

8-BIT SINGLE CHIP MICROCONTROLLERS

GMS81C5016
GMS81C5024
GMS81C5032

USER'S MANUAL

Jun. 2001

Ver 3.02



CONTENTS

| | |
|--|-----------|
| 1. OVERVIEW | 1 |
| 1.1 Description | 1 |
| 1.2 Features | 1 |
| 1.3 Development Tools | 1 |
| 2. BLOCK DIAGRAM | 3 |
| 3. PIN ASSIGNMENT (Top View) | 4 |
| 4. PACKAGE DIMENSION | 7 |
| 4.1 28 SOP PIN DIMENSION (DIMENSIONS IN INCH) | 7 |
| 4.2 28 Skinny DIP PIN DIMENSION (DIMENSIONS IN INCH) | 7 |
| 4.3 40 PDIP Pin Dimension (dimension in inch) | 8 |
| 4.4 44 PLCC Pin Dimension (dimension in mm) | 8 |
| 4.5 44 QFP Pin Dimension (dimension in mm) | 9 |
| 5. PIN FUNCTION | 10 |
| 6. PORT STRUCTURES | 12 |
| 6.1 R0 Ports | 12 |
| 6.2 R1 Ports (R10, R11, R12, R13, R14) | 12 |
| 6.3 R1 Ports (R15, R16, R17) | 13 |
| 6.4 R2, R3, R4 Ports | 13 |
| 6.5 REMOUT Port | 14 |
| 6.6 Xin, Xout Ports | 14 |
| 6.7 RESET Port | 14 |
| 6.8 TEST Port | 15 |
| 7. ELECTRICAL CHARACTERISTICS | 16 |
| 7.1 Absolute maximum ratings (Ta=25 'C) | 16 |
| 7.2 Recommended Operating Ranges | 16 |
| 7.3 DC characteristics (VDD=2.2~5.5, Vss=0, Ta=0~70 'C) | 17 |
| 7.4 REMOUT Port Ioh characteristics graph | 18 |
| 7.5 REMOUT port Iol characteristics graph | 18 |
| 7.6 AC characteristics (VDD=2.2~5.5V, Vss=0V, Ta=0~70'C) | 19 |
| 8. MEMORY ORGANIZATION | 21 |
| 8.1 Registers | 21 |
| 8.2 Program Memory | 24 |
| 8.3 Data Memory | 27 |
| 8.4 Addressing Mode | 30 |

| | |
|--|-----------|
| 9. I/O PORTS | 34 |
| 9.1 R0 Ports | 34 |
| 9.2 R1 Ports | 34 |
| 9.3 R2 Port | 36 |
| 10. CLOCK GENERATOR | 39 |
| 10.1 Operation Mode | 41 |
| 11. TIMER | 42 |
| 11.1 Basic Interval Timer | 42 |
| 11.2 Timer0, Timer1, Timer2 | 44 |
| 12. INTERRUPTS | 57 |
| 12.1 Interrupt priority and sources. | 57 |
| 12.2 INTERRUPT CONTROL REGISTER | 58 |
| 12.3 INTERRUPT ACCEPT MODE | 59 |
| 12.4 INTERRUPT PROCESSING SEQUENCE | 60 |
| 12.5 SOFTWARE INTERRUPT (Interrupt by Break (BRK) Instruction) | 61 |
| 12.6 MULTIPLE INTERRUPT | 62 |
| 12.7 Key Scan Input Processing | 62 |
| 13. WATCH DOG TIMER | 66 |
| 13.1 Control of WDT | 66 |
| 13.2 WDT Interrupt Interval | 67 |
| 14. STANDBY FUNCTION | 69 |
| 14.1 Sleep Mode | 69 |
| 14.2 STOP MODE | 69 |
| 14.3 STANDBY MODE RELEASE | 71 |
| 14.4 RELEASE OPERATION OF STANDBY MODE | 72 |
| 15. OSCILLATION CIRCUIT | 74 |
| 16. RESET FUNCTION | 75 |
| 16.1 EXTERNAL RESET | 75 |
| 16.2 POWER ON RESET | 75 |
| 16.3 Low Voltage Detection Mode | 76 |
| 16.4 Low Voltage Indicator Register (LVIR) | 79 |

GMS81C5016

GMS81C5024

GMS81C5032

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

1. OVERVIEW

1.1 Description

The GMS81C5016/24/32 is an advanced CMOS 8-bit microcontroller with 16K/24K/32K bytes of ROM. The device is one of Hynix 800 family. The Hynix GMS81C5016/24/32 is a powerful microcontroller which provides a highly flexible and cost effective solution to many UR & Keyboard applications. The GMS81C5016/24/32 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 GMS81C5016/24/32 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**

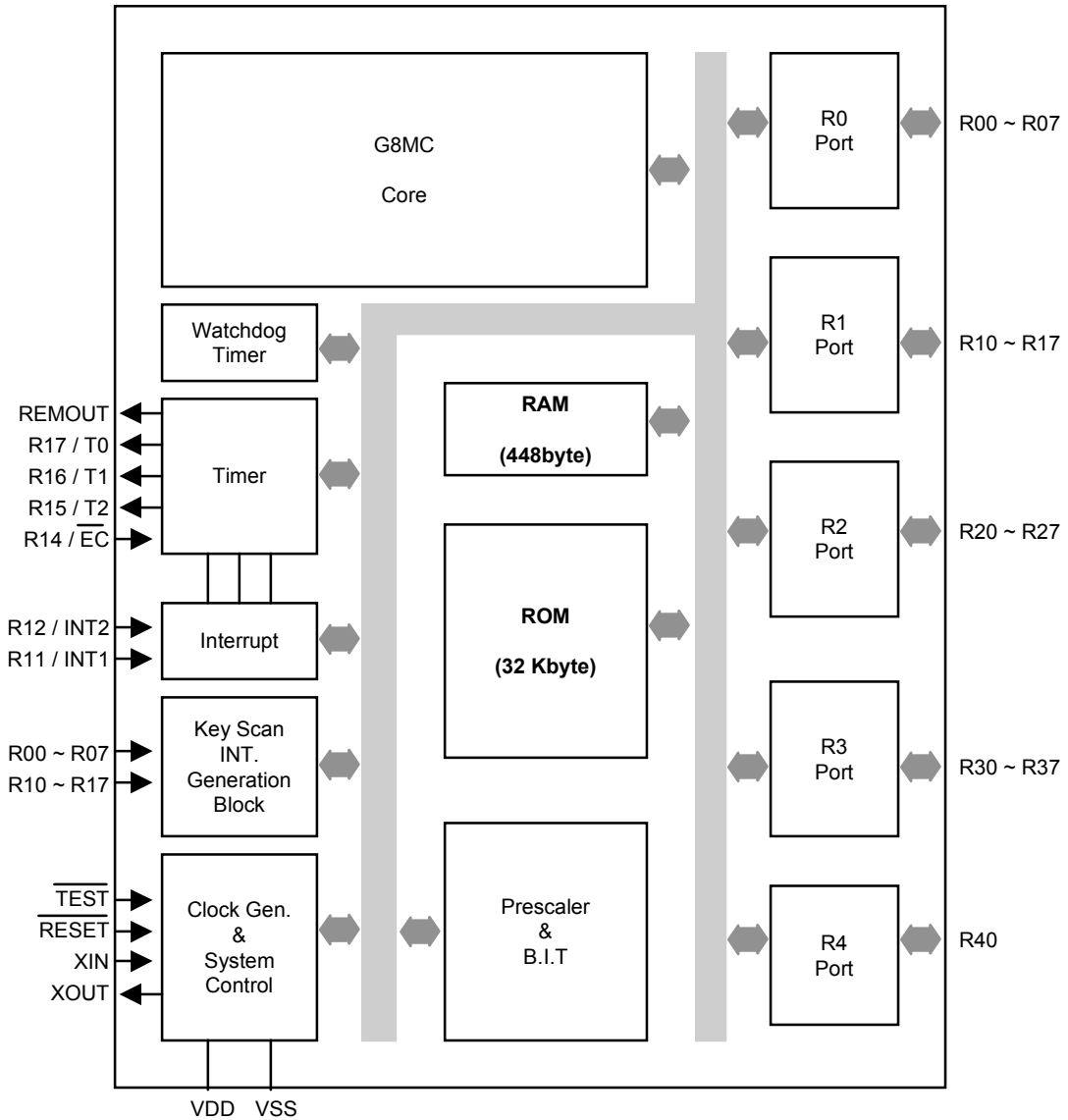
| | 28 PIN | 40 PIN | 44 PIN |
|--------|--------|--------|--------|
| INPUT | 3 | 3 | 3 |
| OUTPUT | 2 | 2 | 2 |
| I/O | 21 | 33 | 33 |
- **Operating Voltage**
 - 2.2 ~ 5.5 V @ 4MHz
 - 2.4 ~ 5.5 V @ 4MHz (OTP GMS87C50XX)
- **Timer**
 - Timer / Counter 16 Bit * 1 ch
 - 8 Bit * 2 ch
- **Basic Interval Timer 8 Bit * 1 ch**
- **Watch Dog Timer 6Bit * 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 1second after Power on Reset)**

1.3 Development Tools

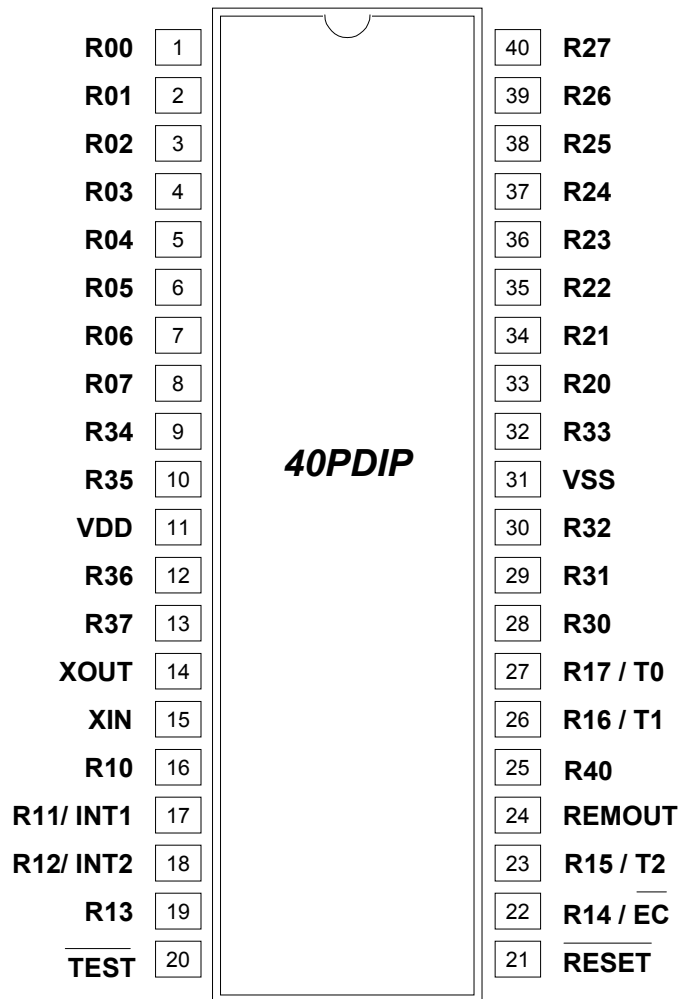
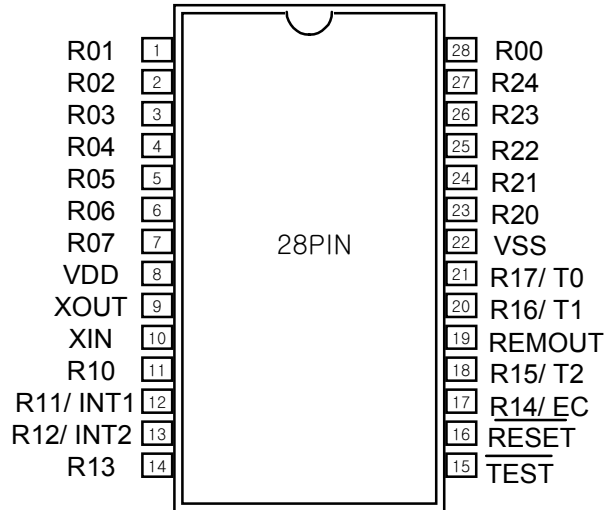
The GMS81C5016/24/32 is supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr™.

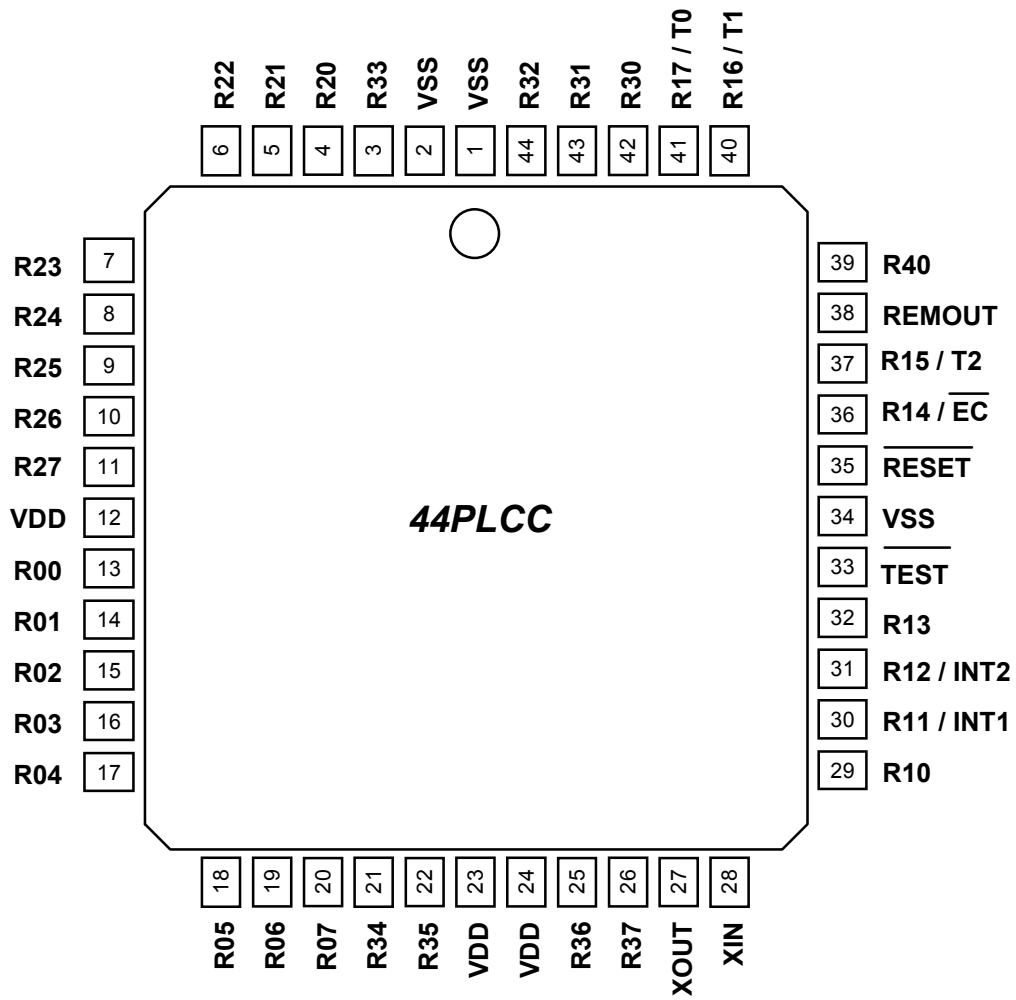
| | |
|-----------------------------|--------------------------|
| In Circuit Emulators | CHOICE-Dr. (with EVA81C) |
| Assembler | Hynix Macro Assembler |

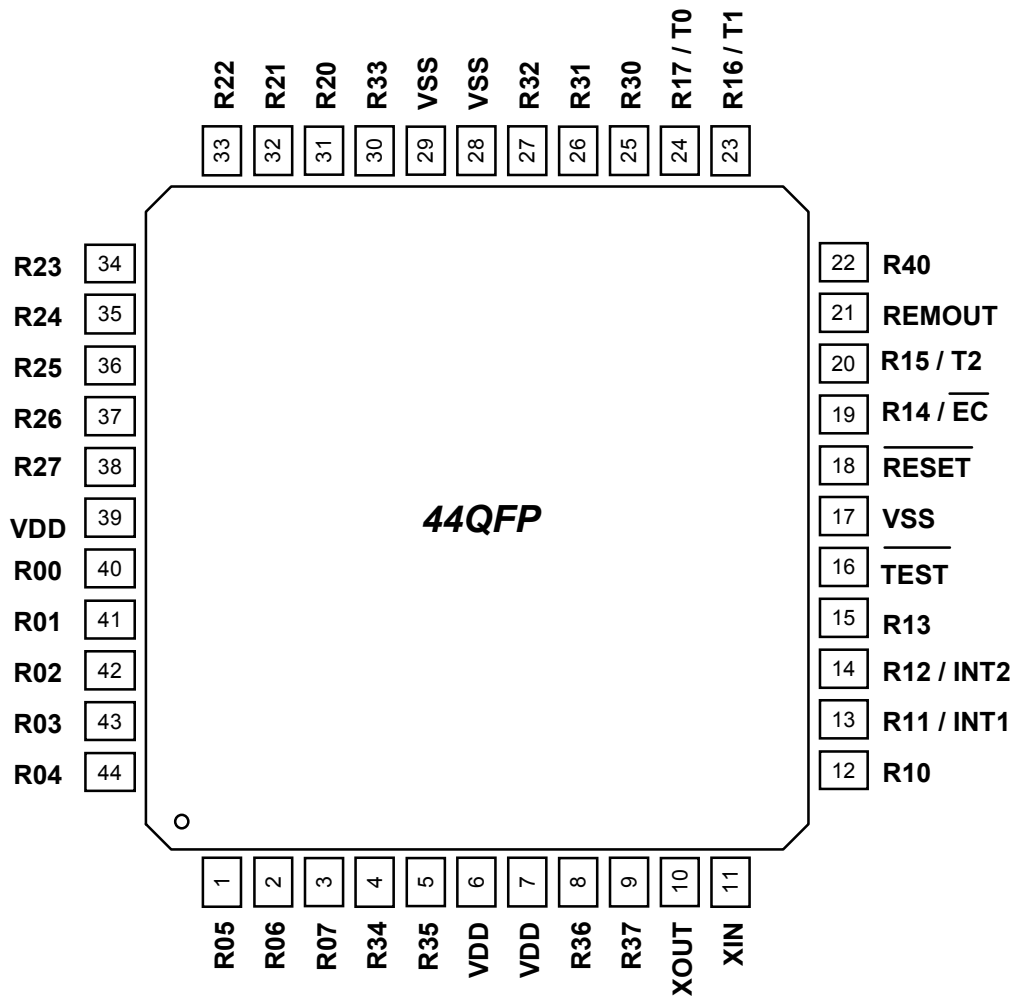
2. BLOCK DIAGRAM



3. PIN ASSIGNMENT (Top View)

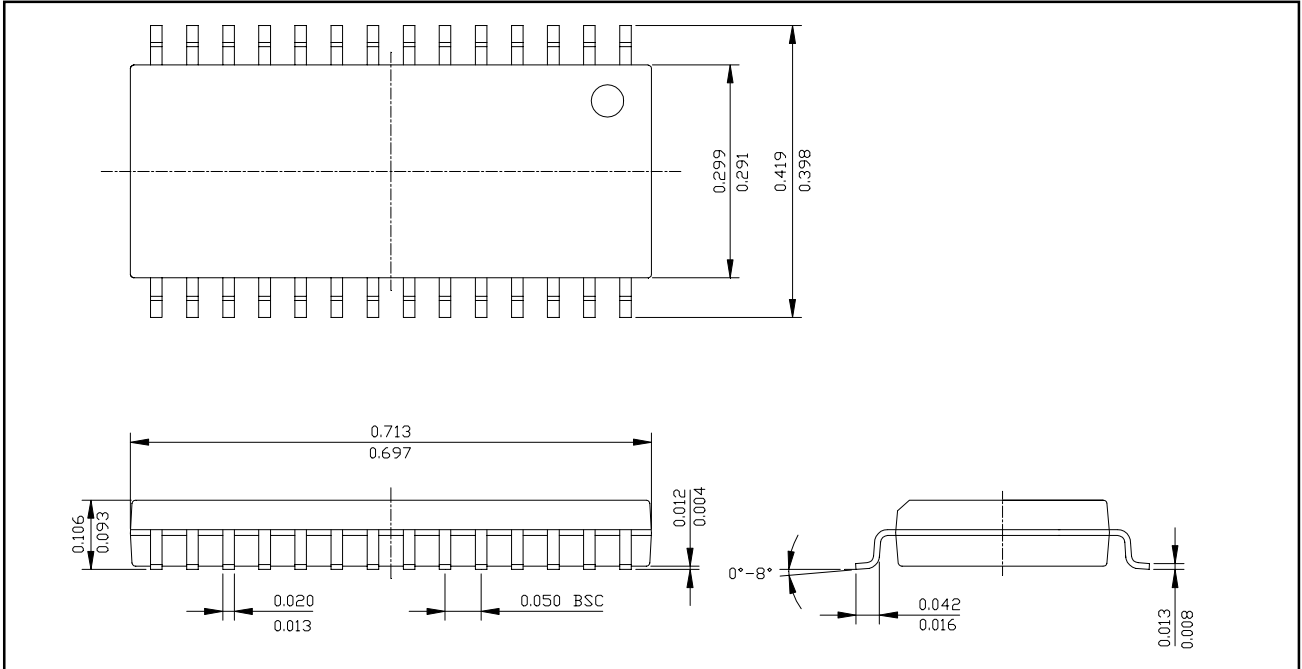




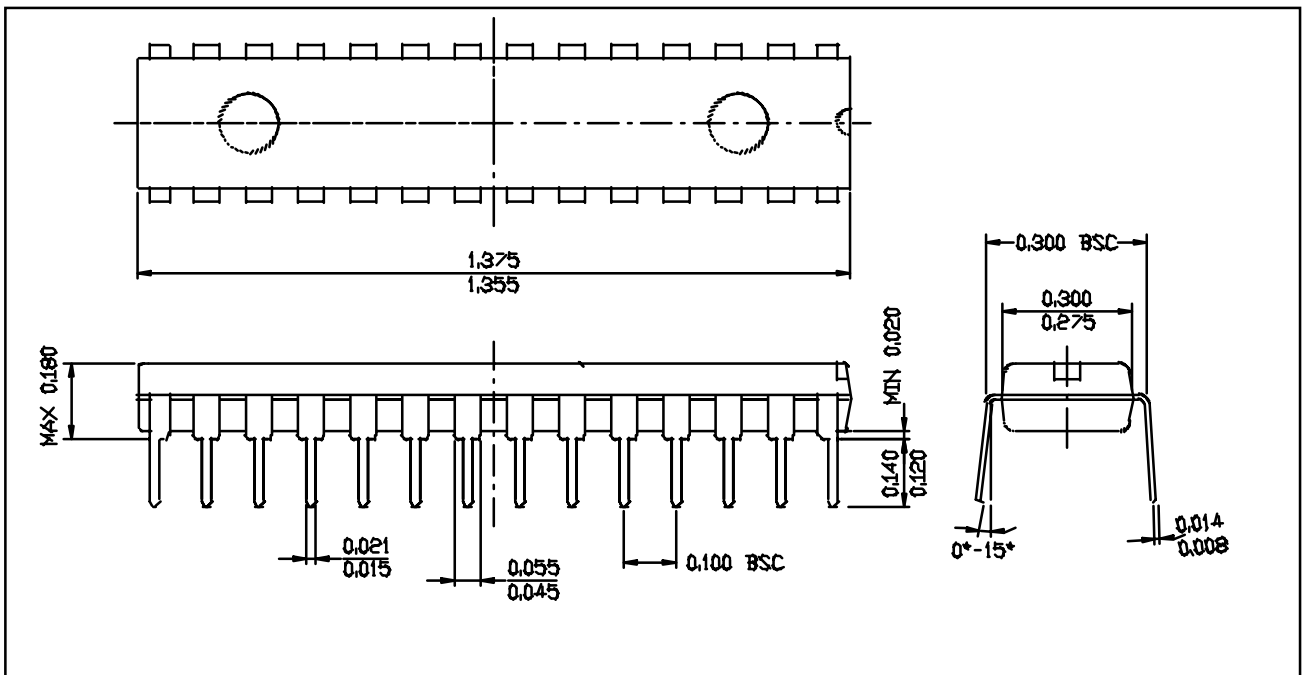


4. PACKAGE DIMENSION

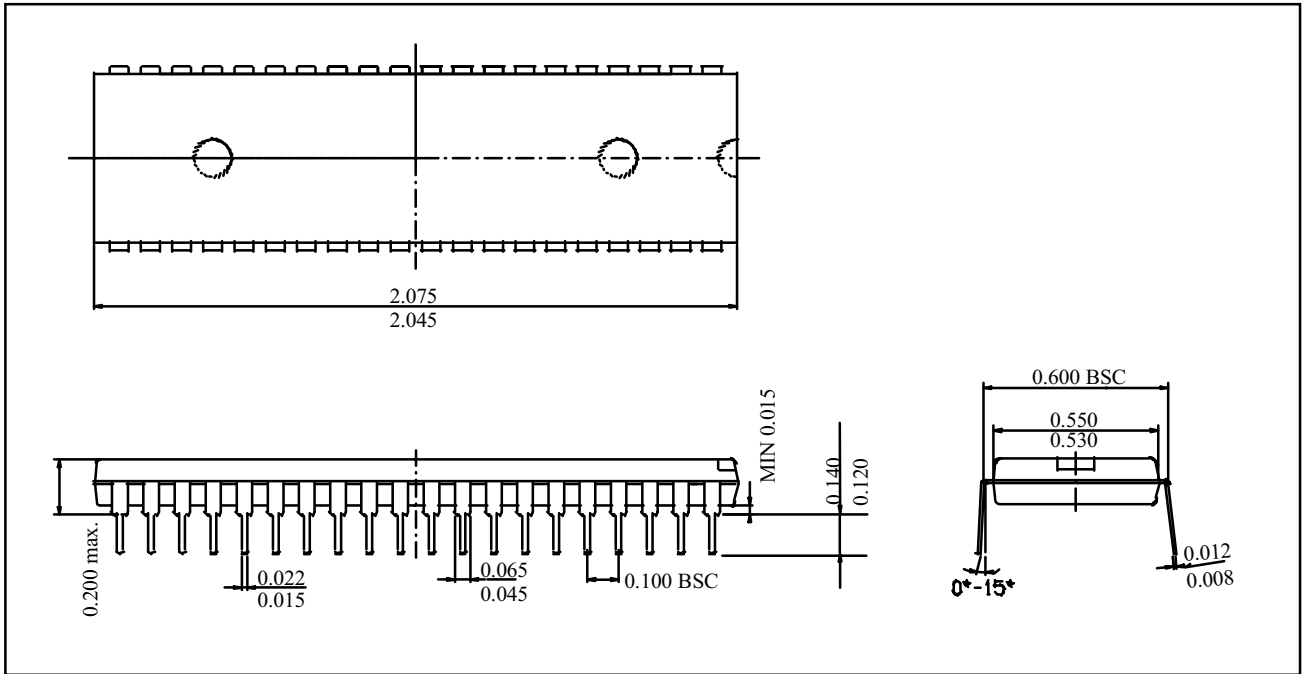
4.1 28 SOP PIN DIMENSION (DIMENSIONS IN INCH)



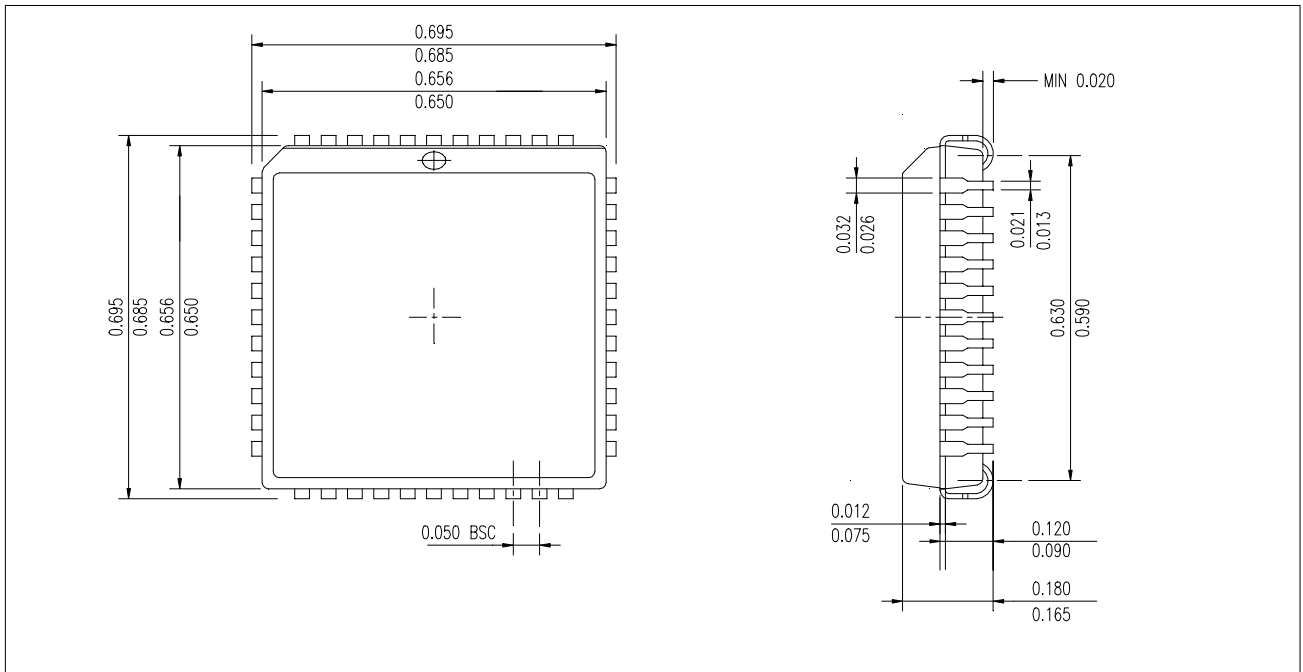
4.2 28 Skinny DIP PIN DIMENSION (DIMENSIONS IN INCH)



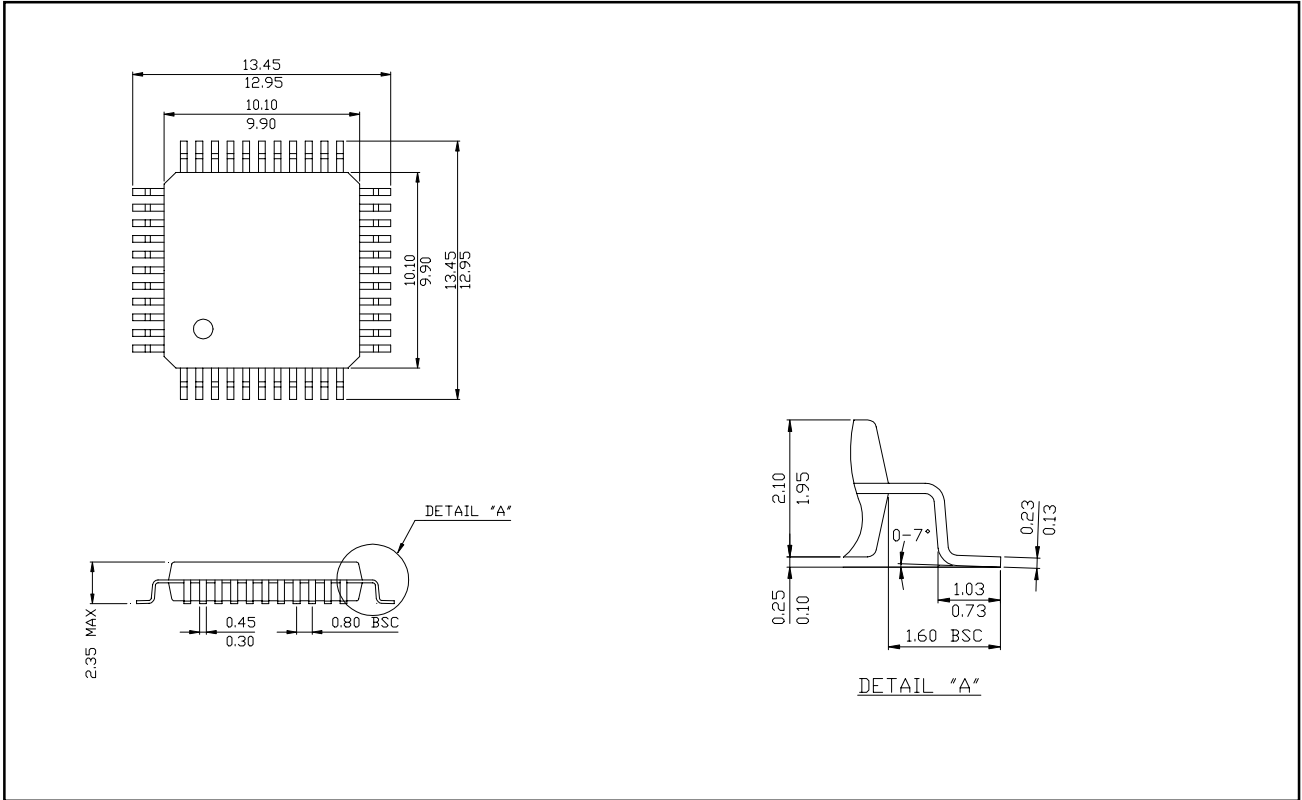
4.3 40 PDIP Pin Dimension (dimension in inch)



4.4 44 PLCC Pin Dimension (dimension in mm)



4.5 44 QFP Pin Dimension (dimension in mm)



5. PIN FUNCTION

V_{DD}: Supply voltage.

