1 PRODUCT OVERVIEW

SAM87RI PRODUCT FAMILY

Samsung's SAM87Ri family of 8-bit single-chip CMOS microcontrollers offers a fast and efficient CPU, a wide range of integrated peripherals, and various mask-programmable ROM sizes.

A address/data bus architecture and a large number of bit-configurable I/O ports provide a flexible programming environment for applications with varied memory and I/O requirements. Timer/counters with selectable operating modes are included to support real-time operations.

KS86C4004/C4104 MICROCONTROLLER

The KS86C4004/C4104 single-chip 8-bit microcontroller is fabricated using an advanced CMOS process. It is built around the powerful SAM87Ri CPU core. The KS86C4004/C4104 is a versatile microcontroller, with its A/D converter and a zero-crossing detection capability it can be used in a wide range of general purpose applications.

Stop and Idle power-down modes were implemented to reduce power consumption. To increase on-chip register space, the size of the internal register file was logically expanded. The KS86C4004/C4104 has 4-Kbytes of program memory on-chip (ROM) and 208-bytes of general purpose register area RAM.

Using the SAM87Ri design approach, the following peripherals were integrated with the SAM87Ri core:

— Four configurable I/O ports (KS86C4004: 22 pins, KS86C4104: 16 pins)
— Six interrupt sources with one vector and one interrupt level
— Two 8-bit timer/counter with various operating modes
— Analog to digital converter (KS86C4004: 8-bit, 8-channel, KS86C4104: 10-bit, 5-channel)
— One zero cross detection module

The KS86C4004/C4104 microcontroller is ideal for use in a wide range of electronic applications requiring simple timer/counter, PWM, ADC, ZCD and capture functions. KS86C4004 is available in a 30-pin SDIP and a 32-pin SOP package. KS86C4104 is available in a 24-pin SDIP and a 24-pin SOP package.

OTP

The KS86P4004/P4104 is an OTP (one time programmable) version of the KS86C4004/C4104 microcontroller. The KS86P4004/P4104 has on-chip 4-Kbyte one-time programmable EEPROM instead of masked ROM. The KS86P4004/P4104 is fully compatible with the KS86C4004/C4104, in function, in D.C. electrical characteristics and in pin configuration.

FEATURES

CPU

- SAM87Ri CPU core

Memory

- 4-Kbyte internal program memory (ROM)
- 208-byte general purpose register area (RAM)

Instruction Set

- 41 instructions
- IDLE and STOP instructions added for power-down modes.

Instruction Execution Time

- 600 ns at 10 MHz f_{OSC} (minimum)

Interrupts

- 6 interrupt sources with one vector and one level interrupt structure

Oscillation Frequency

- 1 MHz to 10 MHz external crystal oscillator
- Maximum 10 MHz CPU clock
- 4 MHz RC oscillator

General I/O

- Four I/O ports (22 pins for KS86C4004, 16 pins for KS86C4104)
- Bit programmable ports

A/D Converter

- Eight analog input pins
- 8-bit conversion resolution (KS86C4004)
- 10-bit conversion resolution (KS86C4104)

Timer/Counter

- One 8-bit basic timer for watchdog function
- One 8-bit timer/counter with three operating modes (10-bit PWM 1ch)
- One 8-bit timer/counter for the zero-crossing detection circuit

Zero-Crossing Detection Circuit

- Zero-crossing detection circuit that generates a digital signal in synchronism with an AC signal input

Buzzer Frequency Range

- 200 Hz to 20 kHz signal can be generated

Operating Temperature Range

- – 40°C to + 85°C

Operating Voltage Range

- 2.7 V to 5.5 V

OTP Interface Protocol Spec

- Serial OTP

Package Types

- 30-pin SDIP, 32-pin SOP for KS86C4004/P4004
- 24-pin SDIP, 24-pin SOP for KS86C4104/P4104

BLOCK DIAGRAM

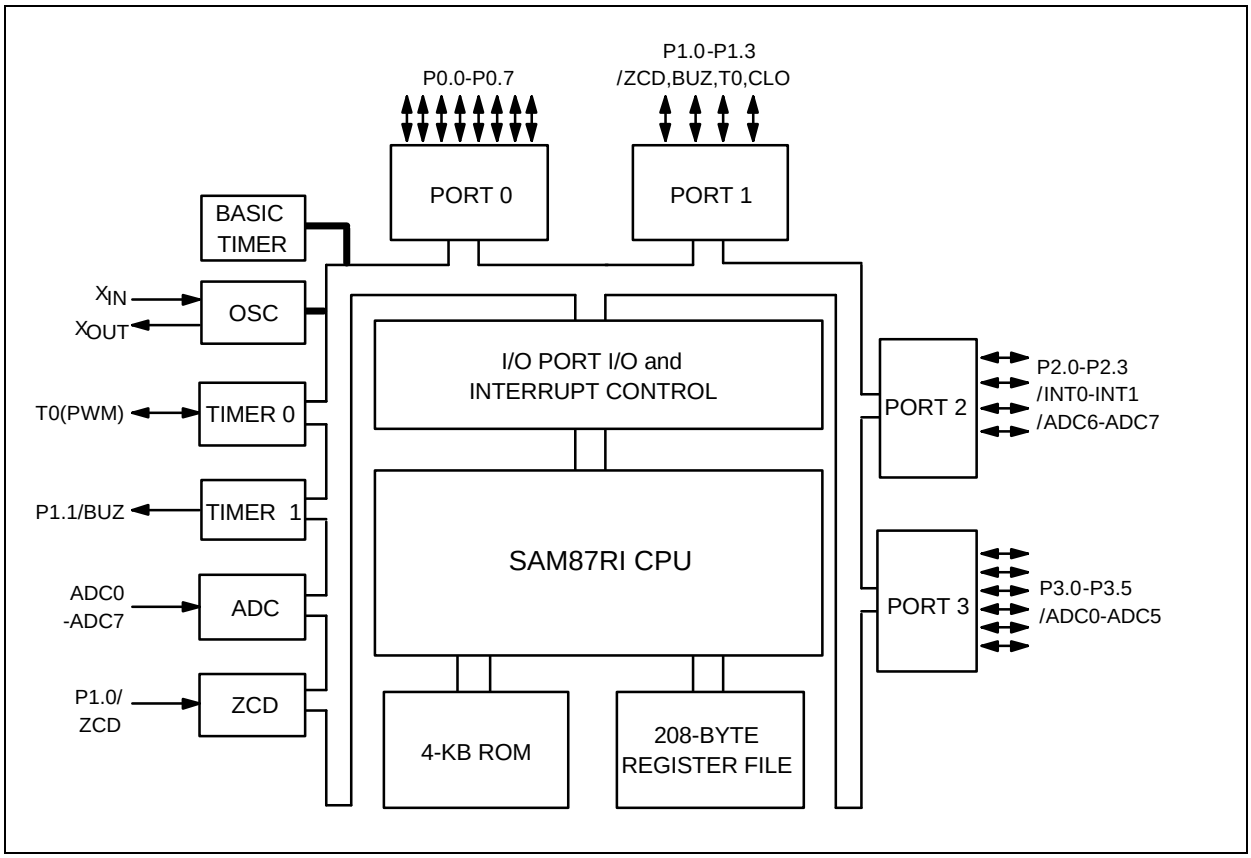


Figure 1-1. Block Diagram

PIN ASSIGNMENTS

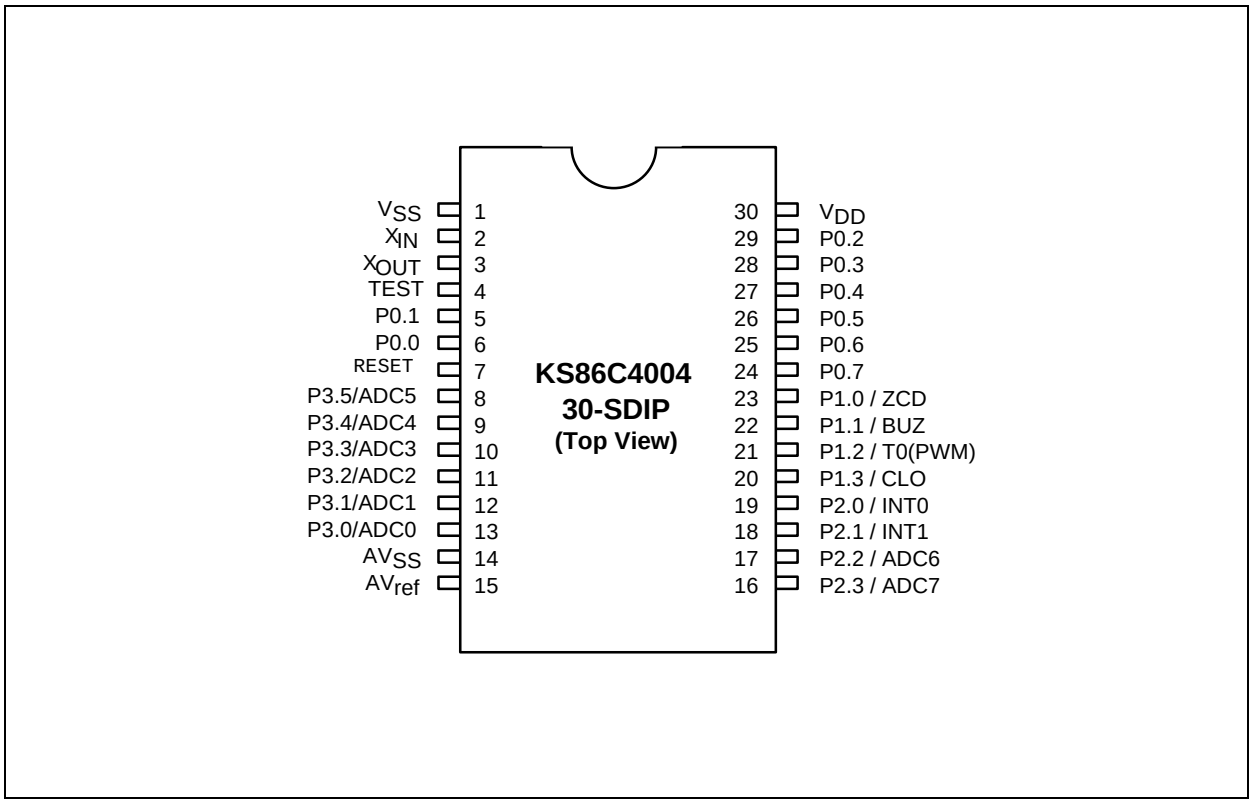


Figure 1-2. Pin Assignment Diagram (30-Pin SDIP Package)

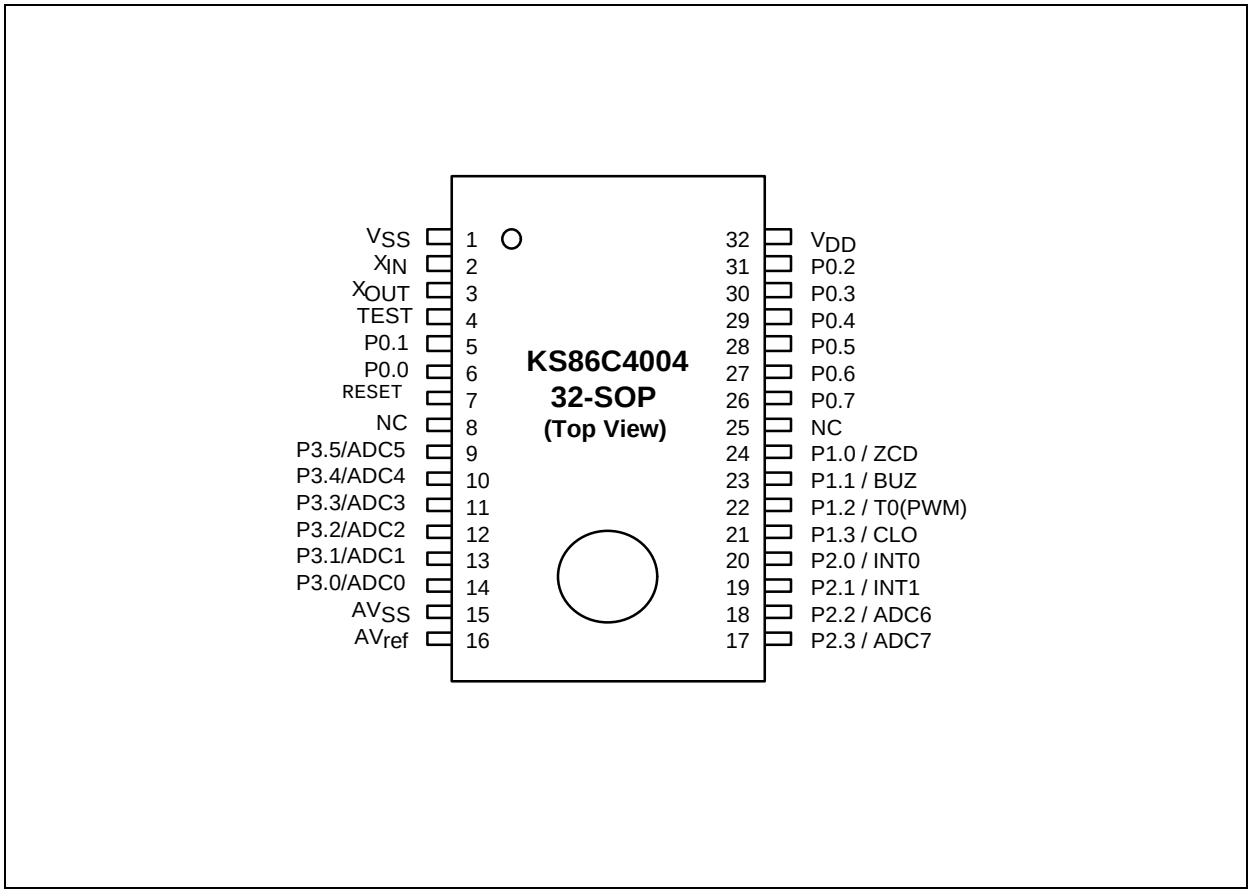


Figure 1-3. Pin Assignment Diagram (32-Pin SOP Package)

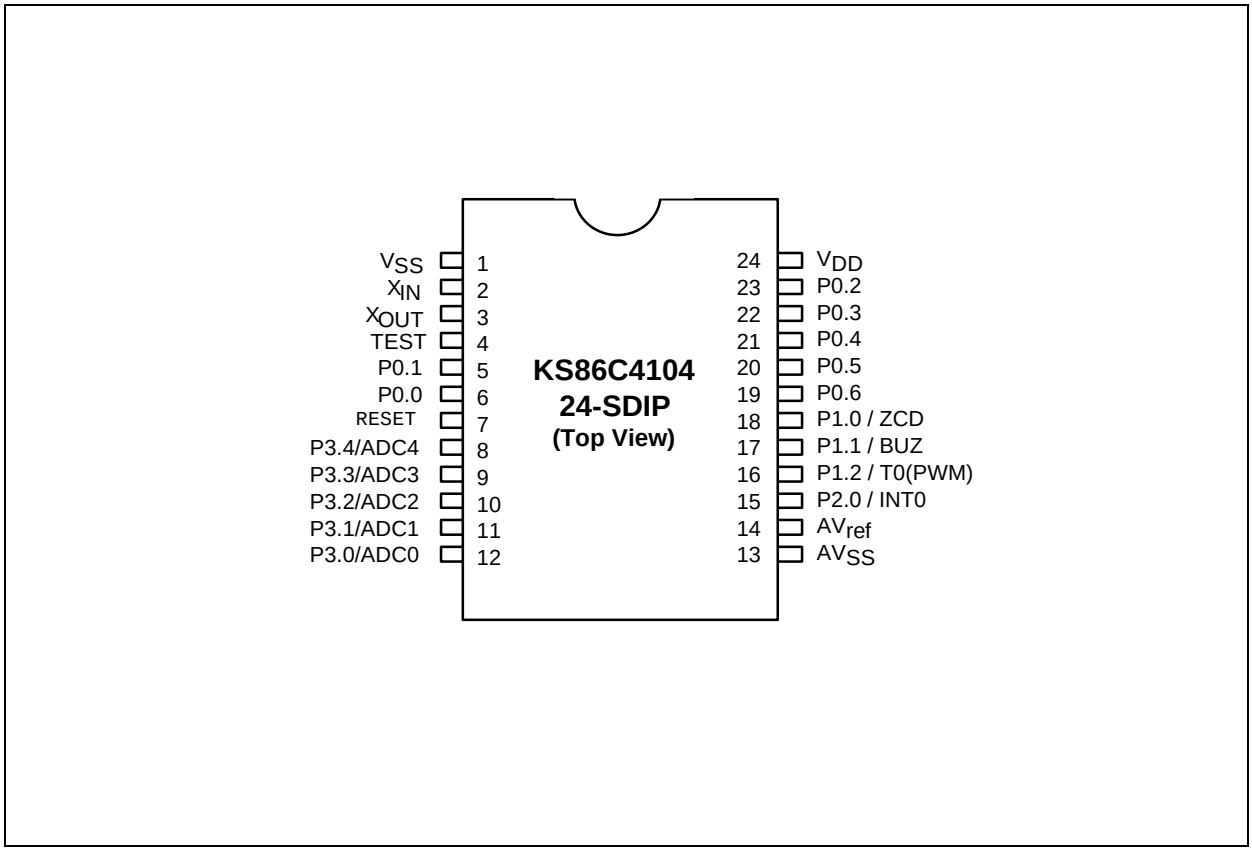


Figure 1-4. Pin Assignment Diagram (24-Pin SDIP Package)

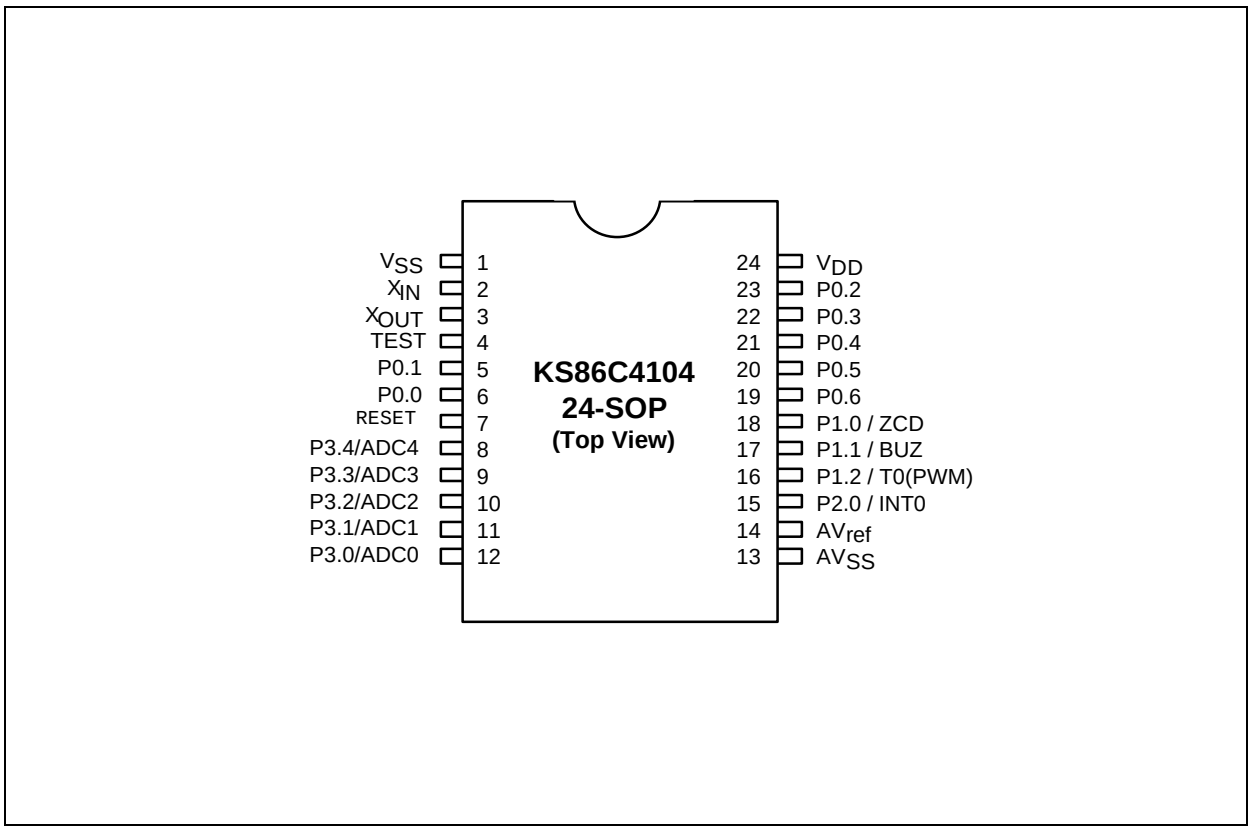


Figure 1-5. Pin Assignment Diagram (24-Pin SOP Package)

PIN DESCRIPTIONS

Table 1-1. KS86C4004/C4104 Pin Descriptions

Pin Names	Pin Type	Pin Description	Circuit Type	Share Pins
P0.0-P0.7	I/O	Bit-programmable I/O port for normal input or push-pull, open-drain output. Pull-up resistors are assignable by software.	E-2	
P1.0-P1.3	I/O	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are assignable by software. Port 1 pins can also be used as alternative functions.	F D D D	ZCD BUZ T0(PWM) CLO
P2.0-P2.3	I/O	Bit-programmable I/O port for Schmitt trigger input or push-pull, open drain output. Pull up resistors are assignable by software. Port 2 can also be used as external interrupt, A/D input.	E E-1	INT0–INT1 ADC6–ADC7
P3.0-P3.5	I/O	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are assignable by software. Port 3 pins can also be used as A/D converter input.	F	ADC0–ADC5
X_{IN}, X_{OUT}	–	Crystal/ceramic, or RC oscillator signal for system clock.	–	–
INT0–INT1	I	External interrupt input.	E	P2.0–P2.1
RESET	I	System RESET signal input pin.	B	–
TEST	I	Test signal input pin (for factory use only: must be connected to V_{SS})	–	–
V_{DD}, V_{SS}	–	Voltage input pin and ground	–	–
AV_{REF}, AV_{SS}	–	A/D converter reference voltage input and ground	–	–
ZCD	I	Zero crossing detector input	F	P1.0
BUZ	O	200 Hz–20 kHz frequency output for buzzer sound	D	P1.1
T0	I/O	Timer 0 capture input or 10-bit PWM output	D	P1.2
CLO	O	System clock output port	D	P1.3
ADC0–ADC7	I	A/D converter input	F E-1	P3.0–P3.5 P2.2–P2.3

NOTE: Port 0.7, P1.3, P2.1–P2.3 and P3.5 is not available in KS86C4104/P4104 .

