLR00	W <00F6 h>

SRLC1 Register

	7							0	
SRLC1	KLR17	KLR16	KLR15	KLR14	KLR13	KLR12	KLR11	KLR10	W <00F7 h>

SRLC0		SRLC1		Key Input Level
KLR07	0	KLR17	0	Low
	1		1	High
KLR06	0	KLR16	0	Low
	1		1	High
KLR05	0	KLR15	0	Low
	1		1	High
KLR04	0	KLR14	0	Low
	1		1	High
KLR03	0	KLR13	0	Low
	1		1	High
KLR02	0	KLR12	0	Low
	1		1	High
KLR01	0	KLR11	0	Low
	1		1	High
KLR00	0	KLR10	0	Low
	1		1	High

12. WATCH DOG TIMER

Watch Dog Timer (WDT) consists of 6-bit binary counter, 6-bit comparator, and Watch Dog Timer Register (WDTR).

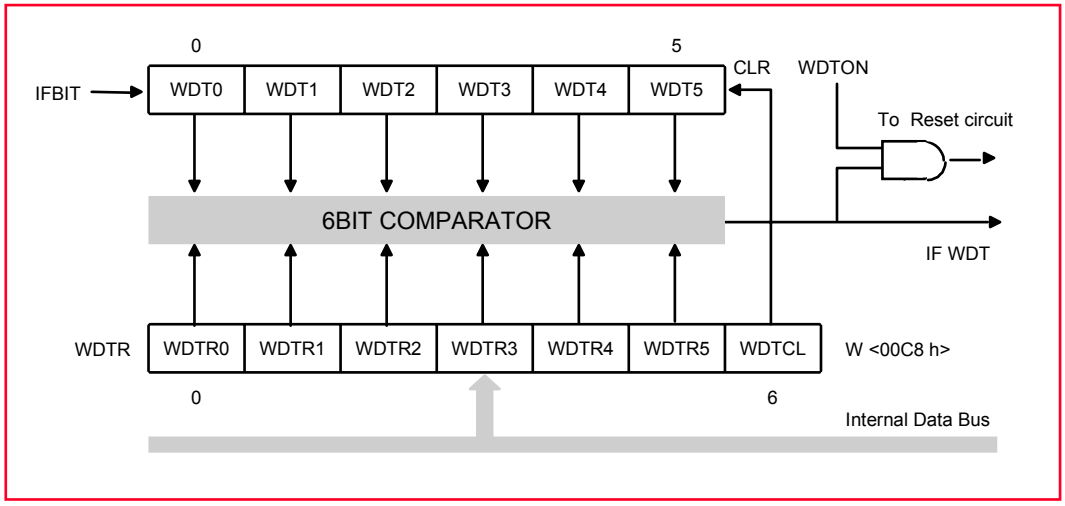


Figure 12-1 Block diagram of Watch Dog Timer

12.1 Control of WDT

Watch Dog Timer can be used as 6-bit general Timer or specific Watch dog timer by setting bit5 (WDTON) of Clock Control Register (CKCTLR).

Clock Control Register

	7							0	
CKCTLR	-	-	WDTON	ENPCK	BTCL	BTS2	BTS1	BTS0	W <00C7 h>

WDTON	Watch Dog Timer Function Control
0	6-bit Timer
1	Watch Dog Timer

By assigning bit6(WDTCL) of WDTR, 6-bit counter can be cleared.

Watch DOG Timer Register

	7							0	
WDTR	-	WDTCL	WDTR5	WDTR4	WDTR3	WDTR2	WDTR1	WDTR0	W <00C8 h>

Determine Interval of IFWDT
Interval of IFWDT = Value of WDTR × Interval of IFBIT

WDTCL	Watch Dog Timer Operation
0	free-run
1	Automatically cleared, after one machine cycle

12.2 Interrupt Interval

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) * (WDTR value)

-Interval of IFWDT : 512 us * 1 = 512 us (MIN>)

-65,536us * 63 = 4,128,768 us (MAX>)

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

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

-Reset WDTR value = 0F h,15

-interval of WDT = 65,536 * 15 = 983040 us

(about 1second)

Clock Control Register

	7							0	
CKCTLR	-	-	WDTON	ENPCK	BTCL	BTS2	BTS1	BTS0	W <00C7 h>

BTS2	BTS1	BTS0	WDT Input clock	Max. Interval of WDT Output (*note1)
0	0	0	512 us	32,756 us
0	0	1	1,024 us	64,512 us
0	1	0	2,048 us	129,024 us
0	1	1	4,096 us	258,048 us
1	0	0	8,192 us	516,096 us
1	0	1	16,384 us	1,032,192 us
1	1	0	32,768 us	2,064,384 us
1	1	1	65,536 us	4,128,768 us

Note: *When WDTR Register value is 63 (3F h) (Caution) : Do not use ``0`` for WDTR Register value.*

Device comes into the reset state by WDT.