V_{SS}: Circuit ground.

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

RESET: Reset the MCU.

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

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

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

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

In addition, R1 serves the functions of the various follow-

ing 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) |

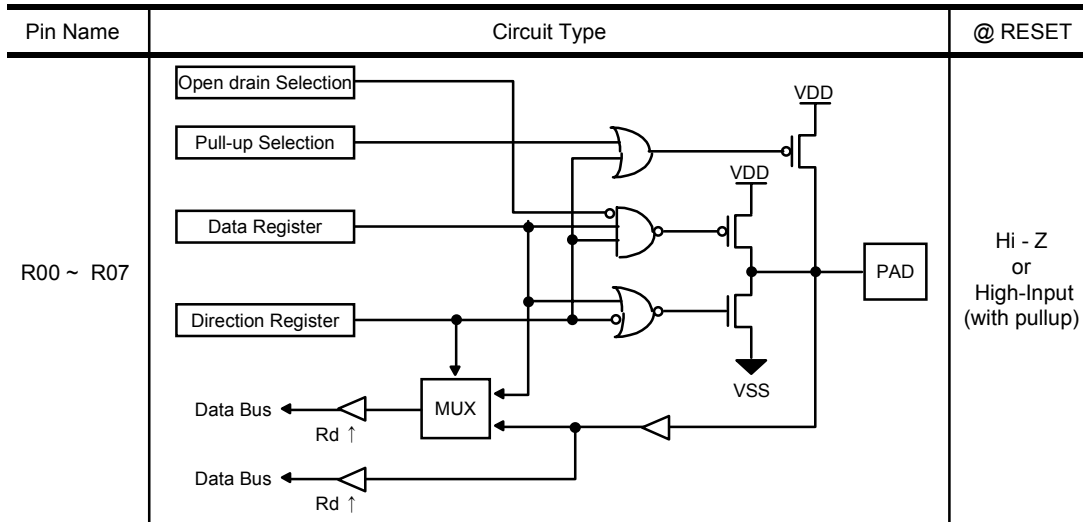
R20~R22, R30~R37 : R2 & R3 is a 8-bit CMOS bidirectional I/O port. Each pins 1 or 0 written to the their Port Direction Register can be used as outputs or inputs.

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 outputs or inputs.

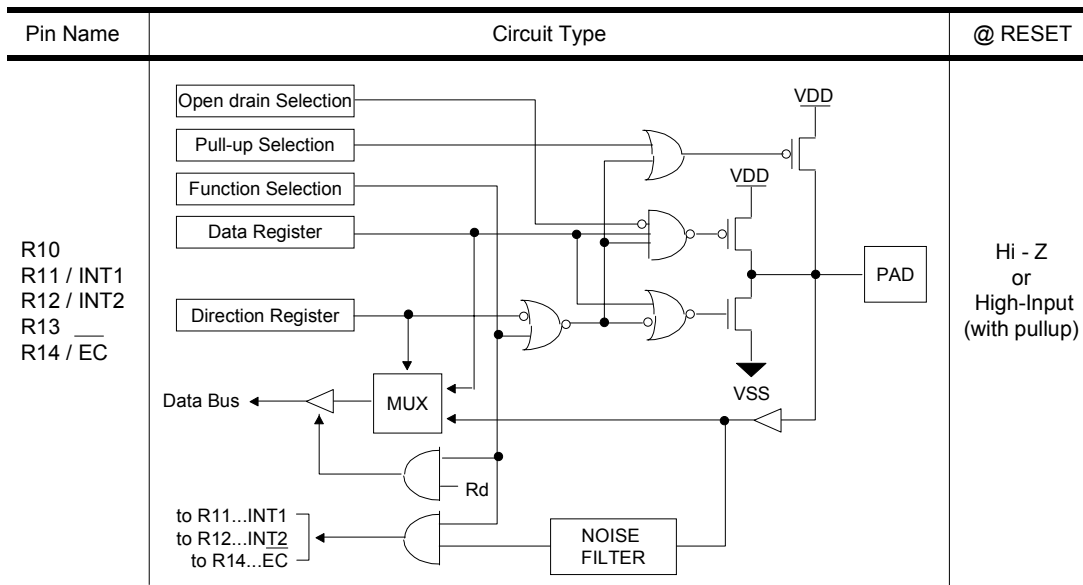
| PIN NAME | INPUT/ OUTPUT | Pin Numbers | | | | Function | @ RESET | @ STOP |
|----------|------------------|-------------|--------|----------|----------|--|------------|----------------------|
| | | 28Pin | 40PDIP | 44PLCC | 44QFP | | | |
| R00 | I/O | 28 | 1 | 13 | 41 | <ul style="list-style-type: none"> - 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 | | |
| 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 | | | |
| 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 |
| RESET | I | 16 | 21 | 35 | 18 | - Includes pull-up resistor | 'L' level | state of before STOP |
| 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 | | |

6. PORT STRUCTURES

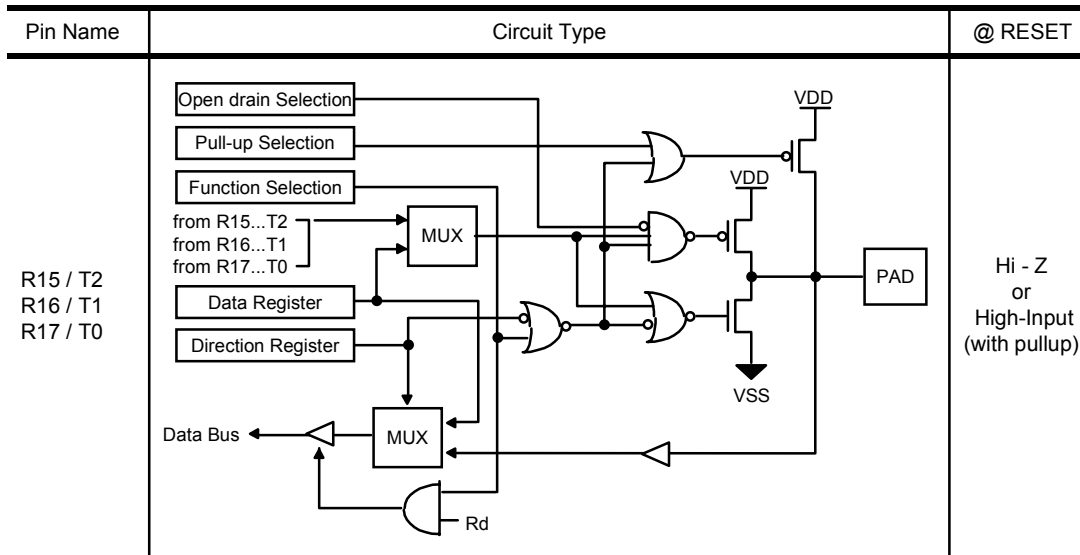
6.1 R0 Ports



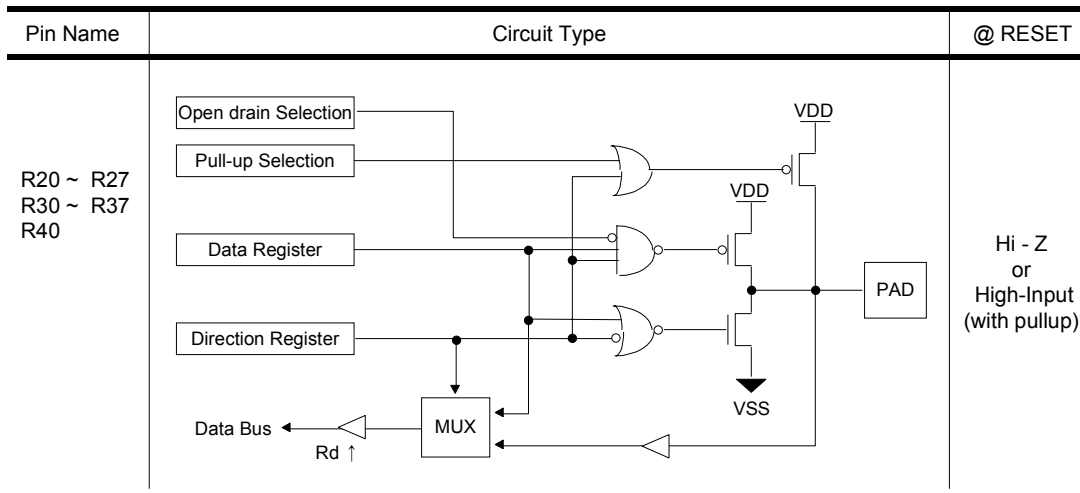
6.2 R1 Ports (R10, R11, R12, R13, R14)



6.3 R1 Ports (R15, R16, R17)



6.4 R2, R3, R4 Ports



6.5 REMOUT Port

| Pin Name | Circuit Type | @ RESET |
|----------|--------------|-----------|
| REMOUT | | Low level |

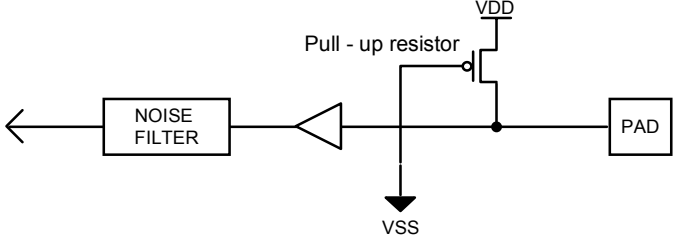
6.6 Xin, Xout Ports

| Pin Name | Circuit Type | @ RESET |
|-------------|--------------|-------------|
| Xin Xout | | oscillation |

6.7 RESET Port

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

6.8 TEST Port

| Pin Name | Circuit Type | @ RESET |
|--|--|-------------------|
| <p>$\overline{\text{TEST}}$</p> |  | <p>High level</p> |

7. ELECTRICAL CHARACTERISTICS

7.1 Absolute maximum ratings (Ta=25 'C)

| 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 | °C |
| Storage Temperature | Tstg | -65 ~ 150 | °C |
| 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 functional operation of the device at any other conditions above those indicated in

the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability

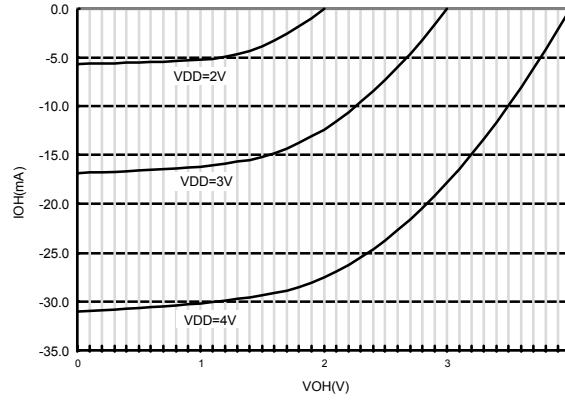
7.2 Recommended Operating Ranges

| 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 | °C |

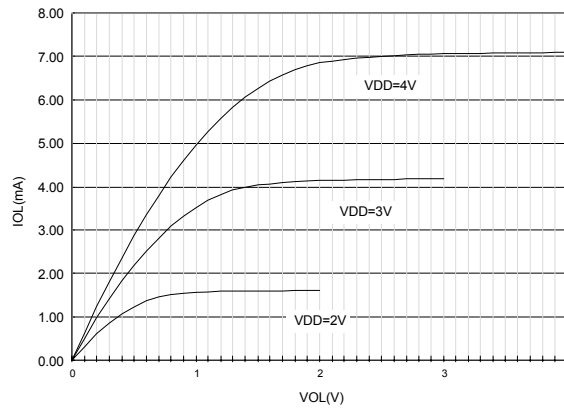
7.3 DC characteristics (VDD=2.2~5.5, Vss=0, Ta=0~70 'C)

| Parameter | Symbol | Condition | | Specification | | | Unit | |
|-----------------------------------|--------------------|---|----------------------------------|------------------------|------|--------------------|------|----|
| | | | | min | typ | max | | |
| high level input voltage | V _{IH1} | R11, R12, R14, 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, 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 , RESET | V _{IH} =V _{DD} | | | 1 | uA | |
| low level input leakage current | I _{IL} | R0 ~ R4 ,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} | 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 | V _{RET} | | | 0.7 | | | V | |

7.4 REMOUT Port Ioh characteristics graph



7.5 REMOUT port Iol characteristics graph



7.6 AC characteristics (VDD=2.2~5.5V, Vss=0V, Ta=0~70°C)

| 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)

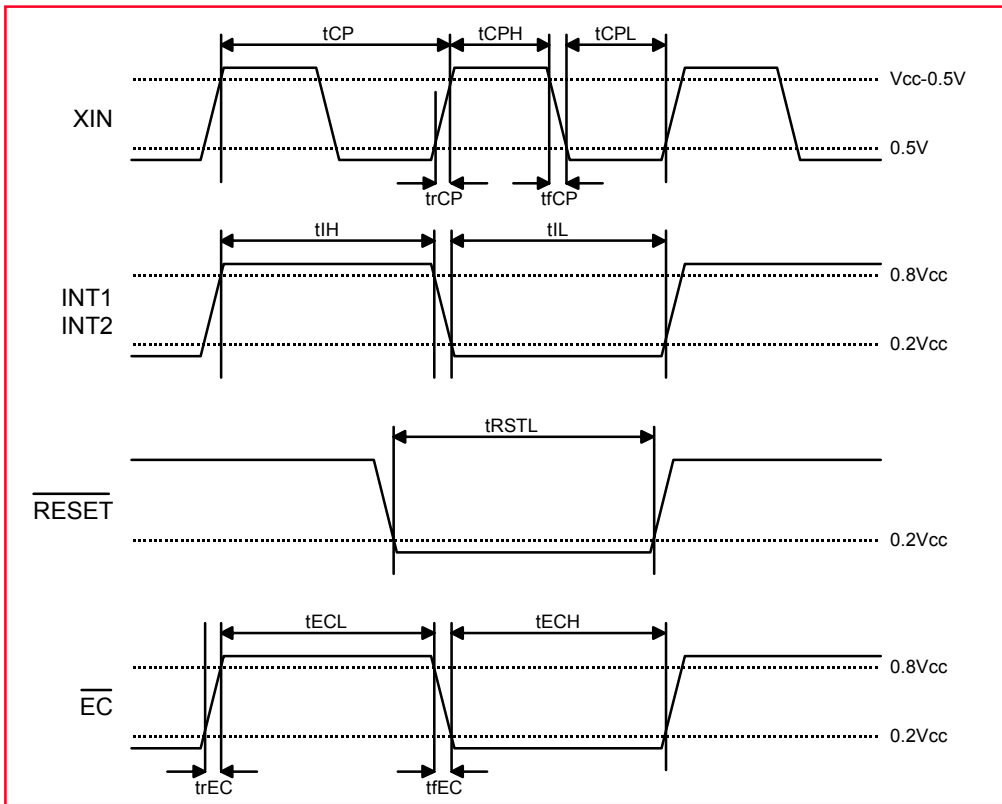


Figure 7-1 Clock, Interrupt, \overline{RESET} , \overline{EC} Input Timing

8. MEMORY ORGANIZATION

The GMS81C5016/24/32 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 to up to 448 bytes including the stack area.

8.1 Registers

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

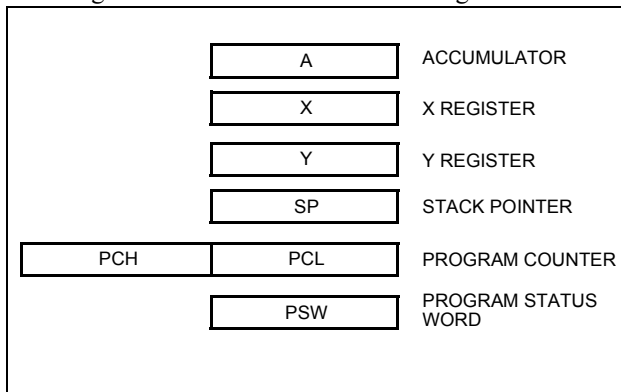


Figure 8-1 Configuration of Registers

Accumulator:

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

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

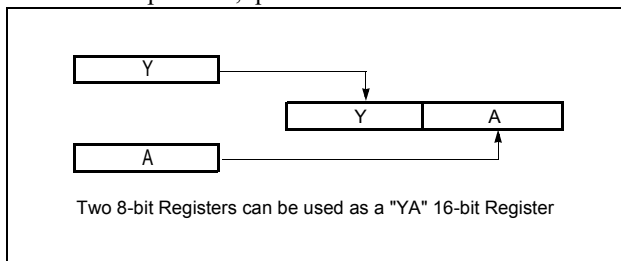


Figure 8-2 Configuration of YA 16-bit Register

X, Y Registers:

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

* X Register : In the case of division instruction, execute as register.

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

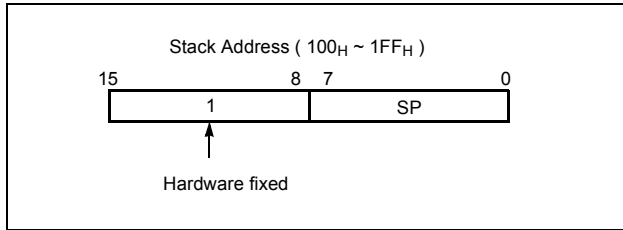
Stack Pointer:

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

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

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

used.



Caution:

The Stack Pointer must be initialized by software because its value is undefined after RESET.

Example: To initialize the SP

```
LDX    #0FFH
TXSP                      ; SP ← FFH
```

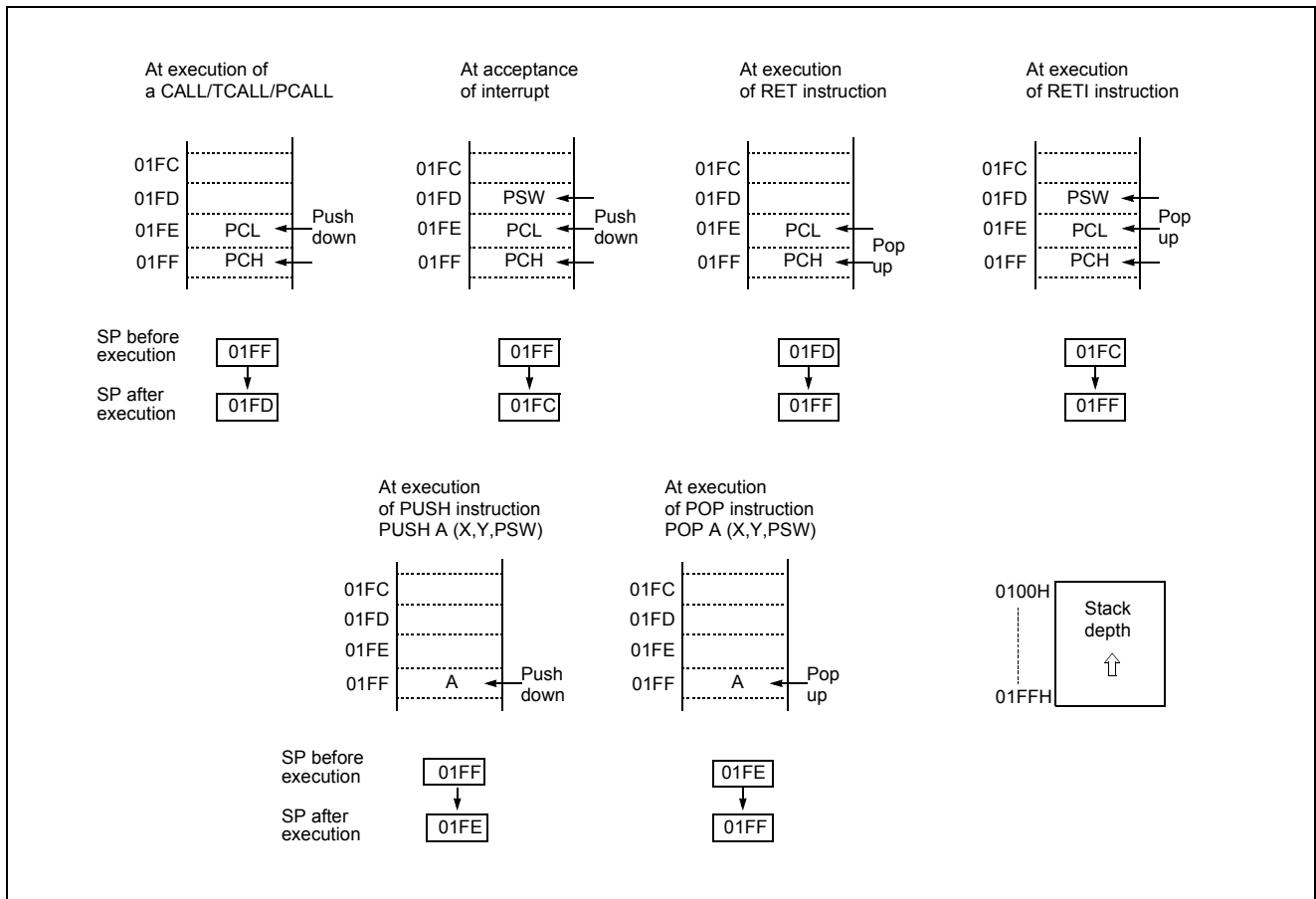


Figure 8-3 Stack Operation

Program Counter:

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

Program Status Word:

The Program Status Word (PSW) contains several bits that

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

[Carry flag C]

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

[Zero flag Z]

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

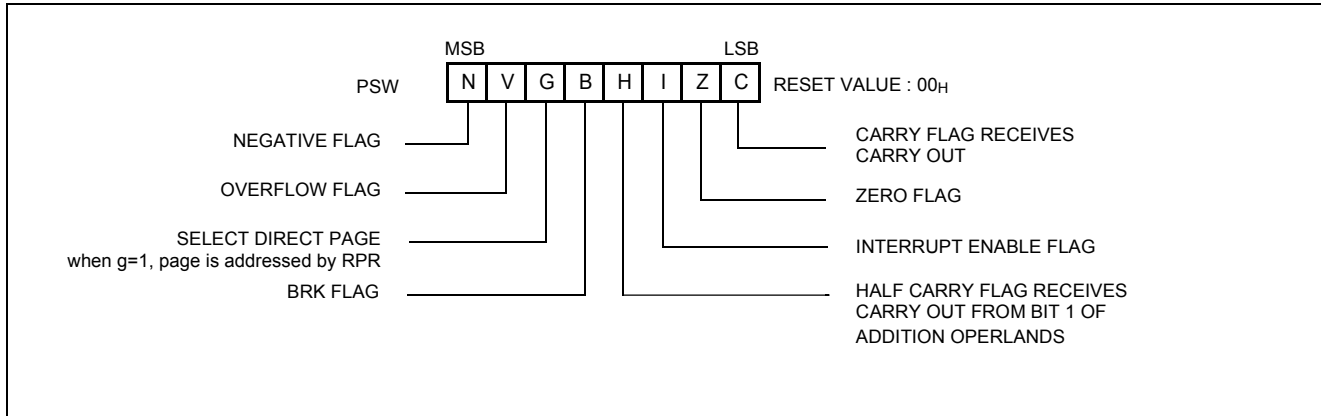


Figure 8-4 PSW (Program Status Word) Register

[Interrupt disable flag I]

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

[Half carry flag H]

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

[Break flag B]

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

[Direct page flag G]

This flag assigns RAM page for direct addressing mode. In

the direct addressing mode, addressing area is from zero page 00H to 0FFH 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(7FH) or -128(80H). The CLRV instruction clears the overflow flag. There is no set instruction. When the BIT instruction is executed, bit 6 of memory is copied to this flag.

[Negative flag N]

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

8.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 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 8-5 , shows a map of Program Memory. After reset, the CPU begins execution from reset vector which is stored in address FFFE_H and FFFF_H as shown in Figure 8-6 .