PIN CIRCUITS

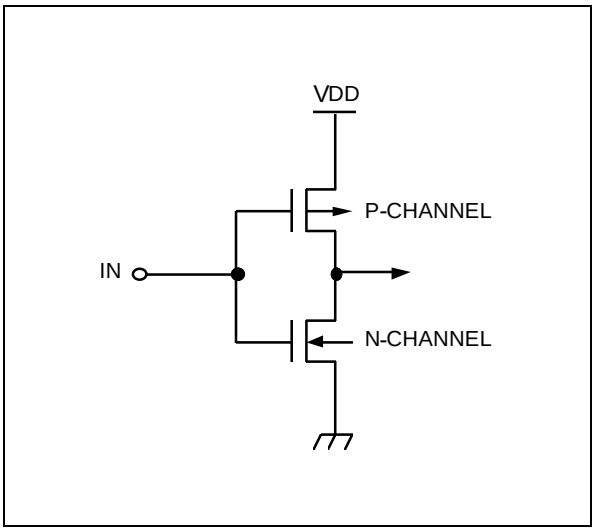


Figure 1-6. Pin Circuit Type A

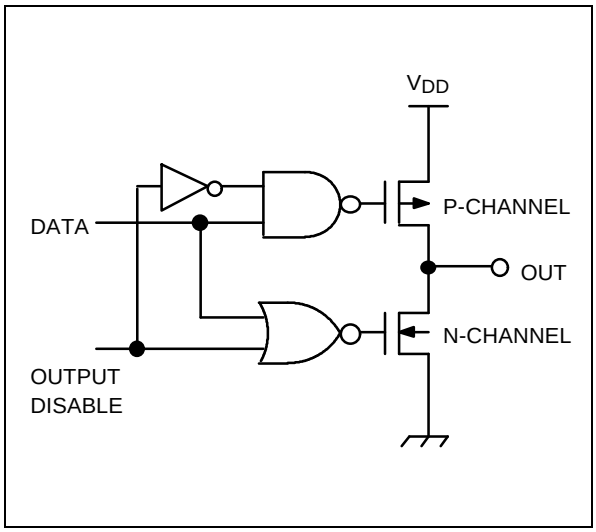


Figure 1-8. Pin Circuit Type C

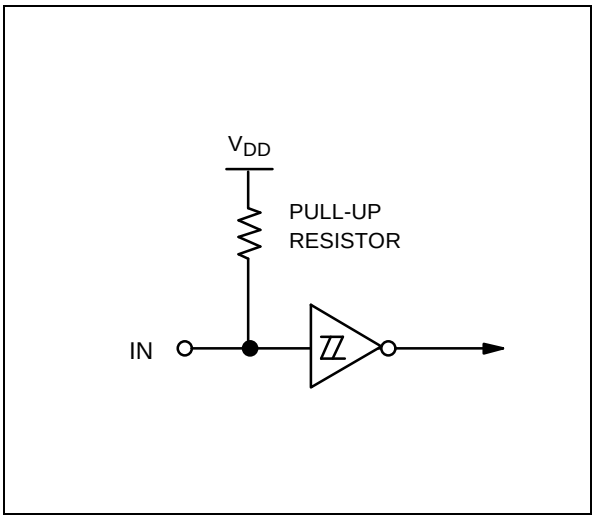


Figure 1-7. Pin Circuit Type B

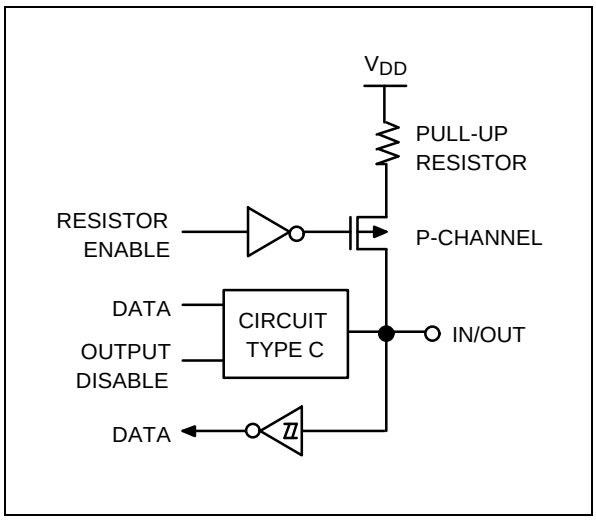


Figure 1-9. Pin Circuit Type D

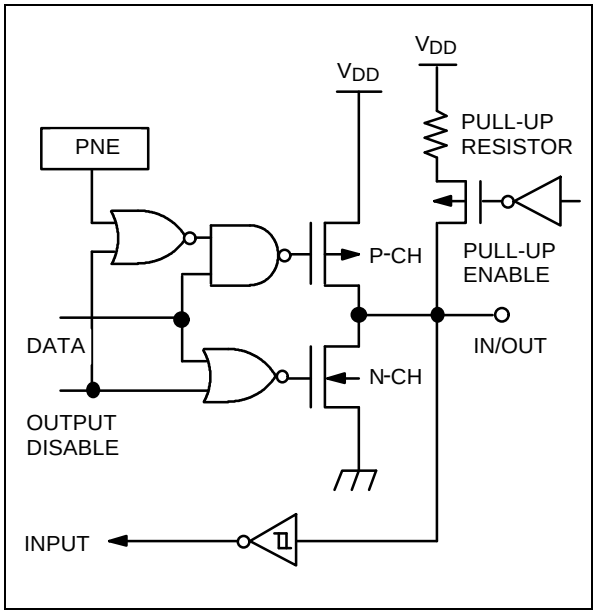


Figure 1-10. Pin Circuit Type E

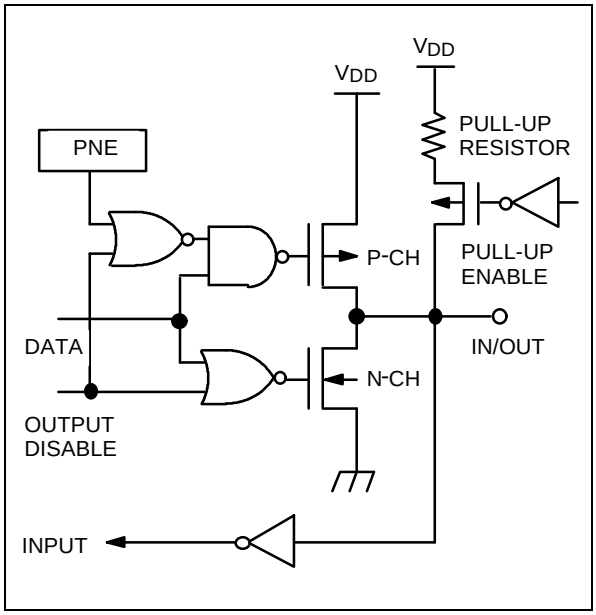


Figure 1-10. Pin Circuit Type E-2

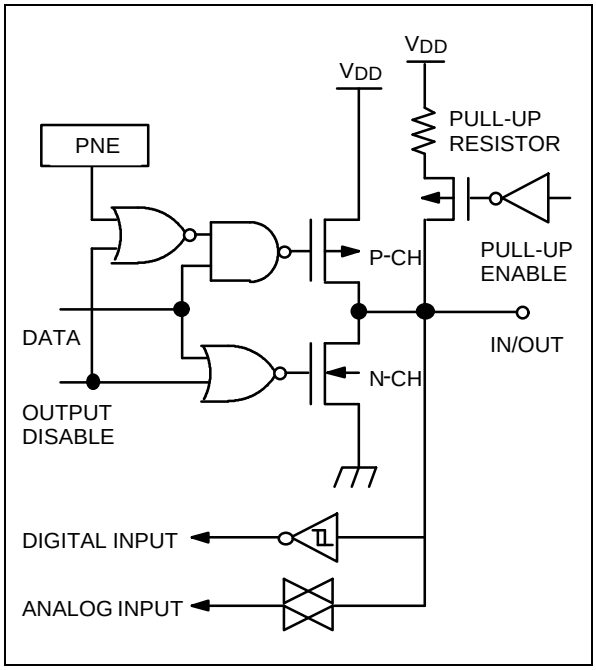


Figure 1-11. Pin Circuit Type E-1

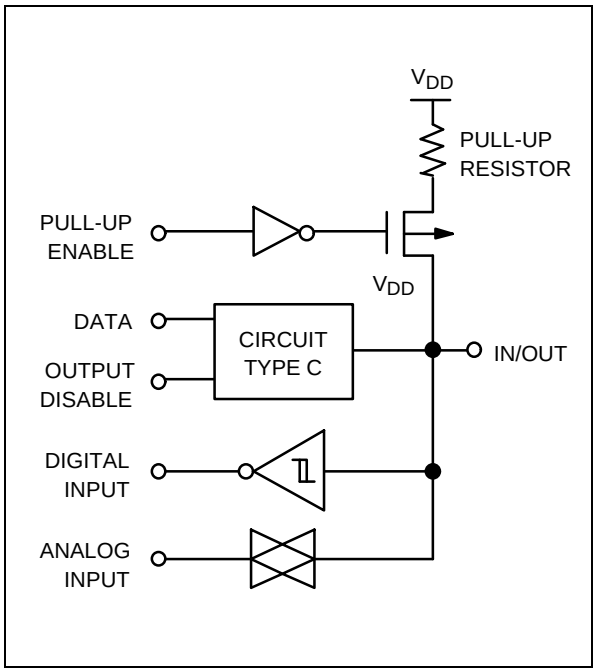


Figure 1-12. Pin Circuit Type F

2 ADDRESS SPACES

OVERVIEW

The KS86C4004/C4104 microcontroller has two kinds of address space:

— Internal program memory (ROM)

— Internal register file

A 12-bit address bus supports program memory operations. A separate 8-bit register bus carries addresses and data between the CPU and the internal register file.

The KS86C4004/C4104 has 4-Kbytes of mask-programmable on-chip program memory: which is configured as the Internal ROM mode, all of the 4-Kbyte internal program memory is used.

The KS86C4004/C4104 microcontroller has 208 general-purpose registers in its internal register file. Thirty-two bytes in the register file are mapped for system and peripheral control functions.

PROGRAM MEMORY (ROM)

Normal Operating Mode

The KS86C4004/C4104 has 4-Kbytes (locations 0H–0FFFH) of internal mask-programmable program memory.

The first 2-bytes of the ROM (0000H–0001H) are interrupt vector address.

Unused locations (0002H–00FFH) can be used as normal program memory.

The program reset address in the ROM is 0100H.

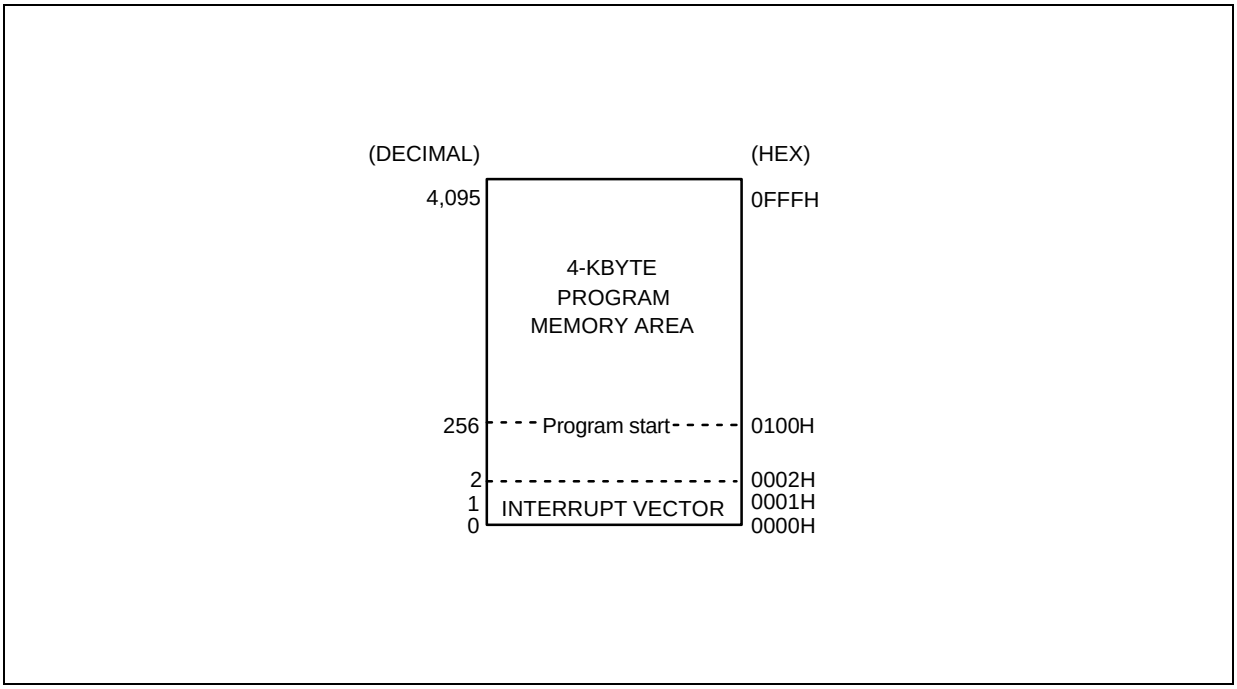


Figure 2-1. Program Memory Address Space

REGISTER ARCHITECTURE

The upper 64-bytes of the KS86C4004/C4104's internal register file are addressed as working registers, system control registers and peripheral control registers. The lower 192-bytes of internal register file(00H–BFH) is called the *general purpose register space*. The total addressable register space is thereby 256-bytes. 240 registers in this space can be accessed; 208 are available for general-purpose use.

For many SAM87Ri microcontrollers, the addressable area of the internal register file is further expanded by additional register pages at the general purpose register space (00H–BFH). This register file expansion is not implemented in the KS86C4004/C4104, however.

The specific register types and the area (in bytes) that they occupy in the internal register file are summarized in Table 2-1.

Table 2-1. Register Type Summary

Register Type	Number of Bytes
CPU and system control registers	10
Peripheral, I/O, and clock control and data registers	22
General-purpose registers (including the 16-bit common working register area)	208
Total Addressable Bytes	**240**

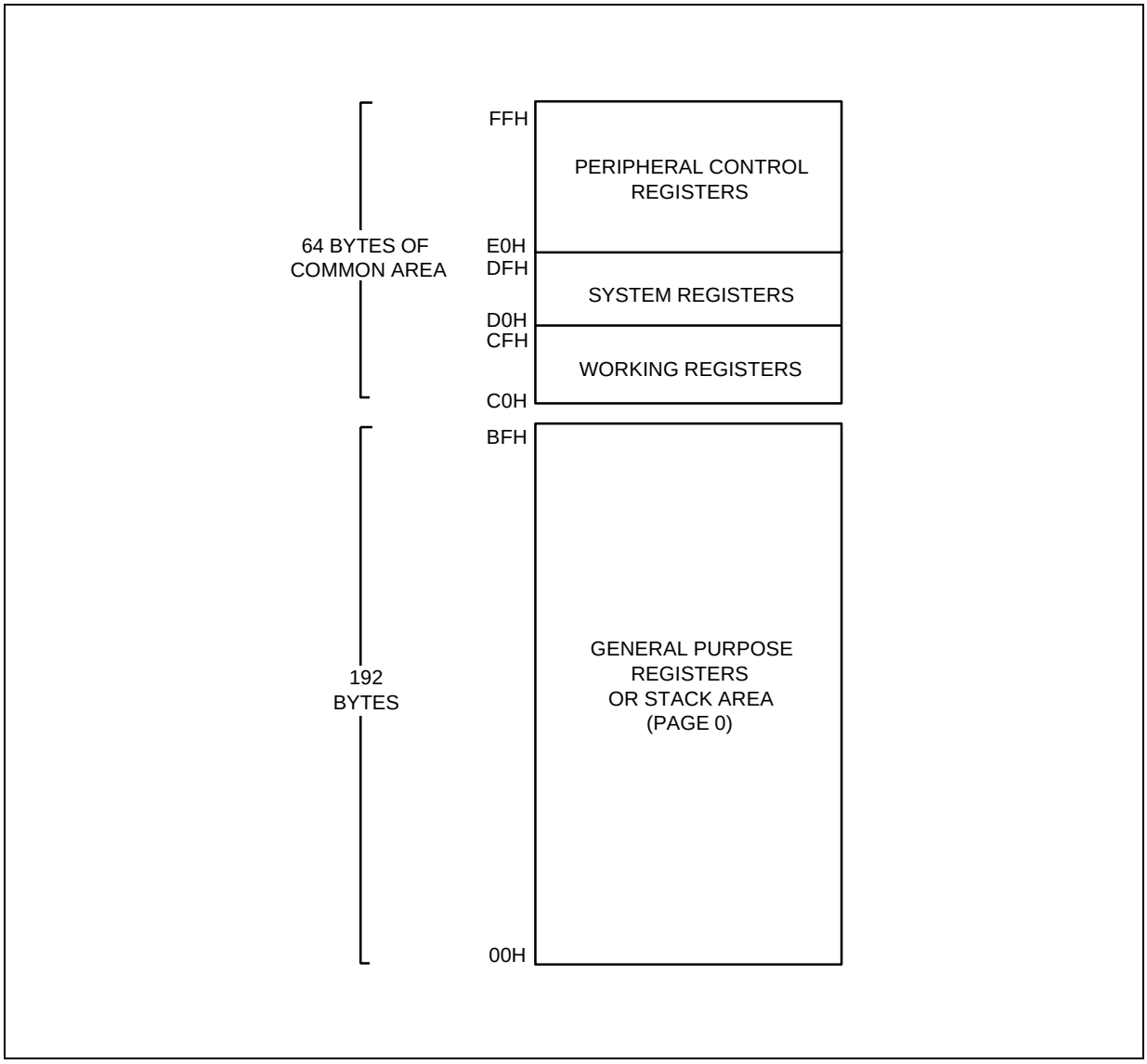


Figure 2-2. Internal Register File Organization

COMMON WORKING REGISTER AREA (C0H–CFH)

The SAM87Ri register architecture provides an efficient method of working register addressing that takes full advantage of shorter instruction formats to reduce execution time.

This16-byte address range is called common area. That is, locations in this area can be used as working registers by operations that address any location on any page in the register file. Typically, these working registers serve as temporary buffers for data operations between different pages. However, because the KS86C4004/C4104 uses only page 0, you can use the common area for any internal data operation.

The Register (R) addressing mode can be used to access this area

Registers are addressed either as a single 8-bit register or as a paired 16-bit register. In 16-bit register pairs, the address of the first 8-bit register is always an even number and the address of the next register is an odd number. The most significant byte of the 16-bit data is always stored in the even-numbered register; the least significant byte is always stored in the next (+ 1) odd-numbered register.

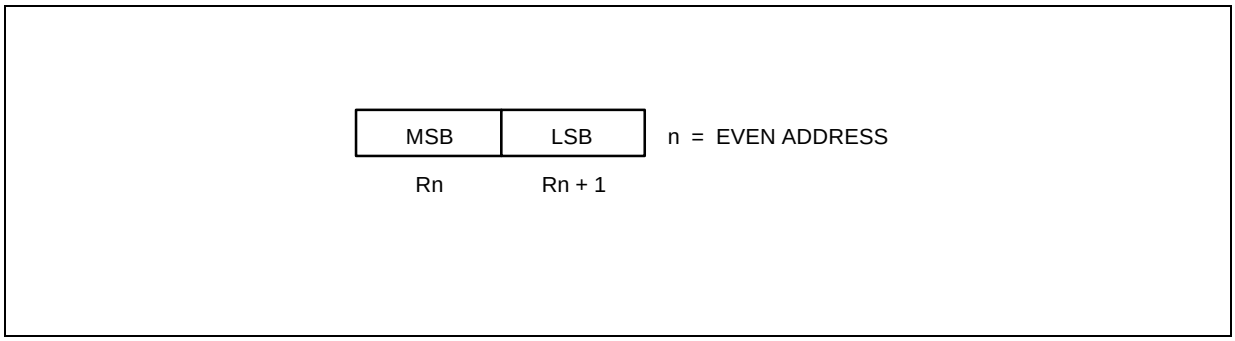


Figure 2-3. 16-Bit Register Pairs

☞ PROGRAMMING TIP — Addressing the Common Working Register Area

As the following examples show, you should access working registers in the common area, locations C0H–CFH, using working register addressing mode only.

Examples:

```
1. LD    0C2H,40H          ;  Invalid addressing mode!
   Use working register addressing instead:
   LD    R2,40H            ;  R2 (C2H) ← the value in location 40H

2. ADD   0C3H,#45H         ;  Invalid addressing mode!
   Use working register addressing instead:
   ADD   R3,#45H           ;  R3 (C3H) ← R3 + 45H
```

SYSTEM STACK

KS86-series microcontrollers use the system stack for subroutine calls and returns and to store data. The PUSH and POP instructions are used to control system stack operations. The KS86C4004/4104 architecture supports stack operations in the internal register file.

Stack Operations

Return addresses for procedure calls and interrupts and data are stored on the stack. The contents of the PC are saved to stack by a CALL instruction and restored by the RET instruction. When an interrupt occurs, the contents of the PC and the FLAGS register are pushed to the stack. The IRET instruction then pops these values back to their original locations. The stack address is always decremented *before* a push operation and incremented *after* a pop operation. The stack pointer (SP) always points to the stack frame stored on the top of the stack, as shown in Figure 2-4.

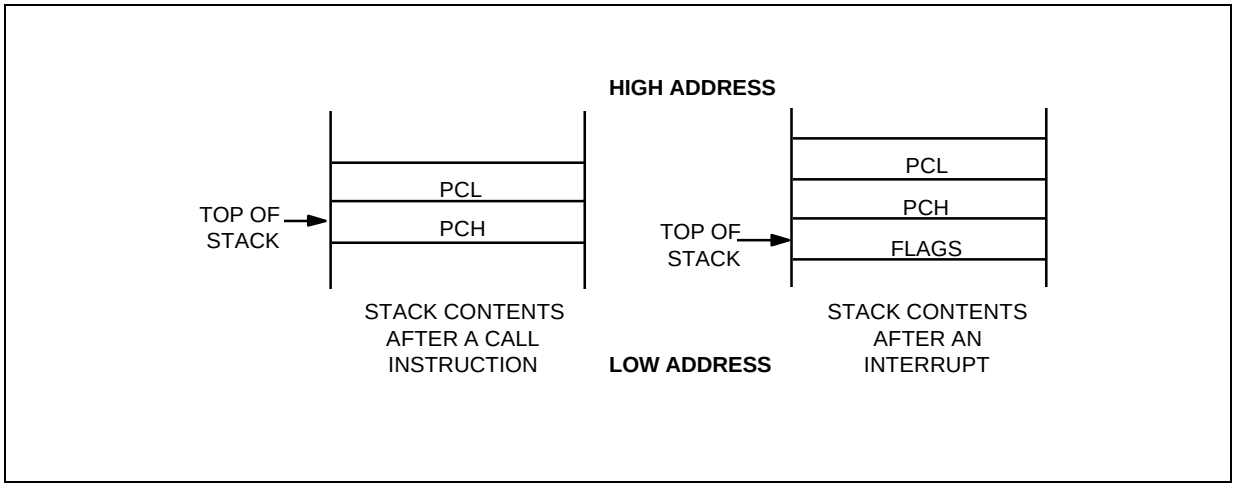


Figure 2-4. Stack Operations

Stack Pointer (SP)

Register location D9H contains the 8-bit stack pointer (SP) that is used for system stack operations. After a reset, the SP value is undetermined.

Because only internal memory space is implemented in the KS86C4004/C4104, the SP must be initialized to an 8-bit value in the range 00H–0C0H.

NOTE

In case a Stack Pointer is initialized to 00H, it is decreased to FFH when stack operation starts. This means that a Stack Pointer access invalid stack area.