13. STANDBY FUNCTION

To save power consumption, there is STOP modes. In this modes, the execution of program stops.

13.1 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. B.I.T and a part of peripheral hardware execute, but prescaler is output which provide clock to peripherals can be stopped by program. (Except, PS10 can't be stopped.) In SLEEP mode, more consuming power can be saved by not using other peripheral hardware except for B.I.T. By setting ENPCK (peripheral clock control bit) of CKCTLR (clock control register) to ``0``, peripheral hardware halts, and SLEEP mode is entered. To release SLEEP mode by BITR (basic interval timer interrupt), bit10 of prescaler should be selected as B.I.T input clock before SLEEP mode is entered. ``NOP`` instruction should follow after setting of SLEEP mode for rising precharge time of data bus line.

(ex) setting of SLEEP mode : set the bit of SLEEP mode register (SLPM)

NOP : NOP instruction

SLEEP MODE CONTROL Register

	7							0	
SLPM	-	-	-	-	-	-	-	SLPM0	W <00F0 h>

SLPM0	condition
0	sleep mode release
1	sleep mode

Colck Control Register

	7							0	
CKCTLR	-	-	WDTON	ENPCK	BTCL	BTS2	BTS1	BTS0	W <00C8 h>

ENCPK	Peripheral Clock
0	stopped
1	provided

13.2 Stop Mode

STOP mode can be entered by STOP instruction during program. In STOP mode, oscillator stops to make all clocks stop, which leads to less power consumption. All registers and RAM data are preserved. ``NOP`` instruction should follow after STOP instruction for rising precharge time of Data Bus line.