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

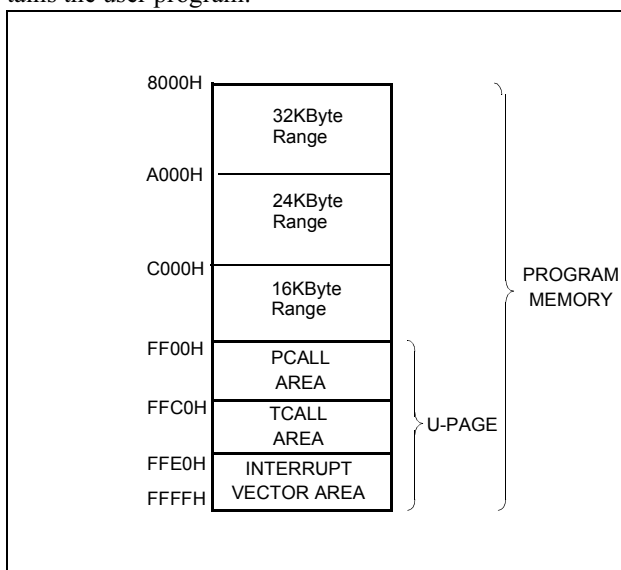


Figure 8-5 Program Memory Map

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

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

Example: Usage of TCALL

```

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

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

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

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

NOTE:
 "-" means reserved area.

Figure 8-6 Interrupt Vector Area

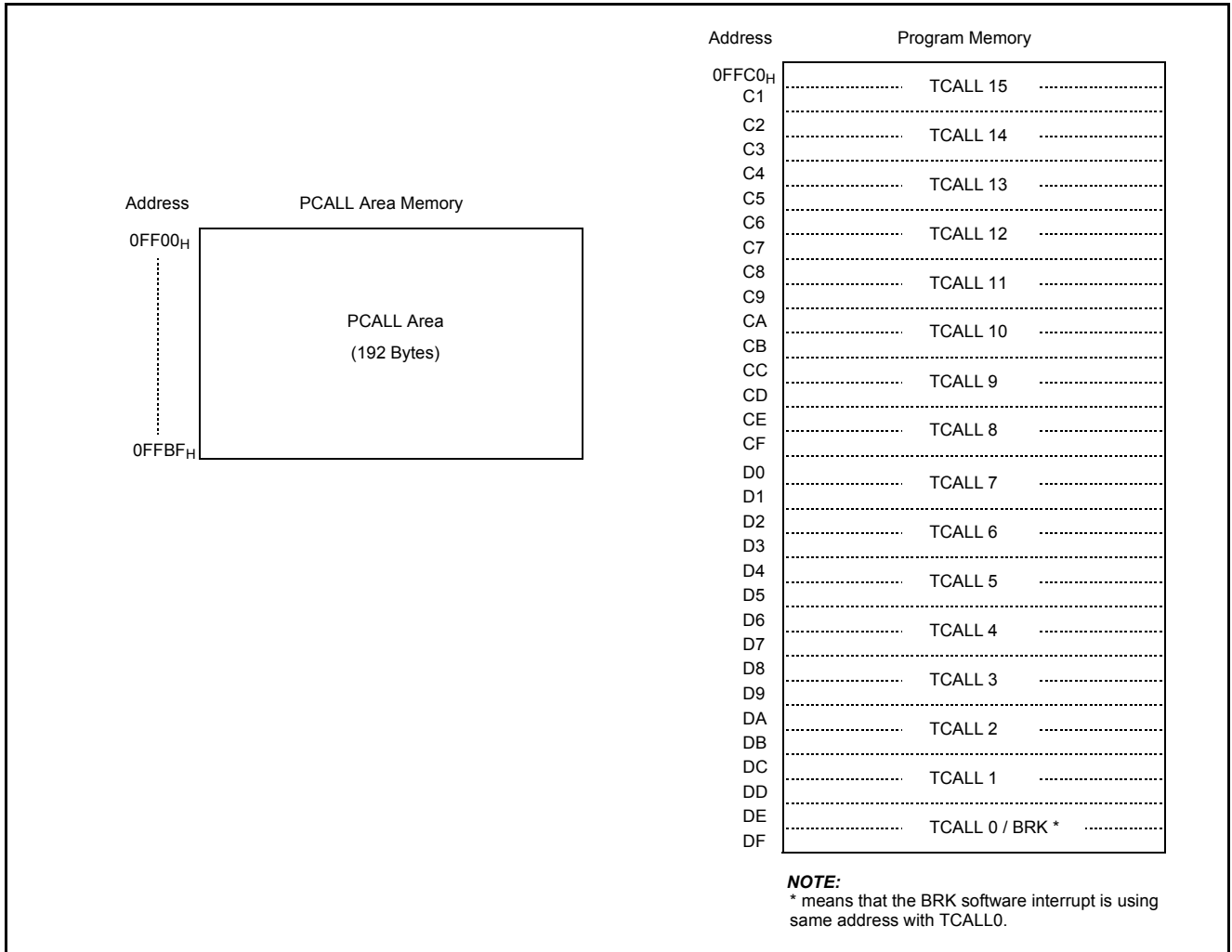
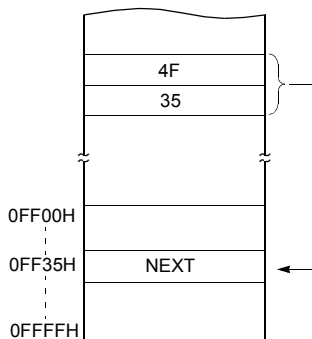


Figure 8-7 PCALL and TCALL Memory Area

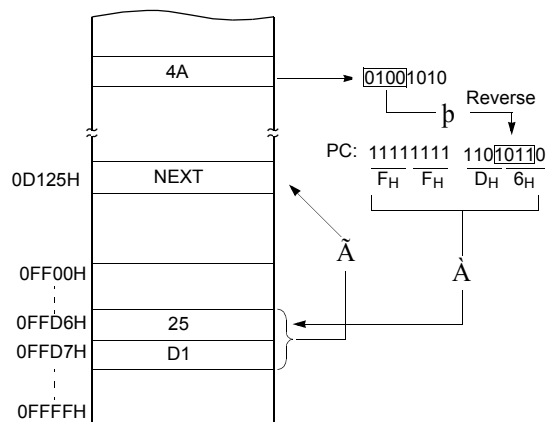
PCALL → rel

4F35 PCALL 35H



TCALL → n

4A TCALL 4



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

```

ORG      0FFE0H

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

ORG      08000H

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

          LDM              R0, #0        ;Normal Port 0
          LDM              R0DD,#1000_0010B ;Normal Port Direction
          LDM              PUR0,#1000_0010B ;Pull Up Selection Set
          LDM              PMR0,#0000_0001B ;R0 port / int
          :
          :
          LDM              PCOR,#1       ;Enable Peripheral clock
          :
          :

```

8.3 Data Memory

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

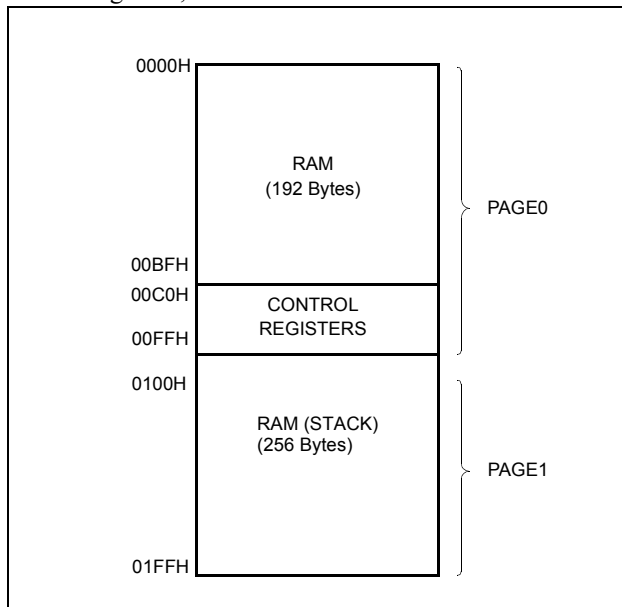


Figure 8-8 Data Memory Map

User Memory

The GMS81C5016/24/32 has 448×8 bits for the user memory (RAM).

Control Registers

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

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

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

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

Example; To write at CKCTRL

```
LDM CLCTRL,#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 8-3 on page 22.

| 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 | 0000000b |
| 00C2h | PORT R1 DATA REG. | R/W | R1 | undefined |
| 00C3h | PORT R1 DATA DIRECTION REG. | W | R1DD | 0000000b |
| 00C4h | PORT R2 DATA REG. | R/W | R2 | undefined |
| 00C5h | PORT R2 DATA DIRECTION REG. | W | R2DD | 0000000b |
| 00C6h | reserved | | | |

| | | | | |
|-------|--------------------------------|-----|--------|-----------|
| 00C7h | CLOCK CONTROL REG. | W | CKCTRL | --110111b |
| | BASIC INTERVAL REG. | R | BTR | undefined |
| 00C8h | WATCH DOG TIMER REG. | W | WDTR | -0001111b |
| 00C9h | PORT R1 MODE REG. | W | PMR1 | 0000000b |
| 00CAh | INT. MODE REG. | R/W | IMOD | -0000000b |
| 00CBh | EXT. INT. EDGE SELECTION | W | IEDS | 0000000b |
| 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 | 0000000b |
| 00D1h | TIMER1 (8bit) MODE REG. | R/W | TM1 | 0000000b |
| 00D2h | TIMER2 (8bit) MODE REG. | R/W | TM2 | 0000000b |
| 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 | 0000000b |
| 00DBh | Reserved | | | |
| 00DCh | STANDBY MODE RELEASE REG0 | W | SMPR0 | 0000000b |
| 00DDh | STANDBY MODE RELEASE REG0 | W | SMPR1 | 0000000b |
| 00DEh | PORT R1 OPEN DRAIN ASSIGN REG. | W | R1ODC | 0000000b |
| 00DFh | PORT R2 OPEN DRAIN ASSIGN REG. | W | R2ODC | 0000000b |
| 00E0h | PORT R3 OPEN DRAIN ASSIGN REG. | W | R3ODC | 0000000b |
| 00E1h | PORT R4 OPEN DRAIN ASSIGN REG. | W | R4ODC | -----0b |
| 00E2h | Reserved | | | |
| 00E3h | Reserved | | | |
| 00E4h | PORT R0 OPEN DRAIN ASSIGN REG. | W | R0ODC | 0000000b |
| 00E5h | PORT R3 DATA REG. | R/W | R3 | undefined |
| 00E6h | PORT R3 DATA DIRECTION REG. | W | R3DD | 0000000b |

| | | | | |
|-------|------------------------------------|-----|-------|-----------|
| 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 | 0000000b |
| 00F7h | STANDBY RELEASE LEVEL CONT. REG. 1 | W | SRLC1 | 0000000b |
| 00F8h | PORT R0 PULL-UP REG. CONT. REG. | W | R0PC | 0000000b |
| 00F9h | PORT R1 PULL-UP REG. CONT. REG. | W | R1PC | 0000000b |
| 00FAh | PORT R2 PULL-UP REG. CONT. REG. | W | R2PC | 0000000b |
| 00FBh | PORT R3 PULL-UP REG. CONT. REG. | W | R3PC | 0000000b |
| 00FCh | PORT R4 PULL-UP REG. CONT. REG. | W | R4PC | ----- 0b |
| 00FDh | Reserved | | | |
| 00FEh | Reserved | | | |
| 00FFh | Reserved | | | |

8.4 Addressing Mode

The GMS81C5016/24/32 uses six addressing modes;

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

(1) Register Addressing

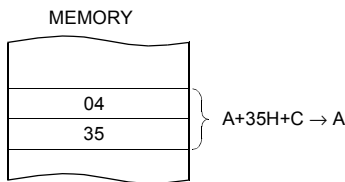
Register addressing accesses the A, X, Y, C and PSW.

(2) Immediate Addressing → #imm

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

Example:

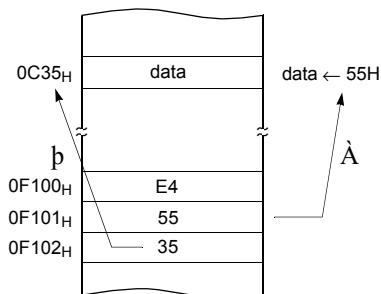
```
0435   ADC   #35H
```



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
```

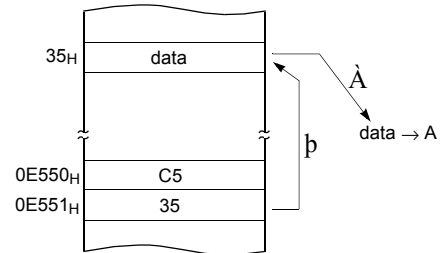


(3) Direct Page Addressing → dp

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

Example; G=0

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



(4) Absolute Addressing → !abs

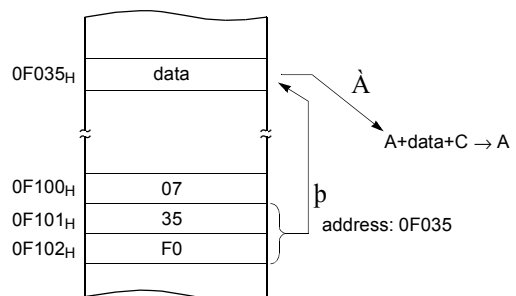
Absolute addressing sets corresponding memory data to Data , i.e. second byte(Operand I) of command becomes lower level address and third byte (Operand II) becomes upper level address.

With 3 bytes command, it is possible to access to whole memory area.

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

Example;

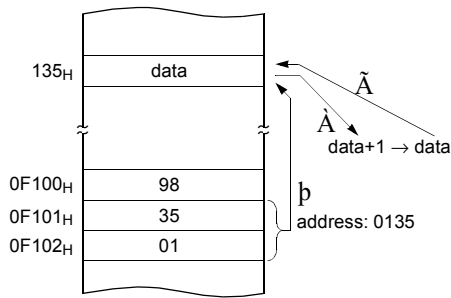
```
0735F0   ADC   !0F035H     ;A ←ROM[0F035H]
```



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

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

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



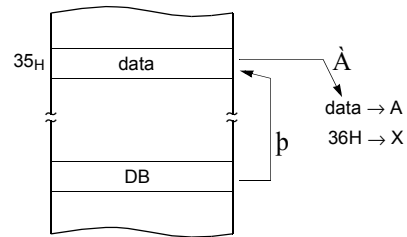
X indexed direct page, auto increment → {X}+

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=35H

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



(5) Indexed Addressing

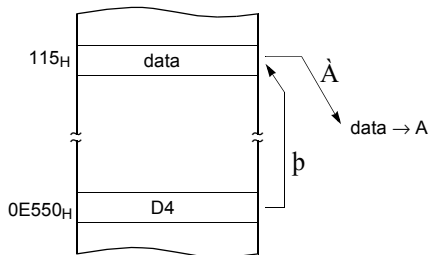
X indexed direct page (no offset) → {X}

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

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

Example; X=15H, G=1, RPR=01H