☞ PROGRAMMING TIP — Standard Stack Operations Using PUSH and POP

The following example shows you how to perform stack operations in the internal register file using PUSH and POP instructions:

```
LD      SP,#0C0H        ;  SP ← C0H (Normally, the SP is set to 0C0H by the
                        ;  initialization routine)
•
•
•
PUSH    SYM             ;  Stack address 0BFH ← SYM
PUSH    R15             ;  Stack address 0BEH ← R15
PUSH    20H             ;  Stack address 0BDH ← 20H
PUSH    R3              ;  Stack address 0BCH ← R3
•
•
•
POP     R3              ;  R3 ← Stack address 0BCH
POP     20H             ;  20H ← Stack address 0BDH
POP     R15             ;  R15 ← Stack address 0BEH
POP     SYM             ;  SYM ← Stack address 0BFH
```

NOTES

3 ADDRESSING MODES

OVERVIEW

Instructions that are stored in program memory are fetched for execution using the program counter. Instructions indicate the operation to be performed and the data to be operated on. *Addressing mode* is the method used to determine the location of the data operand. The operands specified in SAM87Ri instructions may be condition codes, immediate data, or a location in the register file, program memory, or data memory.

The SAM87Ri instruction set supports six explicit addressing modes. Not all of these addressing modes are available for each instruction. The addressing modes and their symbols are as follows:

— Register (R)
— Indirect Register (IR)
— Indexed (X)
— Direct Address (DA)
— Relative Address (RA)
— Immediate (IM)

REGISTER ADDRESSING MODE (R)

In Register addressing mode, the operand is the content of a specified register (see Figure 3-1). Working register addressing differs from Register addressing because it uses an 16-byte working register space in the register file and an 4-bit register within that space (see Figure 3-2).

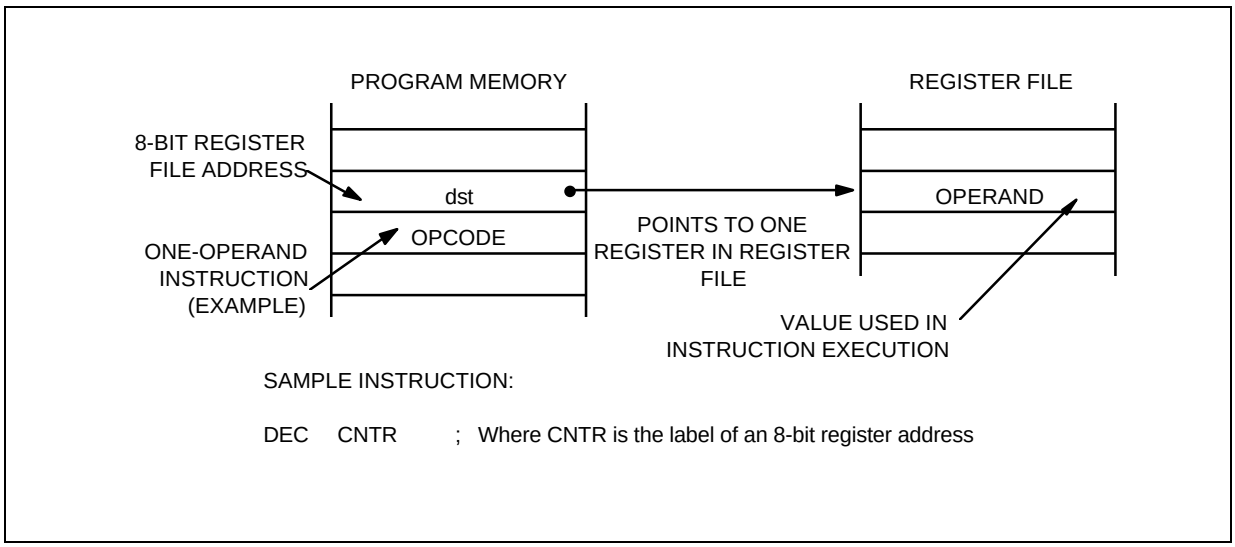


Figure 3-1. Register Addressing

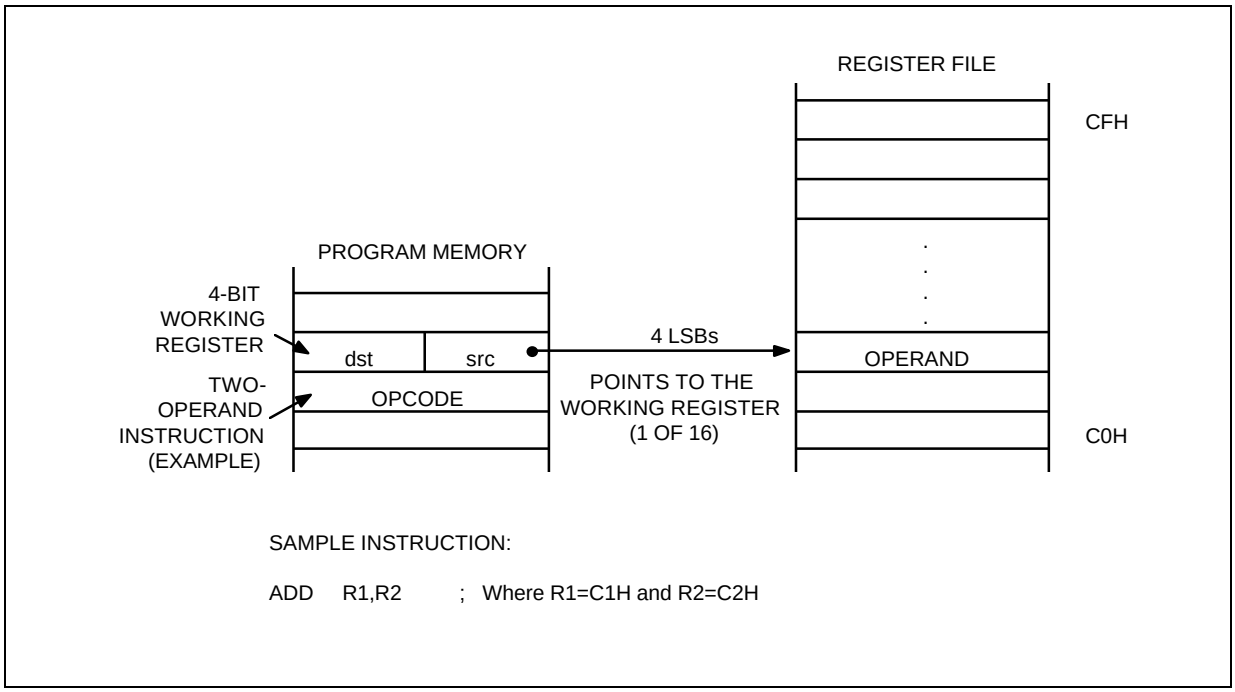


Figure 3-2. Working Register Addressing

INDIRECT REGISTER ADDRESSING MODE (IR)

In Indirect Register (IR) addressing mode, the content of the specified register or register pair is the address of the operand. Depending on the instruction used, the actual address may point to a register in the register file, to program memory (ROM), or to an external memory space (see Figures 3-3 through 3-6).

You can use any 8-bit register to indirectly address another register. Any 16-bit register pair can be used to indirectly address another memory location.

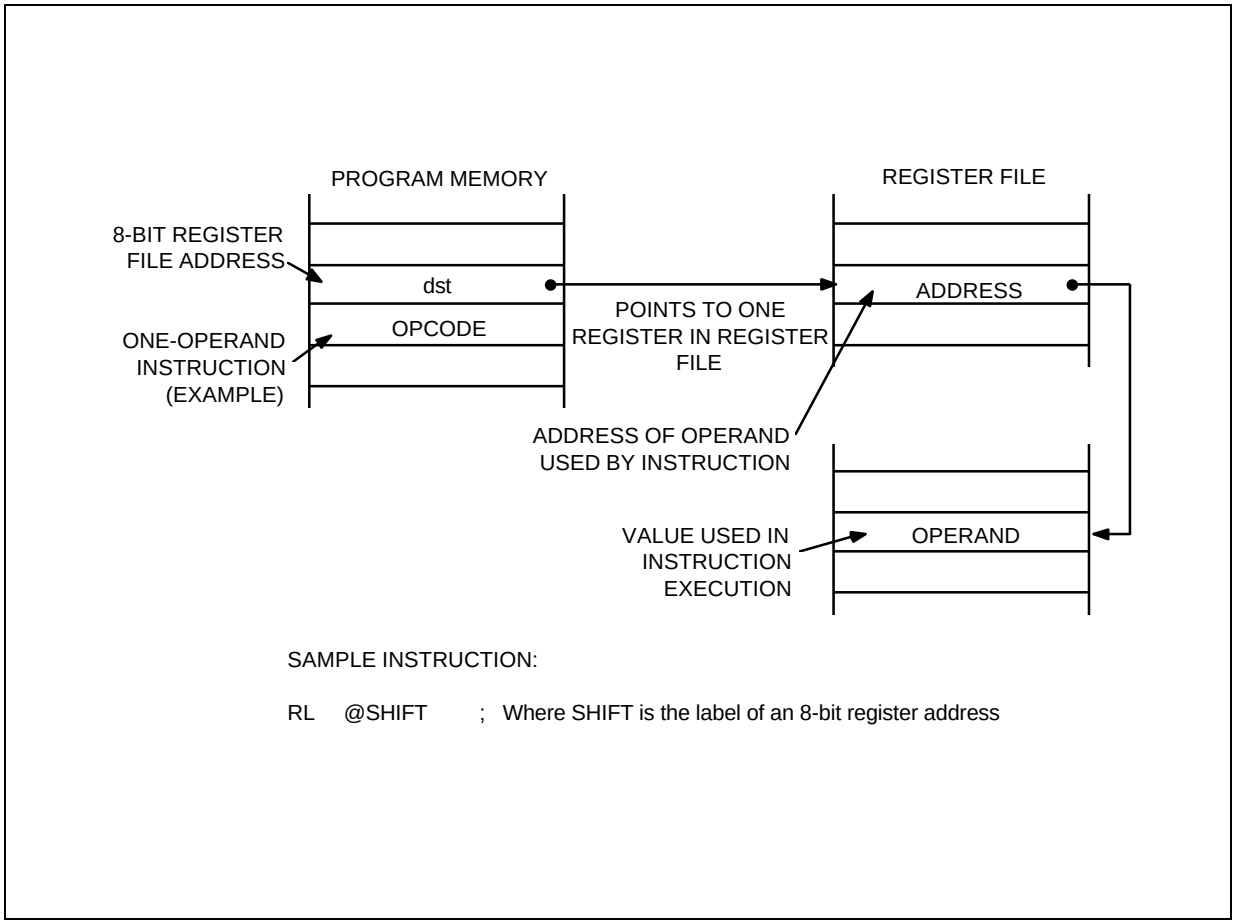


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

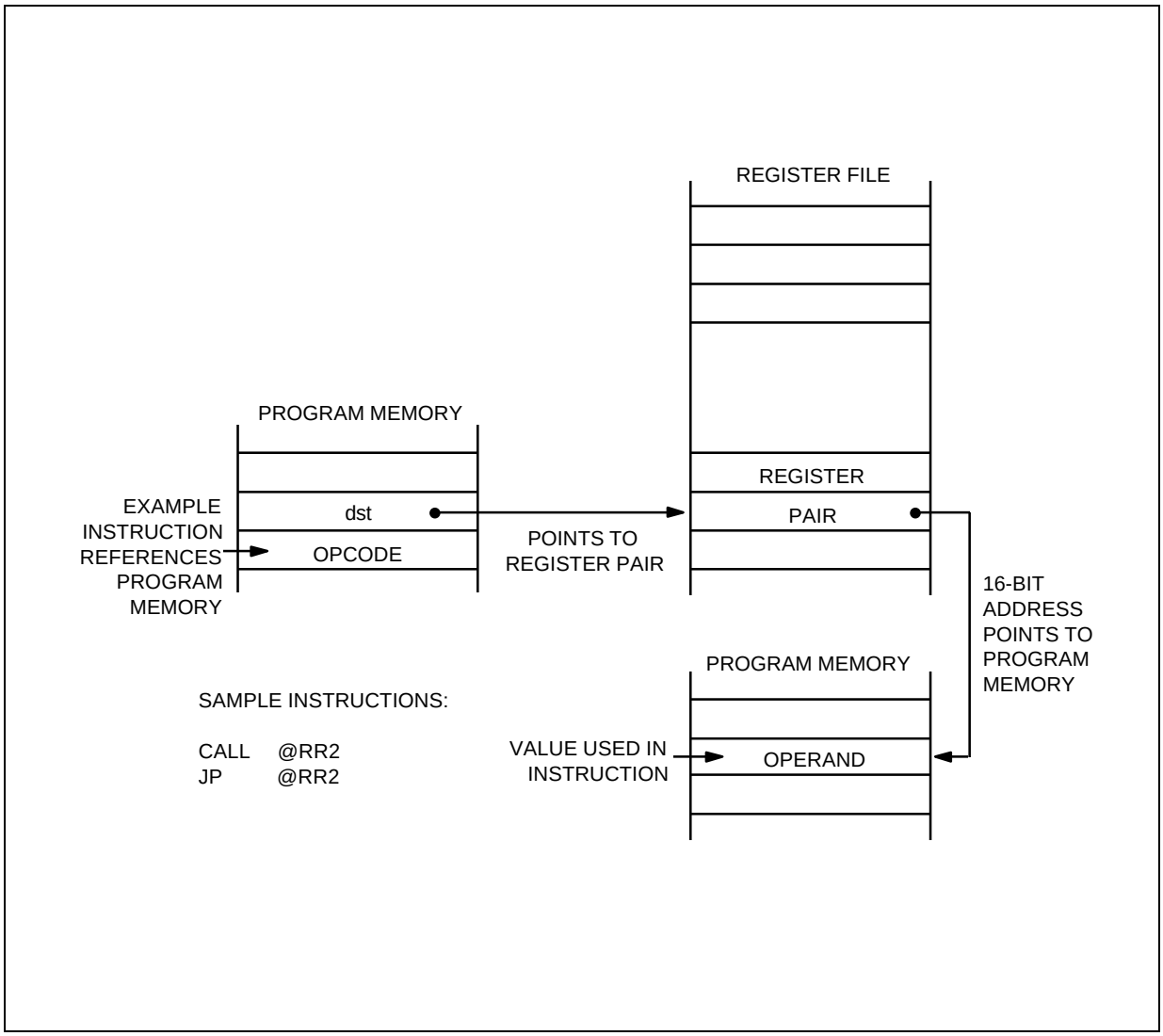


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

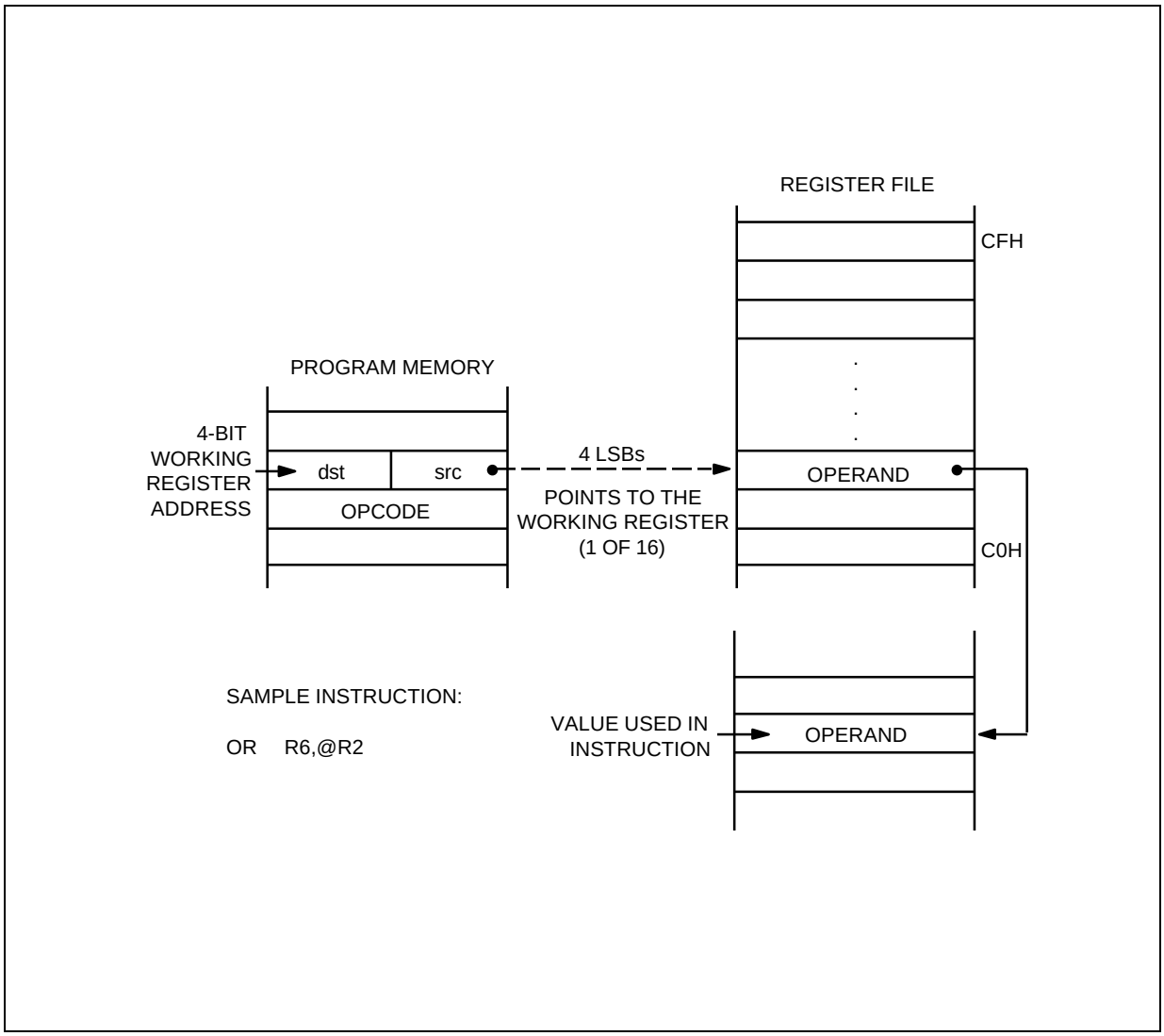


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Concluded)

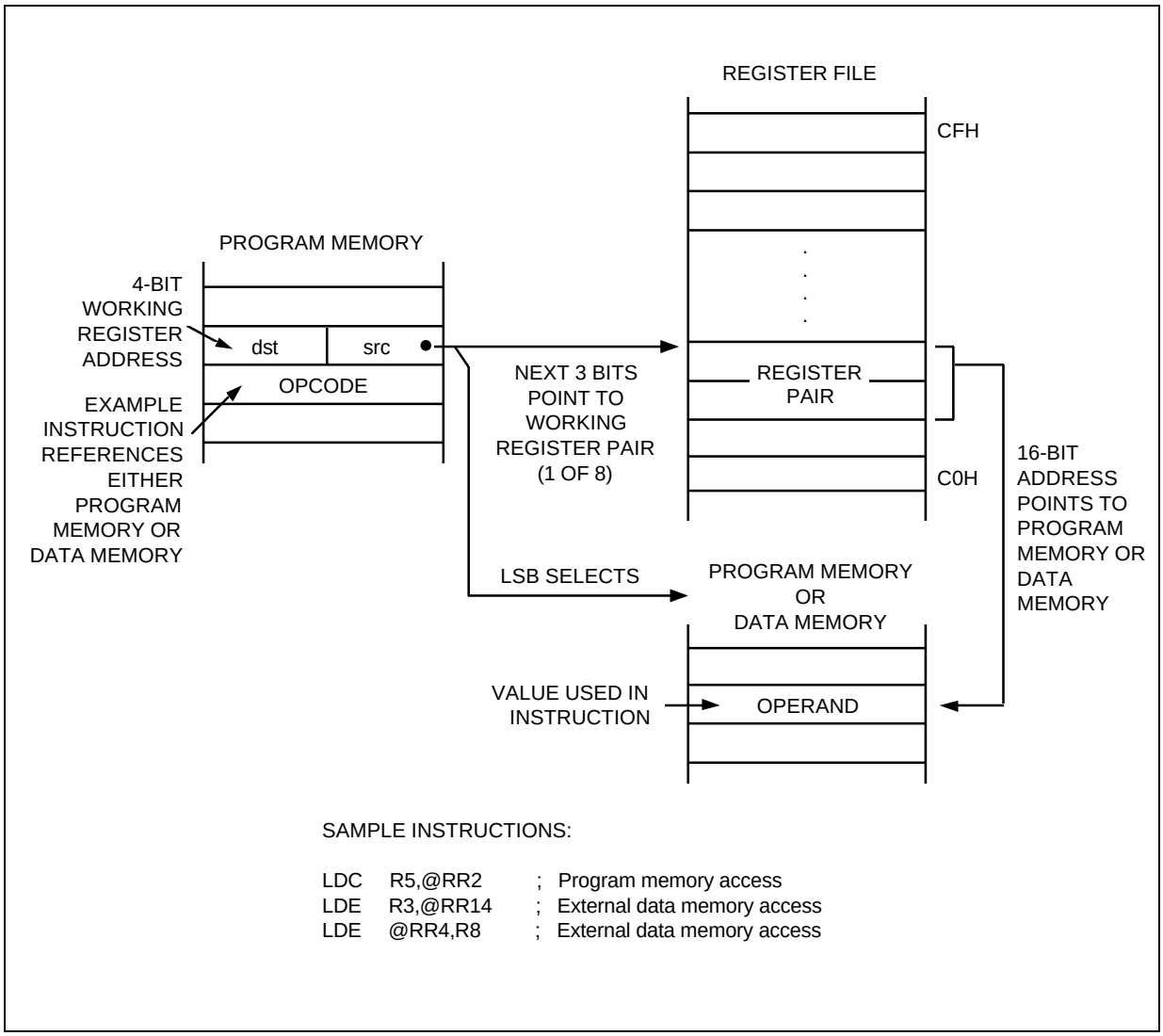


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

INDEXED ADDRESSING MODE (X)

Indexed (X) addressing mode adds an offset value to a base address during instruction execution in order to calculate the effective operand address (see Figure 3-7). You can use Indexed addressing mode to access locations in the internal register file or in external memory.

In short offset Indexed addressing mode, the 8-bit displacement is treated as a signed integer in the range – 128 to + 127. This applies to external memory accesses only (see Figure 3-8).

For register file addressing, an 8-bit base address provided by the instruction is added to an 8-bit offset contained in a working register. For external memory accesses, the base address is stored in the working register pair designated in the instruction. The 8-bit or 16-bit offset given in the instruction is then added to the base address (see Figure 3-9).

The only instruction that supports Indexed addressing mode for the internal register file is the Load instruction (LD). The LDC and LDE instructions support Indexed addressing mode for internal program memory, external program memory, and for external data memory, when implemented.