(ex) STOP : STOP instruction execution

NOP : NOP instruction

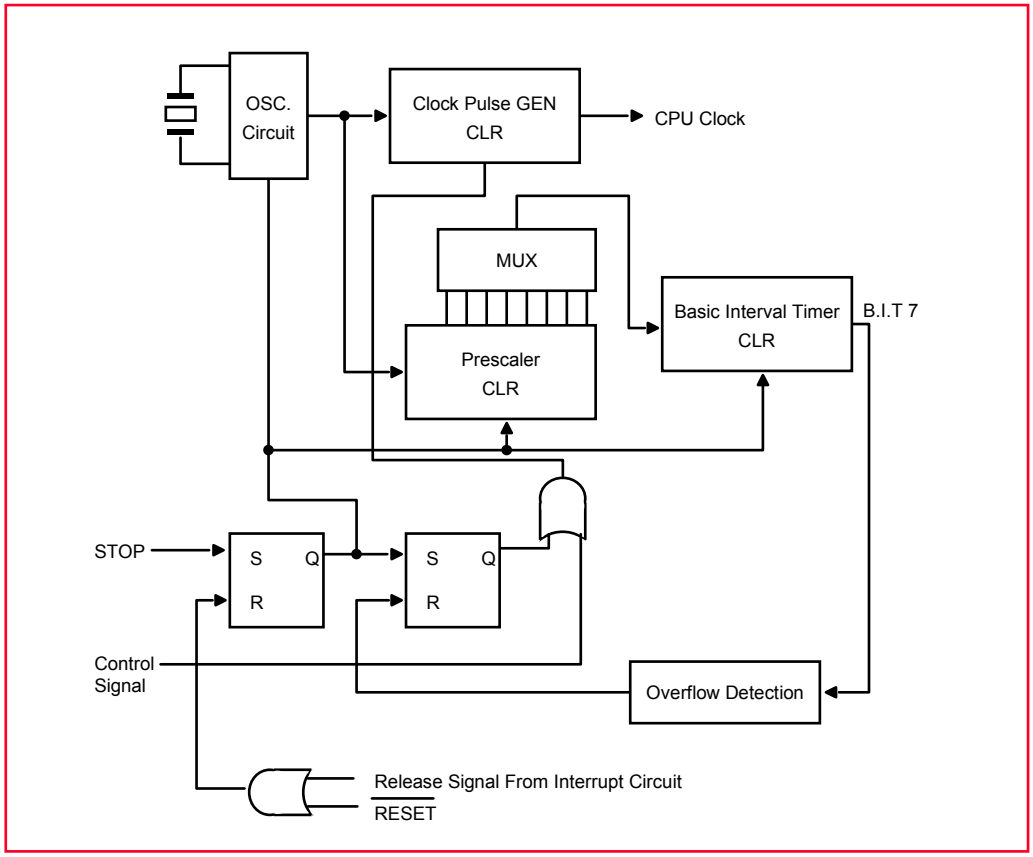


Figure 13-1 Block Diagram of Standby Circuit

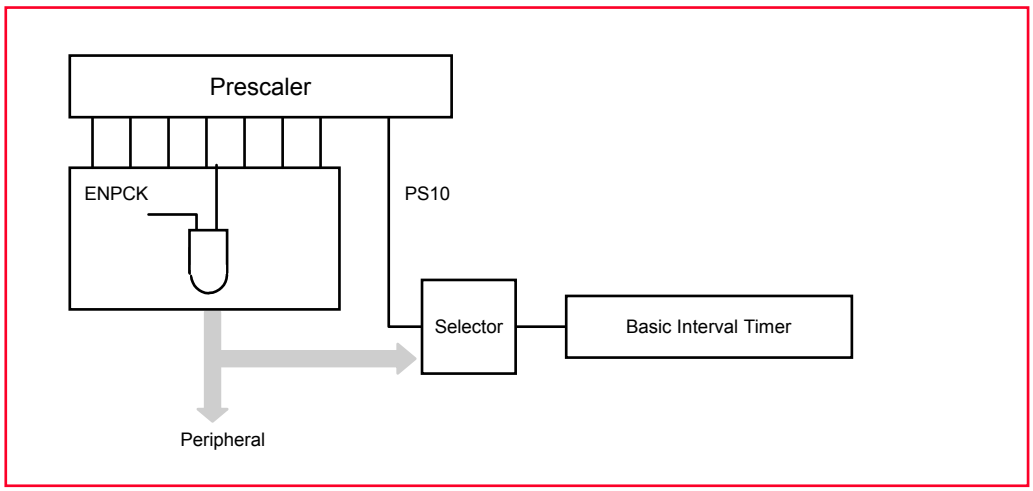


Figure 13-2 ENPCK and Basic Interval Timer Clock

13.3 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 ENPCK is set to ``1``.

Release Signal	SLEEP	STOP
$\overline{\text{RESET}}$	O	O
KSCN (key input)	O	O
INT1 , INT2	O	O
B.I.T	O	X

Table 13-1 Standby Mode Register

Release Factor	Release Method
$\overline{\text{RESET}}$	By $\overline{\text{RESET}}$ Pin = Low level, Standby mode is release 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 B.I.T is executed only by bit10 of prescaler (PS10), SLEEP mode can be release. Interrupt release SLEEP mode, when BIT interrupt enable flag is ``1``. When standby mode is released at interrupt. Mask enable flag = ``0``, program executes from the next instruction of SLEEP instruction. When ``1``, enters each interrupt service routine.