```
D4 LDA {X} ; ACC←RAM[X].
```



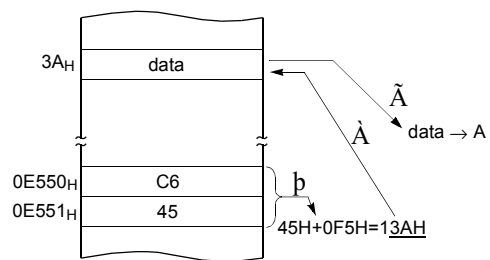
X indexed direct page (8 bit offset) → dp+X

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=0F5H

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



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

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

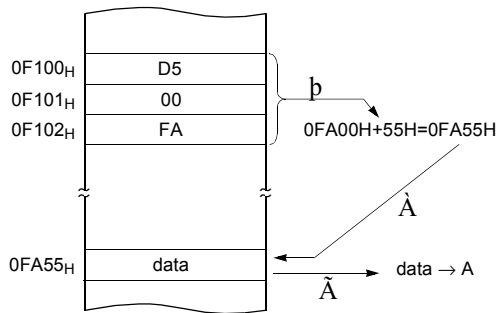
This is same with above (2). Use Y register instead of X.

Y indexed absolute → !abs+Y

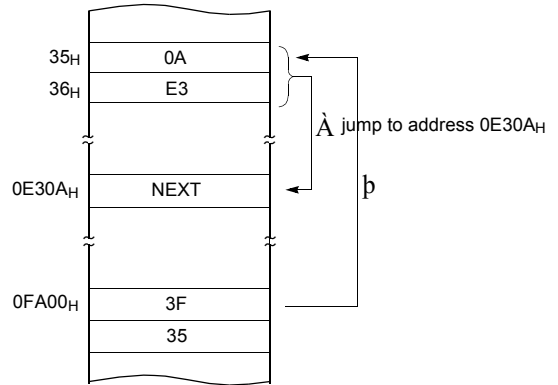
Sets the value of 16-bit absolute address plus Y-register data as Memory. This addressing mode can specify memory in whole area.

Example; Y=55H

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



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



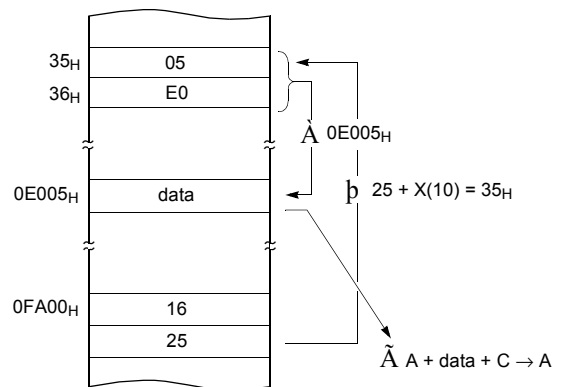
X indexed indirect → [dp+X]

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=10H

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



(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

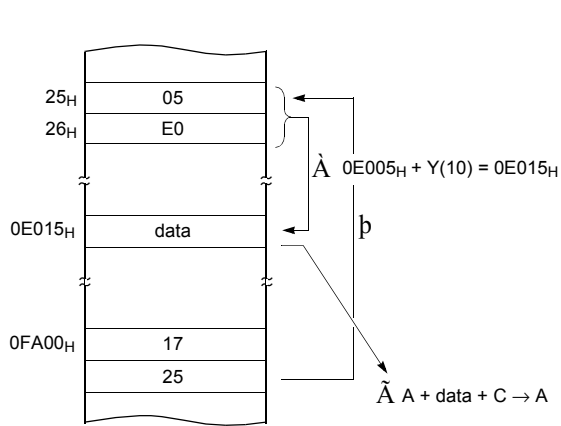
Y indexed indirect → [dp]+Y

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=10H

1725 ADC [25H]+Y



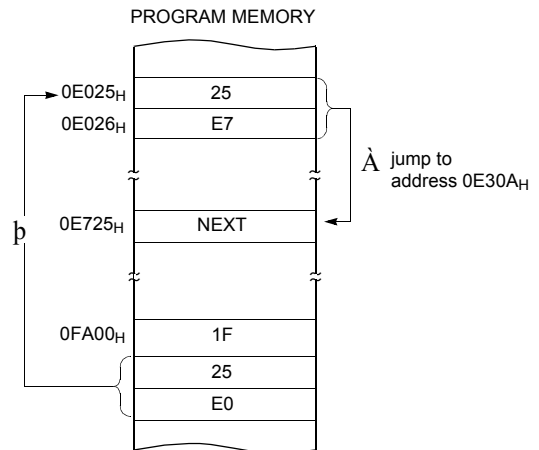
Absolute indirect → [!abs]

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

JMP

Example; G=0

1F25E0 JMP [!0C025H]



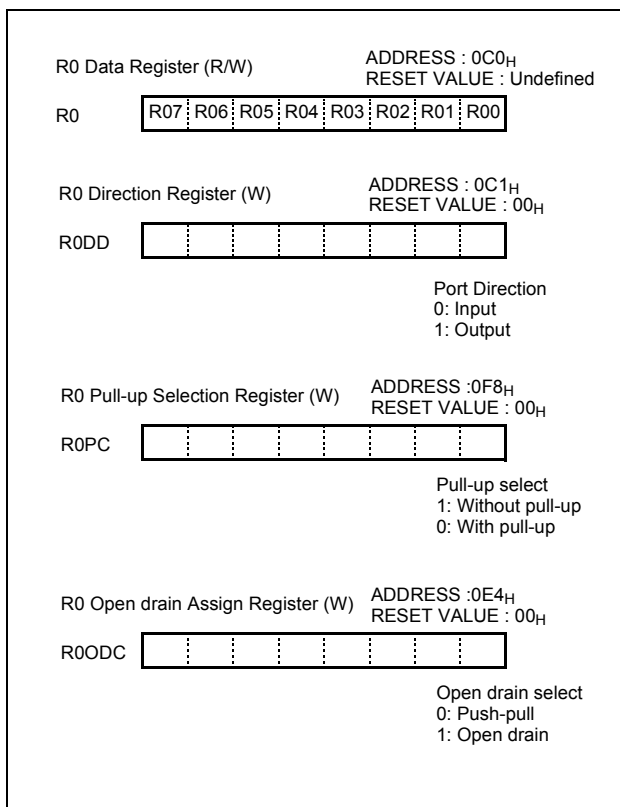
9. I/O PORTS

The GMS81C5016/24/32 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 can be selectable by program. Each port contains data direction register which controls I/O and data register which stores port data.

9.1 R0 Ports

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

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



(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 set as the output state by R0DD, and data is written in R0, data is outputted into R0 pin. When set as the input state, input state of pin is read. The initial value of R0 is unknown in reset state.

(3) R0 Open drain Assign Register (R0ODC)

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

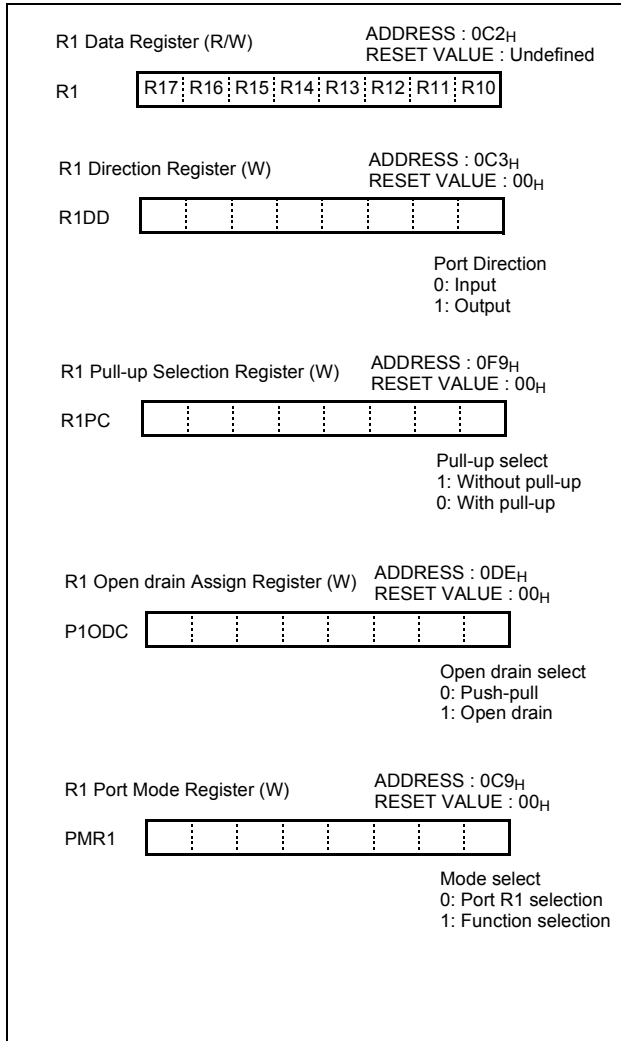
(4) R0 Pull-up Resistor Control Register (R0PC)

R0 pull-up resistor control register (R0PC) is 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R0PC is selected as ``1``, pull-up ia 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.

9.2 R1 Ports

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

R1 has internal pull-ups that is independently connected or disconnected by register R1PC. The control registers for R1 are shown 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 set as the output state by R1DD, and data is written in R1, data is outputted into R1 pin. When set as the input state, input state of pin is read. The initial value of R1 is unknown in reset state.

(3) R1 Mode Register (PMR1)

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

PMR1 is write-only register and initialized as ``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 9-1 Selection mode of PMR1

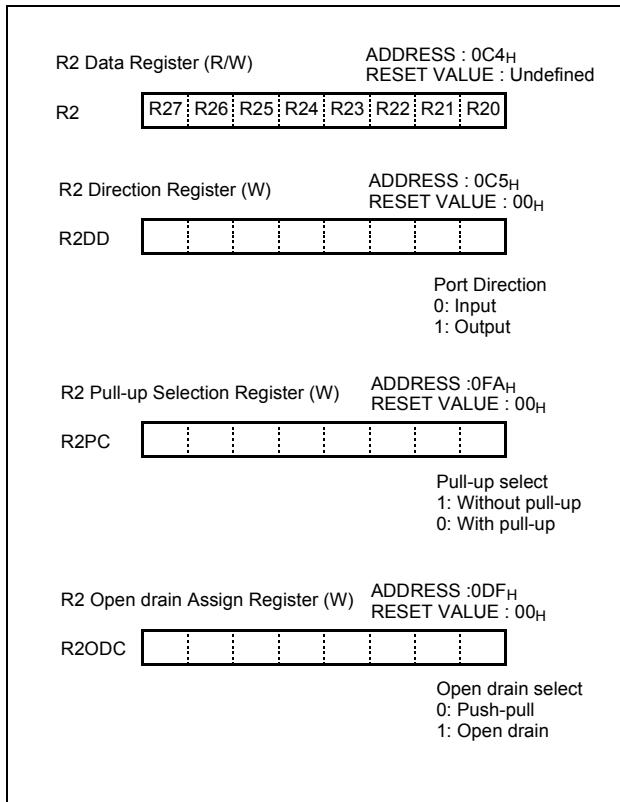
(4) R1 Pull-up Resistor Control Register (R1PC)

R1 pull-up resistor control register (R1PC) is 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R1PC is selected as ``1``, pull-up ia 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.

9.3 R2 Port

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

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



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

R2 I/O Data Direction Register (R2DD) is 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 set as the output state by R2DD, and data is written in R2, data is outputted into R2 pin. When set as the input state, input state of pin is read. The initial value of R2 is unknown in reset state.

(3) R2 Open drain Assign Register (R2ODC)

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

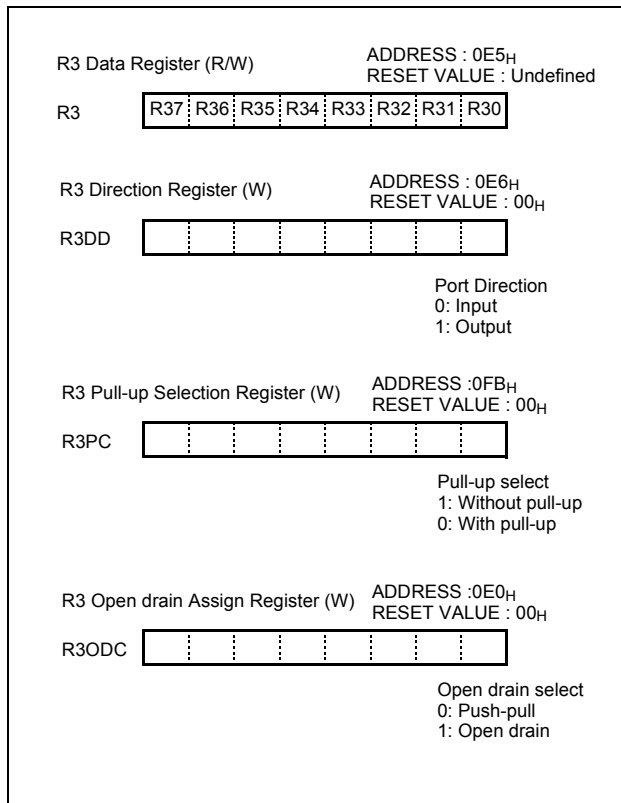
(4) R2 Pull-up Resistor Control Register (R2PC)

R2 pull-up resistor control register (R2PC) is 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R2PC is selected as ``1``, pull-up ia disabled and if selected as ``0``, it is enabled. R2PC is write-only register and initialized as ``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 bidirectional I/O port (address 0E5H). Each I/O pin can independently used as an input or an output through the R3DD register (address 0E6H).

R3 has internal pull-ups that is independently connected or disconnected by R3PC (address 0FBH). 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 set as the output state by R3DD, and data is written in R3, data is outputted into R3 pin. When 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 8bit register, and can assign R3 port as open drain output port 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.

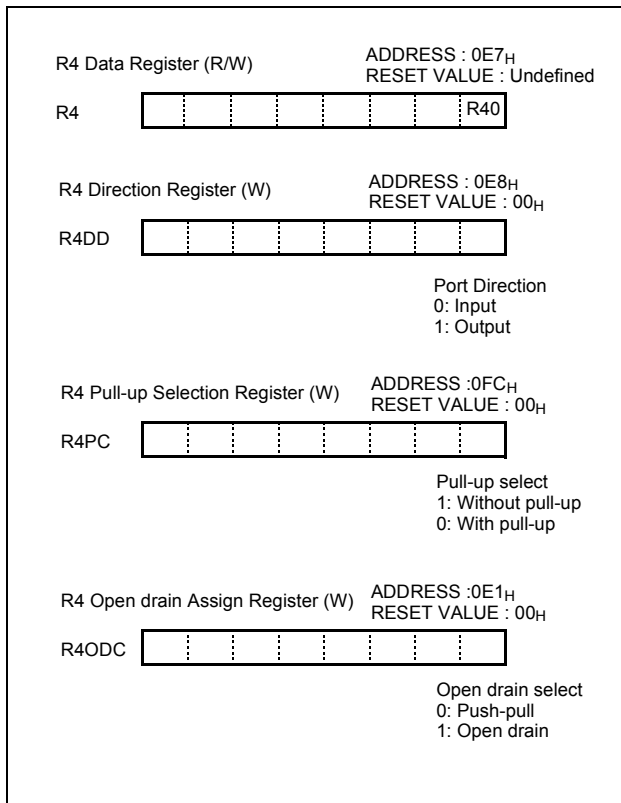
(4) R3 Pull-up Resistor Control Register (R3PC)

R3 pull-up resistor control register (R3PC) is 8-bit register and can control pull-up on or off each bit, if corresponding port is selected as input. If R3PC is selected as ``1``, pull-up ia 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 bidirectional I/O port (address 0E7H). Each I/O pin can independently used as an input or an output through the R4DD register (address 0E8H).

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 set as the output state by R4DD, and data is written in R4, data is outputted into R4 pin. When 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 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 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.

10. 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.

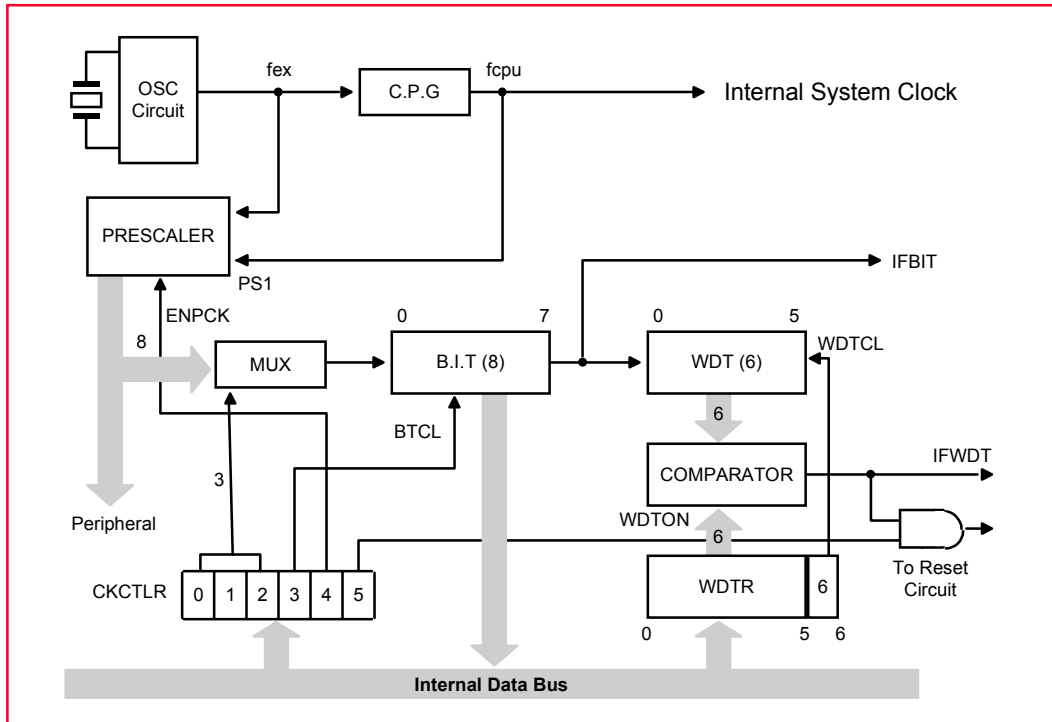


Figure 10-1 Block Diagram of Clock Generator

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.

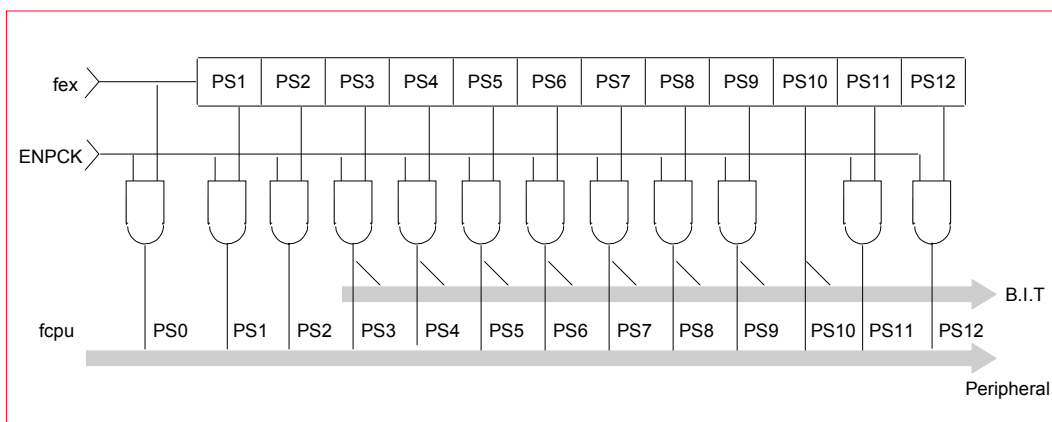


Figure 10-2 Block diagram of Prescaler

| 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 10-1 ps output period

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

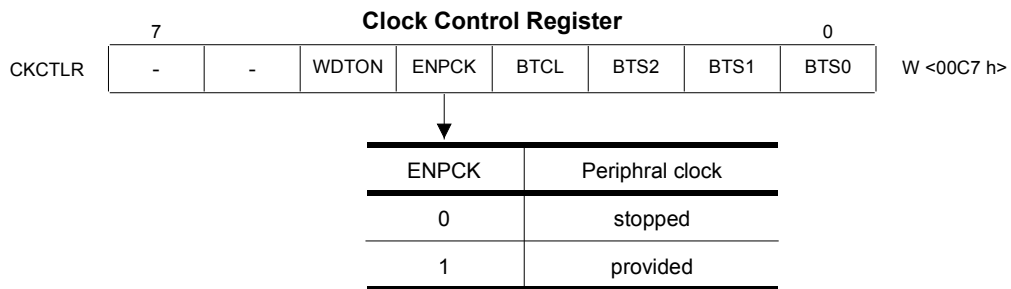


Figure 10-3 Clock Control Register

10.1 Operation Mode

The system clock controller starts or stops the main-frequency clock oscillator. The Figure 10-4 shows the operating mode transition diagram.

Main-clock operating mode

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

voked.

STOP mode

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

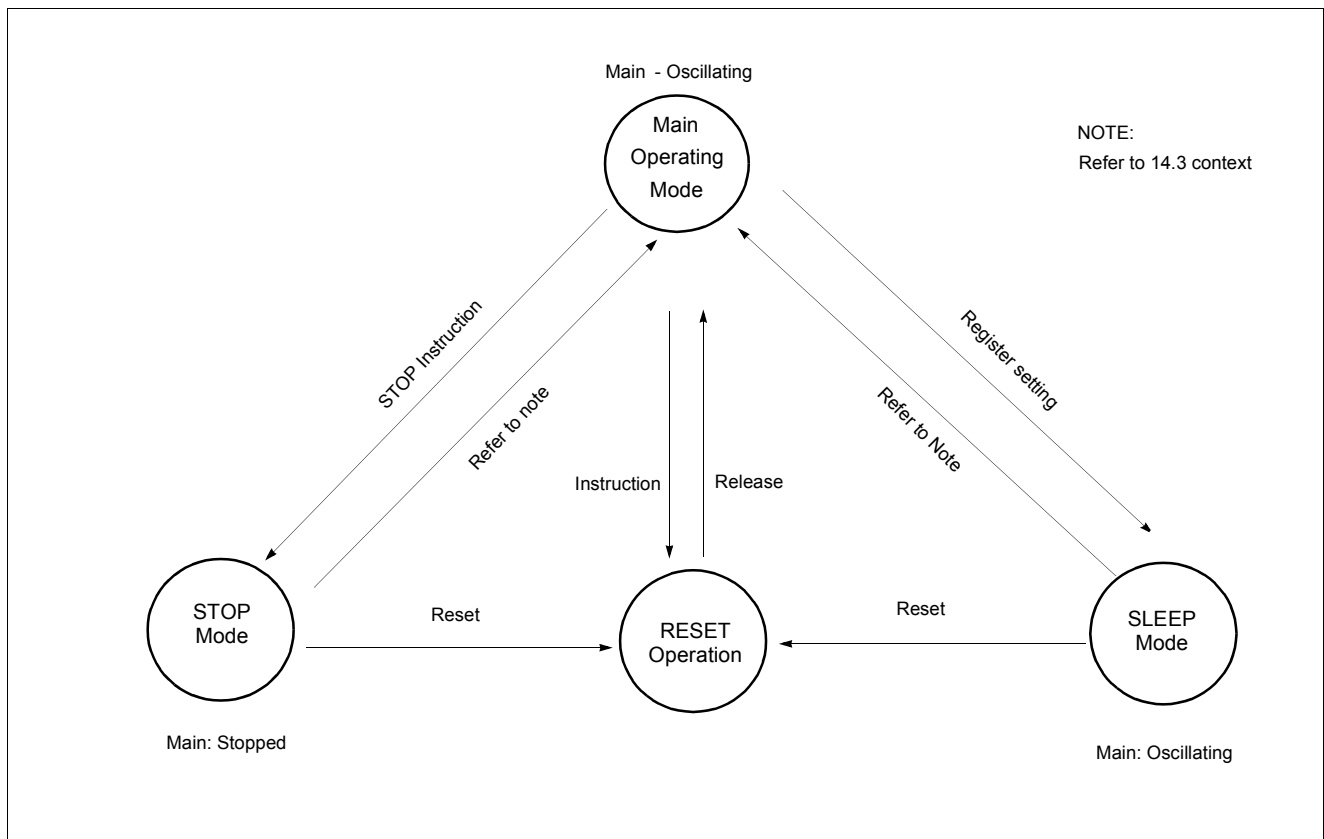


Figure 10-4 Operating Mode

11. TIMER

11.1 Basic Interval Timer

The GMS81C5016/24/32 has one 8-bit Basic Interval Timer that is free-run and can not stop. Block diagram is shown in Figure 11-1 .

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

-8bit binary 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.

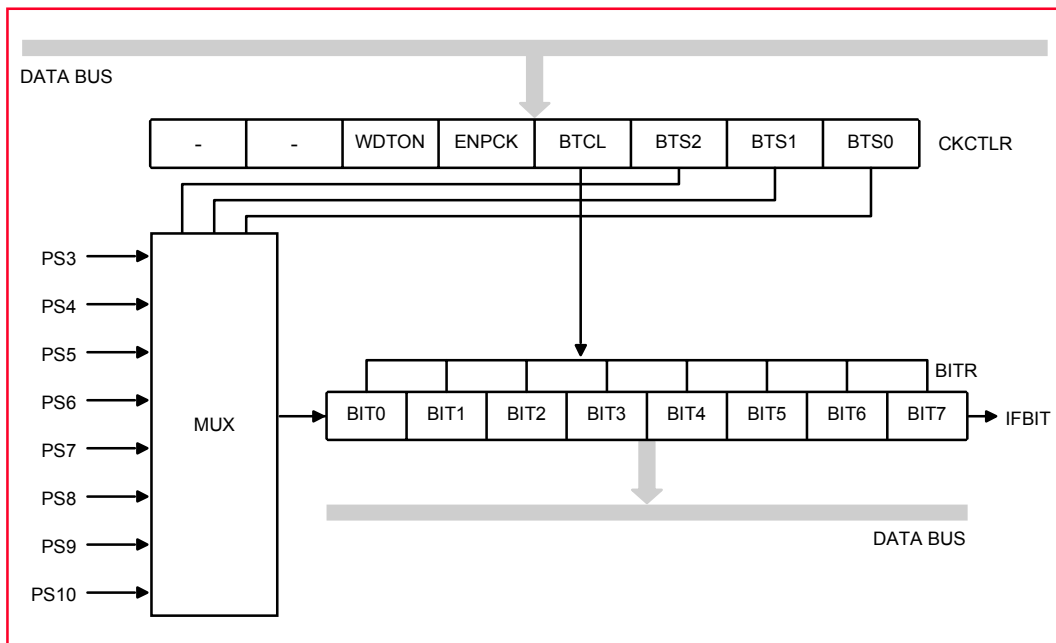


Figure 11-1 Block Diagram of Basic Interval Timer

(1) Control of B.I.T

The Basic Interval Timer is controlled by the clock control register (CKCTRL) shown in Figure 11-2 . If bit3(BTCL) of CKCTRL 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.

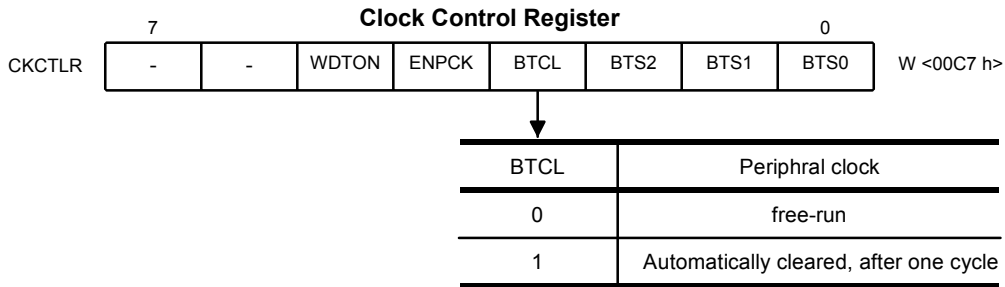


Figure 11-2 BTCL mode of B.I.T

(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 gener-

ate 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 11-3 .

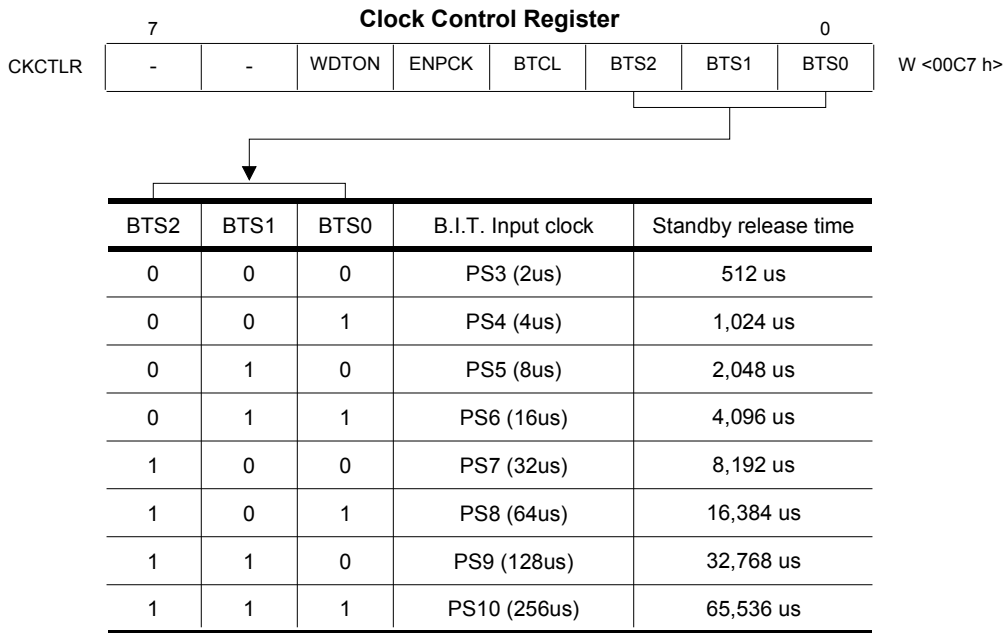


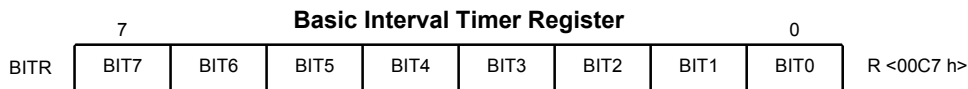
Figure 11-3 Basic Interval Timer Interrupt Time

(3) Reading Basic Interval Timer

By 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 reg-

ister is written, then CKCTLR register with same address is written.



11.2 Timer0, Timer1, Timer2

(1) Timer Operation Mode

Timer consists of 16bit binary counter Timer0 (T0), 8bit binary Timer1 (T1), Timer2 (T2), Timer Data Register, Timer Mode Register (TM01, TM0, TM1, TM2) and control circuit. Timer Data Register Consists of Timer0 High-MSB Data Register (T0HMD), Timer0 High-LSB Data Register (T0HLD), Timer0 Low-MSB Data Register (T0LMD), Timer0 Low-LSB Data Register (T0LLD),

Timer1 High Data Register (T1HD), Timer1 Low Data Register (T1LD), Timer2 Data Register (T2DR). Any of the PS0 ~ PS5, PS11 and external event input 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 | <ul style="list-style-type: none"> - 16-bit Interval Timer - 16-bit Event Counter - 16-bit Input Capture - 16-bit rectangular-wave output | <ul style="list-style-type: none"> - Single/Modulo-N Mode - Timer Output Initial Value Setting - Timer0~Timer1 combination Logic Output - One Interrupt Generating Every 2nd Counter Overflow |
| Timer1 | <ul style="list-style-type: none"> - 8-bit Interval Timer - 8-bit rectangular-wave output | |
| Timer2 | <ul style="list-style-type: none"> - 8-bit Interval Timer - 8-bit rectangular-wave output - Modulo-N Mode | |

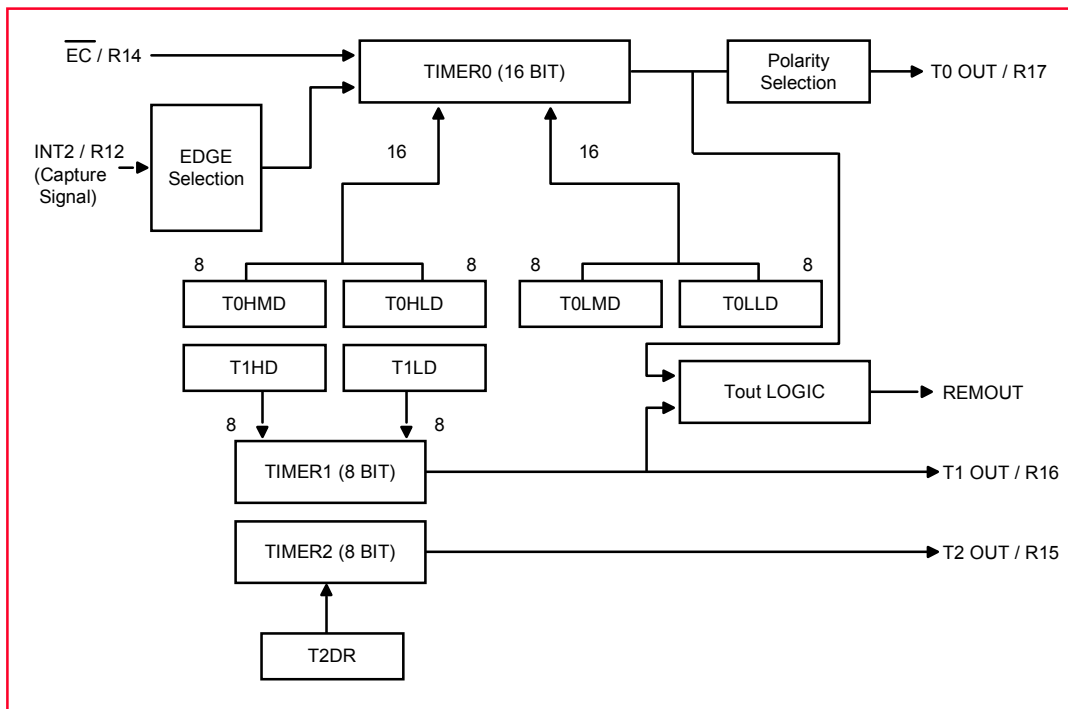


Figure 11-4 Timer / Counter Block diagram

(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 |

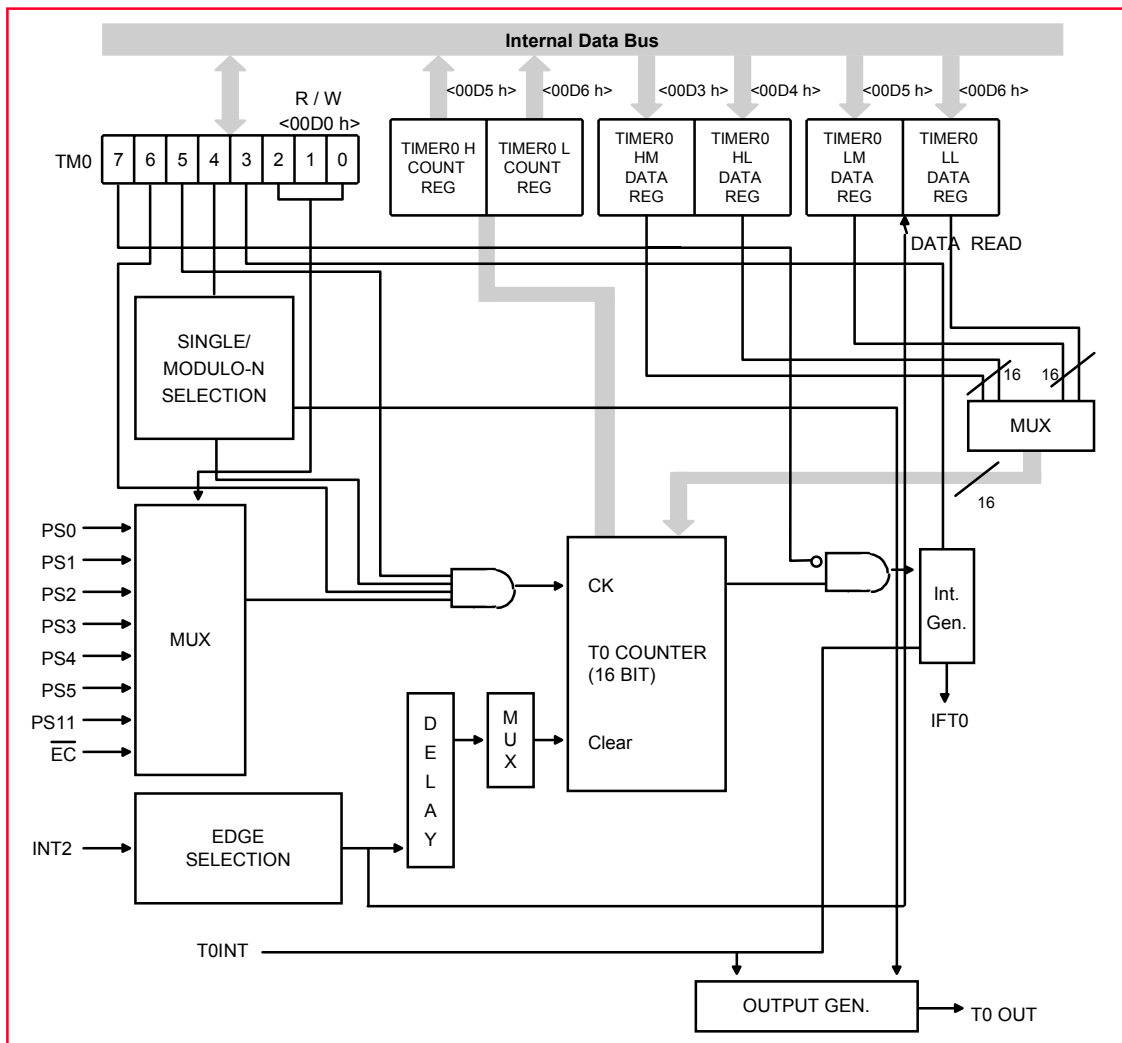


Figure 11-5 Block Diagram of Timer0

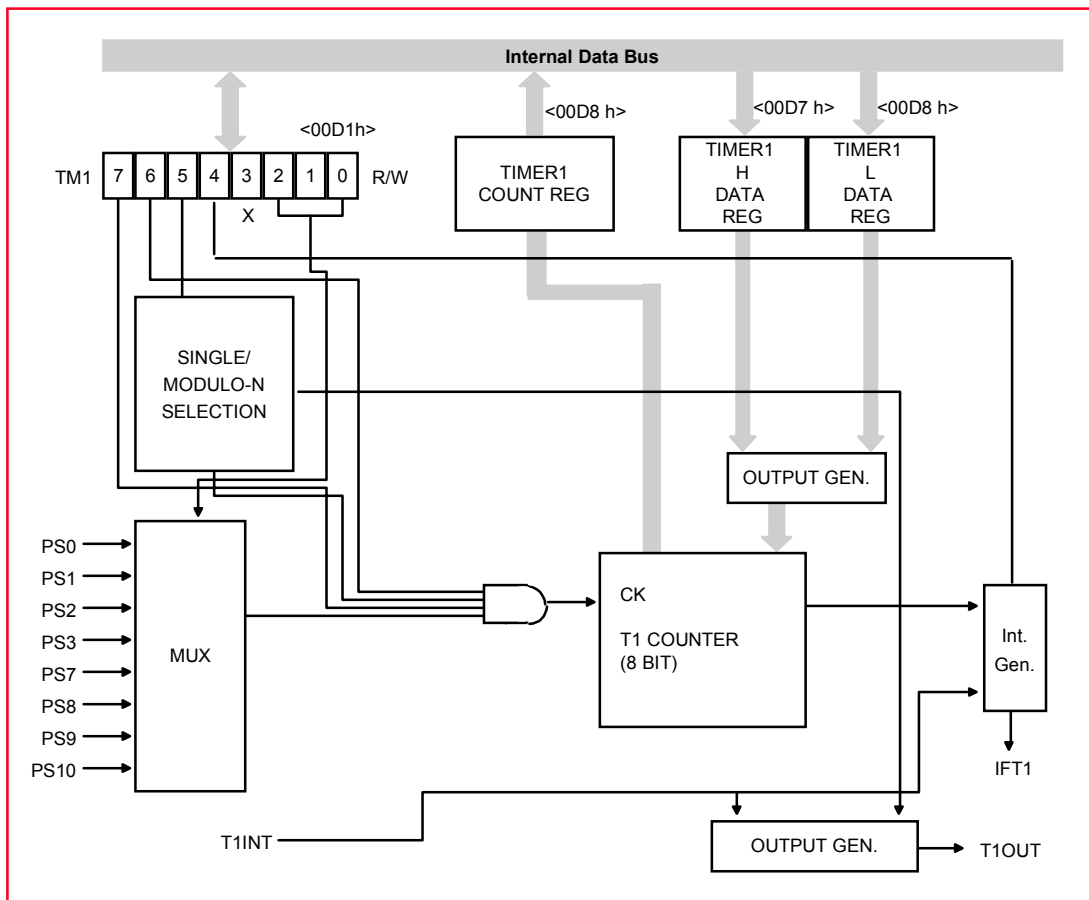


Figure 11-6 Block Diagram of Timer1

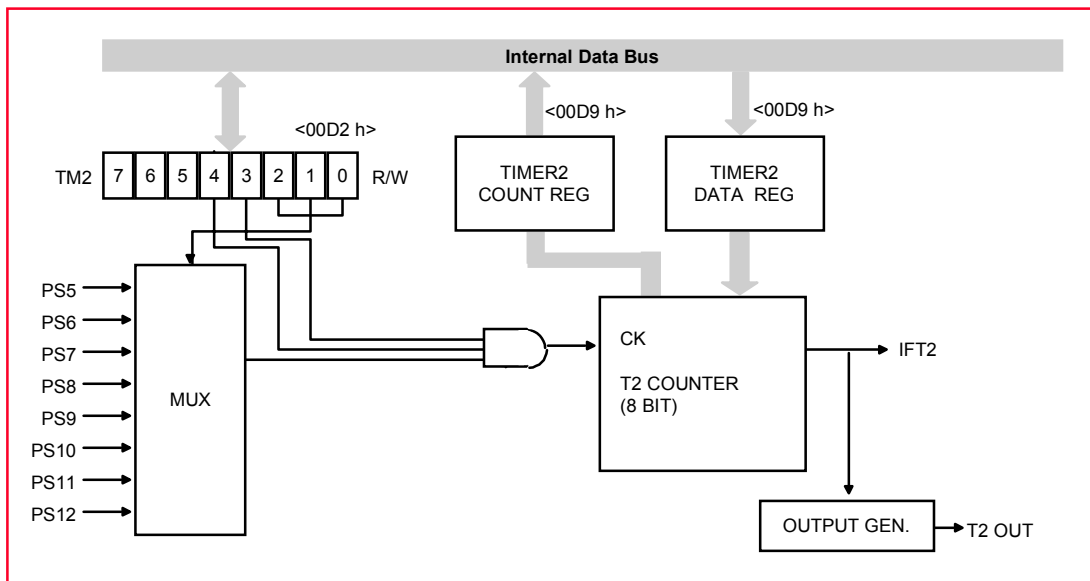


Figure 11-7 Block Diagram of Timer2

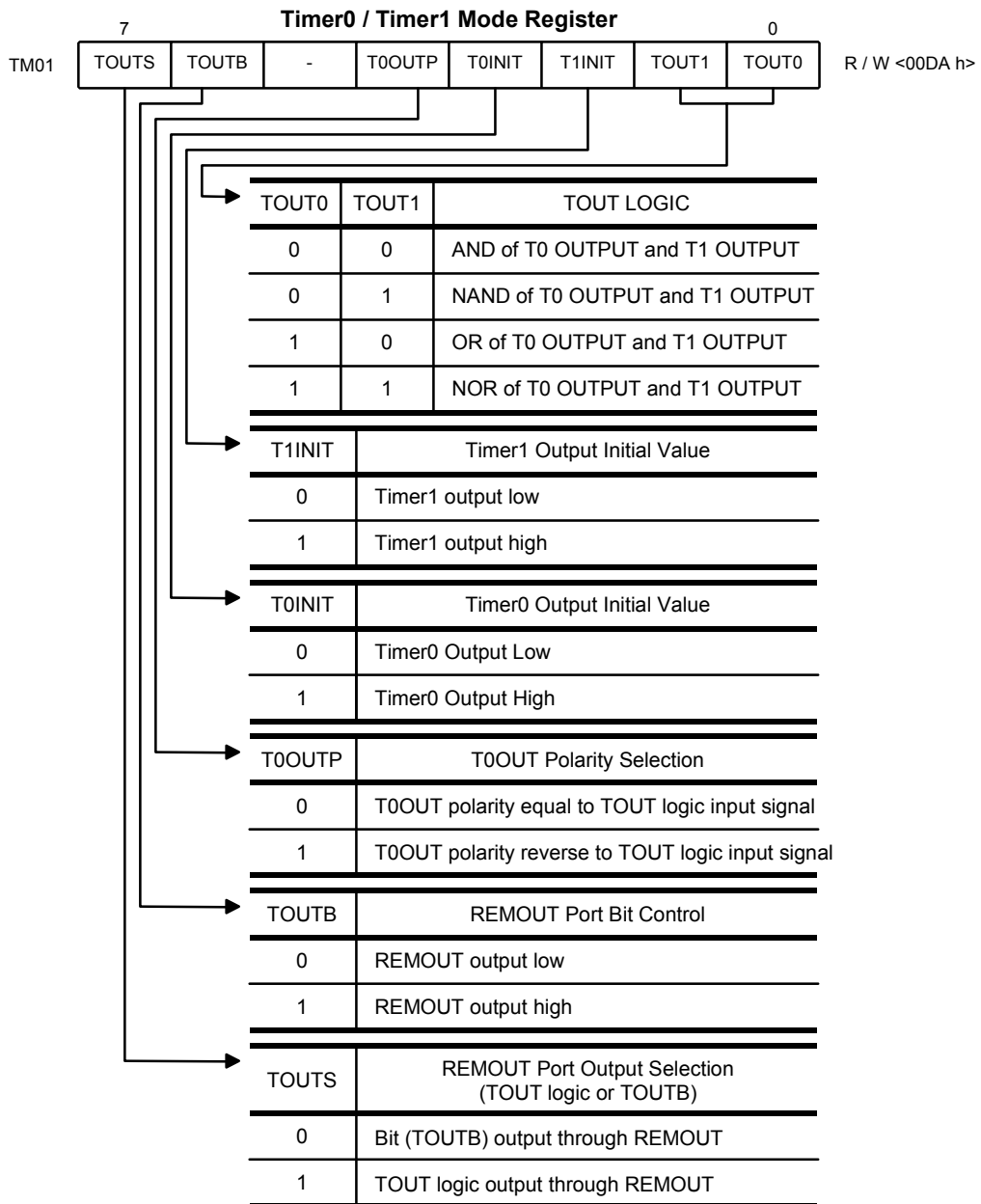
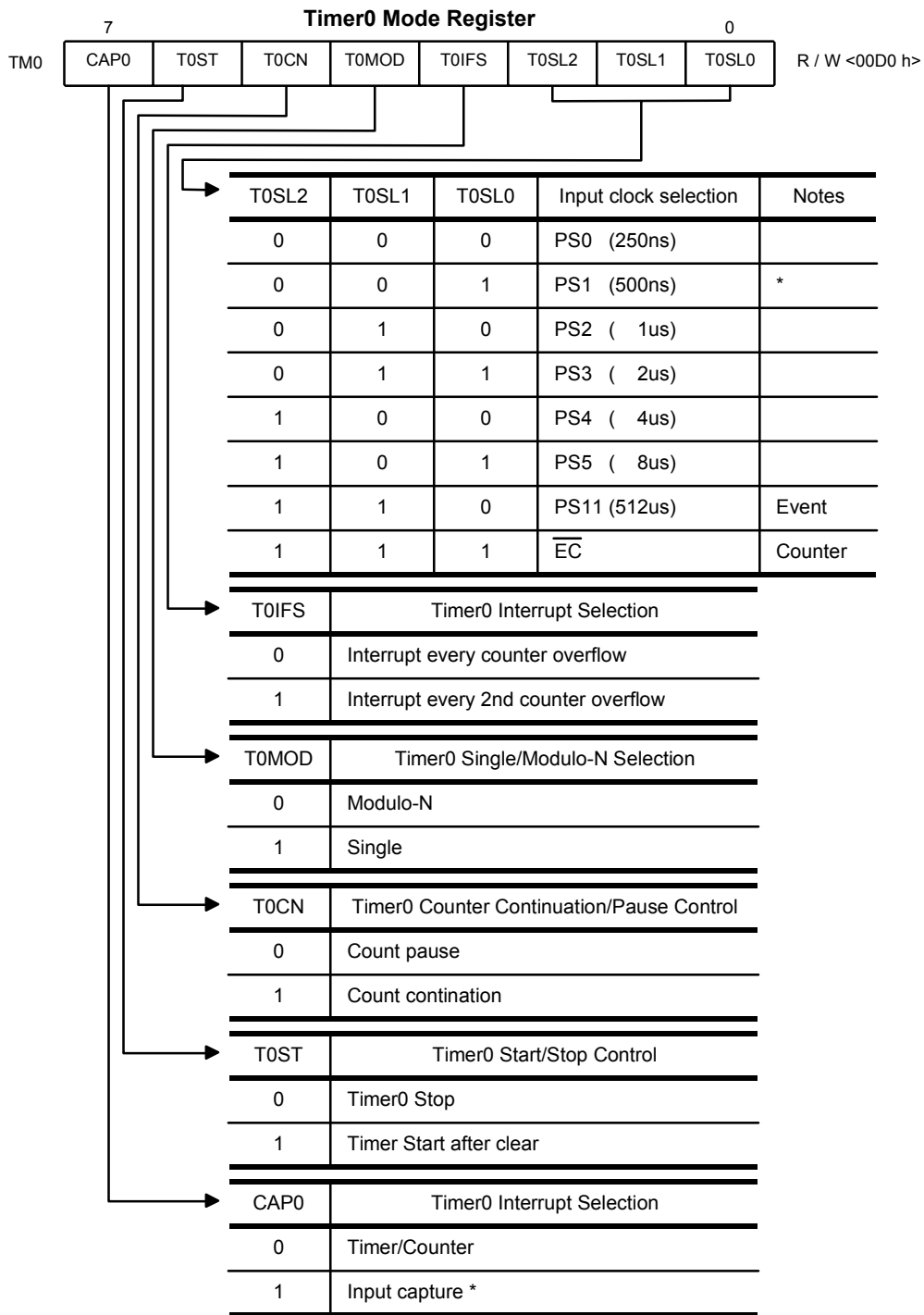


Figure 11-8 Timer0 / Timer1 Mode Register



* PS1 : not supporting input capture.

Figure 11-9 Timer0 Mode Register

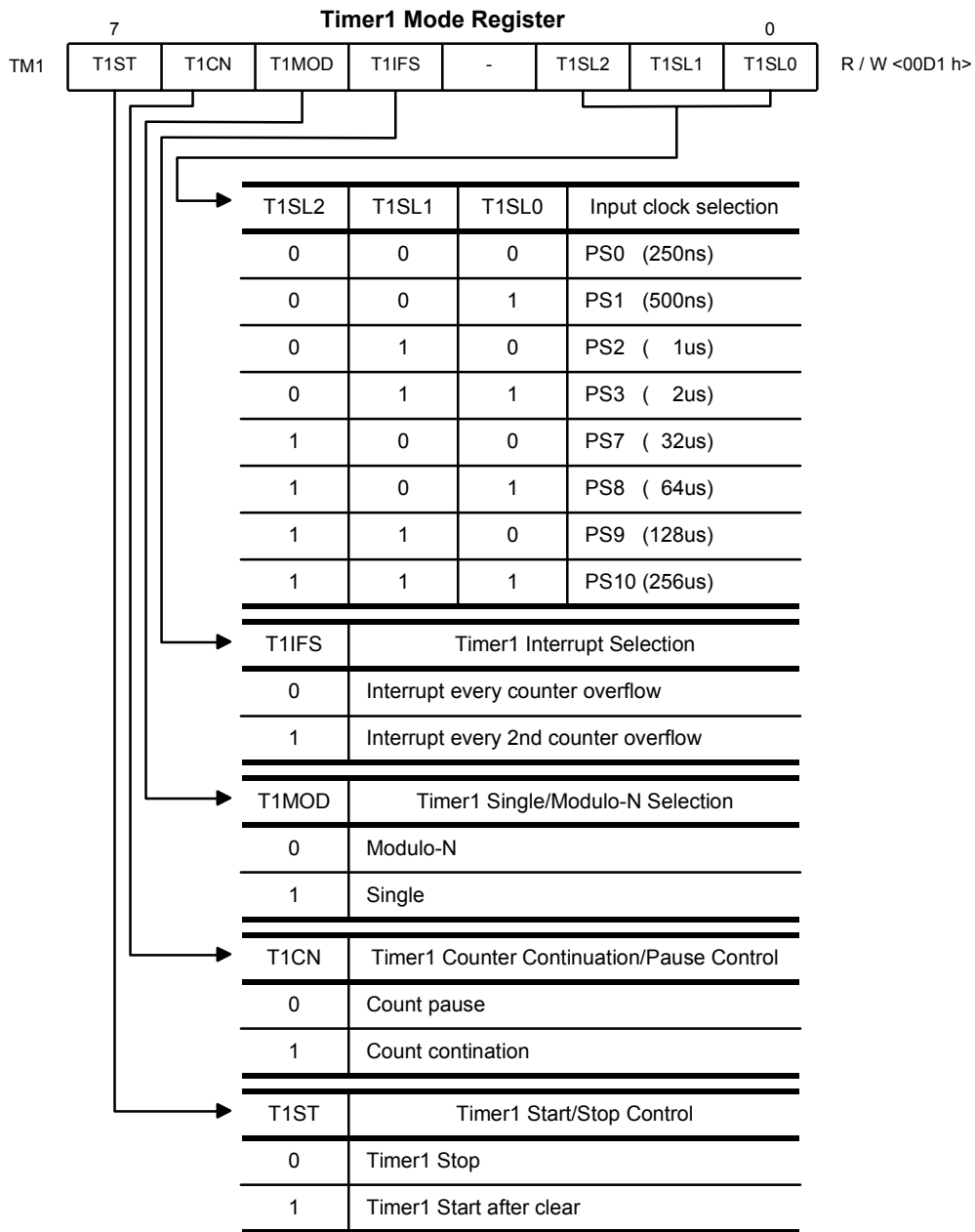


Figure 11-10 Timer1 Mode Register

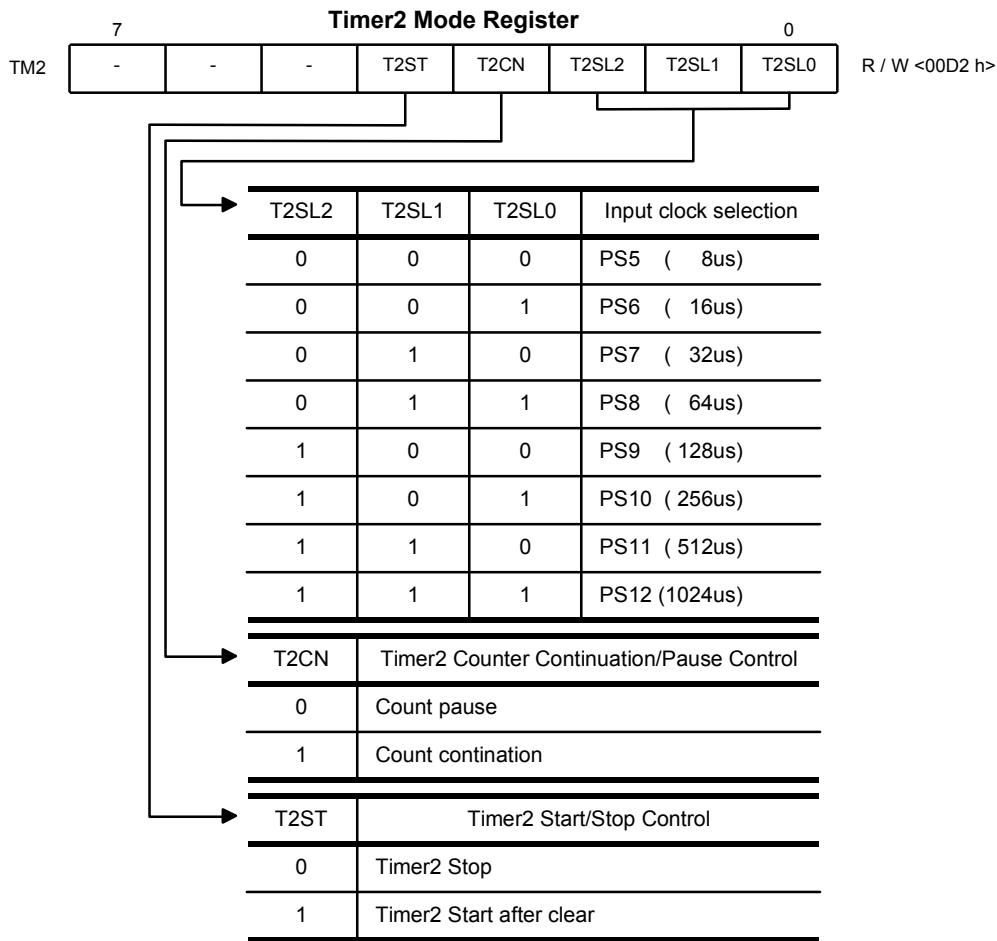


Figure 11-11 Timer2 Mode Register

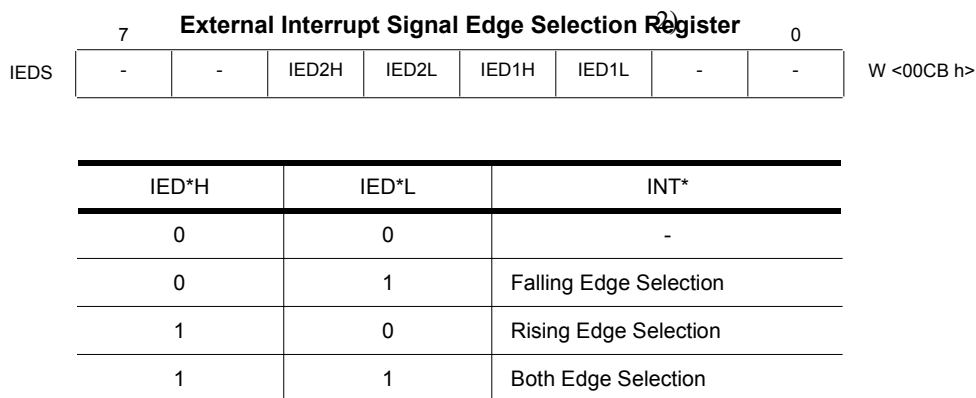


Figure 11-12 External Interrupt Signal Edge Selection Register

(3) Timer0, Timer1

TIMER0 and TIMER1 have an up-counter. When value of the up-counter reaches the content of Timer Data Register

(TDR), the up-counter is cleared to `00 h`, and interrupt (IFT0, IFT1) is occurred at the next clock.

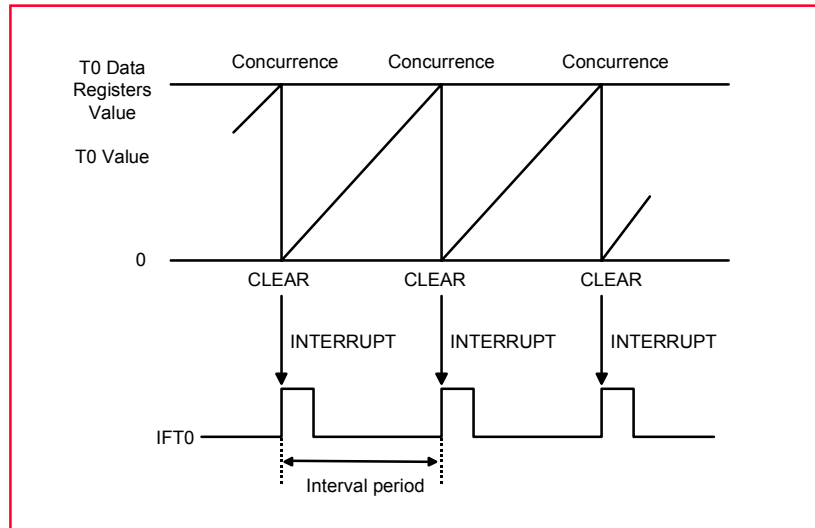


Figure 11-13 Operatiion of Timer0

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 of CAP0 to `1`, the period of signal from INT2 can be measured and then, event counter value for INT2 can be read. During counting-up, value of counter can be read.

Timer execution is stopped by the reset signal (RESET

= `L`)

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 earlier read upper 8-bit data are matched with the later read upper 8-bit data, read 16-bit data are correct. If not, caution should be taken 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