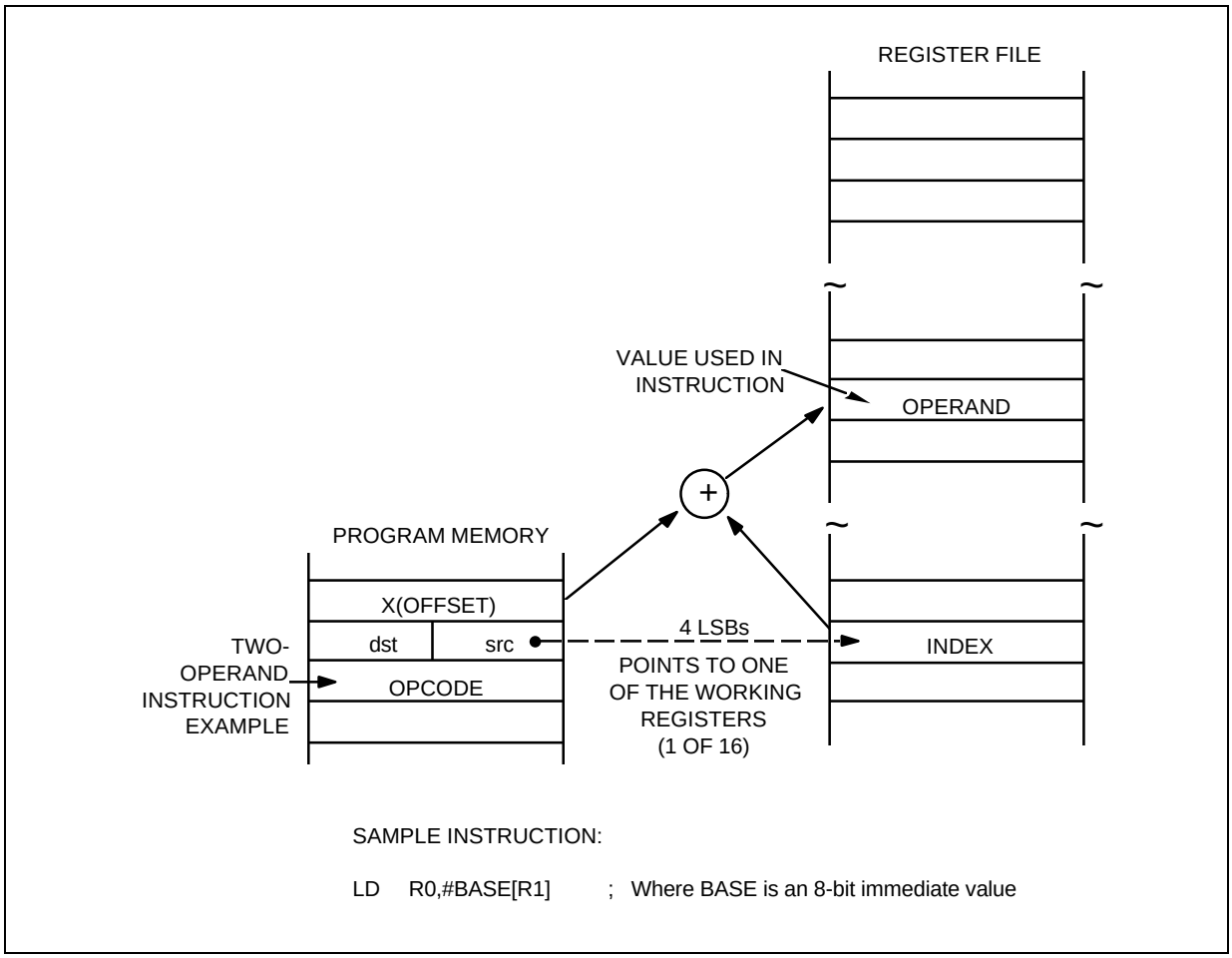
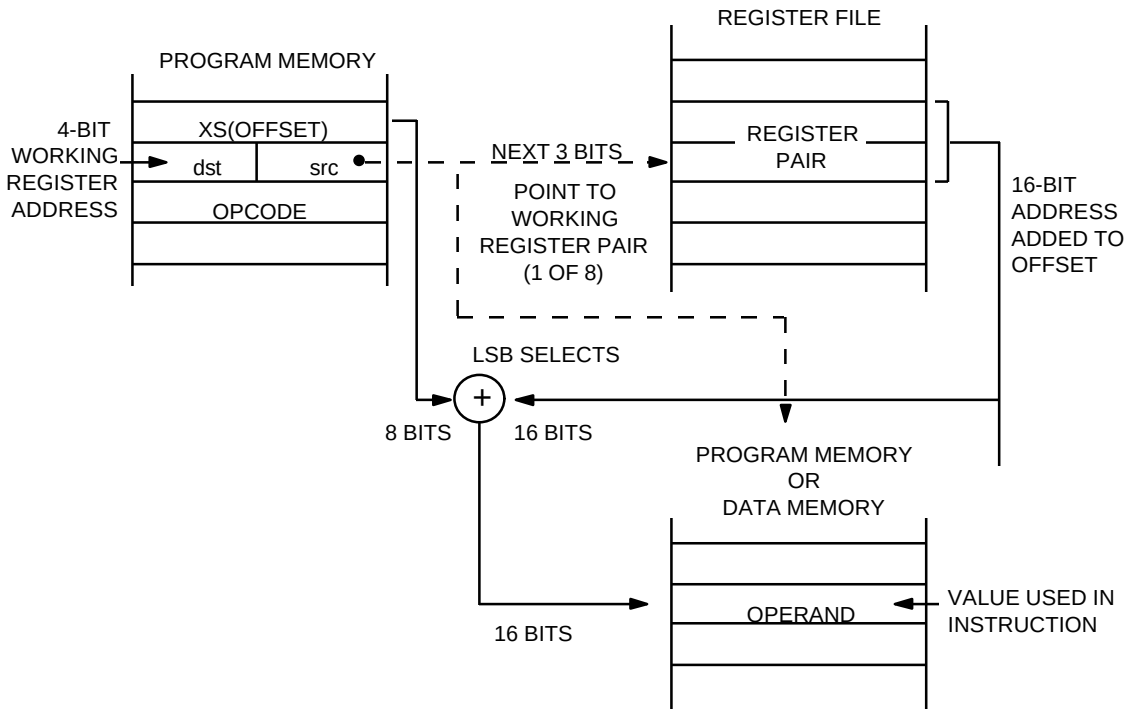


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

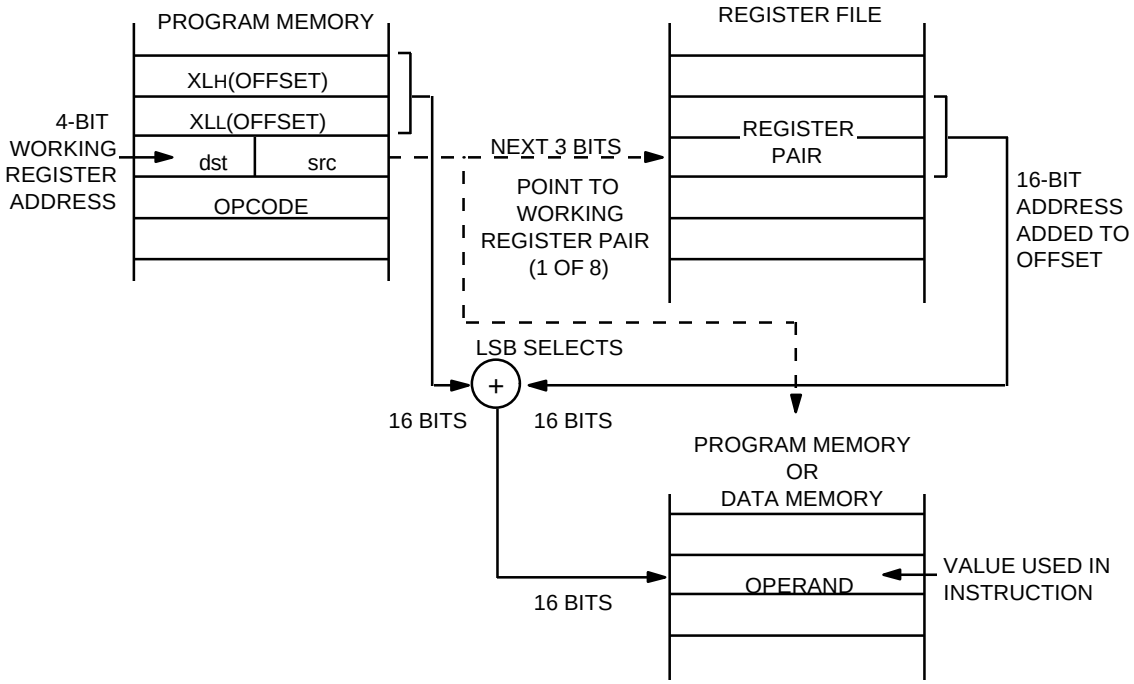


SAMPLE INSTRUCTIONS:

```
LDC     R4,#04H[RR2]      ;  The values in the program address (RR2 + #04H)
                             are loaded into register R4.

LDE     R4,#04H[RR2]      ;  Identical operation to LDC example, except that
                             external program memory is accessed.
```

Figure 3-8. Indexed Addressing to Program or Data Memory with Short Offset

INDEXED ADDRESSING MODE (Concluded)



SAMPLE INSTRUCTIONS:

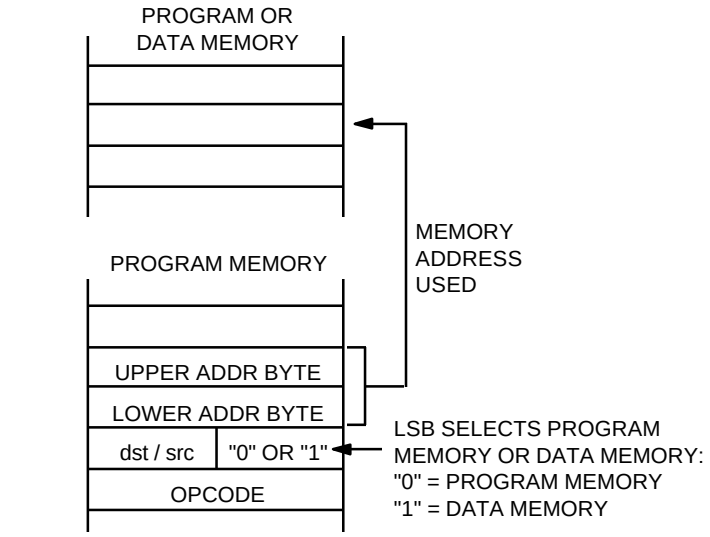
```
LDC     R4,#1000H[RR2]      ;  The values in the program address (RR2 + #1000H)
                               are loaded into register R4.

LDE     R4,#1000H[RR2]      ;  Identical operation to LDC example, except that
                               external program memory is accessed.
```

Figure 3-9. Indexed Addressing to Program or Data Memory with Long Offset

DIRECT ADDRESS MODE (DA)

In Direct Address (DA) mode, the instruction provides the operand's 16-bit memory address. Jump (JP) and Call (CALL) instructions use this addressing mode to specify the 16-bit destination address that is loaded into the PC whenever a JP or CALL instruction is executed.

The LDC and LDE instructions can use Direct Address mode to specify the source or destination address for Load operations to program memory (LDC) or to external data memory (LDE), if implemented.

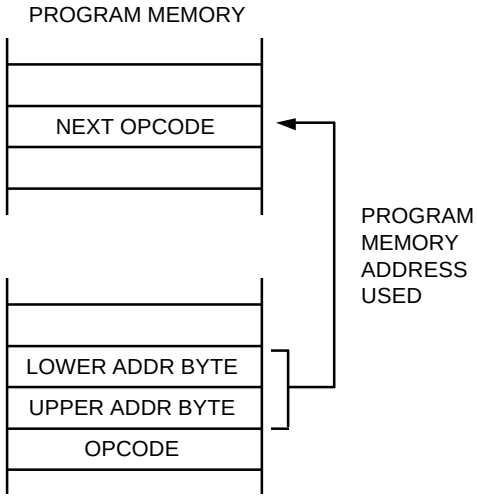


SAMPLE INSTRUCTIONS:

```
LDC    R5,1234H      ;  The values in the program address (1234H)
                        are loaded into register R5.

LDE    R5,1234H      ;  Identical operation to LDC example, except that
                        external program memory is accessed.
```

Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)



SAMPLE INSTRUCTIONS:

```
JP      C,JOB1      ;  Where JOB1 is a 16-bit immediate address

CALL    DISPLAY     ;  Where DISPLAY is a 16-bit immediate address
```

Figure 3-11. Direct Addressing for Call and Jump Instructions

RELATIVE ADDRESS MODE (RA)

In Relative Address (RA) mode, a two's-complement signed displacement between – 128 and + 127 is specified in the instruction. The displacement value is then added to the current PC value. The result is the address of the next instruction to be executed. Before this addition occurs, the PC contains the address of the instruction immediately following the current instruction.

The instructions that support RA addressing is JR.

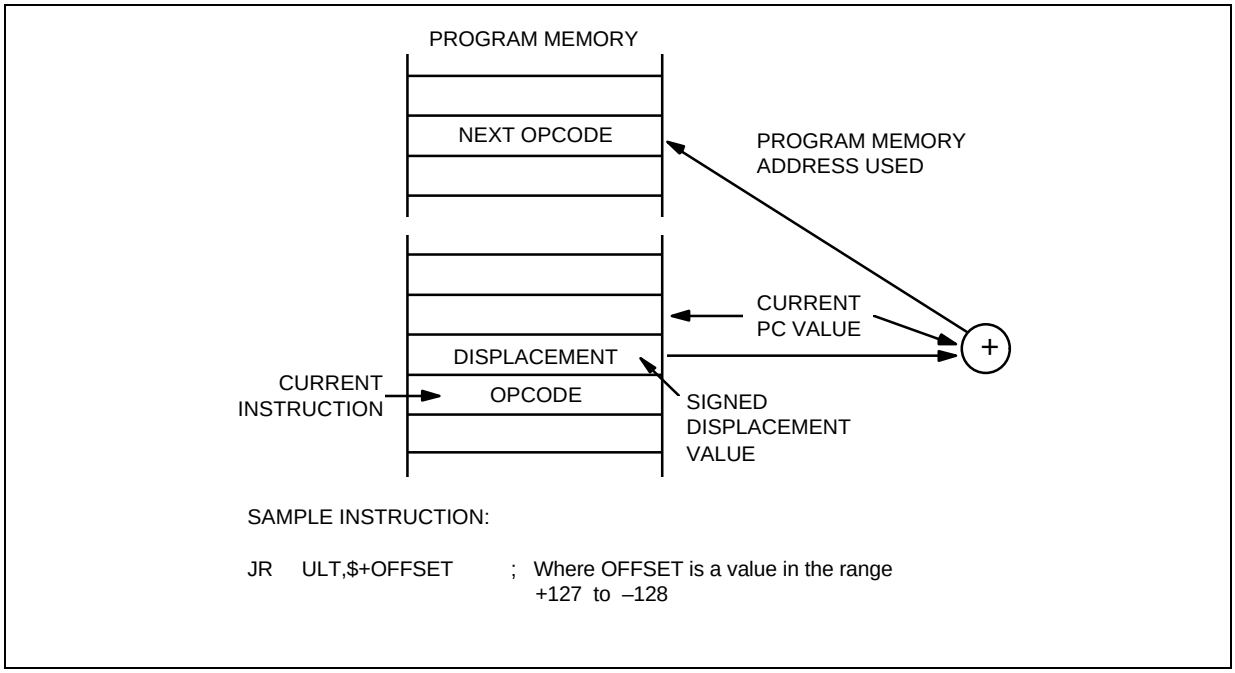


Figure 3-12. Relative Addressing

IMMEDIATE MODE (IM)

In Immediate (IM) addressing mode, the operand value used in the instruction is the value supplied in the operand field itself. Immediate addressing mode is useful for loading constant values into registers.

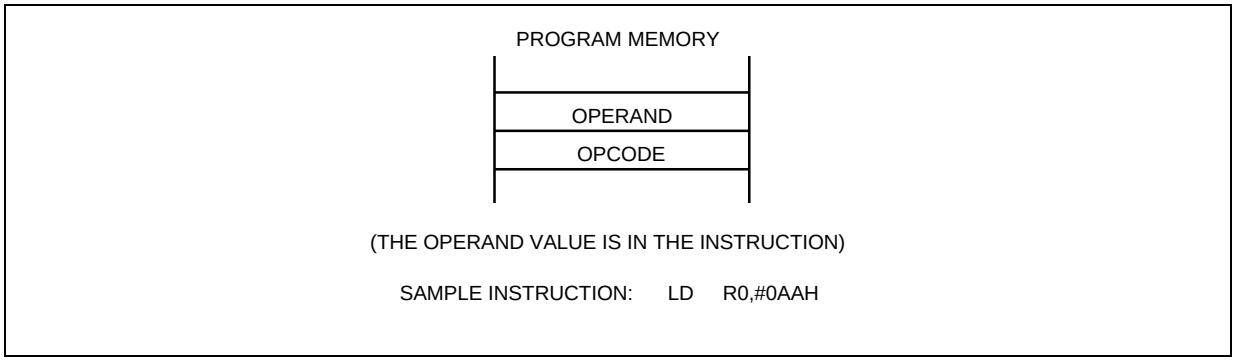


Figure 3-13. Immediate Addressing

4 CONTROL REGISTERS

OVERVIEW

In this section, detailed descriptions of the KS86C4004/C4104 control registers are presented in an easy-to-read format. These descriptions will help familiarize you with the mapped locations in the register file. You can also use them as a quick-reference source when writing application programs.

System and peripheral registers are summarized in Table 4-1. Figure 4-1 illustrates the important features of the standard register description format.

Control register descriptions are arranged in alphabetical order according to register mnemonic. More information about control registers is presented in the context of the various peripheral hardware descriptions in Part II of this manual.

Table 4-1. System and Peripheral control Registers

Register Name	Mnemonic	Hex	R/W
Timer 0 counter register	T0CNT	D0H	R
Timer 0 data register	T0DATA	D1H	R/W
Timer 0 control register (high)	T0CONH	D2H	R/W
Timer 0 control register (low)	T0CONL	D3H	R/W
Clock control register	CLKCON	D4H	R/W
System flags register	FLAGS	D5H	R/W
Locations D6H–D8H are not mapped.			
Stack pointer register	SP	D9H	R/W
Location DAH is not mapped.			
Location DBH is reserved.			
Basic timer control register	BTCON	DCH	R/W
Basic timer counter	BTCNT	DDH	R
Location DEH is reserved.			
System mode register	SYM	DFH	R/W

Table 4-1. System and Peripheral Control Registers (Continued)

Register Name	Mnemonic	Hex	R/W
Port 0 data register	P0	E0H	R/W
Port 1 data register	P1	E1H	R/W
Port 2 data register	P2	E2H	R/W
Port 3 data register	P3	E3H	R/W
Locations E4H–E5H are not mapped.			
Port 0 control register	P0CON	E6H	R/W
Port 0 pull-up resistor enable register	P0PUR	E7H	R/W
Port 0 N-channel open-drain mode register	P0PNE	E8H	R/W
Port 1 control register	P1CON	E9H	R/W
Port 2 control register	P2CON	EAH	R/W
Port 2 open-drain, pull-up resistor enable	P2DPUR	EBH	R/W
Port 2 interrupt pending register	P2PND	ECH	R/W
Location EDH is not mapped.			
Port 3 control register (high byte)	P3CONH	EEH	R/W
Port 3 control register (low byte)	P3CONL	EFH	R/W
Locations F0H–F1H are not mapped.			
Timer 1 counter register	T1CNT	F2H	R
Timer 1 control register	T1CON	F3H	R/W
Timer 1 data register	T1DATA	F4H	R/W
Zero-crossing detector control register	ZCMOD	F5H	R/W
8-bit prescaler for buzzer output	BUZPS	F6H	R/W
A/D control register	ADCON	F7H	R/W
A/D converter data register (high byte)	ADDATAH	F8H	R
A/D converter data register (low byte)	ADDATAL	F9H	R
PWM extension data register	PWMEX	FAH	R/W
PWM extension counter register	T0EXCNT	FBH	R
Locations FCH–FFH are not mapped.			

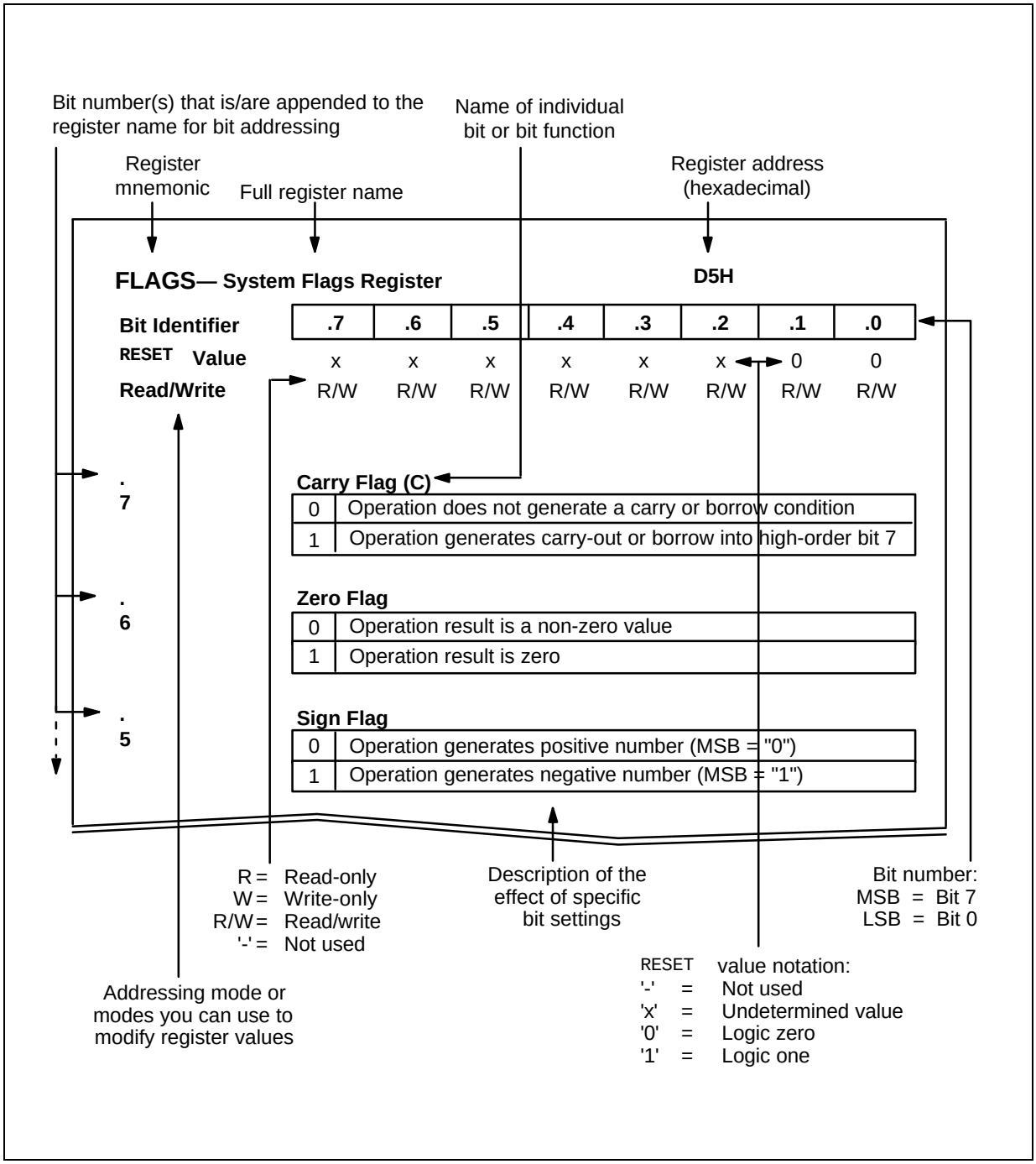


Figure 4-1. Register Description Format

ADCON — A/D Converter Control Register F7H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	0	0	0	0	–	–	0
Read/Write	–	R/W	R/W	R/W	R/W	–	–	R/W

.7 **Not used for KS86C4004/P4004/C4104/P4104**

.6–.4 **A/D Converter Input Pin Selection Bits**

0	0	0	ADC0 (P3.0)
0	0	1	ADC1 (P3.1)
0	1	0	ADC2 (P3.2)
0	1	1	ADC3 (P3.3)
1	0	0	ADC4 (P3.4)
1	0	1	ADC5 (P3.5) **(Not used for KS86C4104/P4104)**
1	1	0	ADC6 (P2.2) **(Not used for KS86C4104/P4104)**
1	1	1	ADC7 (P2.3) **(Not used for KS86C4104/P4104)**

.3 **End-of-Conversion Status Bit**

0	A/D conversion is in progress
1	A/D conversion complete

.2 and .1 **Not used for KS86C4004/P4004/C4104/P4104**

.0 **Conversion Start Bit**

0	No meaning
1	A/D conversion start

BTCON — Basic Timer Control Register DCH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7–.4 **Watchdog Timer Function Enable Bit**

1	0	1	0	Disable watchdog timer function
Others				Enable watchdog timer function

.3 and .2 **Basic Timer Input Clock Selection Bits**

0	0	$f_{OSC}/4096$
0	1	$f_{OSC}/1024$
1	0	$f_{OSC}/128$
1	1	Invalid setting

.1 **Basic Timer 8-bit Counter Clear Bit** (note)

0	No effect
1	Clear basic timer counter value

.0 **Basic Timer Divider Clear Bit** (note)

0	No effect
1	Clear both dividers

NOTE: When you write a "1" to BTCON.0 (or BTCON.1), the basic timer divider (or basic timer counter) is cleared. The bit is then cleared automatically to "0".