Table 13-2 Standby Mode Release

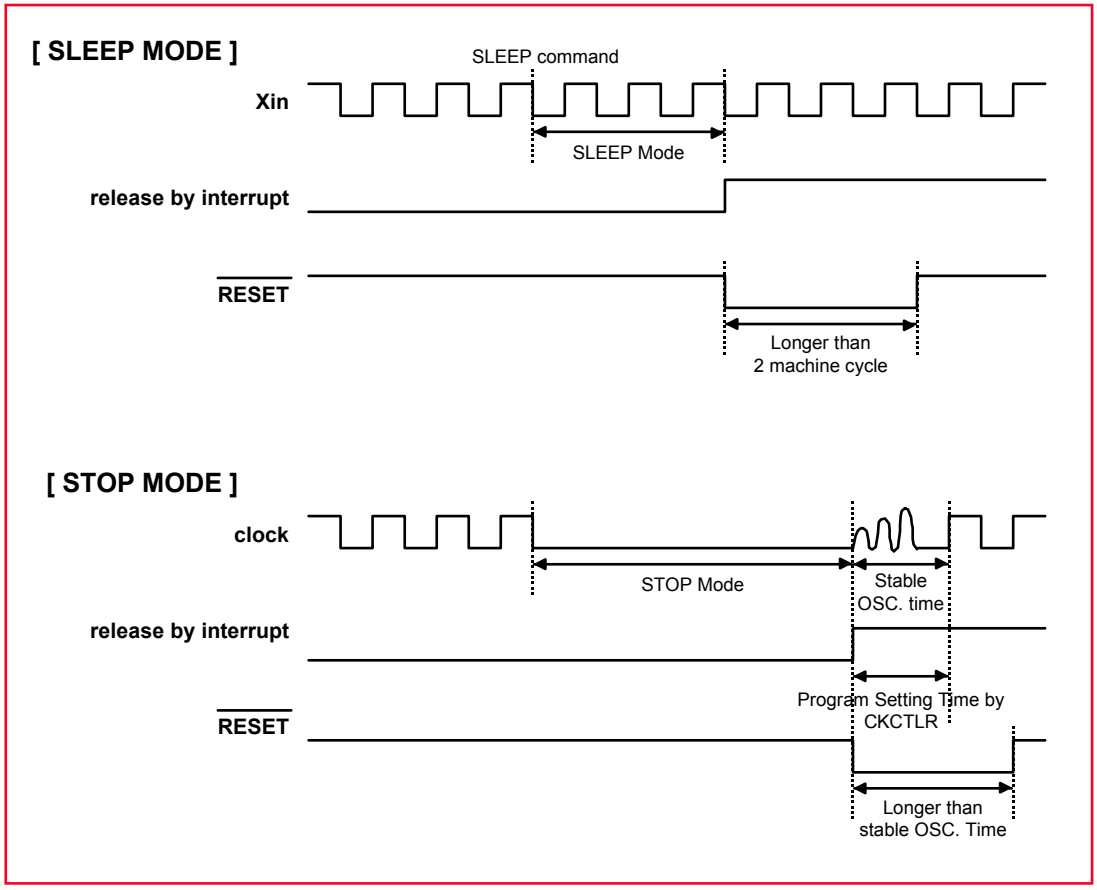


Figure 13-3 Release Timing of Standby Mode

13.4 Release Operation of Standby Mode

After standby mode is released, the operation begins according to the content of related interrupt register just before standby mode start (Figure 13-4)

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

Released by only an interrupt whose 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 jumps to the relevant interrupt service routine.

Note: *When STOP instruction is used, B.I.T should guarantee the stabilization oscillation time. Thus, just before STOP mode is entered, either clock of bit10 (PS10) of prescaler is selected or peripheral hardware clock control bit (ENPCK) is set to ``1``, Therefore the clock necessary for stabilization oscillation time should be input into B.I.T. 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.*