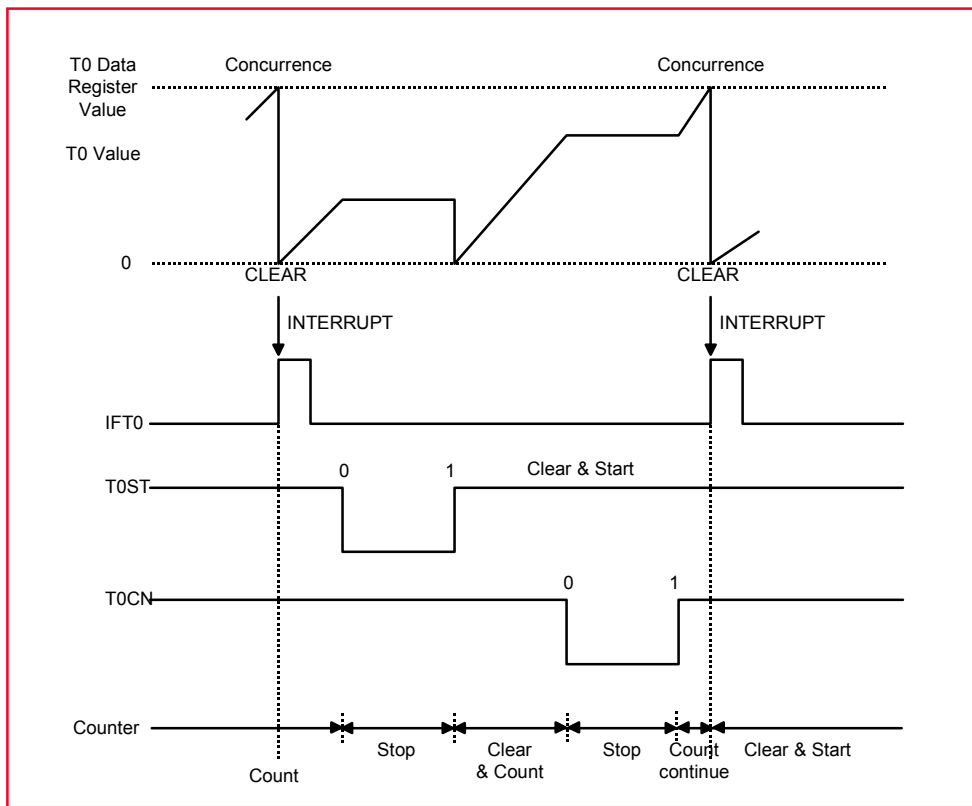


Figure 11-14 Start/Stop operation of Timer0

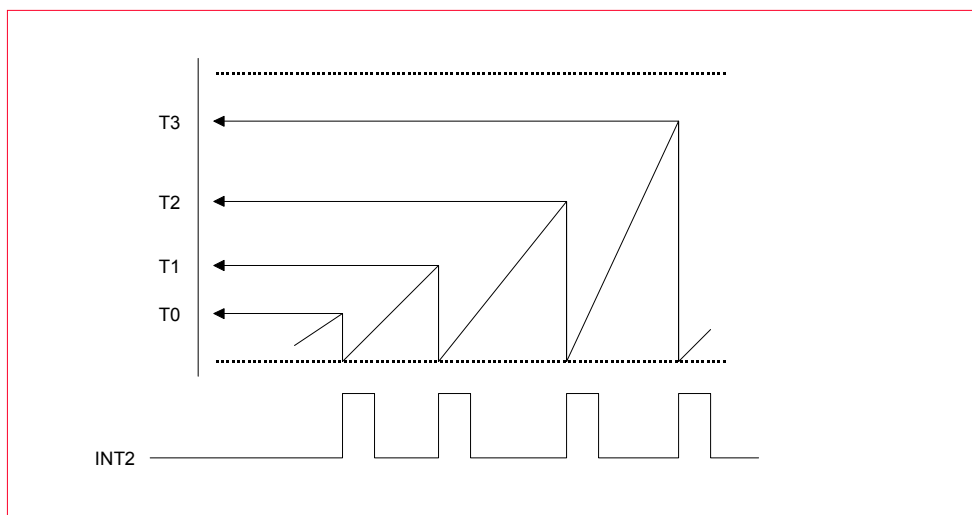


Figure 11-15 Input capture operation of Timer0

*** Single/Modulo-N Mode**

Timer0 (Timer1) can select initial (T0INIT, T1INIT of TM01) 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 High? High -Data Register is transferred and to be ``High``. Single Mode can be set by Mode Select bit (T0MOD, T1MOD) of Timer Mode Register (TM0, TM1) to ``1``. When used as Single Mode, Timer counts up and compares with value of Data Register. If the result is same, Time Out interrupt occurs and level of Timer Output port toggle, then counter stops as reset state. When used as Modulo-N Mode, T0MOD (T1MOD) should be set ``0``. Counter counts up until the

value of Data Register and occurs Time-out interrupt. The level of Timer Output port toggle and repeats process of counting the value which is selected in Data Register. During Modulo-N Mode, If interrupt select bit (T0IFS, T1IFS) of Mode Register is ``0``, Interrupt occurs on every Time-out. If it is ``1``, Interrupt occurs every second time-out.

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

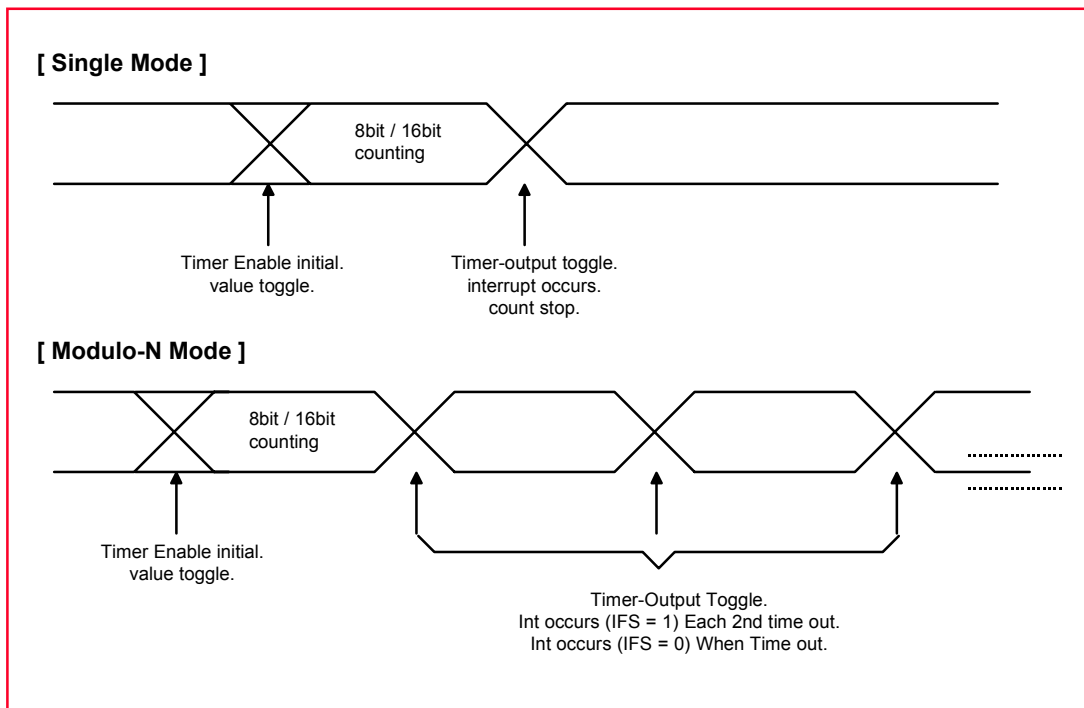


Figure 11-16 Operation Diagram for Single/Modulo-N Mode

(4) Timer 2

Timer2 operates as a up-counter. The content 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.

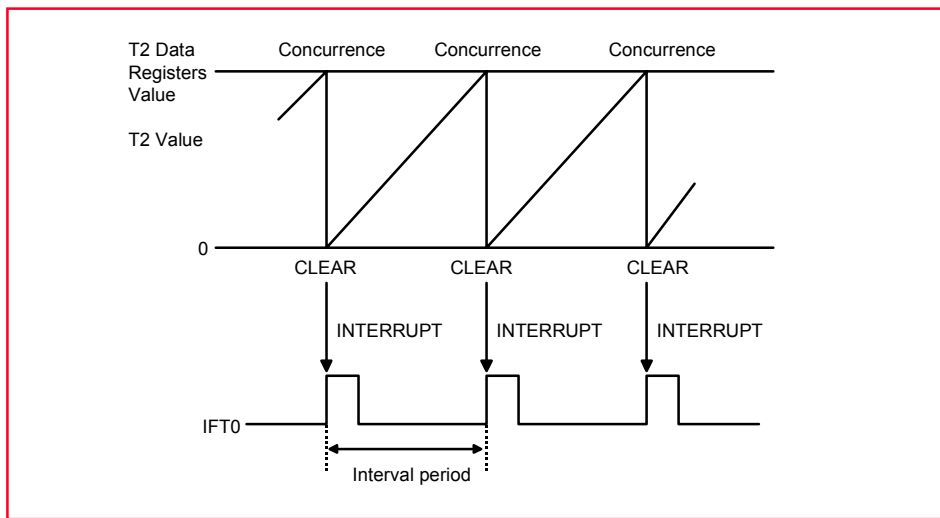


Figure 11-17 Operation of Timer2

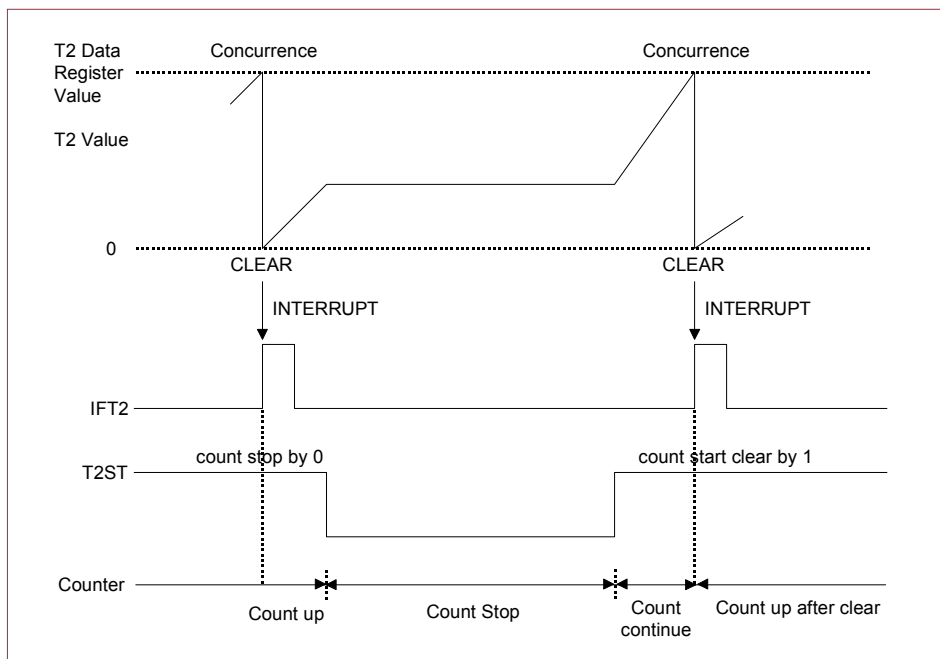


Figure 11-18 Start/Stop of Timer2

12. INTERRUPTS

The GMS81C5016/24/32 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 12-1 .

The GMS81C5016/24/32 contains 8 interrupt sources; 3 externals and 5 internals. Nested interrupt services with priority control is also possible. Software interrupt is non-maskable interrupt, the others are all maskable interrupts.

- 8 interrupt source (2Ext, 3Timer, BIT, WDT and Key Scan)

- 8 interrupt vector
- Nested interrupt control is possible
- Programmable interrupt mode
- Hardware accept mode
- Software selection accept mode
- Read and write of interrupt request flag are possible.
- In interrupt accept, request flag is automatically cleared.

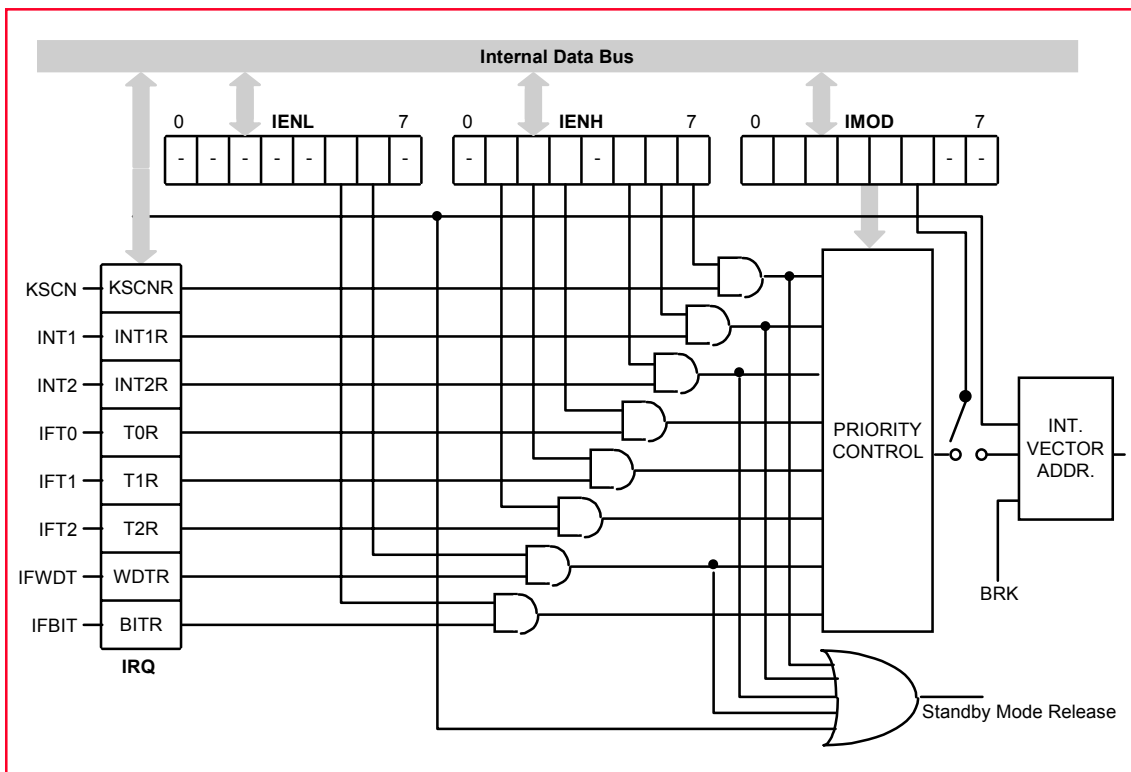


Figure 12-1 Block Diagram of Interrupt

12.1 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 12-1.

| | Mask | Priority | Interrupt Source | INT Vector High | INT Vector Low |
|-----------------------|--------------|----------|-----------------------------|-----------------|----------------|
| Hardware Interrupt | non-maskable | - | RST (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 (Watchdog Timer) | FFE9 | FFE8 |
| | | 7 | BITR (Basic Interval Timer) | FFE7 | FFE6 |
| | - | - | BRK instruction | FFDF | FFDE |

Table 12-1 Interrupt Priority & Source

12.2 INTERRUPT CONTROL REGISTER

I flag of PSW is a interrupt mask enable flag. When I flag = ``0``, all interrupts become disable. When I flag = ``1``, interrupts can be selectively enabled and disabled by contents of corresponding Interrupt Enable Register. When interrupt is occurred, interrupt request flag is set, and Interrupt request is detected at the edge of interrupt signal. The accepted interrupt request flag is automatically cleared

during interrupt cycle process. The interrupt request flag maintains ``1`` until the interrupt is accepted or is cleared in program. In reset state, interrupt request flag register (IRQH, IRQL) is cleared to ``0``. It is possible to read the state of interrupt register and to manipulate the contents of register and to generate interrupt. (Refer to software interrupt).

| | | | | | | | | |
|------|-------|-------|-------|---|-----|-----|-----|-------------|
| 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

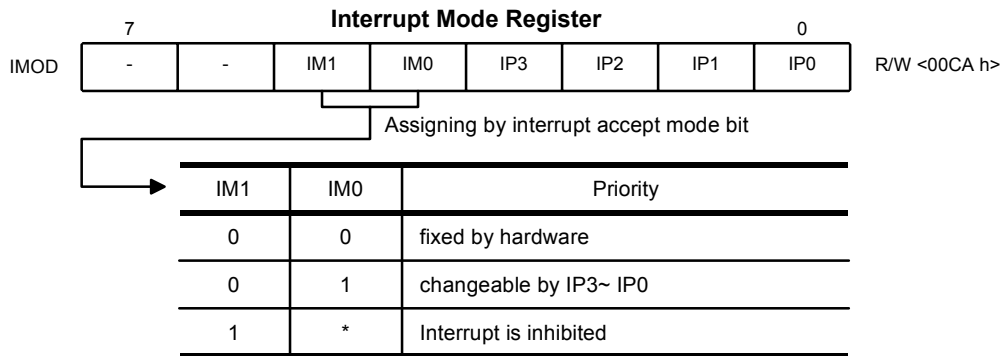
IENH : INTERRUPT ENABLE REGISTER HIGH

IRQL : INTERRUPT REQUEST REGISTER LOW

IRQH : INTERRUPT REQUEST REGISTER HIGH

12.3 INTERRUPT ACCEPT MODE

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



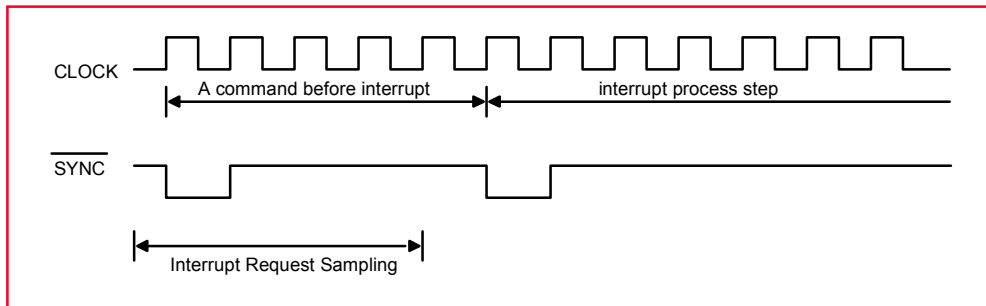
(1) Selection of Interrupt by IP3-IP0

The condition allow for accepting interrupt is set state of the interrupt mask enable flag and

the interrupt enable bit must be '1'. In Reset state, these IP3 - IP0 registers become all '0'.

| 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 12-2 Interrupt Selection by IP3 - IP0

(2) Interrupt Timing**Figure 12-2 Interrupt Enable Accept Timing**

*Interrupt Request sampling time

-Maximum 12 machine cycle (When execute DIV instruction)

-Minimum 0 machine cycle

*Interrupt preprocess step is 8 machine cycle

*Interrupt overhead

-Maximum $1 + 12 + 8 = 21$ machine cycle

-Minimum $1 + 0 + 8 = 9$ machine cycle

(3) The valid timing after executing Interrupt control instructions

I flag is valid just after executing of EI/DI on the contrary. Interrupt Enable register is valid one instruction after con-

trolling interrupt Enable Register.

12.4 INTERRUPT PROCESSING SEQUENCE

When an interrupt is accepted, the on-going process is stopped 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 occurred. 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, $SP \leftarrow SP$
- 2) Store lower byte of Program Counter, $SP \leftarrow SP - 1$
- 3) Store Program Status Word, $SP \leftarrow SP - 2$
- 4) After resetting of I-flag, clear accepted Interrupt Request Flag. (Set B-flag for BRK Instruction)
- 5) Call Interrupt service routine

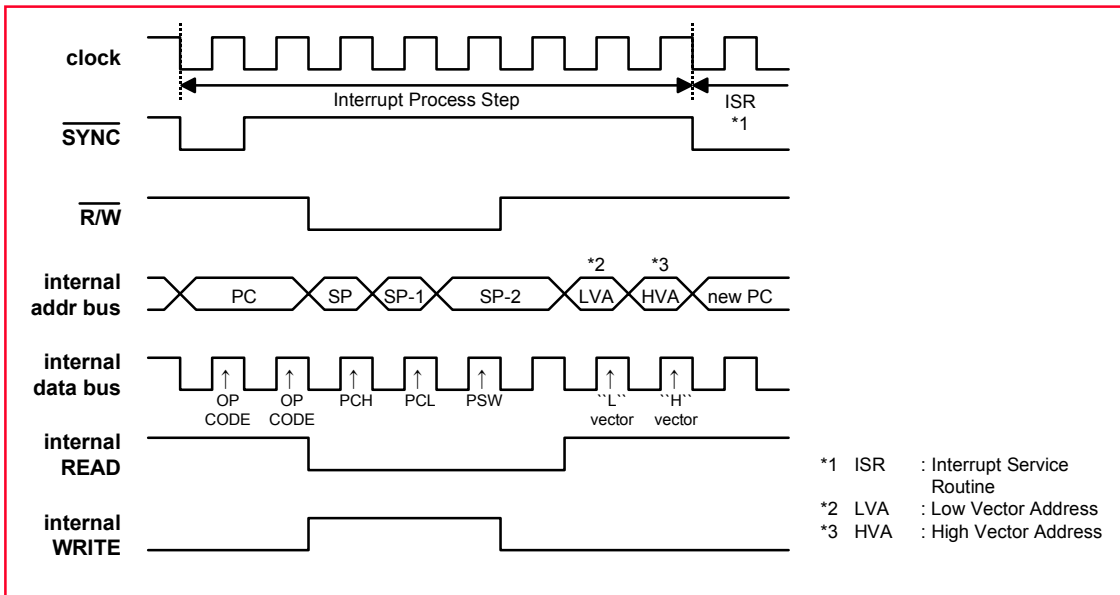
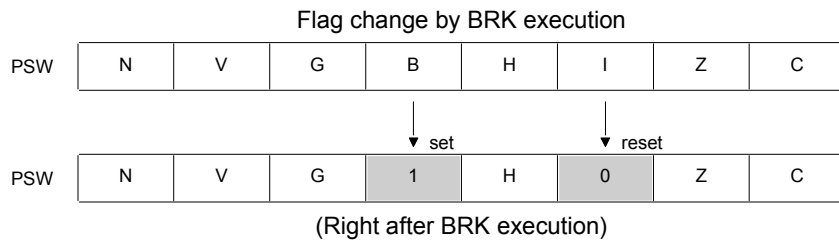


Figure 12-3 Interrupt Processing Step Timing

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

Software interrupt is available just by writing ``Break(BRK)`` instruction. The values of PC and PSW is

stacked by BRK instruction and then B flag of PSW is set and I flag is reset.



Interrupt vector of BRK instruction is shared by vector of Table Call (TCALL0). When both instruction of BRK and TCALL0 are used, as shown in Figure 12-4 each process-

ing routine is judged by contents of B flag. There is no instruction to reset directly B flag.

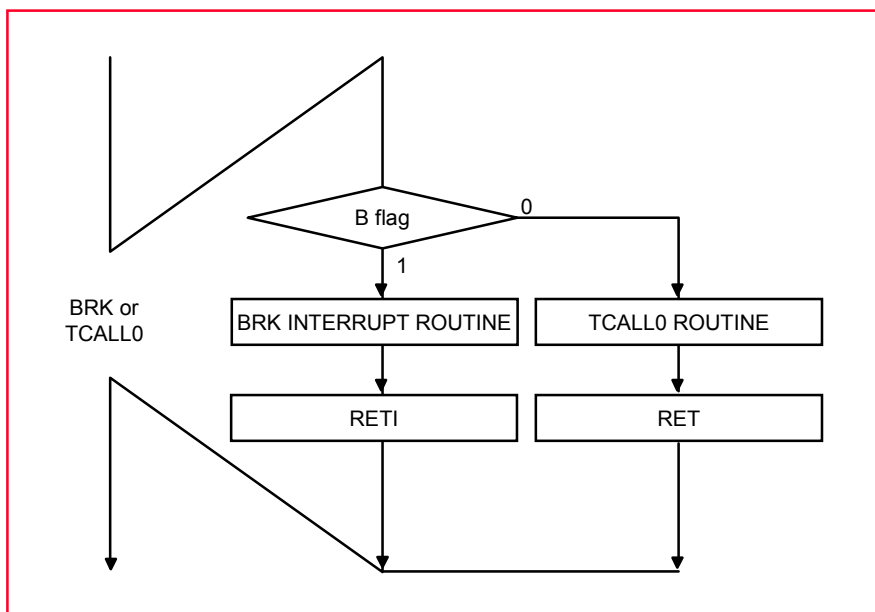


Figure 12-4 Execution of BRK or TCALL0

12.6 MULTIPLE INTERRUPT

If there is an interrupt, Interrupt Mask Enable Flag is automatically cleared before entering the Interrupt Service Routine. 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.

12.7 Key Scan Input Processing

(1) Standby Mode Release Register (SMRR)

Key Scan Interrupt is generated by detecting low or high Input from each Input pin (R0, R1) is one of the sources which release standby (SLEEP, STOP) mode. Key Scan ports are all 16bit which are controlled by Standby Mode Release Register (SMRR0, SMRR1). Key Input is considered as Interrupt, therefore, KSCNE bit of IEHN should be

set for correct interrupt executing, SLEEP mode and STOP mode, the rest of executing is the same as that of external Interrupt. Each SMRR Register bit is allowed for each port (for Bit= '0', no Key Input, for Bit= '1', Key Input available). At reset, SMRR becomes '00 h'. So, there is no Key Input source.

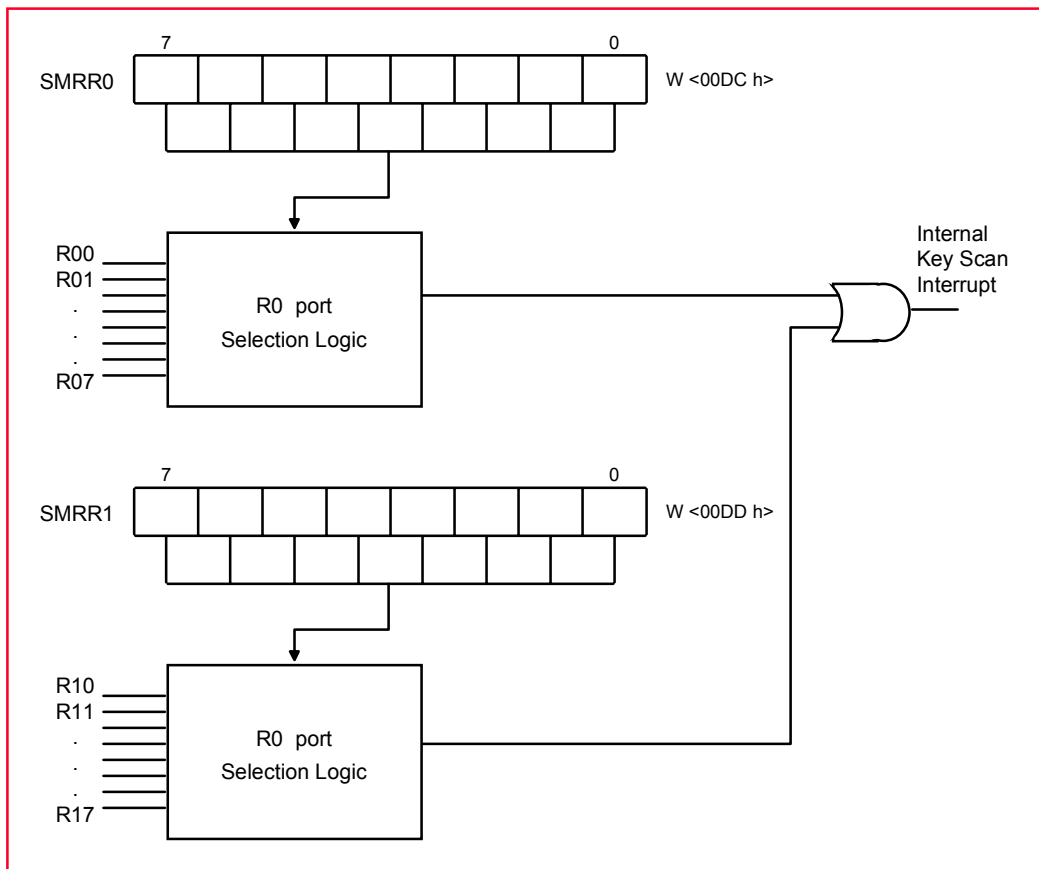
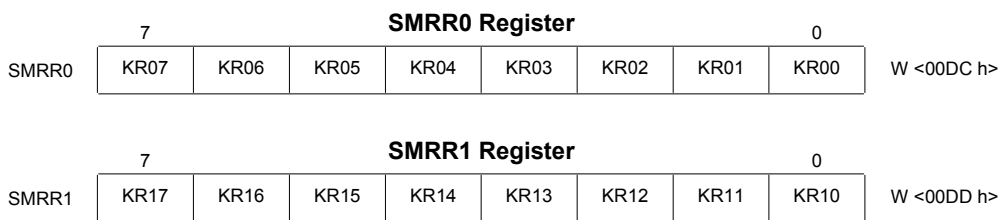
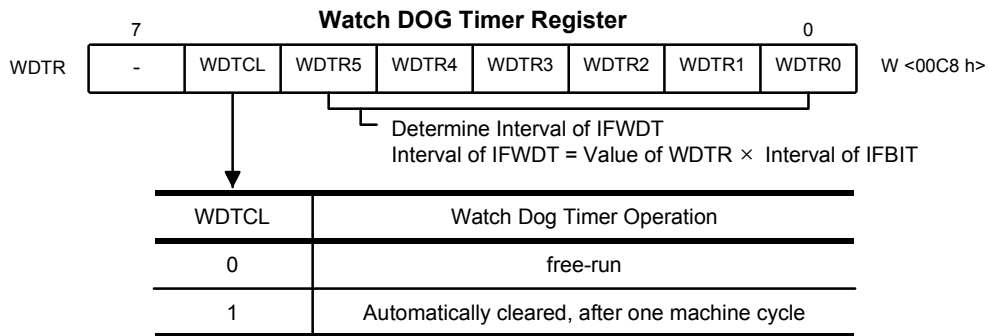


Figure 12-5 Key Scan Block



| 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 |



13.2 WDT 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, Input clock cycle is possible 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 WDT overflow.

-Reset WDTR value = 0F h,15

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

(about 1second)

14. STANDBY FUNCTION

To save power consumption, there is STOP modes. In this

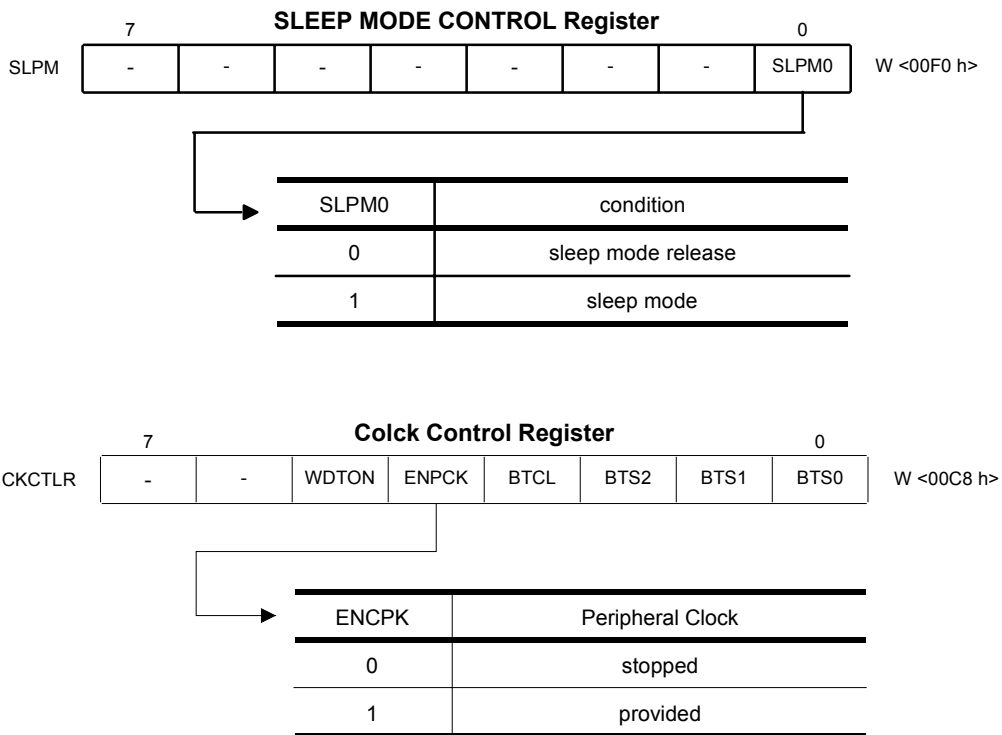
modes, the execution of program stops.

14.1 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's output which provide clock to peripherals can be stopped by program. (Except, PS10 can't stopped.) In SLEEP mode, more consuming power can be saved by not using other peripheral hardware except for B.I.T. By setting ENPCK (peripheral clock control bit) of CKCTRLR (clock control register) to '0', peripheral hardware halted, and SLEEP mode is entered. To

release SLEEP mode by BINTR (basic interval timer interrupt), bit10 of prescaler should be selected as B.I.T input clock before entering SLEEP mode. 'NOP' instruction should be follows setting of SLEEP mode for rising pre-charge time of data bus line.

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



14.2 STOP MODE

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

time of Data Bus line.

(ex) STOP : STOP instruction execution
NOP : NOP instruction

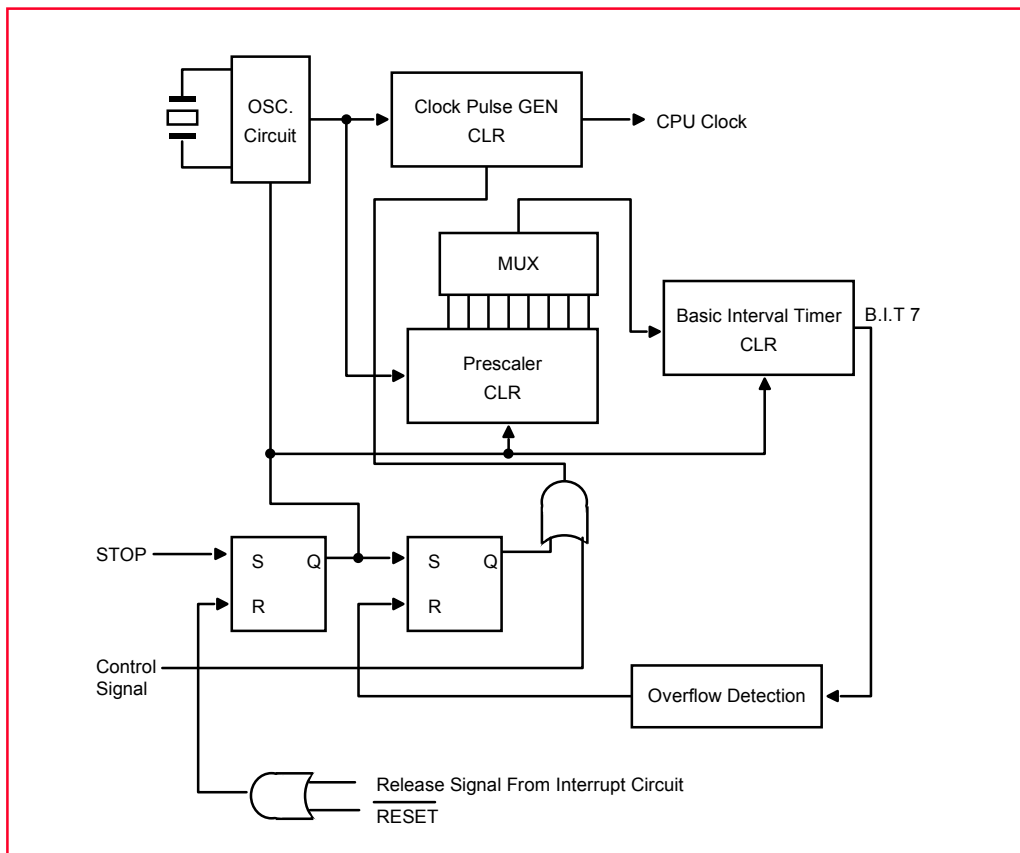


Figure 14-1 Block Diagram of Standby Circuit

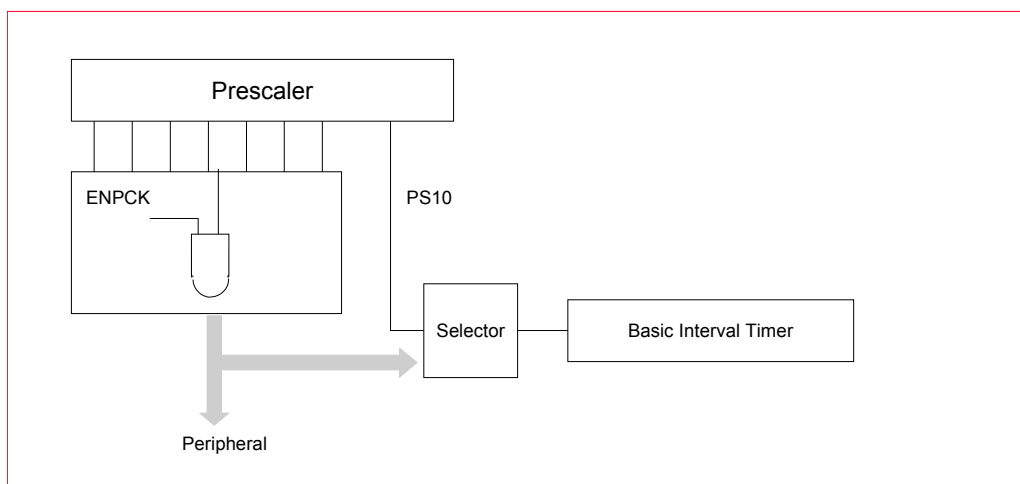


Figure 14-2 ENPCK and Basic Interval Timer Clock

14.3 STANDBY MODE RELEASE

Release of STANDBY mode is executed by RESET input and Interrupt signal. Register value is defined when Reset. When there is a release signal of STOP mode (Interrupt,

RESET input), the instruction execution starts after stabilization oscillation time is set by value of BTS2 ~ BTS0 and set ENPCK to ``1``.

| Release Signal | SLEEP | STOP |
|------------------|-------|------|
| RESET | O | O |
| KSCN (key input) | O | O |
| INT1 , INT2 | O | O |
| B.I.T | O | X |

Table 14-1 Standby Mode Register

| Release Factor | Release Method |
|---------------------------------|---|
| RESET | By 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 14-2 Standby Mode Release

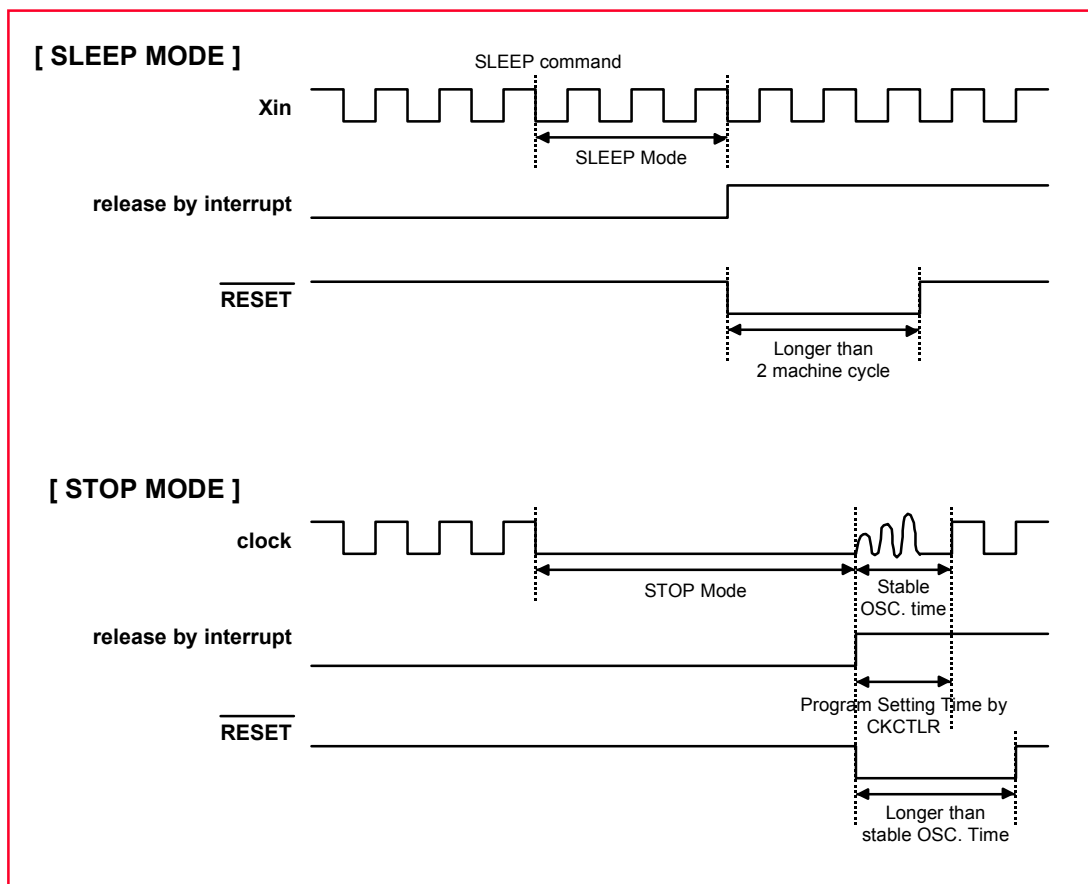


Figure 14-3 Release Timing of Standby Mode

14.4 RELEASE OPERATION OF STANDBY MODE

After standby mode is released, the operation begins according to content of related interrupt register just before standby mode start (Figure 14-4)

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

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

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

Released by only interrupt which each interrupt enable flag

= ``1``, and jump to the relevant interrupt service routine.

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

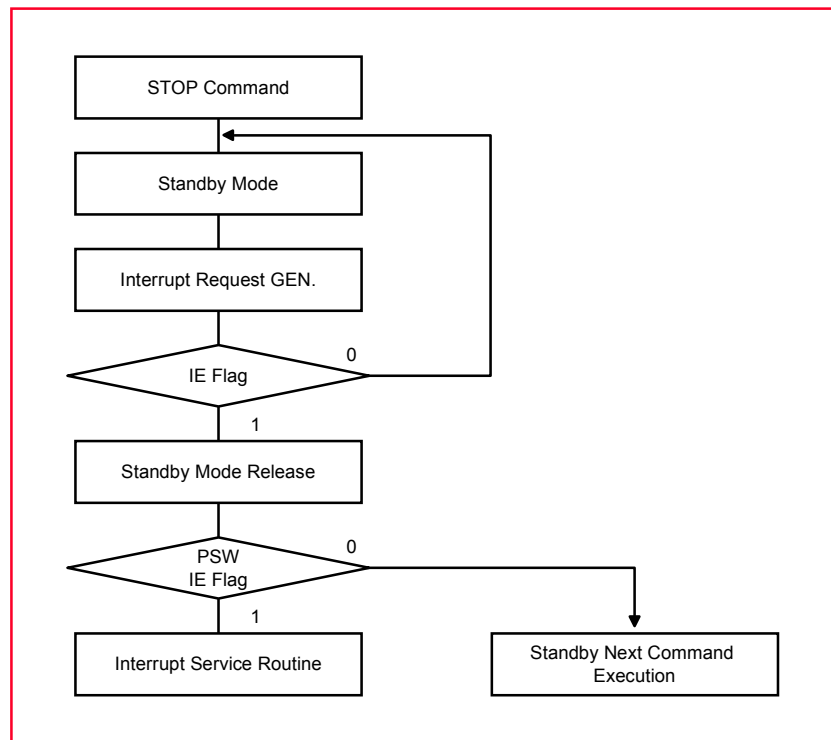


Figure 14-4 Standby Mode Release Flow

| 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 14-3 Operation State in Standby Mode