BUZPS — 6-Bit Prescaler for Buzzer Output F6H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 Buzzer Output Enable Bit

0	Disable buzzer output (buzzer off)
1	Enable buzzer output (buzzer on)

.6 Buzzer Clock Selection Bit

0	Divided by 128 (fx/128)
1	Divided by 32 (fx/32)

.5–.0 6-Bit Prescaler

0	0	0	0	0	0	divided by 2 [fx/(128 or 32)]
0	0	0	0	0	1	divided by 4 [fx/(128 or 32)]
0	0	0	0	1	0	divided by 6 [fx/(128 or 32)]
0	0	0	0	1	1	divided by 8 [fx/(128 or 32)]
• • •						divided by 2x(n+1) [fx/(128 or 32)]
1	1	1	1	1	1	divided by 128 [fx/(128 or 32)]

CLKCON —System Clock Control Register D4H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	–	–	0	0	–	–	–
Read/Write	R/W	–	–	R/W	R/W	–	–	–

.7 **Oscillator IRQ Wake-up Function Enable Bit**

0	Enable IRQ for main system oscillator wake-up function
1	Disable IRQ for main system oscillator wake-up function

.6 and .5 **Not used for KS86C4004/P4004/C4104/P4104**

.4 and .3 **CPU Clock (System Clock) Selection Bits** (1)

0	0	Divide by 16 (f_{OSC}/16)
0	1	Divide by 8 (f_{OSC}/8)
1	0	Divide by 2 (f_{OSC}/2)
1	1	Non-divided clock (f_{OSC}) (2)

.2–.0 **Not used for KS86C4004/P4004/C4104/P4104**

NOTES:

1. After a reset, the slowest clock (divided by 16) is selected as the system clock. To select faster clock speeds, load the appropriate values to CLKCON.3 and CLKCON.4.
2. f_{OSC} means oscillator frequency

FLAGS — System Flags Register D5H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 Carry Flag (C)

0	Operation does not generate a carry or borrow condition
1	Operation generates a carry-out or borrow into high-order bit 7

.6 Zero Flag (Z)

0	Operation result is a non-zero value
1	Operation result is zero

.5 Sign Flag (S)

0	Operation generates a positive number (MSB = "0")
1	Operation generates a negative number (MSB = "1")

.4 Overflow Flag (V)

0	Operation result is $\leq +127$ or ≥ -128
1	Operation result is $> +127$ or < -128

.3–.0 Not used for KS86C4004/P4004/C4104/P4104

P0CON — Port 0 Control Register E6H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 Port 0, P0.7 Configuration Bits (Not used for KS86C4104/P4104)

0	Normal input
1	Push-pull output

.6 Port 0, P0.6 Configuration Bits

0	Normal input
1	Push-pull output

.5 Port 0, P0.5 Configuration Bits

0	Normal input
1	Push-pull output

.4 Port 0, P0.4 Configuration Bits

0	Normal input
1	Push-pull output

.3 Port 0, P0.3 Configuration Bits

0	Normal input
1	Push-pull output

.2 Port 0, P0.2 Configuration Bits

0	Normal input
1	Push-pull output

.1 Port 0, P0.1 Configuration Bits

0	Normal input
1	Push-pull output

.0 Port 0, P0.0 Configuration Bits

0	Normal input
1	Push-pull output

P0PUR — Port 0 Pull-up Resistor Enable Register E7H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 Port 0.7 Pull-up Resistor Enable Bit (Not used for KS86C4104/P4104)

0	Disable pull-up
1	Enable pull-up

.6 Port 0.6 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

.5 Port 0.5 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

.4 Port 0.4 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

.3 Port 0.3 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

.2 Port 0.2 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

.1 Port 0.1 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

.0 Port 0.0 Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

P0PNE — Port 0 N-Channel Open-drain Mode Register E8H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 Port 0.7 N-Channel Open-drain Enable Bit (Not used for KS86C4104/P4104)

0	Configure P0.7 as a push-pull
1	Configure P0.7 as a n-channel open-drain

.6 Port 0.6 N-Channel Open-drain Enable Bit

0	Configure P0.6 as a push-pull
1	Configure P0.6 as a n-channel open-drain

.5 Port 0.5 N-Channel Open-drain Enable Bit

0	Configure P0.5 as a push-pull
1	Configure P0.5 as a n-channel open-drain

.4 Port 0.4 N-Channel Open-drain Enable Bit

0	Configure P0.4 as a push-pull
1	Configure P0.4 as a n-channel open-drain

.3 Port 0.3 N-Channel Open-drain Enable Bit

0	Configure P0.3 as a push-pull
1	Configure P0.3 as a n-channel open-drain

.2 Port 0.2 N-Channel Open-drain Enable Bit

0	Configure P0.2 as a push-pull
1	Configure P0.2 as a n-channel open-drain

.1 Port 0.1 N-Channel Open-drain Enable Bit

0	Configure P0.1 as a push-pull
1	Configure P0.1 as a n-channel open-drain

.0 Port 0.0 N-Channel Open-drain Enable Bit

0	Configure P0.0 as a push-pull
1	Configure P0.0 as a n-channel open-drain

P1CON — Port 1 Control Register E9H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6 **Port 1, P1.3/CLO Configuration Bits (Not used for KS86C4104/P4104)**

0	0	Schmitt trigger input
0	1	Schmitt trigger input; pull-up resistor enable
1	0	Push-pull output
1	1	Alternative function (CLO output)

.5 and .4 **Port 1, P1.2/T0 Configuration Bits**

0	0	Schmitt trigger input (or T0 Capture input)
0	1	Schmitt trigger input; pull-up resistor enable
1	0	Push-pull output
1	1	Alternative function (T0 output: match or PWM)

.3 and .2 **Port 1, P1.1/BUZ Configuration Bits**

0	0	Schmitt trigger
0	1	Schmitt trigger input; pull-up resistor enable
1	0	Push-pull output
1	1	Alternative function (BUZ output)

.1 and .0 **Port 1, P1.0 /ZCD Configuration Bits**

0	0	Schmitt trigger
0	1	Schmitt trigger input; pull-up resistor enable
1	0	Push-pull output
1	1	Alternative function (ZCD input); ZCD enable

P2CON — Port 2 Control Register EAH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6 **Port 2, P2.3/AD7 Configuration Bits (Not used for KS86C4104/P4104)**

0	0	Schmitt trigger input
0	1	Schmitt trigger input
1	0	Push-pull output
1	1	A/D converter input (AD7); Schmitt trigger input off

.5 and .4 **Port 2, P2.2/AD6 Configuration Bits (Not used for KS86C4104/P4104)**

0	0	Schmitt trigger input
0	1	Schmitt trigger input
1	0	Push-pull output
1	1	A/D converter input (AD6); Schmitt trigger input off

.3 and .2 **Port 2, P2.1/INT1 Configuration Bits (Not used for KS86C4104/P4104)**

0	0	Schmitt trigger input; INT1 interrupt disable
0	1	Schmitt trigger input; interrupt on falling edge
1	0	Push-pull output
1	1	Schmitt trigger input; interrupt on rising edge

.1 and .0 **Port 2, P2.0/INT0 Configuration Bits**

0	0	Schmitt trigger input; INT0 interrupt disable
0	1	Schmitt trigger input; interrupt on falling edges
1	0	Push-pull output
1	1	Schmitt trigger input; interrupt on rising edges

P2DPUR — Port 2 Open-drain Enable & Pull-up Resistor Enable Register EBH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 Port 2.3/AD7, Open-drain Enable Bit (Not used for KS86C4104/P4104)

0	Configure P2.3 as a push-pull
1	Configure P2.3 as a n-channel open drain

.6 Port 2.2/AD6, Open-drain Enable Bit (Not used for KS86C4104/P4104)

0	Configure P2.2 as a push-pull
1	Configure P2.2 as a n-channel open drain

.5 Port 2.1/INT1, Open-drain Enable Bit (Not used for KS86C4104/P4104)

0	Configure P2.1 as a push-pull
1	Configure P2.1 as a n-channel open drain

.4 Port 2.0/INT0, Open-drain Enable Bit

0	Configure P2.0 as a push-pull
1	Configure P2.0 as a n-channel open drain

.3 Port 2.3/AD7, Pull-up Resistor Enable Bit (Not used for KS86C4104/P4104)

0	Disable pull-up
1	Enable pull-up

.2 Port 2.2/AD6, Pull-up Resistor Enable Bit (Not used for KS86C4104/P4104)

0	Disable pull-up
1	Enable pull-up

.1 Port 2.1/INT1, Pull-up Resistor Enable Bit (Not used for KS86C4104/P4104)

0	Disable pull-up
1	Enable pull-up

.0 Port 2.0/INT0, Pull-up Resistor Enable Bit

0	Disable pull-up
1	Enable pull-up

NOTE: In order to use the open-drain output mode the push-pull output bit in the P2CON should be set first.

P2PND — Port 2 Interrupt Pending Register ECH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	–	–	0	0
Read/Write (NOTE)	–	–	–	–	–	–	R/W	R/W

.7–.2 **Not used for KS86C4004/P4004/C4104/P4104**

.1 **P2.1/INT1, Interrupt Pending Bit (Not used for KS86C4104/P4104)**

0	No interrupt pending *(when read)*
0	Clear this pending bit *(when write)*
1	Interrupt is pending *(when read)*/No effect *(when write)*

.0 **P2.0/INT0, Interrupt Pending Bit**

0	No interrupt pending *(when read)*
0	Clear this pending bit *(when write)*
1	Interrupt is pending *(when read)*/No effect *(when write)*

NOTES:

1. To clear an interrupt pending condition at a port2 pin, you must write a “0” to the corresponding P2PND bit location.
2. To avoid programming errors, we recommend using load instructions when manipulating P2PND values.

P3CONH — Port 3 Control Register (High Byte) EEH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	0	0	0	0
Read/Write	–	–	–	–	R/W	R/W	R/W	R/W

.7–.4 **Not used for KS86C4004/P4004/C4104/P4104**

.3 and .2 **Port 3, P3.5/ADC5 Configuration Bits (Not used for KS86C4104/P4104)**

0	0	Schmitt trigger input
0	1	Schmitt trigger input, pull-up resistor enabled
1	0	Push-pull output
1	1	A/D converter input (ADC5); Schmitt trigger input off

.1 and .0 **Port 3, P3.4/ADC4 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input, pull-up resistor enabled
1	0	Push-pull output
1	1	A/D converter input (ADC4); Schmitt trigger input off

P3CONL — Port 3 Control Register (Low Byte) EFH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6 **Port 3, P3.3/ADC3 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input, pull-up resistor enabled
1	0	Push-pull output
1	1	A/D converter input (ADC3); Schmitt trigger input off

.5 and .4 **Port 3, P3.2/ADC2 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input, pull-up resistor enabled
1	0	Push-pull output
1	1	A/D converter input (ADC2); Schmitt trigger input off

.3 and .2 **Port 3, P3.1/ADC1 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input, pull-up resistor enabled
1	0	Push-pull output
1	1	A/D converter input (ADC1); Schmitt trigger input off

.1 and .0 **Port 3, P3.0/ADC0 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input, pull-up resistor enabled
1	0	Push-pull output
1	1	A/D converter input (ADC0); Schmitt trigger input off

SYM — System Mode Register DFH

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	–	0	0	0
Read/Write	–	–	–	–	–	R/W	R/W	R/W

.7–.3 **Not used for KS86C4004/P4004/C4104/P4104**

.2 **Global Interrupt Enable Bit** (note)

0	Disable all interrupt (DI instruction)
1	Enable all interrupt (EI Instruction)

.1 and .0 **Page Selection Bits**

0	0	page 0
0	1	page 1 **(not used for KS86C4004/P4004/C4104/P4104)**
1	0	page 2 **(not used for KS86C4004/P4004/C4104/P4104)**
1	1	page 3 **(not used for KS86C4004/P4004/C4104/P4104)**

NOTE: Following a reset, you enable global interrupt processing by executing an EI instruction (not by writing a "1" to SYM.2).

T0CONH — TIMER 0 Control Register (High Byte) D2H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0 (8)
RESET Value	–	–	–	–	–	–	–	0
Read/Write	–	–	–	–	–	–	–	R/W

.7–.1 **Not used for KSC4004/P4004/C4104/P4104**

.0 (8) **Timer 0 Overflow Interrupt Pending Bit (overflow interrupt)**

0	No interrupt pending *(when read)*
0	Clear Pending bit *(when write)*
1	Interrupt is pending *(when read)*

T0CONL — TIMER 0 Control Register (Low Byte) D3H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6 **Timer 0 Input Clock Selection Bits**

0	0	$f_{osc}/4096$
0	1	$f_{osc}/256$
1	0	$f_{osc}/8$
1	1	$f_{osc}/1$

.5 and .4 **Timer 0 Operating Mode Selection Bits**

0	0	Interval mode
0	1	Capture mode (capture on rising edge, counter running, OVF)
1	0	Capture mode (capture on falling edge, counter running, OVF)
0	0	PWM mode (OVF interrupt can occur)

.3 **Timer 0 Counter Clear Bit**

0	No effect
1	clears the timer 0 counter (when write)

.2 **Timer 0 Overflow Interrupt Enable Bit**

0	Disable overflow interrupt
1	Enable overflow interrupt

.1 **Timer 0 Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0 **Timer 0 Interrupt Pending Bit (Capture or Match Interrupt)**

0	No interrupt pending *(when read)*
0	Clear pending bit *(when write)*
1	Interrupt is pending *(when read)*

T1CON — Timer 1 Control Register F3H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	0	0	0	0	0	0
Read/Write	–	–	R/W	R/W	R/W	R/W	R/W	R/W

.7 and .6 **Not used for KS86C4004/P4004/C4104/P4104**

.5 and .4 **Timer 1 Input Clock Selection Bits**

0	0	$f_{osc}/512$
0	1	$f_{osc}/256$
1	0	$f_{osc}/128$
1	1	$f_{osc}/64$

.3 **Timer 1 Counter Automatic Clear Enable Bit**

0	Disable
1	Enable ZCD clear signal to clear the timer 1 counter

.2 **Timer 1 Counter Clear Enable Bit**

0	No effect
1	Clear the timer 1 counter (when write)

.1 **Timer 1 Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0 **Timer 1 Interrupt Pending Bit**

0	No interrupt pending *(when read)*
0	Clear pending bit *(when write)*
1	Interrupt is pending *(when read)*

ZCMOD — Zero Crossing Detection Control Register F5H

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	0	0	0	0
Read/Write	–	–	–	–	R/W	R/W	R/W	R/W

.7–.4 **Not used for KS86C4004/P4004/C4104/P4104**

.3 and .2 **Interrupt Mode Selection Bits**

0	0	Interrupt on falling edge
0	1	Interrupt on rising edge
1	0	Interrupt on both edge
1	1	Not used

.1 **ZCD Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0 **ZCD Interrupt Pending Bit**

0	No interrupt pending *(when read)*
0	Clear pending bit *(when write)*
1	Interrupt is pending *(when read)*

NOTES

5 INTERRUPT STRUCTURE

OVERVIEW

The SAM87Ri interrupt structure has two basic components: a vector, and sources. The number of interrupt sources can be serviced through an interrupt vector which is assigned in ROM address 0000H.

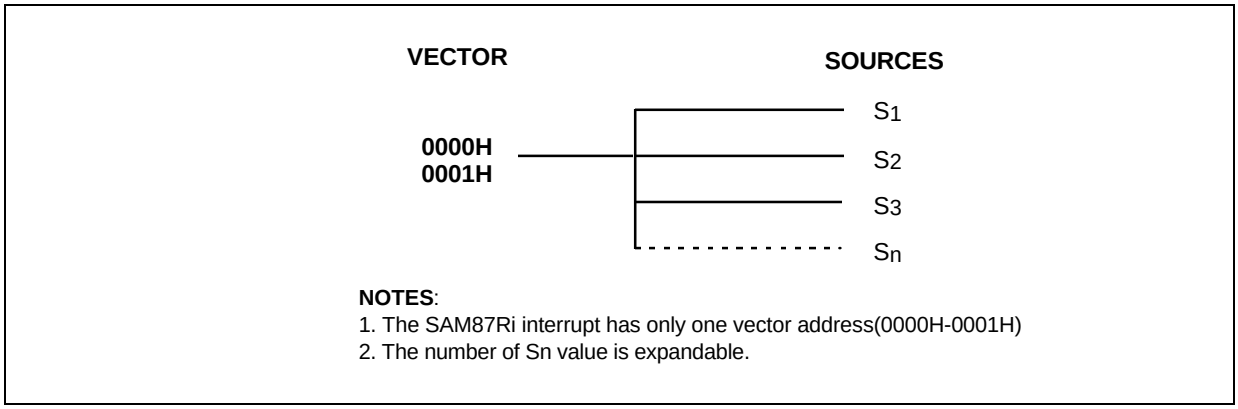


Figure 5-1. KS86-Series Interrupt Type

INTERRUPT PROCESSING CONTROL POINTS

Interrupt processing can be controlled in two ways: either globally, or by specific interrupt level and source. The system-level control points in the interrupt structure are therefore:

— Global interrupt enable and disable (by EI and DI instructions)

— Interrupt source enable and disable settings in the corresponding peripheral control register(s)

ENABLE/DISABLE INTERRUPT INSTRUCTIONS (EI, DI)

The system mode register, SYM (DFH), is used to enable and disable interrupt processing.

SYM.2 is the enable and disable bit for global interrupt processing respectively, by modifying SYM.2. An Enable Interrupt (EI) instruction must be included in the initialization routine that follows a reset operation in order to enable interrupt processing. Although you can manipulate SYM.2 directly to enable and disable interrupts during normal operation, we recommend that you use the EI and DI instructions for this purpose.

INTERRUPT PENDING FUNCTION TYPES

When the interrupt service routine has executed, the application program's service routine must clear the appropriate pending bit before the return from interrupt subroutine (IRET) occurs.

INTERRUPT PRIORITY

Because there is not a interrupt priority register in SAM87Ri, the order of service is determined by a sequence of source which is executed in interrupt service routine.

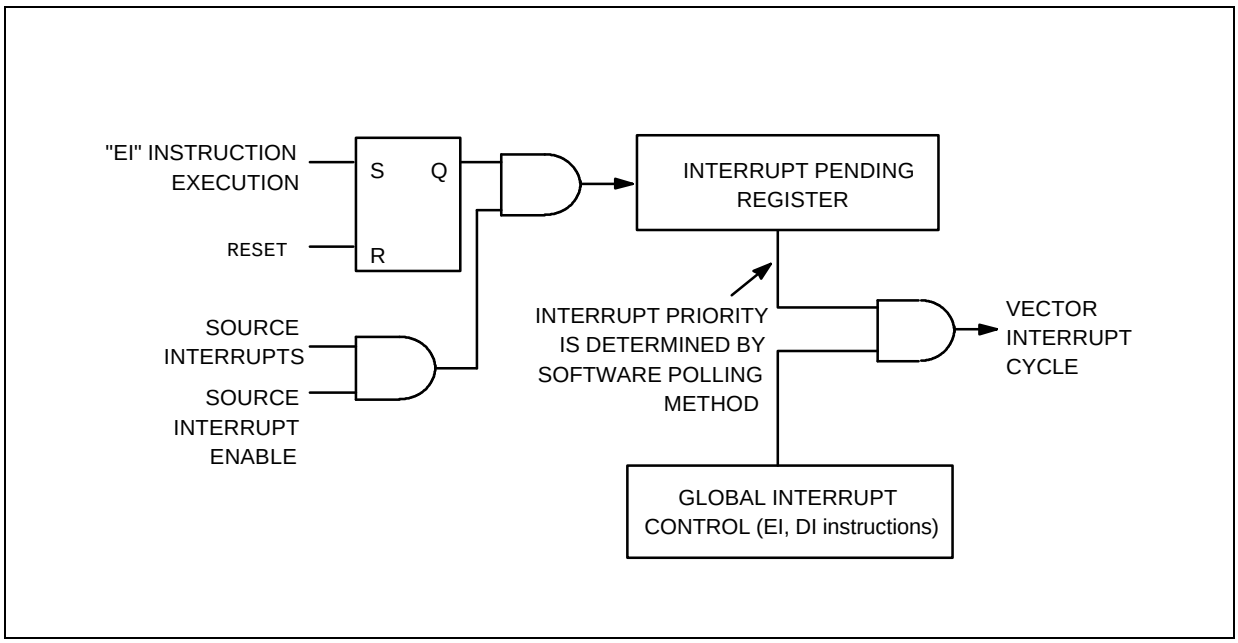


Figure 5-2. Interrupt Function Diagram

INTERRUPT SOURCE SERVICE SEQUENCE

The interrupt request polling and servicing sequence is as follows:

1. A source generates an interrupt request by setting the interrupt request pending bit to "1".
2. The CPU generates an interrupt acknowledge signal.
3. The service routine starts and the source's pending flag is cleared to "0" by software.
4. Interrupt priority must be determined by software polling method.

INTERRUPT SERVICE ROUTINES

Before an interrupt request can be serviced, the following conditions must be met:

— Interrupt processing must be enabled (EI, SYM.2 = "1")

— Interrupt must be enabled at the interrupt's source (peripheral control register)

If all of the above conditions are met, the interrupt request is acknowledged at the end of the instruction cycle. The CPU then initiates an interrupt machine cycle that completes the following processing sequence:

1. Reset (clear to "0") the global interrupt enable bit in the SYM register (DI, SYM.2 = "0") to disable all subsequent interrupts.
2. Save the program counter and status flags to stack.
3. Branch to the interrupt vector to fetch the service routine's address.
4. Pass control to the interrupt service routine.