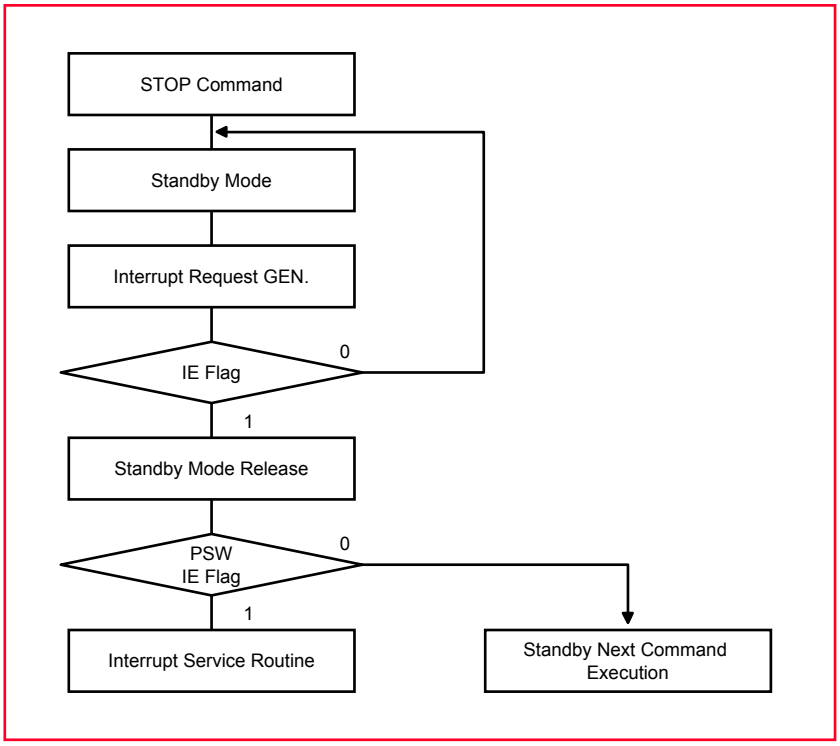


Figure 13-4 Standby Mode Release Flow

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

Table 13-1 Operation State in Standby Mode

14. OSCILLATION CIRCUIT

Oscillation circuit is designed to be used with either a ceramic resonator or crystal oscillator. Fig. 4.2-(a) 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 shown is Fig. 4.2.-(b). In the Standby (STOP) mode, oscillation stops, Xout state goes to ``High``, Xin state goes to ``Low``, and built-in feed back resistor is disabled.

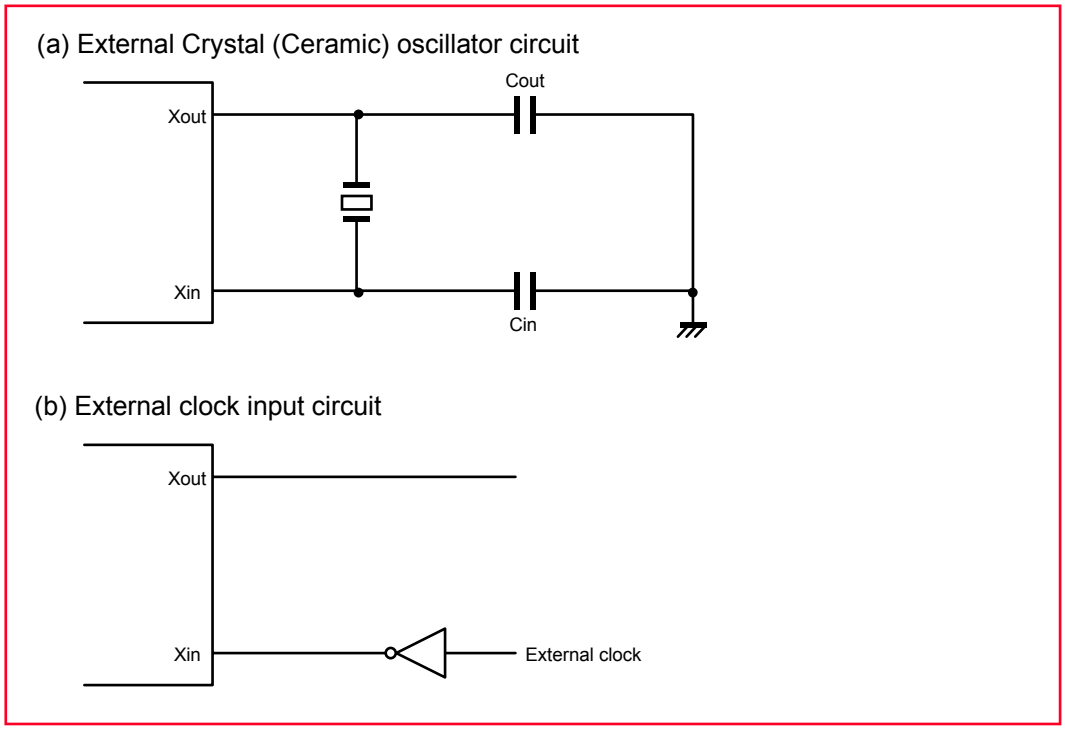


Figure 14-1 Oscillator Configurations

* Recommendable Resonator

Frequency	Resonator Maker	Part Name	Load Capacitor	Operating Voltage
4.0 MHz	CQ	ZTA4.00MG	Cin=Cout=30pF	2.2 ~ 4.0V
	TDK	FCR4.0MC5	Cin=Cout=open	2.2 ~ 4.0V
	TDK	FCR4.0M5	Cin=Cout=33pF	2.2 ~ 4.0V
	TDK	CCR4.0MC3		2.2 ~ 4.0V

* MC type is built in load capacitor. CCR type is chip type.