15. OSCILLATION CIRCUIT

Oscillation circuit is designed to be used either with 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 an 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 in Fig. 4.2-(b). In the Standby (STOP) mode, oscillation stop, Xout state goes to ``High``, Xin state goes to ``Low``, and built-in feedback resistor is disabled.

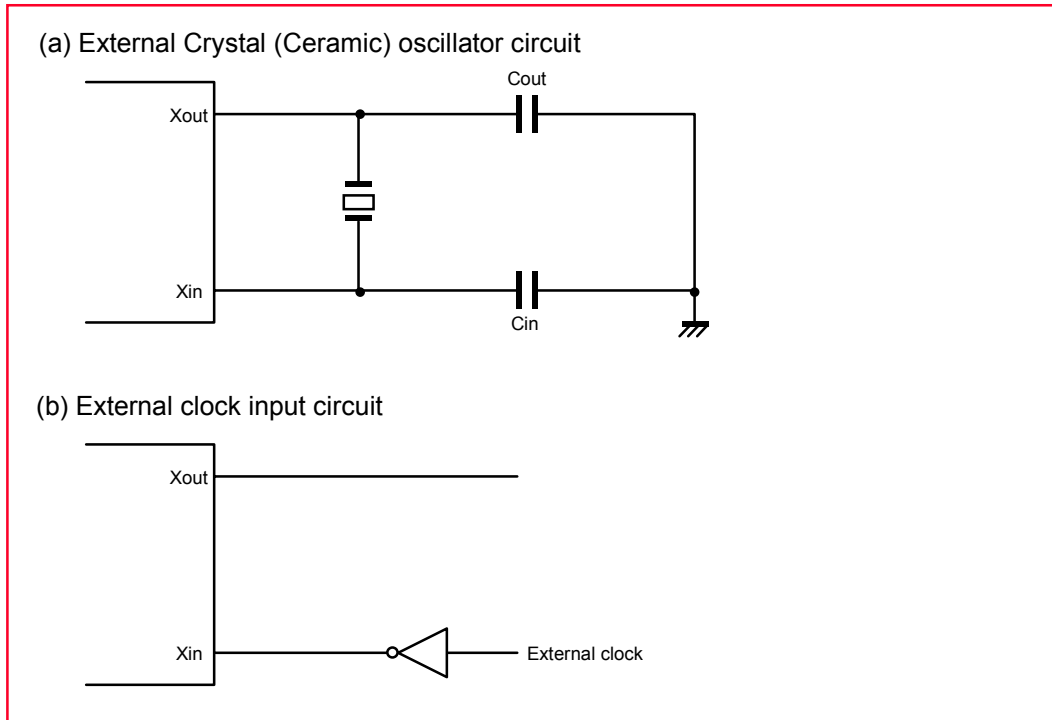


Figure 15-1 Oscillator configurations

* 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 building in load capacitor.CCR type is chip type.

16. RESET FUNCTION

16.1 EXTERNAL RESET

The RESET pin should be held at low for at least 2 machine cycles with the power supply voltage within the operating voltage range and must be connected 0.1uF capacitor for

stable system initialization. The RESET pin contains a Schmitt trigger with an internal pull-up resistor.

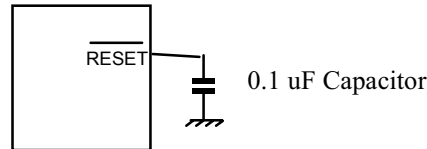


Figure 16-1

16.2 POWER ON RESET

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

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

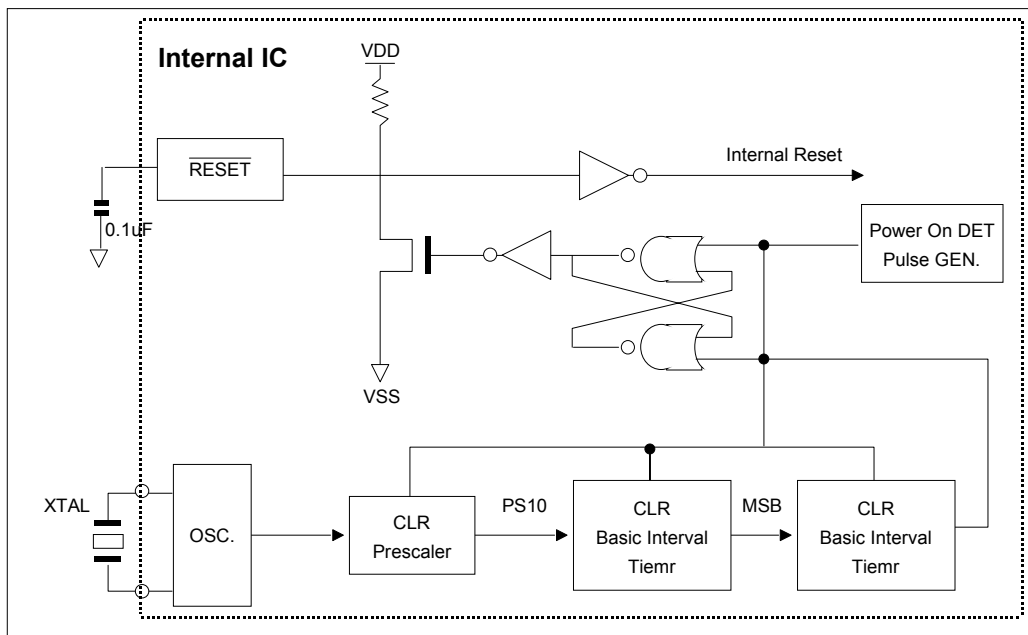


Figure 16-2 Block Diagram of Power On Reset Circuit

Note: Notice ; When Power On Reset, oscillator stabilization time doesn't include OSC. Start time.

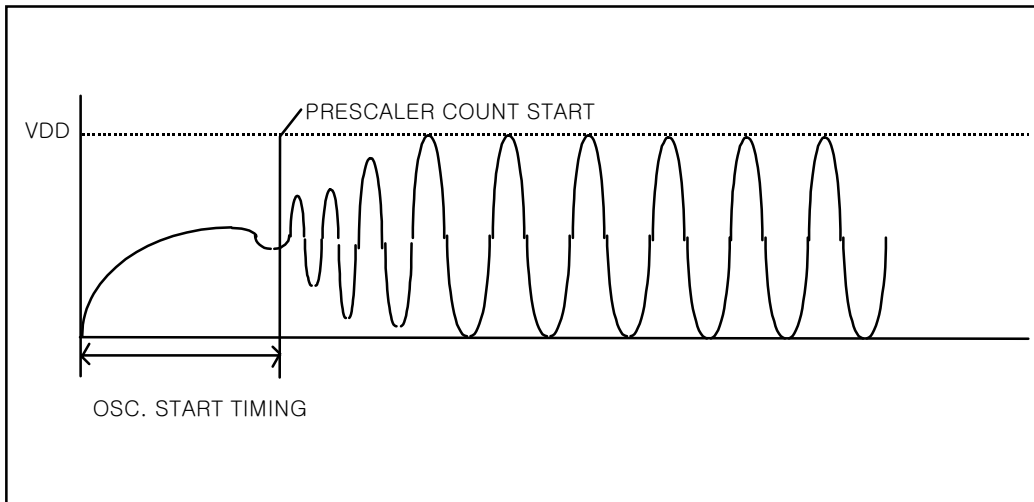


Figure 16-3 Oscillator stabilization diagram

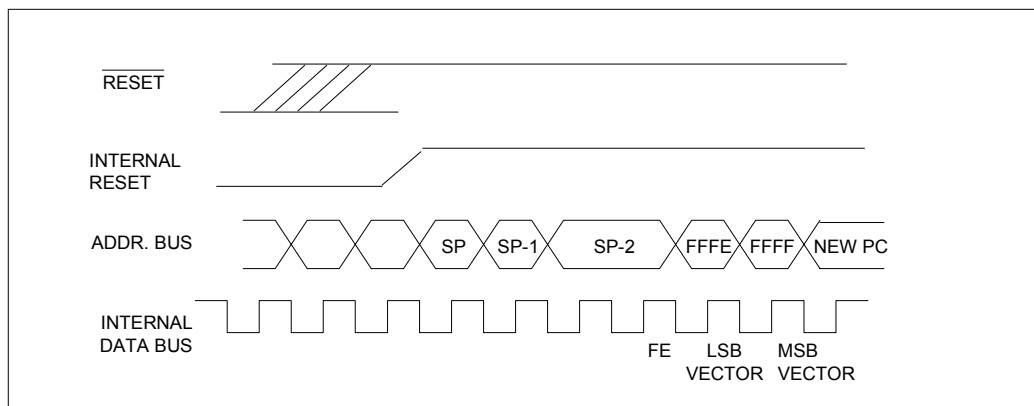


Figure 16-4 Reset Timing by Diagram

16.3 Low Voltage Detection Mode

(1) Low voltage detection condition

An on board voltage comparator checks that 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

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

tor) is selected.

(3) Release of Low Voltage Detection Mode

Reset signal result from new battery(normally 3V) wakes

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

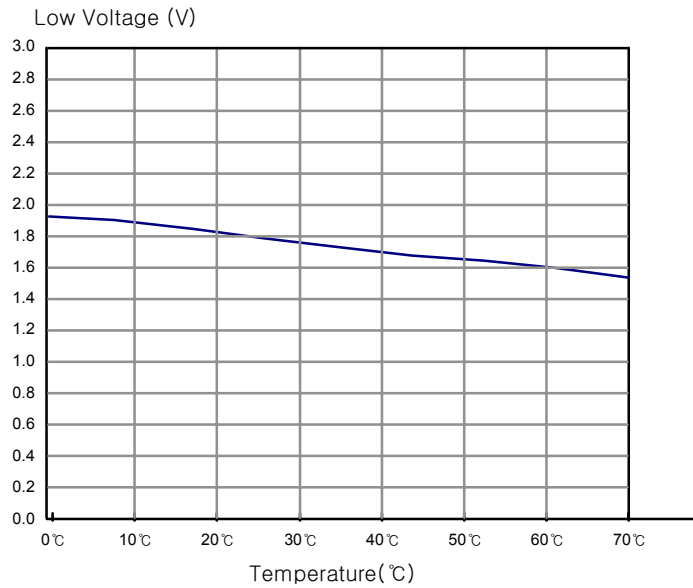


Figure 16-5 Low Voltage vs Temperature

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

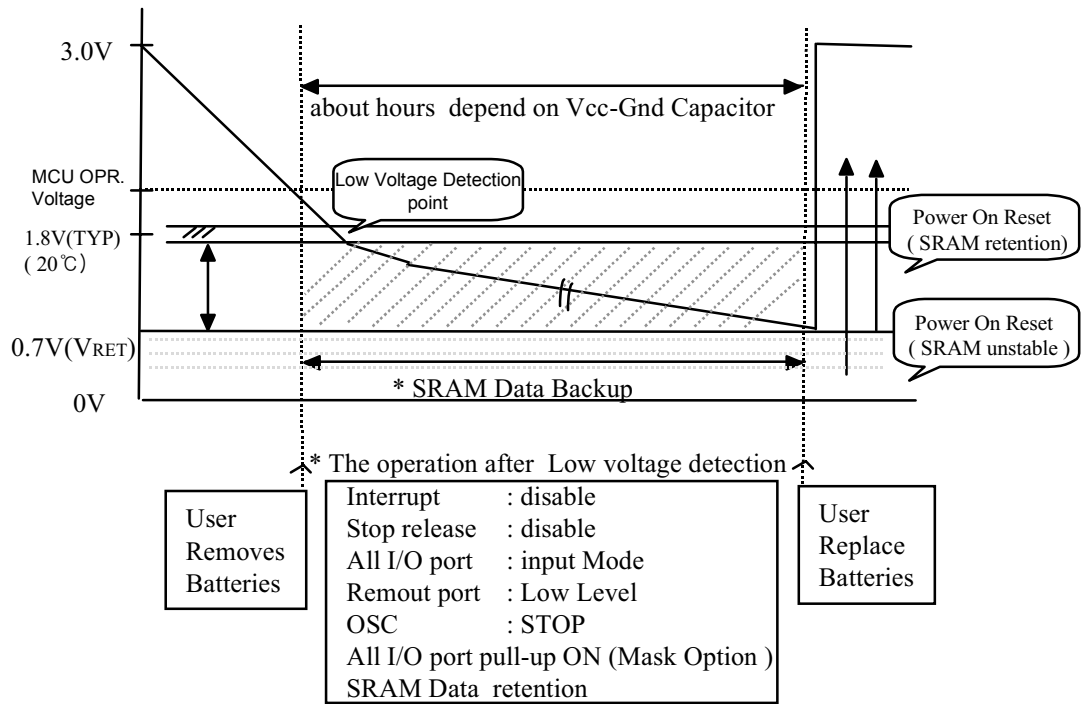


Figure 16-6 Low Voltage Detection and Protection

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

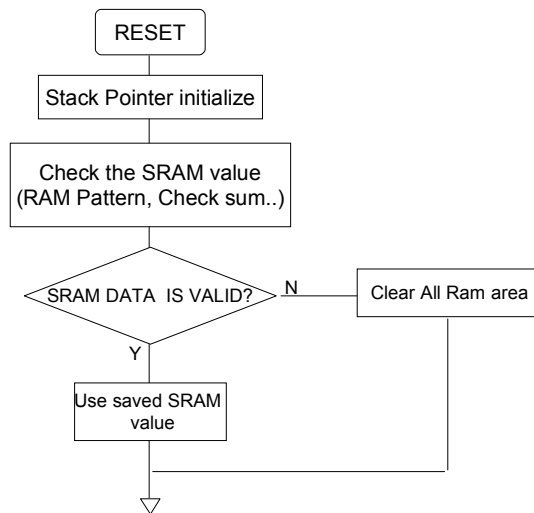


Figure 16-7 S/W Flow Chart Example for SRAM Back-up

16.4 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 16-6),

The bit of LVIR register could be set according to the VDD level sequentially. The VDD dection levels for Indication are two , that is , Bit1 and Bit0 of LVIR Register. The de-tection level of Bit0 is higer than Bit1.

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

Appendix A. Hynix 800 Series Instruction

1. Instruction Map

| LOW HIGH | 0000 00 | 00001 01 | 00010 02 | 00011 03 | 00100 04 | 00101 05 | 00110 06 | 00111 07 | 01000 08 | 01001 09 | 01010 0A | 01011 0B | 01100 0C | 01101 0D | 01110 0E | 01111 0F |
|-------------|------------|----------------|------------------|-------------------|----------------|-------------|-------------|-------------|-------------|-------------|-------------|---------------|-------------|--------------|-------------|----------------|
| 000 | | SET1 dp.bit | BBS A.bit,rel | BBS dp.bit,rel | ADC #imm | ADC dp | ADC dp+X | ADC labs | ASL A | ASL dp | TCALL 0 | SETA1 .bit | BIT dp | POP A | PUSH A | BRK |
| 001 | CLRC | // | // | // | SBC #imm | SBC dp | SBC dp+X | SBC labs | ROL A | ROL dp | TCALL 2 | CLRA1 .bit | COM dp | POP X | PUSH X | BRA rel |
| 010 | CLRG | // | // | // | CMP #imm | CMP dp | CMP dp+X | CMP labs | LSR A | LSR dp | TCALL 4 | NOT1 M.bit | TST dp | POP Y | PUSH Y | PCALL Upage |
| 011 | DI | // | // | // | OR #imm | OR dp | OR dp+X | OR labs | ROR A | ROR dp | TCALL 6 | OR1 OR1B | CMPX dp | POP PSW | PUSH PSW | RET |
| 100 | CLRV | // | // | // | AND #imm | AND dp | AND dp+X | AND labs | INC A | INC dp | TCALL 8 | AND1 AND1B | CMPY dp | CBNE dp+X | TXSP | INC X |
| 101 | SETC | // | // | // | EOR #imm | EOR dp | EOR dp+X | EOR labs | DEC A | DEC dp | TCALL 10 | EOR1 EOR1B | DBNE dp | XMA dp+X | TSPX | DEC X |
| 110 | SETG | // | // | // | LDA #imm | LDA dp | LDA dp+X | LDA labs | TXA | LDY dp | TCALL 12 | LDC LDCB | LDX dp | LDX dp+Y | XCN | DAS |
| 111 | EI | // | // | // | LDM dp,#imm | STA dp | STA dp+X | STA labs | TAX | STY dp | TCALL 14 | STC M.bit | STX dp | STX dp+Y | XAS | STOP |

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

Appendix A. Hynix 800 Series Instruction

2. Alphabetic order table of instruction

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|----------------|---------|----------|----------|--|------------------|
| 1 | ADC #imm | 04 | 2 | 2 | Add with carry. $A \leftarrow A + (M) + C$ | NV -- H - ZC |
| 2 | ADC dp | 05 | 2 | 3 | | |
| 3 | ADC dp + X | 06 | 2 | 4 | | |
| 4 | ADC !abs | 07 | 3 | 4 | | |
| 5 | ADC !abs+Y | 15 | 3 | 5 | | |
| 6 | ADC [dp+X] | 16 | 2 | 6 | | |
| 7 | ADC [dp]+Y | 17 | 2 | 6 | | |
| 8 | ADC {X} | 14 | 1 | 3 | | |
| 9 | ADDW dp | 1D | 2 | 5 | 16-bits add without carry : $YA \leftarrow YA + (dp+1)(dp)$ | NV -- H - ZC |
| 10 | AND #imm | 84 | 2 | 2 | Logical AND $A \leftarrow A \wedge (M)$ | N - - - - - Z - |
| 11 | AND dp | 85 | 2 | 3 | | |
| 12 | AND dp + X | 86 | 2 | 4 | | |
| 13 | AND !abs | 87 | 3 | 4 | | |
| 14 | AND !abs+Y | 95 | 3 | 5 | | |
| 15 | AND [dp+X] | 96 | 2 | 6 | | |
| 16 | AND [dp] + Y | 97 | 2 | 6 | | |
| 17 | AND {X} | 94 | 1 | 3 | | |
| 18 | AND1 M.bit | 8B | 3 | 4 | Bit AND C-flag : $C \leftarrow C \wedge (M.bit)$ | - - - - - C |
| 19 | AND1B M.bit | 8B | 3 | 4 | Bit AND C-flag and NOT : $C \leftarrow C \wedge \sim(M.bit)$ | - - - - - C |
| 20 | ASL A | 08 | 1 | 2 | Arithmetic shift left $ \begin{array}{cccccccc} & C & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ \square & \leftarrow & \leftarrow & \leftarrow & \leftarrow & \leftarrow & \leftarrow & \leftarrow & \leftarrow & \leftarrow \\ & & & & & & & & & \leftarrow "0" \end{array} $ | N - - - - - ZC |
| 21 | ASL dp | 09 | 2 | 4 | | |
| 22 | ASL dp + X | 19 | 2 | 5 | | |
| 23 | ASL !abs | 18 | 3 | 5 | | |
| 24 | BBC A.bit,rel | y2 | 2 | 4/6 | Branch if bit clear : if(bit) = 0, then $PC \leftarrow PC + rel$ | - - - - - |
| 25 | BBC dp.bit,rel | y3 | 3 | 5/7 | | |
| 26 | BBS A.bit,rel | x2 | 2 | 4/6 | Branch if bit clear : if(bit) = 1, then $PC \leftarrow PC + rel$ | - - - - - |
| 27 | BBS dp.bit,rel | x3 | 3 | 5/7 | | |
| 28 | BCC rel | 50 | 2 | 2/4 | Branch if carry bit clear : if(C) = 0, then $PC \leftarrow PC + rel$ | MM - - - - - Z - |
| 29 | BCS rel | D0 | 2 | 2/4 | Branch if carry bit set : If (C) = 1, then $PC \leftarrow PC + rel$ | - - - - - |
| 30 | BEQ rel | F0 | 2 | 2/4 | Branch if equal : if (Z) = 1, then $PC \leftarrow PC + rel$ | - - - - - |
| 31 | BIT dp | 0C | 2 | 4 | Bit test A with memory : $Z \leftarrow A \wedge M, N \leftarrow (M_7), V \leftarrow (M_6)$ | MM - - - - - Z - |
| 32 | BIT !abs | 1C | 3 | 5 | | |
| 33 | BMI rel | 90 | 2 | 2/4 | Branch if minus : if (N) = 1, then $PC \leftarrow PC + rel$ | - - - - - |
| 34 | BNE rel | 70 | 2 | 2/4 | Branch if not equal : if (Z) = 0, then $PC \leftarrow PC + rel$ | - - - - - |
| 35 | BPL rel | 10 | 2 | 2/4 | Branch if not minus : if (N) = 0, then $PC \leftarrow PC + rel$ | - - - - - |
| 36 | BRA rel | 2F | 2 | 4 | Branch always : $PC \leftarrow PC + rel$ | - - - - - |
| 37 | BRK | 0F | 1 | 8 | Software interrupt: $B \leftarrow "1", M(SP) \leftarrow (PC_H), SP \leftarrow SP - 1,$ $M(s) \leftarrow (PC_L), SP \leftarrow S - 1, M(SP) \leftarrow PSW,$ $SP \leftarrow SP - 1, PC_L \leftarrow (0FFDE_H), PC_H \leftarrow (0FFDF_H)$ | - - - 1 - 0 - - |
| 38 | BVC rel | 30 | 2 | 2/4 | Branch if overflow bit clear : If (V) = 0, then $PC \leftarrow PC + rel$ | - - - - - |
| 39 | BVS rel | B0 | 2 | 2/4 | Branch if overflow bit set : If (V) = 1, then $PC \leftarrow PC + rel$ | - - - - - |

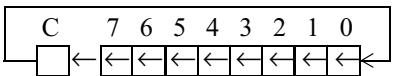
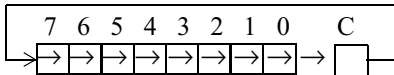
Appendix A. Hynix 800 Series Instruction

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|------------------|---------|----------|----------|--|---------------|
| 40 | CALL !abs | 3B | 3 | 8 | Subroutine call | |
| 41 | CALL [dp] | 5F | 2 | 8 | $M(SP) \leftarrow (PC_H)$, $SP \leftarrow SP-1$, $M(SP) \leftarrow (PC_L)$, $SP \leftarrow SP-1$ if !abs, $PC \leftarrow abs$; if [dp], $PC_L \leftarrow (dp)$, $PC_H \leftarrow (dp+1)$ | ----- |
| 42 | CBNE dp,rel | FD | 3 | 5/7 | Compare and branch if not equal ; | ----- |
| 43 | CBNE dp + X, rel | 8D | 3 | 6/8 | If $A \neq (M)$, then $PC \leftarrow PC + rel$. | ----- |
| 44 | CLR1 dp.bit | y1 | 2 | 4 | Clear bit : $(M.bit) \leftarrow "0"$ | ----- |
| 45 | CLR1A A.bit | 2B | 2 | 2 | Clear A.bit : $(A.bit) \leftarrow "0"$ | ----- |
| 46 | CLRC | 20 | 1 | 2 | Clear C-flag : $C \leftarrow "0"$ | -----0 |
| 47 | CLRG | 40 | 1 | 2 | Clear G-flag : $G \leftarrow "0"$ | --0----- |
| 48 | CLRV | 80 | 1 | 2 | Clear V-flag : $V \leftarrow "0"$ | -0--0--- |
| 49 | CMP #imm | 44 | 2 | 2 | Compare accumulator contents with memory contents | |
| 50 | CMP dp | 45 | 2 | 3 | A - (M) | |
| 51 | CMP dp + X | 46 | 2 | 4 | | |
| 52 | CMP !abs | 47 | 3 | 4 | | |
| 53 | CMP !abs + Y | 55 | 3 | 5 | | |
| 54 | CMP [dp + X] | 56 | 2 | 6 | | |
| 55 | CMP [dp] + Y | 57 | 2 | 6 | | |
| 56 | CMP {X} | 54 | 1 | 3 | | |
| 57 | CMPW dp | 5D | 2 | 4 | Compare YA contents with memory pair contents : $YA - (dp+1)(dp)$ | N-----ZC |
| 58 | CMPX #imm | 5E | 2 | 2 | Compare X contents with memory contents | |
| 59 | CMPX dp | 6C | 2 | 3 | X - (M) | |
| 60 | CMPX !abs | 7C | 3 | 4 | | N-----ZC |
| 61 | CMPY #imm | 7E | 2 | 2 | Compare Y contents with memory contents | |
| 62 | CMPY dp | 8C | 2 | 3 | Y - (M) | |
| 63 | CMPY !abs | 9C | 3 | 4 | | N-----ZC |
| 64 | COM dp | 2C | 2 | 4 | 1's complement : $(dp) \leftarrow \sim(dp)$ | N-----Z- |
| 65 | DAA | DF | 1 | 3 | Decimal adjust for addition | N-----ZC |
| 66 | DAS | CF | 1 | 3 | Decimal adjust for subtraction | N-----ZC |
| 67 | DBNE dp,rel | AC | 3 | 5/7 | Decrement and branch if not equal : | ----- |
| 68 | DBNE Y,rel | 7B | 2 | 4/6 | if $(M) \neq 0$, then $PC \leftarrow PC + rel$. | ----- |
| 69 | DEC A | A8 | 1 | 2 | Decrement | |
| 70 | DEC dp | A9 | 2 | 4 | $M \leftarrow M - 1$ | |
| 71 | DEC dp + X | B9 | 2 | 5 | | |
| 72 | DEC !abs | B8 | 3 | 5 | | |
| 73 | DEC X | AF | 1 | 2 | | |
| 74 | DEC Y | BE | 1 | 2 | | |
| 75 | DECW dp | BD | 2 | 6 | Decrement memory pair : $(dp+1)(dp) \leftarrow \{(dp+1)(dp)\} - 1$ | N-----Z- |
| 76 | DI | 60 | 1 | 3 | Disable interrupts : $I \leftarrow "0"$ | -----0-- |