When the interrupt service routine is completed, an Interrupt Return instruction (IRET) occurs. The IRET restores the PC and status flags and sets SYM.2 to "1" (EI), allowing the CPU to process the next interrupt request.

GENERATING INTERRUPT VECTOR ADDRESSES

The interrupt vector area in the ROM contains the address of the interrupt service routine. Vectored interrupt processing follows this sequence:

1. Push the program counter's low-byte value to stack.
2. Push the program counter's high-byte value to stack.
3. Push the FLAGS register values to stack.
4. Fetch the service routine's high-byte address from the vector address 0000H.
5. Fetch the service routine's low-byte address from the vector address 0001H.
6. Branch to the service routine specified by the 16-bit vector address.

KS86C4004/C4104 INTERRUPT STRUCTURE

The KS86C4004/C4104 microcontroller has six peripheral interrupt sources:

— Timer 0 match/capture interrupt
— Timer 0 overflow interrupt
— Timer 1 match interrupt
— Zero-cross detection
— Two external interrupts for port 2, P2.0–P2.1 (P2.1 not used for KS86C4104/P4104)

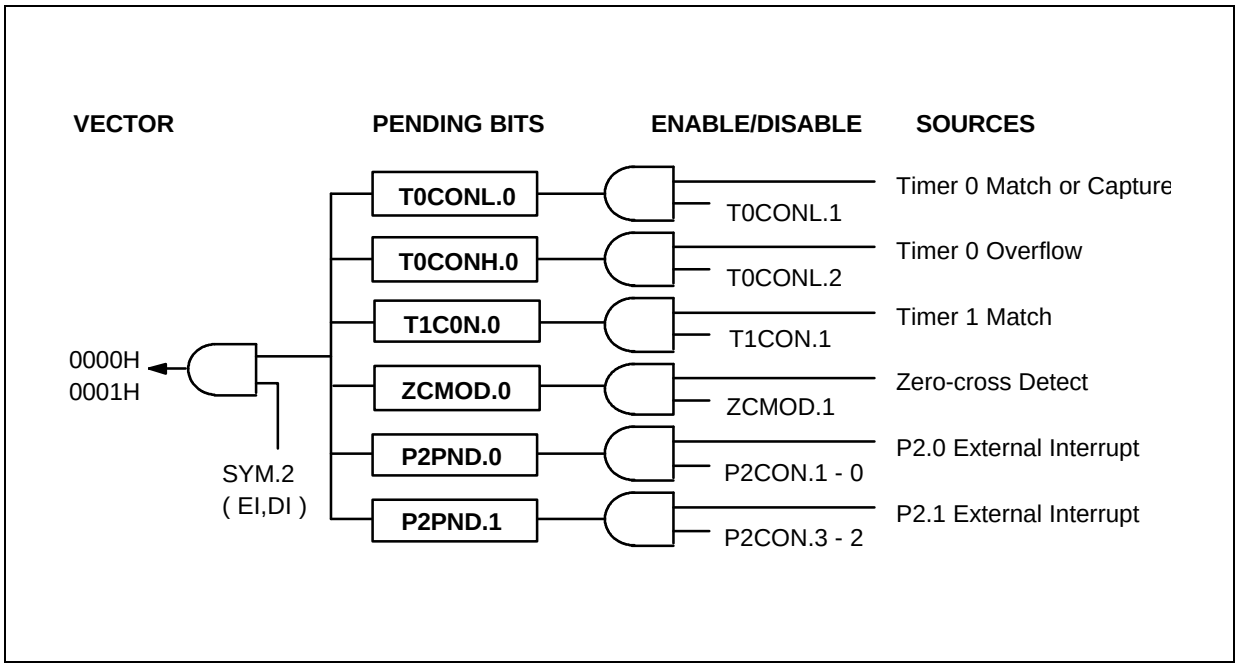


Figure 5-3. KS86C4004/C4104 Interrupt Structure

7 CLOCK CIRCUIT

OVERVIEW

An RC oscillation source provides a typical 4 MHz clock for KS86C4004/C4104. An internal capacitor supports the RC oscillator circuit. An external crystal or ceramic oscillation source provides a maximum 10 MHz clock. The X_{IN} and X_{OUT} pins connect the oscillation source to the on-chip clock circuit. Simplified RC oscillator and crystal/ceramic oscillator circuits are shown in Figures 7-1 and 7-2.

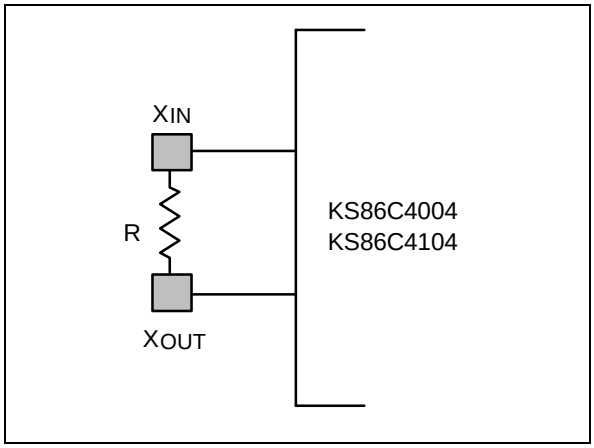


Figure 7-1. Main Oscillator Circuit (RC Oscillator with Internal Capacitor)

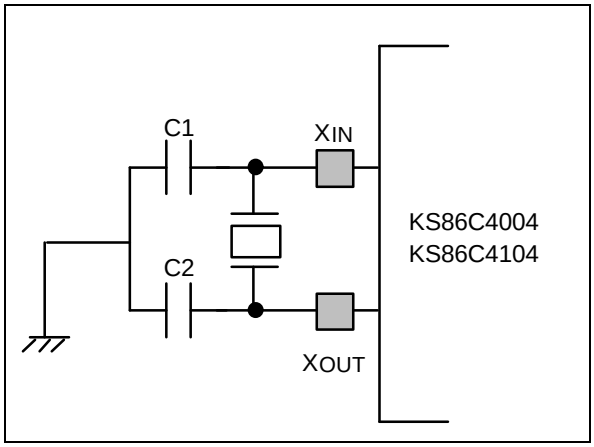


Figure 7-2. Main Oscillator Circuit (Crystal/Ceramic Oscillator)

MAIN OSCILLATOR LOGIC

To increase processing speed and to reduce clock noise, non-divided logic is implemented for the main oscillator circuit. For this reason, very high resolution waveforms (square signal edges) must be generated in order for the CPU to efficiently process logic operations.

CLOCK STATUS DURING POWER-DOWN MODES

The two power-down modes, Stop mode and Idle mode, affect clock oscillation as follows:

— In Stop mode, the main oscillator "freezes", halting the CPU and peripherals. The contents of the register file and current system register values are retained. Stop mode is released, and the oscillator started, by a reset operation or by an external interrupt with RC-delay noise filter (for KS86C4004/C4104, INT0–INT1).

— In Idle mode, the internal clock signal is gated off to the CPU, but not to interrupt control and the timer. The current CPU status is preserved, including stack pointer, program counter, and flags. Data in the register file is retained. Idle mode is released by a reset or by an interrupt (external or internally-generated).

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

The system clock control register, CLKCON, is located in location D4H. It is read/write addressable and has the following functions:

— Oscillator IRQ wake-up function enable/disable (CLKCON.7)

— Oscillator frequency divide-by value: non-divided, 2, 8, or 16 (CLKCON.4 and CLKCON.3)

The CLKCON register controls whether or not an external interrupt can be used to trigger a Stop mode release (This is called the "IRQ wake-up" function). The IRQ wake-up enable bit is CLKCON.7.

After a reset, the external interrupt oscillator wake-up function is enabled, the main oscillator is activated, and the $f_{OSC}/16$ (the slowest clock speed) is selected as the CPU clock. If necessary, you can then increase the CPU clock speed to f_{OSC}, $f_{OSC}/2$ or $f_{OSC}/8$.

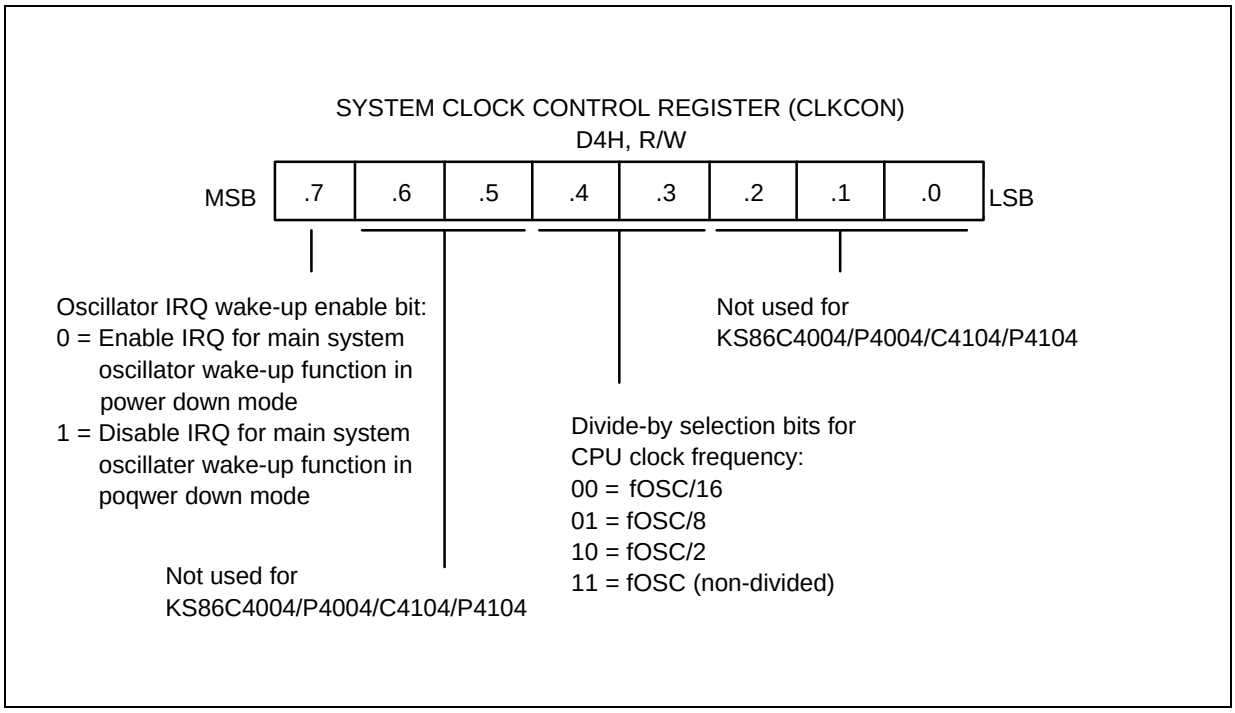


Figure 7-3. System Clock Control Register (CLKCON)

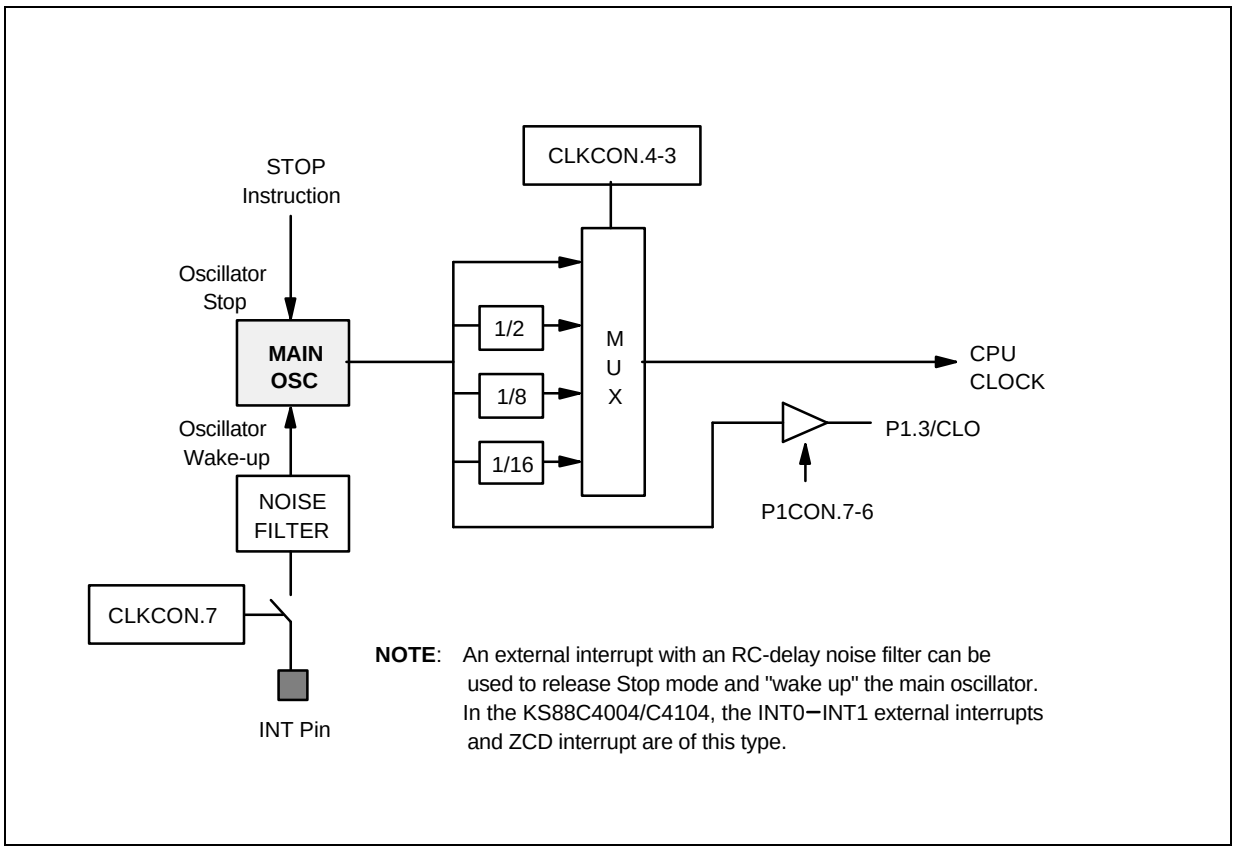


Figure 7-4. System Clock Circuit Diagram

NOTES

8 RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

During a power-on reset, the voltage at V_{DD} is High level and the RESET pin is forced to Low level. The RESET signal is input through a Schmitt trigger circuit where it is then synchronized with the CPU clock. This brings the KS86C4004/4104 into a known operating status.

The RESET pin must be held to Low level for a minimum time interval after the power supply comes within tolerance in order to allow time for internal CPU clock oscillation to stabilize. The minimum required oscillation stabilization time for a reset is approximately 6.55 ms ($\cong 2^{16}/f_{osc}$, f_{osc} = 10 MHz).

When a reset occurs during normal operation (with both V_{DD} and RESET at High level), the signal at the RESET pin is forced Low and the reset operation starts. All system and peripheral control registers are then set to their default hardware reset values (see Table 8-1).

The following sequence of events occurs during a reset operation:

— All interrupts are disabled.
— The watchdog function (basic timer) is enabled.
— Ports 0–3 are set to input mode and all pull-up resistors are disabled.
— Peripheral control and data registers are disabled and reset to their initial values.
— The program counter is loaded with the ROM reset address, 0100H.
— When the programmed oscillation stabilization time interval has elapsed, the address stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

To program the duration of the oscillation stabilization interval, you must make the appropriate settings to the basic timer control register, BTCON, before entering Stop mode. Also, if you do not want to use the basic timer watchdog function (which causes a system reset if a basic timer counter overflow occurs), you can disable it by writing "1010B" to the upper nibble of BTCON.

POWER-DOWN MODES

STOP MODE

Stop mode is invoked by the instruction STOP (opcode 7FH). In Stop mode, the operation of the CPU and all peripherals is halted. That is, the on-chip main oscillator stops and the supply current is reduced to less than 5 µA. All system functions are halted when the clock "freezes", but data stored in the internal register file is retained. Stop mode can be released in one of two ways: by a RESET signal or by an external interrupt.

Using RESET to Release Stop Mode

Stop mode is released when the RESET signal is released and returns to High level. All system and peripheral control registers are then reset to their default values and the contents of all data registers are retained. A reset operation automatically selects a slow clock (1/16) because CLKCON.3 and CLKCON.4 are cleared to "00B". After the oscillation stabilization interval has elapsed, the CPU executes the system initialization routine by fetching the 16-bit address stored in ROM locations 0100H and 0101H.

Using an External Interrupt to Release Stop Mode

Only external interrupts with an RC-delay noise filter circuit can be used to release Stop mode (Clock-related external interrupts cannot be used). External interrupts INT0–INT1 in the KS86C4004/4104 interrupt structure meet this criteria.

Note that when Stop mode is released by an external interrupt, the current values in system and peripheral control registers are not changed. When you use an interrupt to release Stop mode, the CLKCON.3 and CLKCON.4 register values remain unchanged, and the currently selected clock value is used. If you use an external interrupt for Stop mode release, you can also program the duration of the oscillation stabilization interval. To do this, you must make the appropriate control and clock settings *before* entering Stop mode.

The external interrupt is serviced when the Stop mode release occurs. Following the IRET from the service routine, the instruction immediately following the one that initiated Stop mode is executed.

IDLE MODE

Idle mode is invoked by the instruction IDLE (opcode 6FH). In Idle mode, CPU operations are halted while select peripherals remain active. During Idle mode, the internal clock signal is gated off to the CPU, but not to interrupt logic and timer/counters. Port pins retain the mode (input or output) they had at the time Idle mode was entered.

There are two ways to release Idle mode:

1. Execute a reset. All system and peripheral control registers are reset to their default values and the contents of all data registers are retained. The reset automatically selects a slow clock (1/16) because CLKCON.3 and CLKCON.4 are cleared to "00B". If interrupts are masked, a reset is the only way to release Idle mode.
2. Activate any enabled interrupt, causing Idle mode to be released. When you use an interrupt to release Idle mode, the CLKCON.3 and CLKCON.4 register values remain unchanged, and the currently selected clock value is used. The interrupt is then serviced. Following the IRET from the service routine, the instruction immediately following the one that initiated Idle mode is executed.

NOTES

1. Only external interrupts that are not clock-related can be used to release stop mode. To release Idle mode, however, any type of interrupt (that is, internal or external) can be used.
2. Before enter the STOP or IDLE mode, the ZCD (P1CON) and ADC (P2CON, P3CONH, P3CONL) must be disabled. Otherwise, the STOP or IDLE current will be increased significantly.

HARDWARE RESET VALUES

Table 8-1 lists the values for CPU and system registers, peripheral control registers, and peripheral data registers following a reset operation in normal operating mode.

— A "1" or a "0" shows the reset bit value as logic one or logic zero, respectively.

— An "x" means that the bit value is undefined following a reset.

— A dash ("–") means that the bit is either not used or not mapped.

Table 8-1. Register Values after a Reset

Register Name	Mnemonic	Address	Bit Values After RESET							
			7	6	5	4	3	2	1	0
General purpose register (page 0)	–	00H–BFH	x	x	x	x	x	x	x	x
Working registers	R0–R15	C0H–CFH	x	x	x	x	x	x	x	x
Timer 0 counter register	T0CNT	D0H	0	0	0	0	0	0	0	0
Timer 0 data register	T0DATA	D1H	1	1	1	1	1	1	1	1
Timer 0 control register (high)	T0CONH	D2H	–	–	–	–	–	–	–	0
Timer 0 control register (low)	T0CONL	D3H	0	0	0	0	0	0	0	0
Clock control register	CLKCON	D4H	0	–	–	0	0	–	–	–
System flags register	FLAGS	D5H	x	x	x	x	–	–	–	–
Locations D6H–D8H are not mapped.										
Stack pointer register	SP	D9H	x	x	x	x	x	x	x	x
Location DAH is not mapped.										
Location DBH is reserved.										
Basic timer control register	BTCON	DCH	0	0	0	0	0	0	0	0
Basic timer counter	BTCNT	DDH	0	0	0	0	0	0	0	0
Location DEH is reserved.										
System mode register	SYM	DFH	–	–	–	–	–	0	0	0

Table 8-1. Register Values After a Reset (continued)

Bank 0 Register Name	Mnemonic	Address	Bit Values After a Reset							
			7	6	5	4	3	2	1	0
Port 0 data register	P0	E0H	0	0	0	0	0	0	0	0
Port 1 data register	P1	E1H	–	–	–	–	0	0	0	0
Port 2 data register	P2	E2H	–	–	–	–	0	0	0	0
Port 3 data register	P3	E3H	–	–	0	0	0	0	0	0
Locations E4H–E5H are not mapped.										
Port 0 control register	P0CON	E6H	0	0	0	0	0	0	0	0
Port 0 pull-up resistor enable register	P0PUR	E7H	0	0	0	0	0	0	0	0
Port 0 N-channel open-drain mode	P0PNE	E8H	0	0	0	0	0	0	0	0
Port 1 control register	P1CON	E9H	0	0	0	0	0	0	0	0
Port 2 control register	P2CON	EAH	0	0	0	0	0	0	0	0
Port 2 open-drain, pull-up resistor enable	P2DPUR	EBH	0	0	0	0	0	0	0	0
Port 2 interrupt pending register	P2PND	ECH	–	–	–	–	–	–	0	0
Location EDH is not mapped.										
Port 3 control register (high byte)	P3CONH	EEH	–	–	–	–	0	0	0	0
Port 3 control register (low byte)	P3CONL	EFH	0	0	0	0	0	0	0	0
Locations F0H–F1H are not mapped.										
Timer 1 counter register	T1CNT	F2H	0	0	0	0	0	0	0	0
Timer 1 control register	T1CON	F3H	–	–	0	0	0	0	0	0
Timer 1 data register	T1DATA	F4H	1	1	1	1	1	1	1	1
Zero-crossing detector control register	ZCMOD	F5H	–	–	–	–	0	0	0	0
8-bit prescaler for buzzer output	BUZPS	F6H	0	0	0	0	0	0	0	0
A/D control register	ADCON	F7H	–	0	0	0	0	–	–	0
A/D converter data register (high byte)	ADDATAH	F8H	x	x	x	x	x	x	x	x
A/D converter data register (low byte)	ADDATAL	F9H	0	0	0	0	0	0	x	x
PWM extension data register	PWMEX	FAH	–	–	–	–	–	–	0	0
PWM extension counter register	T0EXCNT	FBH	–	–	–	–	–	–	0	0
Locations FCH–FFH are not mapped.										

NOTE: "–" means not mapped, "x" means undefined.