15. RESET FUNCTION

15.1 External Reset

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

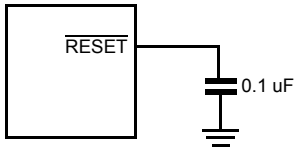


Figure 15-1 The Circuit of External RESET

15.2 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 219 oscillation cycles(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, B.I.T and B.I.T Overflow detection circuit.

(2) Once B.I.T Overflow detection circuit is reset, then, Prescaler starts to count.

(3) Prescaler output is input into B.I.T and PS10 of Prescaler output is automatically selected. If overflow of B.I.T is detected, Overflow detection circuit is set.

(4) Reset circuit generates maximum period of reset pulse from Prescaler and B.I.T.

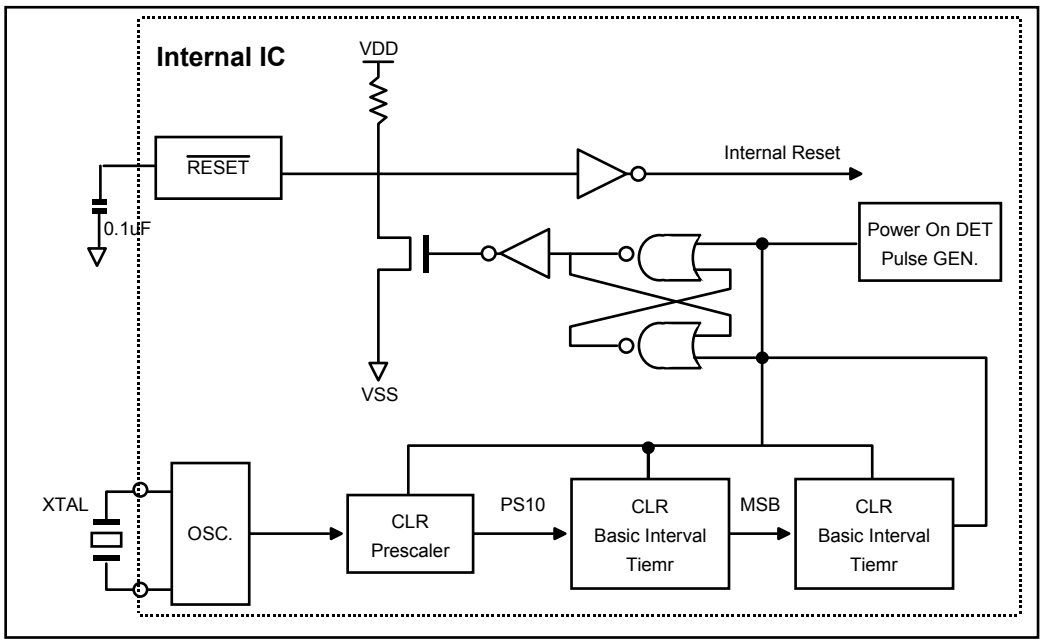


Figure 15-2 Block Diagram of Power On Reset Circuit

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

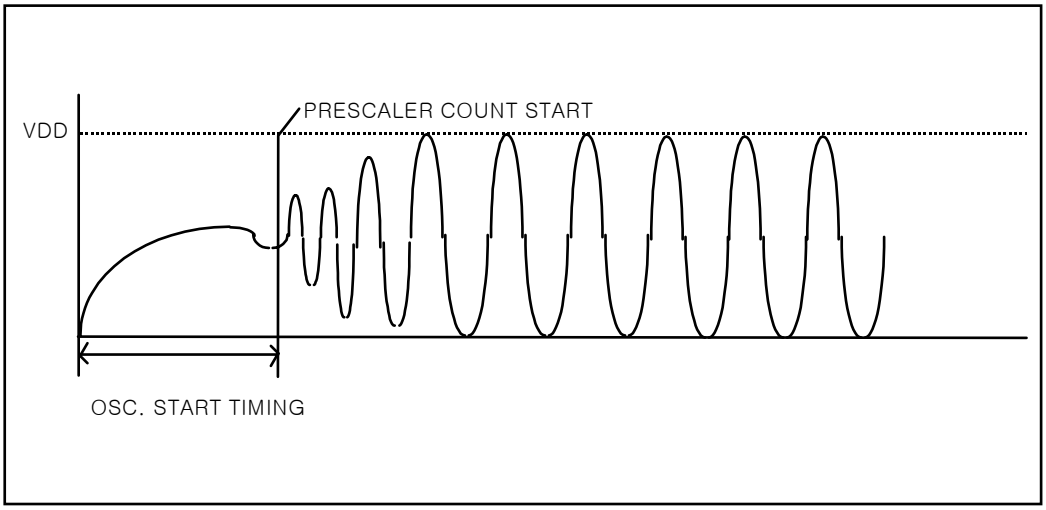


Figure 15-3 Oscillator Stabilization Diagram

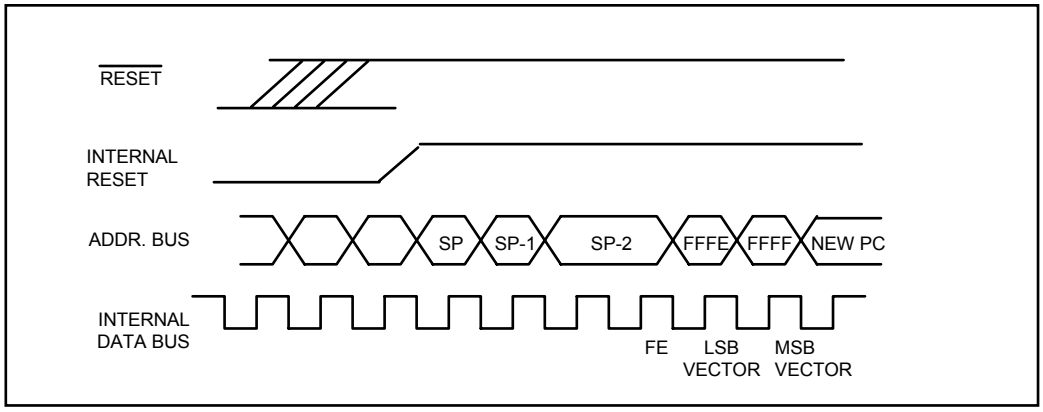


Figure 15-4 Reset Timing by Diagram

15.3 Low Voltage Detection Mode

(1) Low voltage detection condition

An on-board voltage comparator checks if VDD is at the required level to ensure correct operation of the device. If VDD is below a certain level, Low voltage detector forces the device into low voltage detection mode.

(2) Low Voltage Detection Mode

When 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 ports can be selected with Pull-up resistors by Mask option. If there is no information on the Mask option sheet , the default pull up option (all ports are connected to pull-up resistor) is selected. The Option data are equally applied to normal operation mode and LVD mode. In other words, if pull up options are checked 'Y' on the Mask option sheet, port status will be same on pull up in LVD mode and normal operation mode. And if pull up options are checked 'N' , port status will be same on no pull up in LVD mode and normal operation mode.

(3) Release of Low Voltage Detection Mode

Reset signal resulted from new battery(normally 3V) wakes the

low voltage detection mode and comes into normal reset state. It depends on user whether to execute RAM clear routine or not.