| 77 | DIV | 9B | 1 | 12 | Divide : $YA/X \leftarrow Q:A$, $R:Y$ | NV--H-Z- |
| 78 | EI | E0 | 1 | 3 | Enable interrupts : $I \leftarrow "1"$ | -----1-- |

Appendix A. Hynix 800 Series Instruction

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC | | | | | | | | | | |
|-----|---------------|---------|----------|----------|---|-----------------|---|---|---|---|---|---|---|---|---|----------------|
| 79 | EOR #imm | A4 | 2 | 2 | Exclusive OR $A \leftarrow A \oplus (M)$ | N - - - - - Z - | | | | | | | | | | |
| 80 | EOR dp | A5 | 2 | 3 | | | | | | | | | | | | |
| 81 | EOR dp + X | A6 | 2 | 4 | | | | | | | | | | | | |
| 82 | EOR !abs | A7 | 3 | 4 | | | | | | | | | | | | |
| 83 | EOR !abs + Y | B5 | 3 | 5 | | | | | | | | | | | | |
| 84 | EOR [dp + X] | 96 | 2 | 6 | | | | | | | | | | | | |
| 85 | EOR [dp] + Y | 97 | 2 | 6 | | | | | | | | | | | | |
| 86 | EOR {X} | 94 | 1 | 3 | | | | | | | | | | | | |
| 87 | EOR1 M.bit | AB | 3 | 5 | Bit exclusive-OR C-flag : $C \leftarrow C \oplus (M.bit)$ | - - - - - C | | | | | | | | | | |
| 88 | EOR1B M.bit | AB | 3 | 5 | Bit exclusive-OR C-flag and NOT : $C \leftarrow C \oplus \sim(M.bit)$ | - - - - - C | | | | | | | | | | |
| 89 | INC A | 88 | 1 | 2 | Increment $(M) \leftarrow (M) + 1$ | N - - - - - ZC | | | | | | | | | | |
| 90 | INC dp | 89 | 2 | 4 | | | | | | | | | | | | |
| 91 | INC dp + X | 99 | 2 | 5 | | | | | | | | | | | | |
| 92 | INC !abs | 98 | 3 | 5 | | | | | | | | | | | | |
| 93 | INC X | 8F | 1 | 2 | | | | | | | | | | | | |
| 94 | INC Y | 9E | 1 | 2 | | | | | | | | | | | | |
| 95 | INCW dp | 9D | 2 | 6 | Increment memory pair : $(dp+1)(dp) \leftarrow \{(dp+1)(dp)\} + 1$ | N - - - - - Z - | | | | | | | | | | |
| 96 | JMP !abs | 1B | 3 | 3 | Unconditional jump $PC \leftarrow$ jump address | - - - - - | | | | | | | | | | |
| 97 | JMP [!abs] | 1F | 3 | 5 | | | | | | | | | | | | |
| 98 | JMP [dp] | 3F | 2 | 4 | | | | | | | | | | | | |
| 99 | LDA #imm | C4 | 2 | 2 | Load accumulator $A \leftarrow (M)$ | N - - - - - Z - | | | | | | | | | | |
| 100 | LDA dp | C5 | 2 | 3 | | | | | | | | | | | | |
| 101 | LDA dp + X | C6 | 2 | 4 | | | | | | | | | | | | |
| 102 | LDA !abs | C7 | 3 | 4 | | | | | | | | | | | | |
| 103 | LDA !abs + Y | D5 | 3 | 5 | | | | | | | | | | | | |
| 104 | LDA [dp + X] | D6 | 2 | 6 | | | | | | | | | | | | |
| 105 | LDA [dp]+Y | D7 | 2 | 6 | | | | | | | | | | | | |
| 106 | LDA {X} | D4 | 1 | 3 | | | | | | | | | | | | |
| 107 | LDA {X}+ | DB | 1 | 4 | X-register auto-increment : $A \leftarrow (M), X \leftarrow X + 1$ | | | | | | | | | | | |
| 108 | LDC M.bit | CB | 3 | 4 | Load C-flag : $C \leftarrow (M.bit)$ | - - - - - C | | | | | | | | | | |
| 109 | LDCB M.bit | CB | 3 | 4 | Load C-flag with NOT : $C \leftarrow \sim(M.bit)$ | - - - - - C | | | | | | | | | | |
| 110 | LDM dp,#imm | E4 | 3 | 5 | Load memory with immediate data : $(M) \leftarrow imm$ | - - - - - | | | | | | | | | | |
| 111 | LDX #imm | 1E | 2 | 2 | Load X-register $X \leftarrow (M)$ | N - - - - - Z - | | | | | | | | | | |
| 112 | LDX dp | CC | 2 | 3 | | | | | | | | | | | | |
| 113 | LDX dp + Y | CD | 2 | 4 | | | | | | | | | | | | |
| 114 | LDX !abs | DC | 3 | 4 | | | | | | | | | | | | |
| 115 | LDY #imm | 3E | 2 | 2 | Load X-register $Y \leftarrow (M)$ | N - - - - - Z - | | | | | | | | | | |
| 116 | LDY dp | C9 | 2 | 3 | | | | | | | | | | | | |
| 117 | LDY dp + Y | D9 | 2 | 4 | | | | | | | | | | | | |
| 118 | LDY !abs | D8 | 3 | 4 | | | | | | | | | | | | |
| 119 | LDYA dp | 7D | 2 | 5 | Load YA : $YA \leftarrow (dp+1)(dp)$ | N - - - - - Z - | | | | | | | | | | |
| 120 | LSR A | 48 | 1 | 2 | Logical shift right "0" → <table style="display: inline-table; border-collapse: collapse; vertical-align: middle;"> <tr> <td style="border: 1px solid black; padding: 2px;">7</td> <td style="border: 1px solid black; padding: 2px;">6</td> <td style="border: 1px solid black; padding: 2px;">5</td> <td style="border: 1px solid black; padding: 2px;">4</td> <td style="border: 1px solid black; padding: 2px;">3</td> <td style="border: 1px solid black; padding: 2px;">2</td> <td style="border: 1px solid black; padding: 2px;">1</td> <td style="border: 1px solid black; padding: 2px;">0</td> <td style="padding: 0 5px;">→</td> <td style="border: 1px solid black; padding: 2px;">C</td> </tr> </table> | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | → | C | N - - - - - ZC |
| 7 | 6 | 5 | 4 | 3 | | 2 | 1 | 0 | → | C | | | | | | |
| 121 | LSR dp | 49 | 2 | 4 | | | | | | | | | | | | |
| 122 | LSR dp + X | 59 | 2 | 5 | | | | | | | | | | | | |
| 123 | LSR !abs | 58 | 3 | 5 | | | | | | | | | | | | |

Appendix A. Hynix 800 Series Instruction

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|--------------|---------|----------|----------|---|-----------------------------|
| 124 | MUL | 5B | 1 | 9 | Multiply : $YA \leftarrow Y \times A$ | N - - - - - Z - |
| 125 | NOP | FF | 1 | 2 | No operation | - - - - - |
| 126 | NOT1 M.bit | 4B | 3 | 5 | Bit complement : $(M.bit) \leftarrow \sim(M.bit)$ | - - - - - |
| 127 | OR #imm | 64 | 2 | 2 | Logical OR $A \leftarrow A \vee (M)$ | N - - - - - Z - |
| 128 | OR dp | 65 | 2 | 3 | | |
| 129 | OR dp + X | 66 | 2 | 4 | | |
| 130 | OR !abs | 67 | 3 | 4 | | |
| 131 | OR !abs + Y | 75 | 3 | 5 | | |
| 132 | OR [dp + X] | 76 | 2 | 6 | | |
| 133 | OR [dp] + Y | 77 | 2 | 6 | | |
| 134 | OR {X} | 74 | 1 | 3 | | |
| 135 | OR1 M.bit | 6B | 3 | 5 | Bit OR C-flag : $C \leftarrow C \vee (M.bit)$ | - - - - - C |
| 136 | OR1B M.bit | 6B | 3 | 5 | Bit OR C-flag and NOT : $C \leftarrow C \vee \sim(M.bit)$ | - - - - - C |
| 137 | PCALL | 4F | 2 | 6 | U-page call : $M(SP) \leftarrow (PC_H)$, $SP \leftarrow SP - 1$, $M(SP) \leftarrow (PC_L)$, $SP \leftarrow SP - 1$, $PC_L \leftarrow (upage)$, $PC_H \leftarrow "OFF_H"$ | - - - - - |
| 138 | POP A | 0D | 1 | 4 | Pop from stack $SP \leftarrow SP + 1$, Reg. $\leftarrow M(SP)$ | - - - - - (restored) |
| 139 | POP X | 2D | 1 | 4 | | |
| 140 | POP Y | 4D | 1 | 4 | | |
| 141 | POP PSW | 6D | 1 | 4 | | |
| 142 | PUSH A | 0E | 1 | 4 | Push to stack $M(SP) \leftarrow \text{Reg.}$ $SP \leftarrow SP - 1$ | - - - - - |
| 143 | PUSH X | 2E | 1 | 4 | | |
| 144 | PUSH Y | 4E | 1 | 4 | | |
| 145 | PUSH PSW | 6E | 1 | 4 | | |
| 146 | RET | 6F | 1 | 5 | Return from subroutine : $SP \leftarrow SP + 1$, $PC_L \leftarrow M(SP)$, $SP \leftarrow SP + 1$, $PC_H \leftarrow M(SP)$ | - - - - - |
| 147 | RETI | 7F | 1 | 6 | Return from interrupt : $SP \leftarrow SP + 1$, $PSW \leftarrow M(SP)$, $SP \leftarrow SP + 1$, $PC_L \leftarrow M(SP)$, $SP \leftarrow SP + 1$, $PC_H \leftarrow M(SP)$ | (restored) |
| 148 | ROL A | 28 | 1 | 2 | Rotate left through carry  | N - - - - - ZC |
| 149 | ROL dp | 29 | 2 | 4 | | |
| 150 | ROL dp + X | 39 | 2 | 5 | | |
| 151 | ROL !abs | 38 | 3 | 5 | | |
| 152 | ROR A | 68 | 1 | 2 | Rotate right through carry  | N - - - - - ZC |
| 153 | ROR dp | 69 | 2 | 4 | | |
| 154 | ROR dp + X | 79 | 2 | 5 | | |
| 155 | ROR !abs | 78 | 3 | 5 | | |
| 156 | SBC #imm | 24 | 2 | 2 | Subtract with carry $A \leftarrow A - (M) - \sim(C)$ | NV - - - HZC |
| 157 | SBC dp | 25 | 2 | 3 | | |
| 158 | SBC dp + X | 26 | 2 | 4 | | |
| 159 | SBC !abs | 27 | 3 | 4 | | |
| 160 | SBC !abs + Y | 35 | 3 | 5 | | |
| 161 | SBC [dp + X] | 36 | 2 | 6 | | |
| 162 | SBC [dp] + Y | 37 | 2 | 6 | | |
| 163 | SBC {X} | 34 | 1 | 3 | | |

Appendix A. Hynix 800 Series Instruction

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|--------------|---------|----------|----------|---|---------------|
| 164 | SET1 dp.bit | x1 | 2 | 4 | Set bit : (M.bit) ← "1" | ----- |
| 165 | SETA1 A.bit | 0B | 2 | 2 | Set A.bit : (A.bit) ← "1" | ----- |
| 166 | SETC | A0 | 1 | 2 | Set C-flag : C ← "1" | -----1 |
| 167 | SETG | C0 | 1 | 2 | Set G-flag : G ← "1" | --1----- |
| 168 | STA dp | E5 | 2 | 3 | Store accumulator contents in memory (M) ← A | ----- |
| 169 | STA dp + X | E6 | 2 | 4 | | |
| 170 | STA !abs | E7 | 3 | 4 | | |
| 171 | STA !abs + Y | F5 | 3 | 5 | | |
| 172 | STA [dp + X] | F6 | 2 | 6 | | |
| 173 | STA [dp] + Y | F7 | 2 | 6 | | |
| 174 | STA {X} | F4 | 1 | 3 | | |
| 175 | STA {X}+ | FB | 1 | 4 | X-register auto-increment : (M) ← A, X ← X + 1 | |
| 176 | STC M.bit | EB | 3 | 6 | Store C-flag : (M.bit) ← C | ----- |
| 177 | STOP | 00 | 1 | 3 | Stop mode (halt CPU, stop oscillator) | ----- |
| 178 | STX dp | EC | 2 | 4 | Store X-register contents in memory (M) ← X | ----- |
| 179 | STX dp + Y | ED | 2 | 5 | | |
| 180 | STX !abs | FC | 3 | 5 | | |
| 181 | STY dp | E9 | 2 | 4 | Store Y-register contents in memory (M) ← Y | ----- |
| 182 | STY dp + X | F9 | 2 | 5 | | |
| 183 | STY !abs | F8 | 3 | 5 | | |
| 184 | STYA dp | DD | 2 | 5 | Store YA : (dp+1)(dp) ← YA | ----- |
| 185 | SUBW dp | 3D | 2 | 5 | 16-bits subtract without carry : YA ← YA - (dp+1)(dp) | NV--H-ZC |
| 186 | TAX | E8 | 1 | 2 | Transfer accumulator contents to X-register : X ← A | N-----Z- |
| 187 | TAY | 9F | 1 | 2 | Transfer accumulator contents to Y-register : Y ← A | N-----Z- |
| 188 | TCALL n | nA | 1 | 8 | Table call : M(SP) ← (PC _H), SP ← SP -1, M(SP) ← (PC _L), SP ← SP -1 PC _L ← (Table vector L), PC _H ← (Table vector H) | ----- |
| 189 | TCLR1 !abs | 5C | 3 | 6 | Test and clear bits with A : A - (M), (M) ← (M) ^ ~(A) | N-----Z- |
| 190 | TSET1 !abs | 3C | 3 | 6 | Test and set bits with A : A - (M), (M) ← (M) V (A) | N-----Z- |
| 191 | TSPX | AE | 1 | 2 | Transfer stack-pointer contents to X-register : X ← SP | N-----Z- |
| 192 | TST dp | 4C | 2 | 3 | Test memory contents for negative or zero : (dp) - 00 _H | N-----Z- |
| 193 | TXA | C8 | 1 | 2 | Transfer X-register contents to accumulator : A ← X | N-----Z- |
| 194 | TXSP | 8E | 1 | 2 | Transfer X-register contents to stack-pointer : SP ← X | N-----Z- |
| 195 | TYA | BF | 1 | 2 | Transfer Y-register contents to accumulator : A ← Y | N-----Z- |
| 196 | XAX | EE | 1 | 4 | Exchange X-register contents with accumulator : X fA | ----- |
| 197 | XAY | DE | 1 | 4 | Exchange Y-register contents with accumulator : Y fA | ----- |
| 198 | XCN | CE | 1 | 5 | Exchange nibbles within the accumulator: A ₇ ~ A ₄ f A ₃ ~ A ₀ | N-----Z- |
| 199 | XMA dp | BC | 2 | 5 | Exchange memory contents with accumulator (M) f A | N-----Z- |
| 200 | XMA dp + X | AD | 2 | 6 | | |
| 201 | XMA {X} | BB | 1 | 5 | | |
| 202 | XYX | FE | 1 | 4 | | |

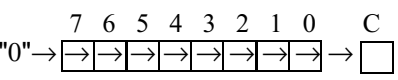
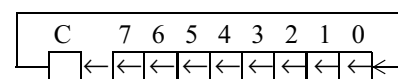
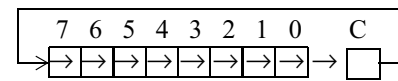