☞ PROGRAMMING TIP — Sample KS86C4004 Initialization Routine

The following sample program suggests how to program the initial program settings for KS86C4004 address space, interrupt, and peripheral function. Program comment guide you through the necessary steps.

```
;-------<< Interrupt vector address >>

                .ORG     0000H
                .VECTOR  00H, COMMON_INT        ; IRQ0/Interrupt vector address

;-------<< Initialize system and peripherals >>

                .ORG    0100H                   ; Reset start address
RESET:          DI                              ; disable interrupt
                LD      BTCON,#00000010B        ; enable watchdog function
                                                ; clock source: fosc/4096 (104 ms overflow at 10 MHz)
                LD      CLKCON,#00011000B       ; CPU clock source select (non-divided)
                LD      SP,#0C0H                ; KS86C4004 Stack pointer initial
                LD      P0CON,#0FFH             ; push-pull output (LED direct drive → Low active)
                LD      P0PUR,#00H              ; disable pull-up resistor
                LD      P0PNE,#00H              ; disable open-drain
                LD      P1CON,#1010101011B      ; P1.0 ZCD input enable/P1.1-3 push-pull
                LD      ZCMOD,#00000010B        ; enable falling edge interrupt
                LD      T1DATA,#81H
                LD      T1CON,#00001100B        ; ZCD clear enable (fosc/512)
                                                ; timer 1 interrupt disable
                LD      P2CON,#11000110B        ; P2.3 A/D input mode/P2.2 input mode
                                                ; P2.1 falling edge interrupt
                LD      P2DPUR,#00010110B       ; P2.0 open-drain output mode
                                                ; P2.2 pull-up enable
                                                ; P2.1 pull-up enable
                LD      P2PND,#00H              ; no interrupt pending
                LD      P3CONH,#0FH             ; AD input mode (AD4,AD5)
                LD      P3CONL,#0AAH            ; P3.0-3 push-pull output mode
                •
                •
                •

;-------<< Initialize data register >>

                LD      R0,#0BFH
RAMCLR:         CLR     @R0                     ; (00h–0BFh) ← #00h
                DEC     R0
                JR      NZ,RAMCLR               ; Initialize data register
                •
                •
                LD      T0DATA,#26H             ; 1 ms interval at 10 MHz system clock
                LD      T0CONH,#00H             ; timer 0 overflow interrupt disable
                LD      T0CONL,#01001010B       ; timer 0 match interrupt enable
                                                ; clock source: fosc/256
                EI                              ; enable interrupt
```

```
;-------<< Main loop >>
MAIN:       •
            CALL    XXX                 ; subroutine call
            CALL    YYY                 ; subroutine call
            •
            •
            LD      BTCON,#02H          ; enable watchdog function
                                        ; basic counter (BTCNT) clear
            JP      T,MAIN              ; for main loop

;-------<< Subroutines >>
XXX:        •
            •
            RET
YYY:        •
            •
            •
            RET
            •
            •
            •

;-------<< Interrupt service routine >>

COMMON_INT: TM      ZCMOD,#00000001B
            JP      NZ,ZCD_INT
            TM      T1CON,#00000001B
            JP      NZ,TIMER1_INT
            TM      T0CONL,#00000001B
            JP      NZ,TIMER0_INT
            TM      P2PND,#00000001B
            JP      NZ,EXT20_INT
            TM      P2PND,#00000010B
            JP      NZ,EXT21_INT
            TM      T0CONH,#00000001B
            JP      T0OVERFLOW_INT
            IRET

TIMER0_INT:     •
            AND     T0CONL,#11110110B   ; timer0 pending bit clear
            •
            •
            IRET                        ; timer0 interrupt return

TIMER1_INT:     •
            AND     T1CON,#11111100B    ; timer1 pending bit clear/timer 1 interrupt disable
            XOR     P1,#00000100B       ; P1.2 toggle
            IRET
```

```
ZCD_INT:        AND     ZCMOD,#11111110B    ; pending bit clear
                XOR     P1,#00001000B       ; P1.3 toggle
                LD      T1CON,#00001010B    ; timer1 interrupt enable (fosc/512)
                                            ; enable ZCD clear signal to clear the timer 1 counter
                IRET

EXT20_INT:      PUSH    R5
                LD      P2PND,#00000010B    ; P2.0 pending bit clear
                •
                LD      R5,#X1
                POP     R5
                IRET

EXT21_INT:      PUSH    R8
                LD      P2PND,#00000001B    ; P2.1 pending bit clear
                •
                AND     R8,#X2
                POP     R8
                IRET

T0OVERFLOW_INT: NOP
                LD      T0CONH,#0           ; overflow pending bit clear
                INC     CAPTURE_BUFH
                IRET
                •
                •
                END
```

NOTES

9 I/O PORTS

OVERVIEW

The KS86C4004/C4104 has four I/O ports (0–3): KS86C4004/P4004, with 22 pins total and KS86C4104/P4104, with 16 pins total. You access these ports directly by writing or reading port data register addresses.

Port 0 can be configured as LED drive. (High current output: typical 15 mA)

Table 9-1. KS86C4004/C4104 Port Configuration Overview

Port	Function Description	Programmability
0	Bit-programmable I/O port for normal input or push-pull, open-drain output. Pull-up resistors are assignable by software	Bit
1	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are assignable by software. Port1 pins can also be used as alternative function. (ZCD, PWM, buzzer)	Bit
2	Bit-programmable I/O port for Schmitt trigger input or push-pull, open drain output. Pull-up resistors are assignable by software. Port2 can also be used as external interrupt, A/D converter input.	Bit
3	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are assignable by software. Port3 pins can also be used as A/D converter input.	Bit

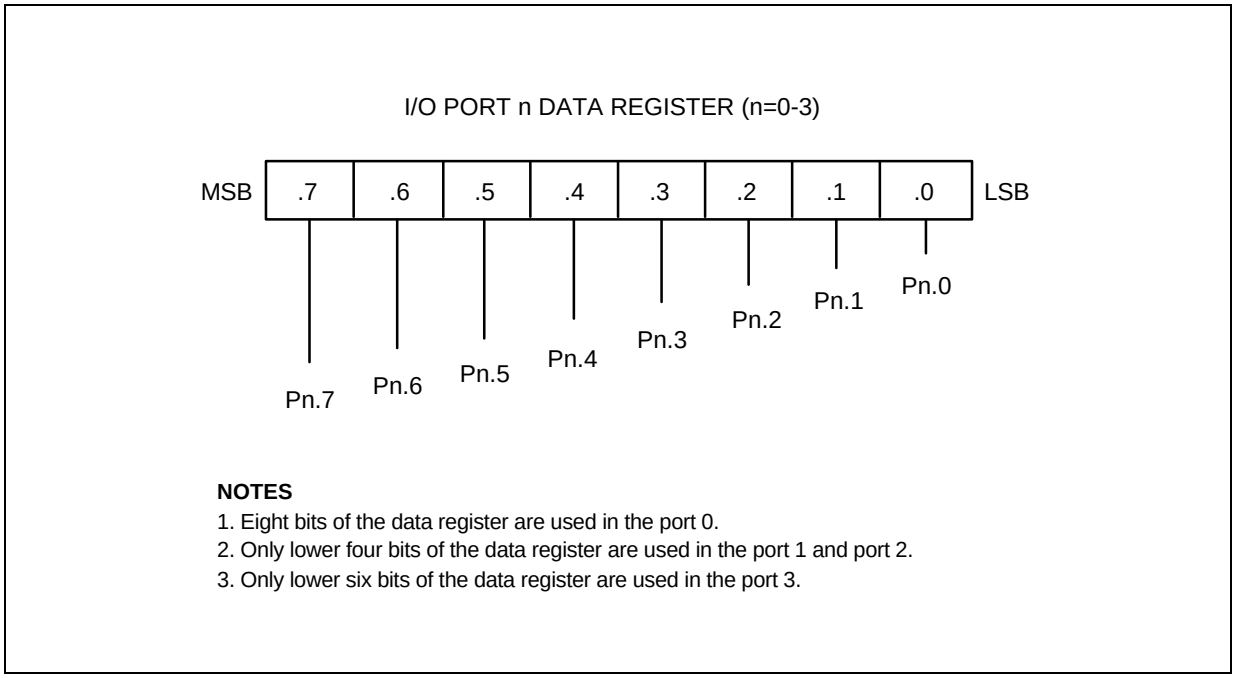
PORT DATA REGISTERS

Table 9-2 gives you an overview of the port data register names, locations, and addressing characteristics. Data registers for ports 0–3 have the structure shown in Figure 9-1.

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Hex	R/W
Port 0 data register	P0	E0H	R/W
Port 1 data register	P1	E1H	R/W
Port 2 data register	P2	E2H	R/W
Port 3 data register	P3	E3H	R/W

NOTE: A reset operation clears the P0–P3 data register to "00H".



NOTES
1. Eight bits of the data register are used in the port 0.
2. Only lower four bits of the data register are used in the port 1 and port 2.
3. Only lower six bits of the data register are used in the port 3.

Figure 9-1. Port Data Register Format

PORT 0

Port 0 is a bit-programmable, general-purpose, I/O ports. You can select normal input or push-pull, open drain output mode. In addition, you can configure a pull-up resistor to individual pins using control register settings. It is designed for high-current functions such as LED direct drive.

You access port 0 directly by writing or reading the corresponding port data register, P0 (E0H). A reset clears the port control register, P0CON, to "00H" configuring port 0 pins as normal inputs.

Two addition resisters are used to control Port 0: P0PUR (E7H) and P0PNE (E8H). By setting bits in the Port 0 open-drain enable register P0PNE, you can configure specific pin as a open-drain output.

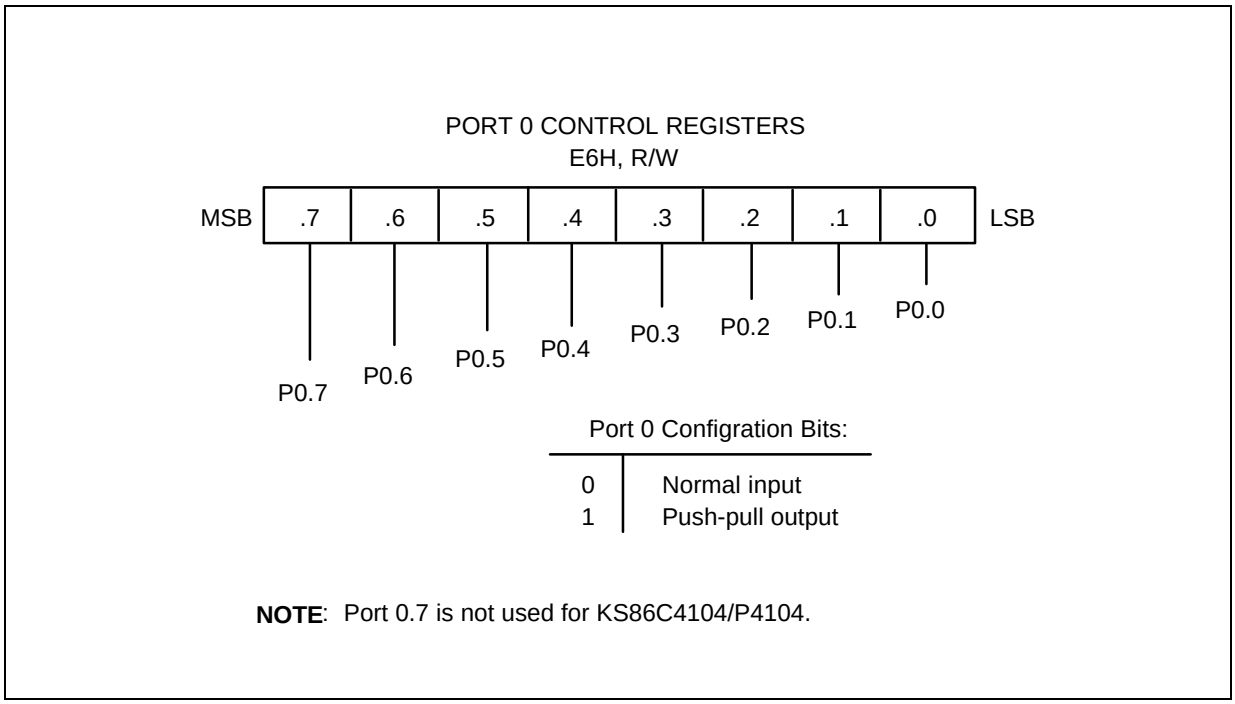


Figure 9-2. Port 0 Control Registers (P0CON)

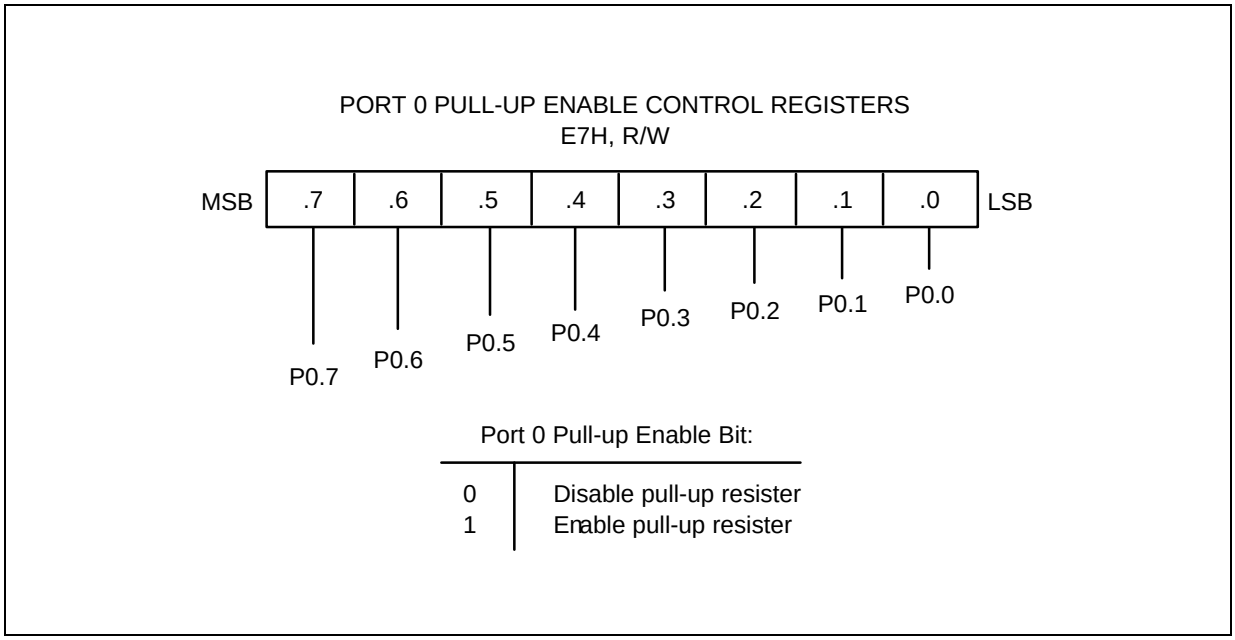
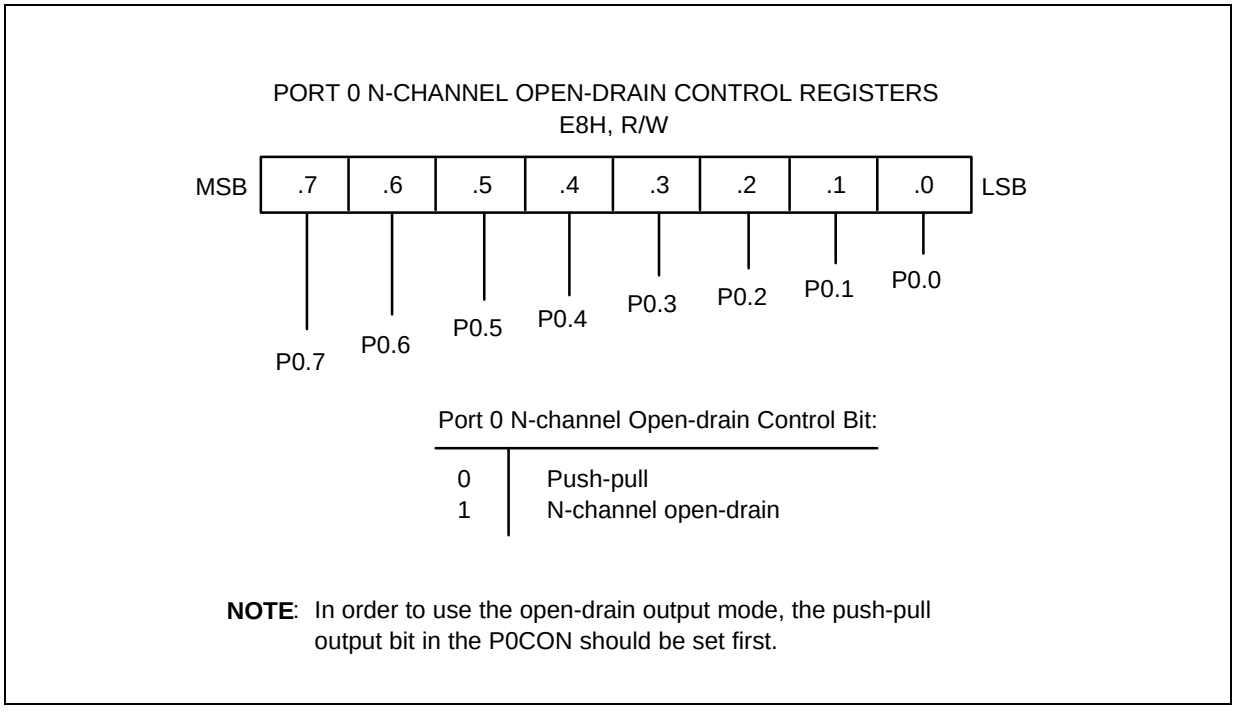


Figure 9-3. Port 0 Pull-up Enable Control Registers (P0PUR)



NOTE: In order to use the open-drain output mode, the push-pull output bit in the P0CON should be set first.

Figure 9-4. Port 0 Control Registers (P0PNE)

PORT 1

Port 1, is a 4-bit I/O port with individually configurable pins. It can be used for general I/O port (Schmitt trigger input mode or push-pull output mode). You can also use port1 as special input (ZCD) or output (BUZ, T0, CLO). In addition, you can configure a pull-up resistor to individual pin using control register settings.

In normal operating mode, a reset clears P1CON to "00H", configuring P1.0–P1.3 as normal Schmitt trigger inputs, but you can also configure P1CON to "0FFH" for alternative functions.

You address port 1 bits directly by writing or reading the port 1 data register, P1 (E1H). The port 1 control register, P1CON is located at addresses E9H.

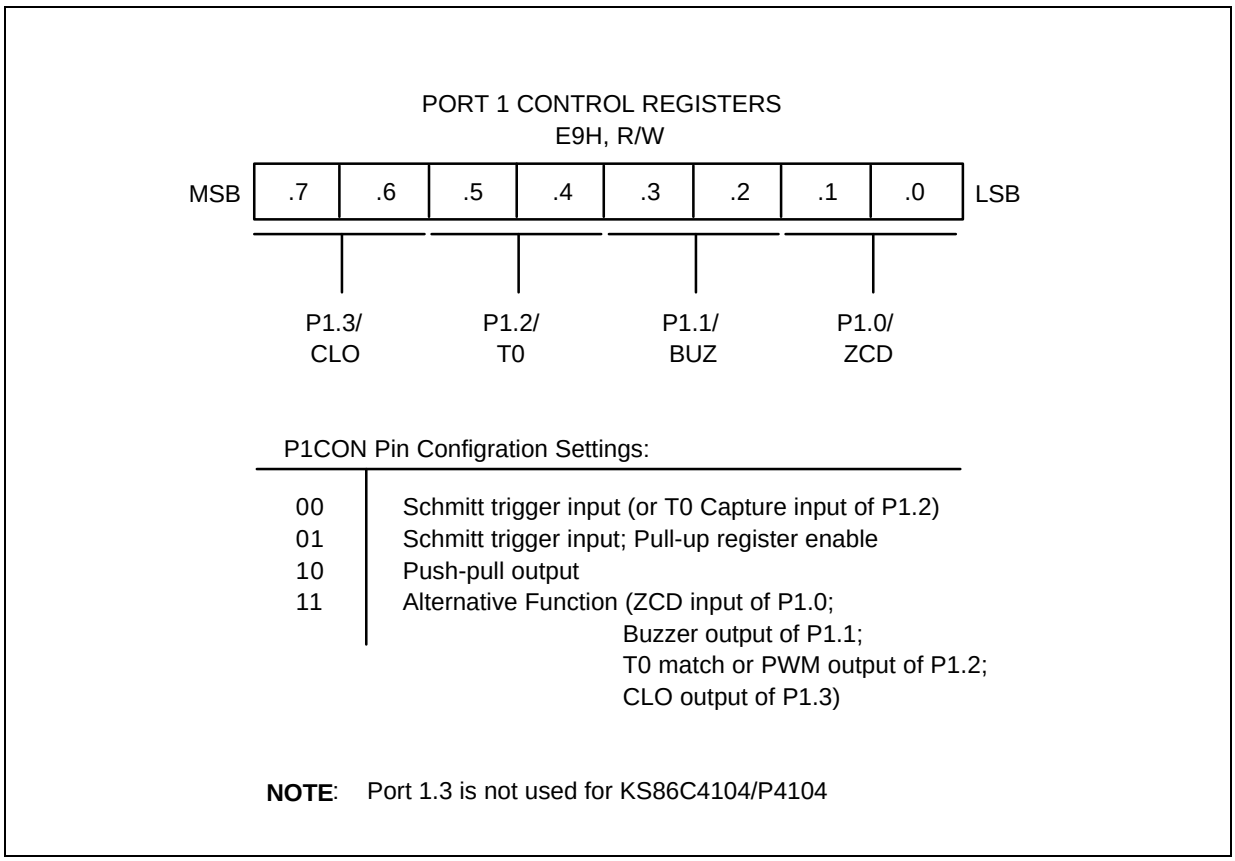


Figure 9-5. Port 1 Control Registers (P1CON)

PORT 2

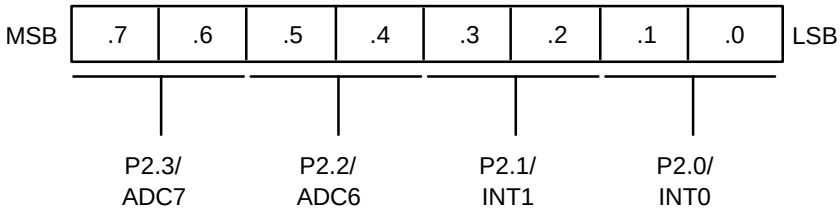
Port 2 is a 4-bit I/O port with individually configurable pins. It can be used for general I/O port (Schmitt trigger input mode or push-pull output mode or N-channel open-drain output mode). You can also use port 2 pins as external interrupt (INT0–INT1) or A/D inputs. In addition, you can configure a pull-up resistor to individual pins using control register settings.

In normal operating mode, a reset clears P2CON to "00H", configuring P2.0–P2.3 as normal Schmitt trigger inputs.

You address port 2 bits directly by writing or reading the port 2 data register, P2 (E2H). The port 2 control register, P2CON is located at addresses EAH.

Two additional registers are used to control Port 2: P2DPUR (EBH) and P2PND (ECH). By setting port 2 open-drain and pull-up resistor enable register, P2DPUR, you can configure specific pins as open-drain or push-pull output. The application program polls the port 2 interrupt pending register, P2PND, to detect interrupt requests. When an interrupt request is acknowledged, the corresponding pending bit must be cleared by the interrupt service routine.