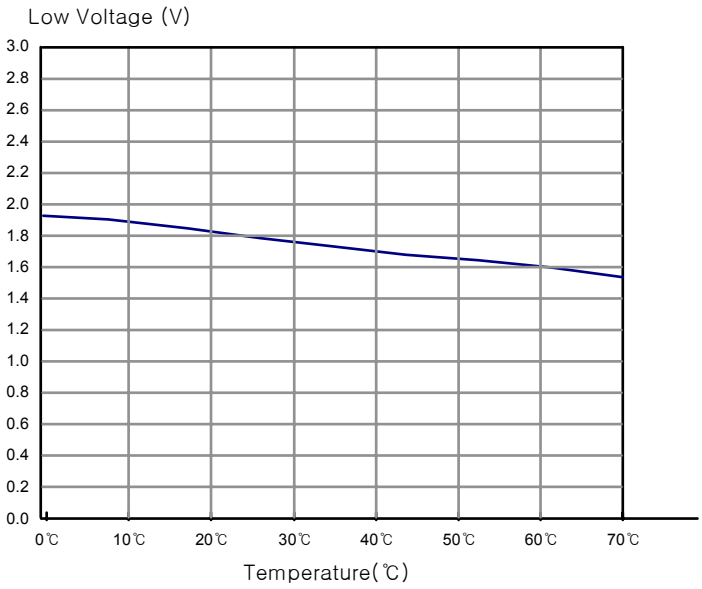


Figure 15-5 Low Voltage vs. Temperature

(4) SRAM BACK-UP after Low Voltage Detection

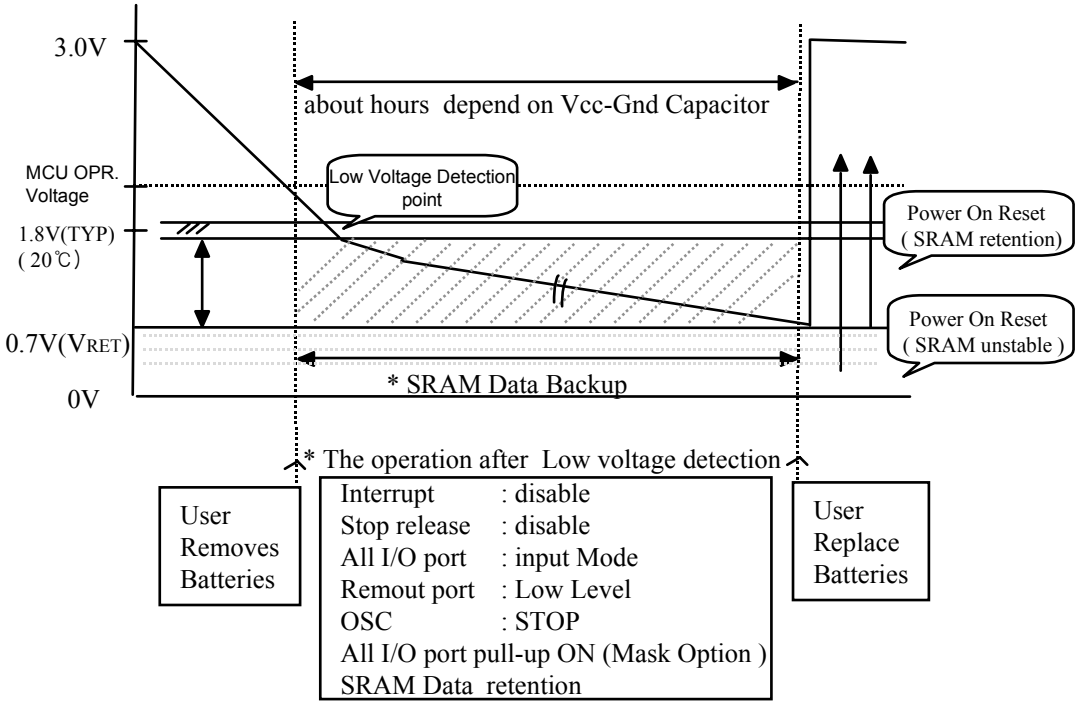


Figure 15-6 Low Voltage Detection and Protection

(5) S/W Flow Chart Example after Reset using SRAM Back-up

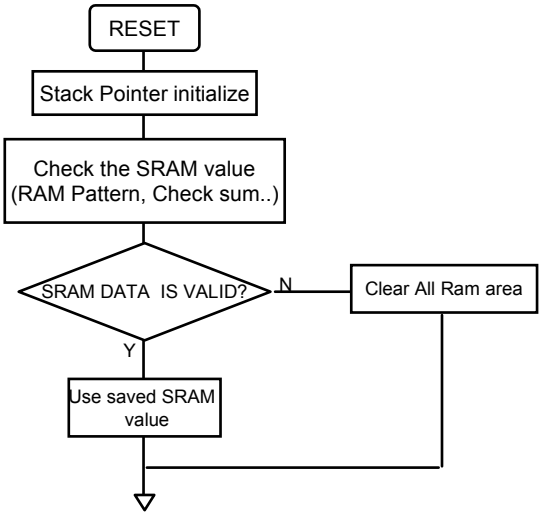


Figure 15-7 S/W Flow Chart Example for SRAM Back-up

15.4 Low Voltage Indicator Register (LVIR)

Low Voltage Indication Register (LVIR) is read only Register. It is useful to display the consumption of Batteries. If VDD power level is below a certain level which is higher than low voltage detection level (refer to Figure 15-6), the LVIR register bit should be set according to the VDD level sequentially. The VDD detection levels for Indication are two, Bit1 and Bit0 of LVIR Register. The detection level of Bit0 is higher than Bit1.

bit	7	6	5	4	3	2	1	0	
LVIR	-	-	-	-	-	-	LVIR1	LVIR0	<00EF h>
initial value	-	-	-	-	-	-	0	0	
R / W	-	-	-	-	-	-	R	R	