Appendix A. Hynix 800 Series Instruction

2.1 Instruction Table by Function

1. Arithmetic/Logic Operation

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

Appendix A. Hynix 800 Series Instruction

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|--------------|---------|----------|----------|--|---------------------------------------|
| 45 | EOR #imm | A4 | 2 | 2 | Exclusive OR $A \leftarrow A \oplus (M)$ | N - - - - Z - |
| 46 | EOR dp | A5 | 2 | 3 | | |
| 47 | EOR dp + X | A6 | 2 | 4 | | |
| 48 | EOR !abs | A7 | 3 | 4 | | |
| 49 | EOR !abs + Y | B5 | 3 | 5 | | |
| 50 | EOR [dp + X] | 96 | 2 | 6 | | |
| 51 | EOR [dp] + Y | 97 | 2 | 6 | | |
| 52 | EOR {X} | 94 | 1 | 3 | | |
| 53 | INC A | 88 | 1 | 2 | Increment $(M) \leftarrow (M) + 1$ | N - - - - ZC |
| 54 | INC dp | 89 | 2 | 4 | | N - - - - Z - |
| 55 | INC dp + X | 99 | 2 | 5 | | |
| 56 | INC !abs | 98 | 3 | 5 | | |
| 57 | INC X | 8F | 1 | 2 | | |
| 58 | INC Y | 9E | 1 | 2 | | |
| 59 | LSR A | 48 | 1 | 2 | Logical shift right "0" →  | |
| 60 | LSR dp | 49 | 2 | 4 | | |
| 61 | LSR dp + X | 59 | 2 | 5 | | |
| 62 | LSR !abs | 58 | 3 | 5 | | |
| 63 | MUL | 5B | 1 | 9 | | Multiply : $YA \leftarrow Y \times A$ |
| 64 | OR #imm | 64 | 2 | 2 | Logical OR $A \leftarrow A \vee (M)$ | N - - - - Z - |
| 65 | OR dp | 65 | 2 | 3 | | |
| 66 | OR dp + X | 66 | 2 | 4 | | |
| 67 | OR !abs | 67 | 3 | 4 | | |
| 68 | OR !abs + Y | 75 | 3 | 5 | | |
| 69 | OR [dp + X] | 76 | 2 | 6 | | |
| 70 | OR [dp] + Y | 77 | 2 | 6 | | |
| 71 | OR {X} | 74 | 1 | 3 | | |
| 72 | ROL A | 28 | 1 | 2 | Rotate left through carry  | N - - - - ZC |
| 73 | ROL dp | 29 | 2 | 4 | | |
| 74 | ROL dp + X | 39 | 2 | 5 | | |
| 75 | ROL !abs | 38 | 3 | 5 | Rotate right through carry  | N - - - - ZC |
| 76 | ROR A | 68 | 1 | 2 | | |
| 77 | ROR dp | 69 | 2 | 4 | | |
| 78 | ROR dp + X | 79 | 2 | 5 | | |
| 79 | ROR !abs | 78 | 3 | 5 | | |
| 80 | SBC #imm | 24 | 2 | 2 | Subtract with carry $A \leftarrow A - (M) - \sim(C)$ | NV - - HZC |
| 81 | SBC dp | 25 | 2 | 3 | | |
| 82 | SBC dp + X | 26 | 2 | 4 | | |
| 83 | SBC !abs | 27 | 3 | 4 | | |
| 84 | SBC !abs + Y | 35 | 3 | 5 | | |
| 85 | SBC [dp + X] | 36 | 2 | 6 | | |
| 86 | SBC [dp] + Y | 37 | 2 | 6 | | |
| 87 | SBC {X} | 34 | 1 | 3 | | |
| 88 | TST dp | 4C | 2 | 3 | Test memory contents for negative or zero : (dp) - 00 _H | N - - - - Z - |
| 89 | XCN | CE | 1 | 5 | Exchange nibbles within the accumulator: $A_7 \sim A_4 \text{ f } A_3 \sim A_0$ | N - - - - Z - |

Appendix A. Hynix 800 Series Instruction

2. Register / Memory Operation

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

Appendix A. Hynix 800 Series Instruction

3. 16-Bit Operation

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

4. Bit Manipulation

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|-------------|---------|----------|----------|--|------------------|
| 1 | AND1 M.bit | 8B | 3 | 4 | Bit AND C-flag : $C \leftarrow C \wedge (M.bit)$ | - - - - - C |
| 2 | AND1B M.bit | 8B | 3 | 4 | Bit AND C-flag and NOT : $C \leftarrow C \wedge \sim(M.bit)$ | - - - - - C |
| 3 | BIT dp | 0C | 2 | 4 | Bit test A with memory : | MM - - - - - Z - |
| 4 | BIT !abs | 1C | 3 | 5 | $Z \leftarrow A \wedge M, N \leftarrow (M_7), V \leftarrow (M_6)$ | |
| 5 | CLR1 dp.bit | y1 | 2 | 4 | Clear bit : $(M.bit) \leftarrow "0"$ | - - - - - - - |
| 6 | CLR1A A.bit | 2B | 2 | 2 | Clear A.bit : $(A.bit) \leftarrow "0"$ | - - - - - - - |
| 7 | CLRC | 20 | 1 | 2 | Clear C-flag : $C \leftarrow "0"$ | - - - - - 0 |
| 8 | CLRG | 40 | 1 | 2 | Clear G-flag : $G \leftarrow "0"$ | - - 0 - - - - |
| 9 | CLR V | 80 | 1 | 2 | Clear V-flag : $V \leftarrow "0"$ | - 0 - - 0 - - - |
| 10 | EOR1 M.bit | AB | 3 | 5 | Bit exclusive-OR C-flag : $C \leftarrow C \oplus (M.bit)$ | - - - - - C |
| 11 | EOR1B M.bit | AB | 3 | 5 | Bit exclusive-OR C-flag and NOT : $C \leftarrow C \oplus \sim(M.bit)$ | - - - - - C |
| 12 | LDC M.bit | CB | 3 | 4 | Load C-flag : $C \leftarrow (M.bit)$ | - - - - - C |
| 13 | LDCB M.bit | CB | 3 | 4 | Load C-flag with NOT : $C \leftarrow \sim(M.bit)$ | - - - - - C |
| 14 | NOT1 M.bit | 4B | 3 | 5 | Bit complement : $(M.bit) \leftarrow \sim(M.bit)$ | - - - - - - - |
| 15 | OR1 M.bit | 6B | 3 | 5 | Bit OR C-flag : $C \leftarrow C \vee (M.bit)$ | - - - - - C |
| 16 | OR1B M.bit | 6B | 3 | 5 | Bit OR C-flag and NOT : $C \leftarrow C \vee \sim(M.bit)$ | - - - - - C |
| 17 | SET1 dp.bit | x1 | 2 | 4 | Set bit : $(M.bit) \leftarrow "1"$ | - - - - - - - |
| 18 | SETA1 A.bit | 0B | 2 | 2 | Set A.bit : $(A.bit) \leftarrow "1"$ | - - - - - - - |
| 19 | SETC | A0 | 1 | 2 | Set C-flag : $C \leftarrow "1"$ | - - - - - 1 |
| 20 | SETG | C0 | 1 | 2 | Set G-flag : $G \leftarrow "1"$ | - - 1 - - - - |
| 21 | STC M.bit | EB | 3 | 6 | Store C-flag : $(M.bit) \leftarrow C$ | - - - - - - - |
| 22 | TCLR1 !abs | 5C | 3 | 6 | Test and clear bits with A : $A - (M), (M) \leftarrow (M) \wedge \sim(A)$ | N - - - - - Z - |
| 23 | TSET1 !abs | 3C | 3 | 6 | Test and set bits with A : $A - (M), (M) \leftarrow (M) \vee (A)$ | N - - - - - Z - |

Appendix A. Hynix 800 Series Instruction

5. Branch / Jump Operation

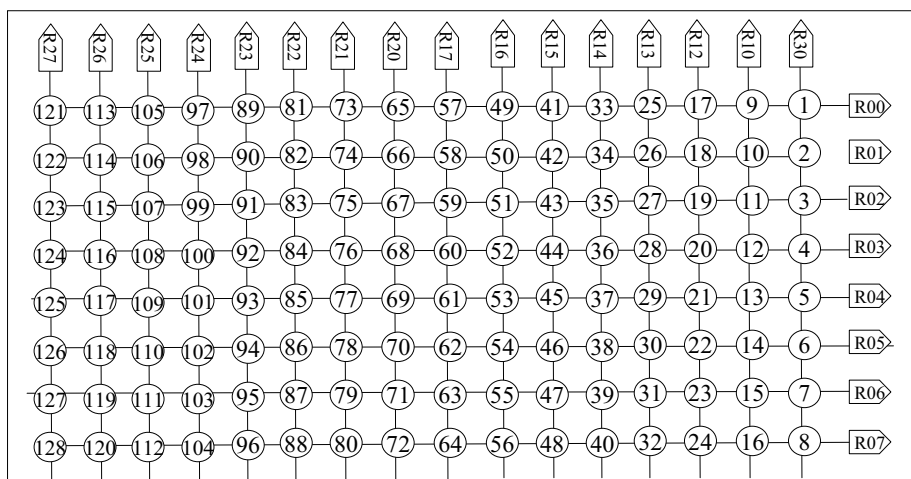
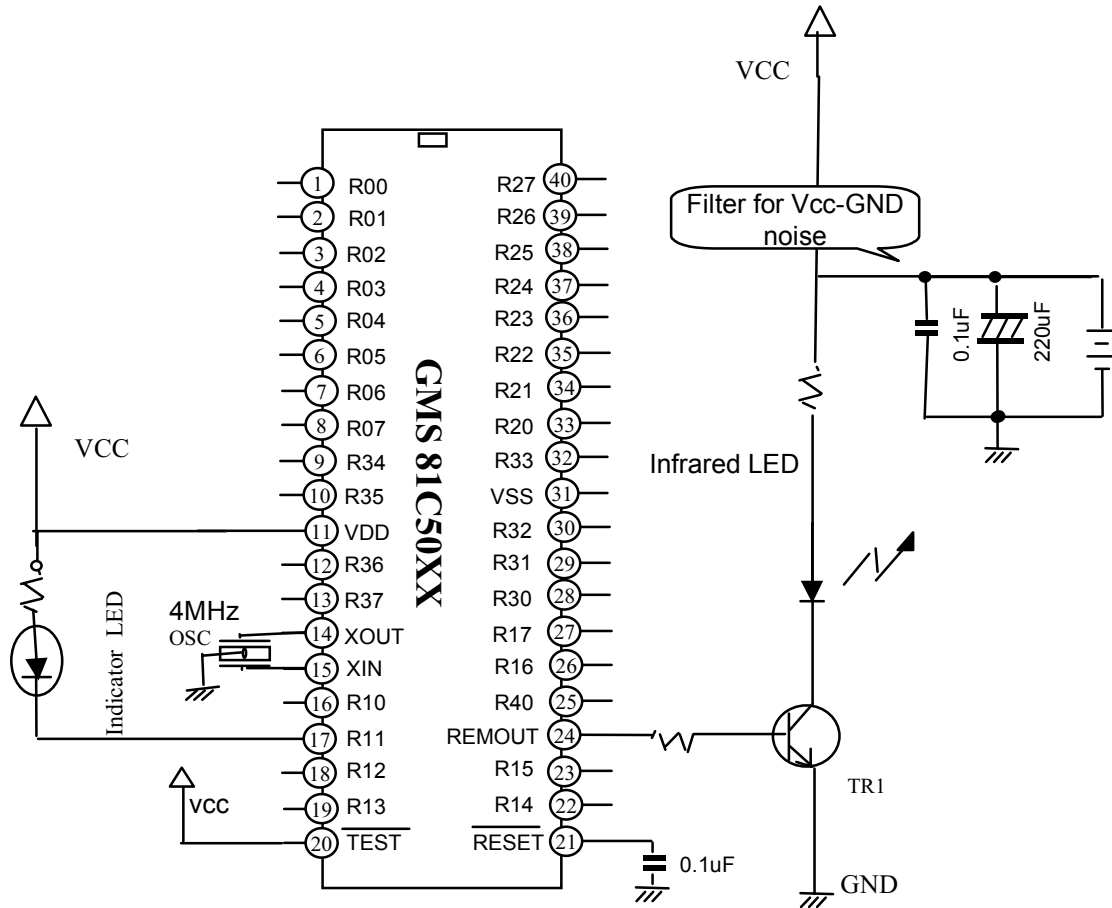
| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|------------------|---------|----------|----------|---|---------------|
| 1 | BBC A.bit,rel | y2 | 2 | 4/6 | Branch if bit clear : | |
| 2 | BBC dp.bit,rel | y3 | 3 | 5/7 | if(bit) = 0, then PC ← PC + rel | ----- |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

Appendix A. Hynix 800 Series Instruction

6. Control Operation & etc.

| NO. | MNEMONIC | OP CODE | BYTE NO. | CYCLE NO | OPERATION | FLAG NVGBHIZC |
|-----|----------|---------|----------|----------|--|-------------------------|
| 1 | BRK | 0F | 1 | 8 | Software interrupt: B ← "1", M(SP) ← (PC _H), SP ← SP - 1, M(s) ← (PC _L), SP ← S - 1, M(SP) ← PSW, SP ← SP - 1, PC _L ← (0FFDE _H), PC _H ← (0FFDF _H) | --- 1 - 0 -- |
| 2 | DI | 60 | 1 | 3 | Disable interrupts : I ← "0" | ----- 0 -- |
| 3 | EI | E0 | 1 | 3 | Enable interrupts : I ← "1" | ----- 1 -- |
| 4 | NOP | FF | 1 | 2 | No operation | ----- |
| 5 | POP A | 0D | 1 | 4 | Pop from stack | ----- (restored) |
| 6 | POP X | 2D | 1 | 4 | SP ← SP + 1, Reg. ← M(SP) | |
| 7 | POP Y | 4D | 1 | 4 | | |
| 8 | POP PSW | 6D | 1 | 4 | | |
| 9 | PUSH A | 0E | 1 | 4 | Push to stack | ----- |
| 10 | PUSH X | 2E | 1 | 4 | M(SP) ← Reg. SP ← SP - 1 | |
| 11 | PUSH Y | 4E | 1 | 4 | | |
| 12 | PUSH PSW | 6E | 1 | 4 | | |
| 13 | RET | 6F | 1 | 5 | Return from subroutine : SP ← SP+1, PC _L ← M(SP), SP ← SP+1, PC _H ← M(SP) | ----- |
| 14 | RETI | 7F | 1 | 6 | Return from interrupt : SP ← SP+1, PSW ← M(SP), SP ← SP+1, PC _L ← M(SP), SP ← SP+1, PC _H ← M(SP) | (restored) |
| 15 | STOP | EF | 1 | 3 | Stop mode (halt CPU, stop oscillator) | ----- |

1. General Circuit Diagram of GMS81C5016/24/32



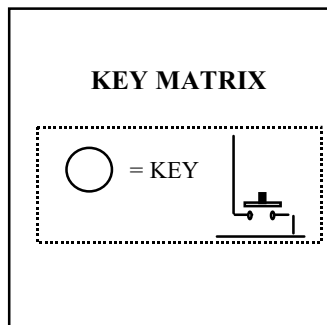


Figure B-1. Circuit Diagram

Note: Normally use the above 100uF capacitor for prevent power drop during pulse is transmitted. If you use the SRAM back-up, use at least 220uF.
We recommend to use ALKALINE battery.

Note: Figure B-1, Circuit Description:
device : GMS81C5016
package : 40PIN PDIP
port R0x : All input port with pull-up resistor
port R1x : All output port with N-MOS Open drain
port R11 : LED Drive port

2. Mask Option List Example Refer to Circuit B-1

GMS81C50 MASK OPTION LIST

Code Name : GMS81C5016 - Uxxx HYUNDAI
MCU Application Team.

1. Device & Package

GMS81C5004 GMS81C5024 Please enter check marks as ✓
 GMS81C5008 GMS81C5032
 GMS81C5016

28PIN : SOP 28 PIN : Skinny DIP
 40PIN : PDIP
 44PIN : PLCC 44PIN : MQFP

2. Inclusion of Pull up Resistor in Low Voltage Detection Option =Y

- R0 PORT Y : Yes N : No

| | | | | | | | | |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Port | R00 | R01 | R02 | R03 | R04 | R05 | R06 | R07 |
| Y/N | y | y | y | y | y | y | y | y |

- R1 PORT Y : Yes N : No

| | | | | | | | | |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Port | R10 | R11 | R12 | R13 | R14 | R15 | R16 | R17 |
| Y/N | y | y | y | y | y | y | y | y |

- R2 PORT Y : Yes N : No

| | | | | | | | | |
|------|-----|-----|-----|-----|-----|--------|--------|--------|
| Port | R20 | R21 | R22 | R23 | R24 | R25 *2 | R26 *2 | R27 *2 |
| Y/N | y | y | y | y | y | y | y | y |

- R3 PORT Y : Yes N : No

| | | | | | | | | |
|------|--------|--------|--------|--------|--------|--------|--------|--------|
| Port | R30 *2 | R31 *2 | R32 *2 | R33 *2 | R34 *2 | R35 *2 | R36 *2 | R37 *2 |
| Y/N | y | y | y | y | y | y | y | y |

- R4 PORT Y : Yes N : No

| | |
|------|--------|
| Port | R40 *2 |
| Y/N | y |

< NOTICE >
 *1 : is not available for 28PIN & 40PIN. So, Default option is Pull-Up.
 *2 : is not available for 28PIN. So, Default Option is Pull-Up.

3. Low Voltage Detection

Y/N Y

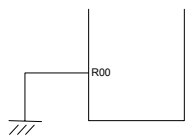
Date : _____
 Company Name : _____
 Section Name : _____
 Signature : _____

Note: Caution: When the power to the MCU would be decreased under LVD, all I/O ports are changed to input ports with pull up resistor. In below cases, you must take care of selecting the pull up in LVD.. You must detach the pull up of I/O port at these cases.

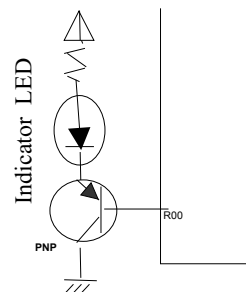
Case1 : When any I/O port is connected to GND, the current

will flow from the Pull up to GND. It cause the large power consumption and RAM would not be retained enough to satisfy your want.

Case2 : The case of using any I/O port for controlling PNP TR., The TR is always turn on by the Pull up of I/O port in LVD mode



< Case 1 >

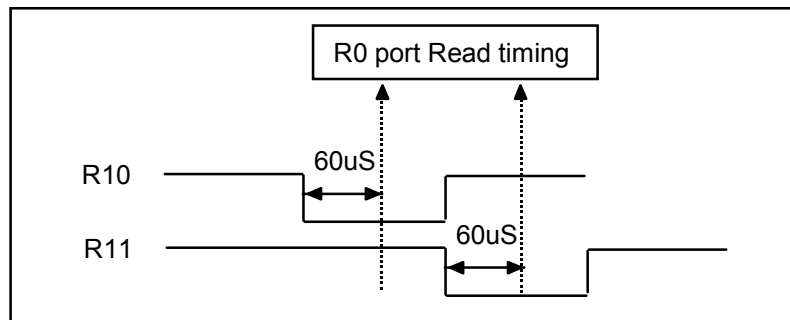


< Case 2 >

3. Key Scan

To secure the key board scanning , read the input port after minimum 60uS delay time from output port set to `Low`. This time delay is for the port rising time depend on the input pull-up resistor .

```
; program example ,See the Figure B-1 circuit.  
ldm R1,#1111_1110b ;R10 port set to LOW  
call delay_60uS ;60uS time delay routine  
lda R0 ;R0 port Read
```



< Fig B-2 , Input with pull-up port read time method >

* Current Consumption

The current consumption during the Pulse transmission depends on the external circuit and each Protocol. Normally , if you used Fig B-1 circuit., the operation current is 15mA

~ 25mA (Max 45mA). But this value is normal case. Some special protocol can be possible to consume more larger current.