PORT 2 CONTROL REGISTERS
EAH, R/W,



P2CON. 7-4 Pin Configration Settings:

00	Schmitt trigger input
01	Schmitt trigger input
10	Push-pull output
11	A/D converter input(ADC7,ADC6); Schmitt trigger input off

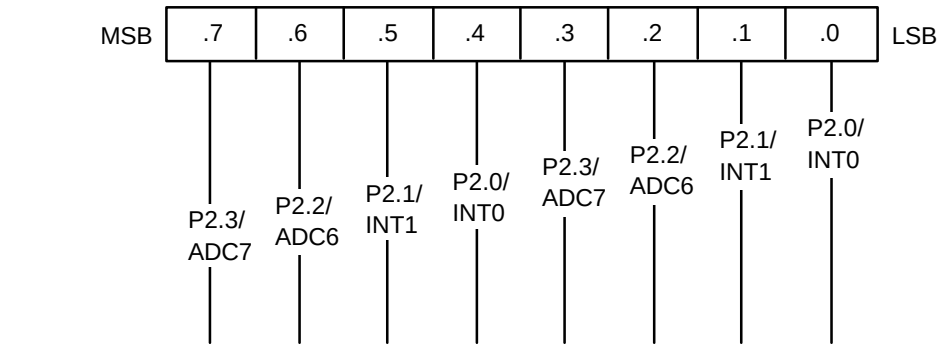
P2CON.3-0 Pin Configration Settings:

00	Schmitt trigger input; Interrupt disable
01	Schmitt trigger input; Interrupt on falling edge
10	Push-pull output
11	Schmitt trigger input; interrupt on rising edge

NOTE: Port 2.1-3 are not used for KS86C4104/P4104.

Figure 9-6. Port 2 Control Registers (P2CON)

PORT 2 OPEN-DRAIN ENABLE & PULL-UP RESISTOR ENABLE REGISTER
EBH, R/W



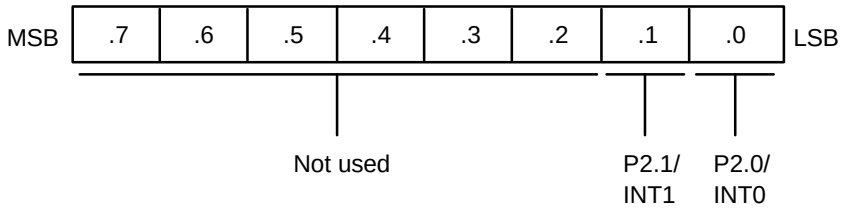
Open-drain Enable Bit	
0	Configure P2.n as a push-pull
1	Configure P2.n as a n-channel open-drain

Pull-up resistor Enable Bit	
0	Disable pull-up
1	Enable pull-up

NOTE: In order to use the open-drain output mode, the push-pull output bit in the P2CON should be set first.

Figure 9-7. Port 2 Open-Drain and Pull-Up Resistor Enable Register (P2DPUR)

PORT 2 INTERRUPT PENDING REGISTERS
ECH, R/W



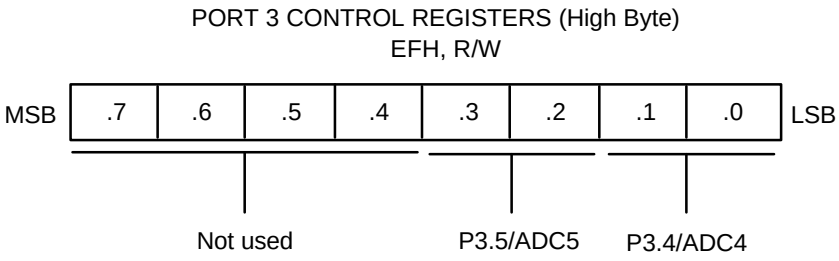
P2PND Interrupt Pending Bit	
0	No interrupt pending (when read)
0	*Clear this pending bit (when write)*
1	Interrupt is pending (when read)

Figure 9-8. Port 2 Interrupt Pending Register (P2PND)

PORT 3

Port 3 is a 6-bit I/O port with individually configurable pins. It can be used for general I/O port. You can also use port 3 pins as A/D I/O inputs. In addition, you can configure a pull-up register to individual pins using control register settings.

In normal operating mode, reset clears P3CONH and P3CONL to "00H", configures P3.0–P3.5 schmitt trigger input mode. Using the P3CONH and P3CONL registers (EEH and EFH respectively), you can alternatively configure the port 3 pins as push-pull outputs, or as A/D converter input pins.



P3CONH Configration Bits

00	Schmitt trigger input
01	Schmitt trigger input, pull-up resistor enabled
10	Push-pull output
11	A/D converter input (ADC5/ADC4); Schmitt trigger input off

NOTE: Port 3.5 is not used for KS86C4104/P4104.

Figure 9-9. Port 3 High-Byte Control Registers (P3CONH)

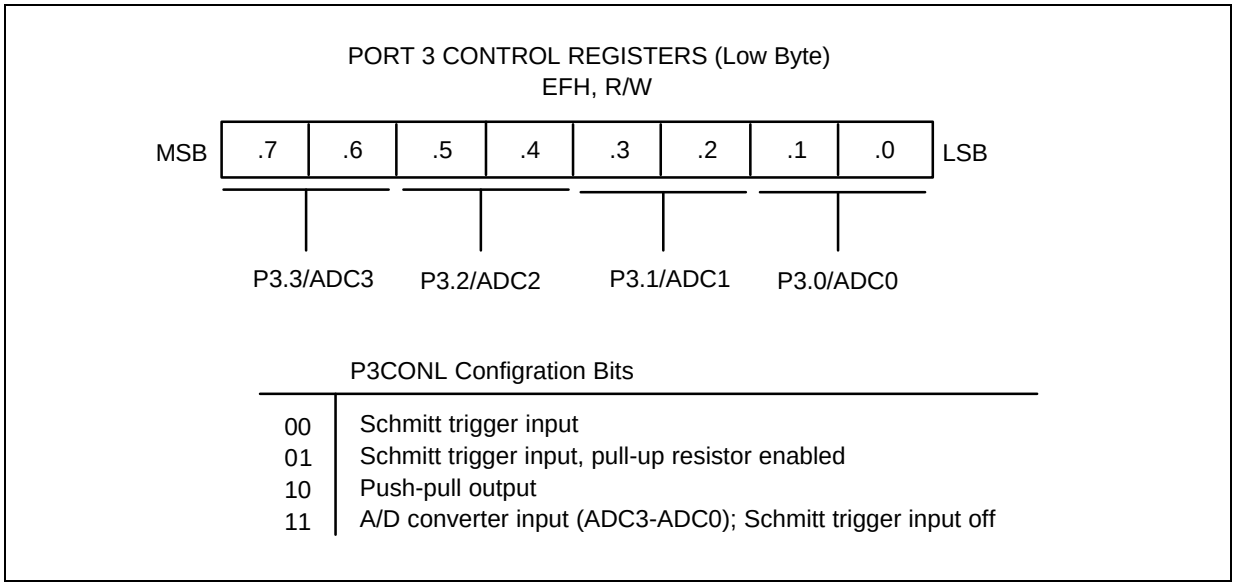


Figure 9-10. Port 3 Low-Byte Control Registers (P3CONL)

☞ PROGRAMMING TIP – Configuring I/O Port Pins to Specification

The following sample program shows how to configure the KS86C4004 I/O ports to specification

Program comments show the effect of the settings:

```
•
•
•

LD      P0CON,#0FFH         ; push-pull output (LED direct drive → Low active)
LD      P0PUR,#00H          ; disable pull-up resistor
LD      P0PNE,#00H          ; disable open-drain
LD      P1CON,#1010101011B  ; P1.0 ZCD input enable/P1.1-3 push-pull
LD      ZCMOD,#00000010B    ; enable falling edge interrupt
LD      T1DATA,#81H
LD      T1CON,#00001100B    ; ZCD clear enable (fosc/512)
                            ; timer 1 interrupt disable
LD      P2CON,#11000110B    ; P2.3 A/D input mode/P2.2 input mode
                            ; P2.1 falling edge interrupt
LD      P2DPUR,#00010110B   ; P2.0 open-drain output mode
                            ; P2.2 pull-up enable
                            ; P2.1 pull-up enable
LD      P2PND,#00H          ; no interrupt pending
LD      P3CONH,#0FH         ; AD input mode (ADC4,ADC5)
LD      P3CONL,#0AAH        ; P3.0-3 push-pull output mode
•
•
•
```

10 BASIC TIMER and TIMERS

MODULE OVERVIEW

The KS86C4004/C4104 has three default timers: an 8-bit *basic timer,* one 8-bit general-purpose timer/counter, called *timer 0,* and one 8-bit timer/counter for the zero-crossing detection circuit called timer 1.

Basic Timer (BT)

You can use the basic timer (BT) in two different ways:

— As a watchdog timer to provide an automatic reset mechanism in the event of a system malfunction.
— To signal the end of the required oscillation stabilization interval after a reset or a Stop mode release.

The functional components of the basic timer block are:

— Clock frequency divider (f_{OSC} divided by 4096, 1024, or 128) with multiplexer
— 8-bit basic timer counter, BTCNT (DDH, read-only)
— Basic timer control register, BTCON (DCH, read/write)

Timer 0

Timer 0 has three operating modes, one of which you select by an appropriate T0CON setting:

— Interval timer mode
— Capture input mode
— 10-bit PWM output mode

Timer 0 has the following functional components:

— Clock frequency divider (f_{OSC} divided by 4096, 256, 8, or 1) with multiplexer
— 10(8)-bit counter (T0CNT + T0EXCNT), 8-bit comparator, and 10(8)-bit data register (T0DATA + PWMEX)
— I/O pin (P1.2, T0 match or PWM) for timer 0 match/PWM output or capture input
— Timer 0 overflow interrupt (T0OVF) and match interrupt (T0INT) generation
— Timer 0 control registers, T0CONH and T0CONL (D2H and D3H respectively)

Timer 1

Timer 1 has one operating mode, interval timer mode. You can clear the timer 1 counter by appropriate setting of T1CON register. If T1CON.3 is set to “1”, T1CNT is cleared by the ZCD edge detection.

Timer 1 has the following components:

— Clock frequency divider (f_{OSC} divided by 512, 256, 128, or 64)
— 8-bit counter (T1CNT), 8-bit compare register, and a 8-bit data register (T1DATA)
— Timer 1 control register, T1CON

BASIC TIMER (BT)

BASIC TIMER CONTROL REGISTER (BTCON)

The basic timer control register, BTCON, is used to select the input clock frequency, to clear the basic timer counter and frequency dividers, and to enable or disable the watchdog timer function.

A reset clears BTCON to "00H". This enables the watchdog function and selects a basic timer clock frequency of $f_{OSC}/4096$. To disable the watchdog function, you must write the signature code "1010B" to the basic timer register control bits BTCON.7–BTCON.4.

The 8-bit basic timer counter, BTCNT, can be cleared during normal operation by writing a "1" to BTCON.1. To clear the frequency dividers for both the basic timer input clock and the timer 0 clock, you write a "1" to BTCON.0.

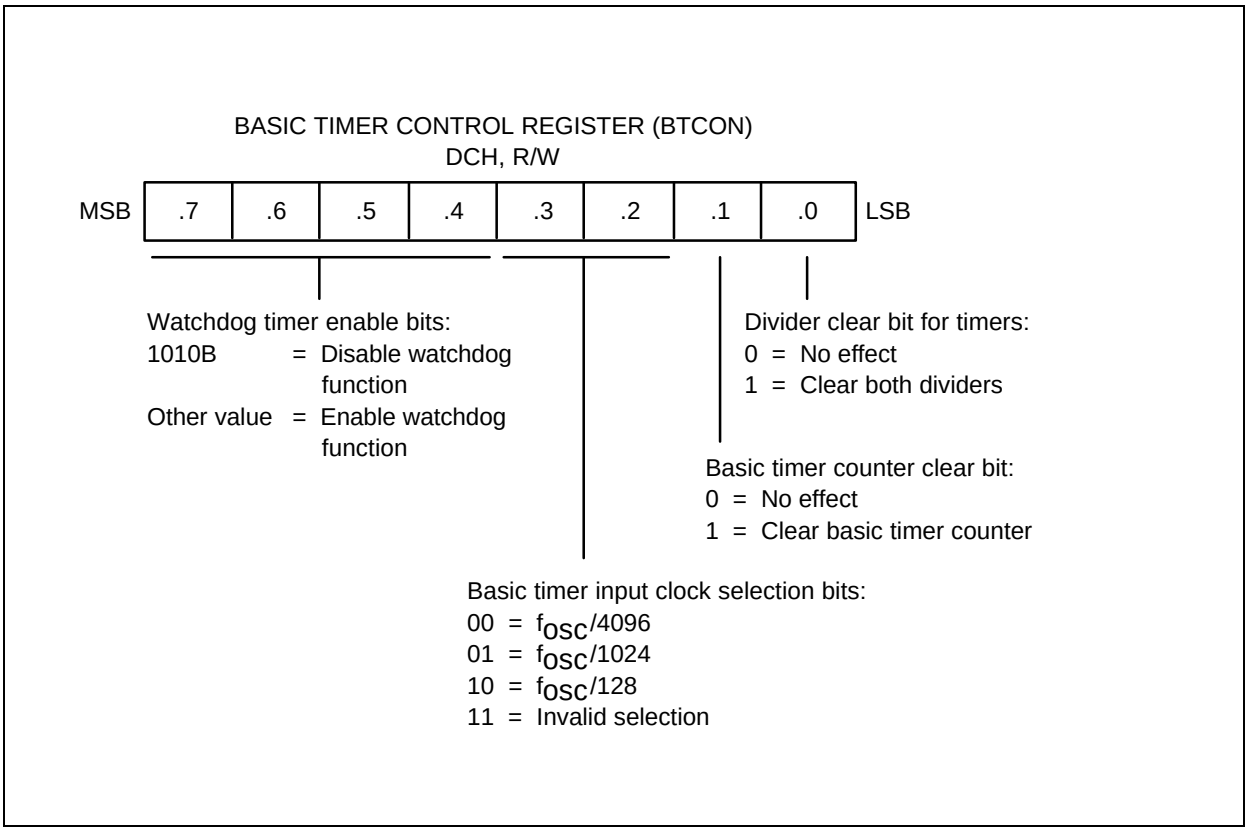


Figure 10-1. Basic Timer Control Register (BTCON)

BASIC TIMER FUNCTION DESCRIPTION

Watchdog Timer Function

You can program the basic timer overflow signal (BTOVF) to generate a reset by setting BTCON.7–BTCON.4 to any value other than "1010B" (The "1010B" value disables the watchdog function). A reset clears BTCON to "00H", automatically enabling the watchdog timer function. A reset also selects the CPU clock (as determined by the current CCON register setting) divided by 4096 as the BT clock.

A reset whenever a basic timer counter overflow occurs. During normal operation, the application program must prevent the overflow, and the accompanying reset operation, from occurring. To do this, the BTCNT value must be cleared (by writing a "1" to BTCON.1) at regular intervals.

If a system malfunction occurs due to circuit noise or some other error condition, the BT counter clear operation will not be executed and a basic timer overflow will occur, initiating a reset. In other words, during normal operation, the basic timer overflow loop (a bit 7 overflow of the 8-bit basic timer counter, BTCNT) is always broken by a BTCNT clear instruction. If a malfunction does occur, a reset is triggered automatically.

Oscillation Stabilization Interval Timer Function

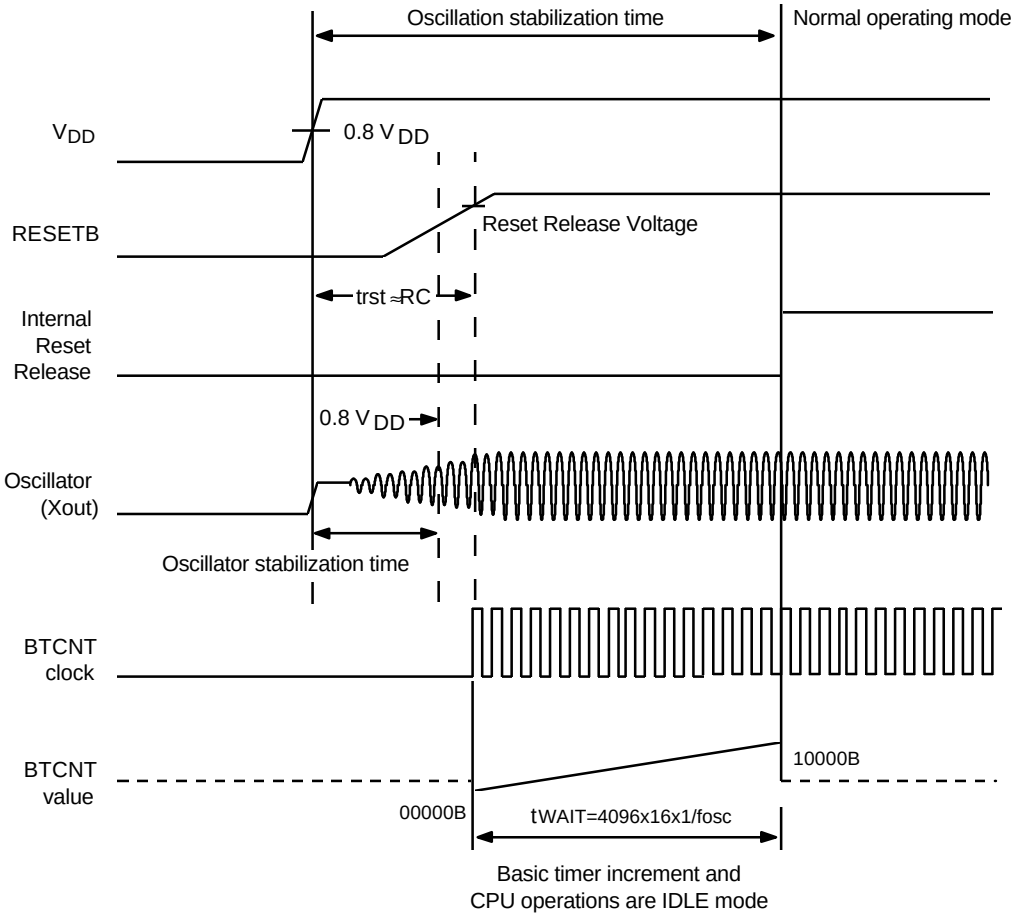
You can also use the basic timer to program a specific oscillation stabilization interval following a reset or when Stop mode has been released by an external interrupt.

In Stop mode, whenever a reset or an external interrupt occurs, the oscillator starts. The BTCNT value then starts increasing at the rate of $f_{OSC}/4096$ (for reset), or at the rate of the preset clock source (for an external interrupt). When BTCNT.4 is set, a signal is generated to indicate that the stabilization interval has elapsed and to gate the clock signal off to the CPU so that it can resume normal operation.

In summary, the following events occur when Stop mode is released:

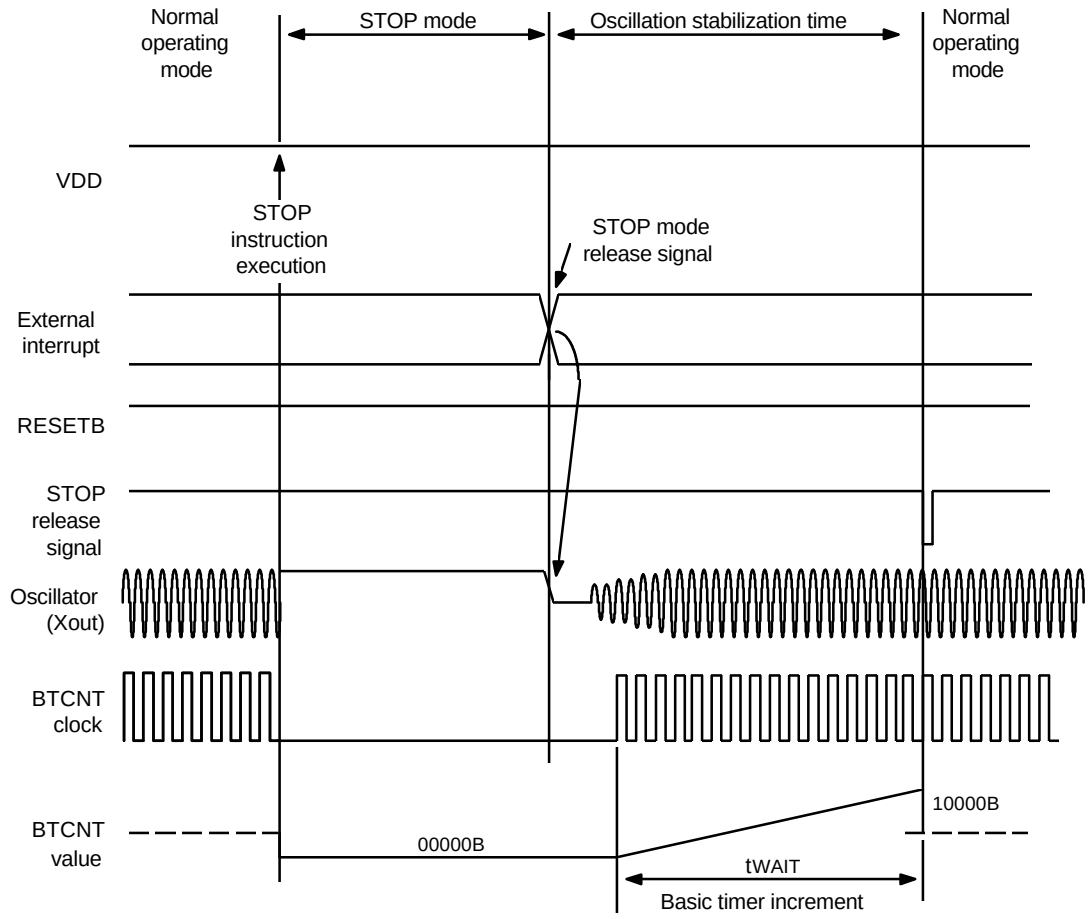
1. During Stop mode, a power-on reset or an external interrupt occurs to trigger the Stop mode release and oscillation starts.
2. If a power-on reset occurred, the basic timer counter will increase at the rate of $f_{OSC}/4096$. If an external interrupt is used to release Stop mode, the BTCNT value increases at the rate of the preset clock source.
3. Clock oscillation stabilization interval begins and continues until bit 4 of the basic timer counter is set.
4. When a BTCNT.4 is set, normal CPU operation resumes.

Figure 10-2 and 10-3 shows the oscillation stabilization time on RESET and STOP mode release



NOTE: Duration of the oscillator stabilization wait time, tWAIT, when it is released by a Power-on-reset is 4096x16/fosc.
trst ≈RC (R is external resister and C is on chip capacitor)

Figure 10-2. Oscillation Stabilization Time on RESET



NOTE: Duration of the oscillator stabilization wait time, tWAIT, it is released by an interrupt is determined by the setting in basic timer control register, BTCON.

BTCON.3	BTCON.2	tWAIT	tWAIT (When fosc is 10MHz)
0	0	4096 x 16 / fosc	6.55 ms
0	1	1024 x 16 / fosc	1.64 ms
1	0	128 x 16 / fosc	0.2 ms
1	1	Invalid setting	–

Figure 10-3. Oscillation Stabilization Time on STOP Mode Release

☞ Programming Tip — Configuring the Basic Timer

This example shows how to configure the basic timer to sample specification

```
;----------<< Interrupt vector address>>

            ORG     0000H
            VECTOR  00H, COMMON_INT         ;   IRQ0/Interrupt vector address

;----------<<Initialize system and peripherals>>

            ORG         0100H               ;   Reset start address
RESET       DI                              ;   disable interrupt
            LD          BTCON,#10100010B    ;   disable watchdog function
                                                ;       clock source: fosc/4096 (104 ms overflow at 10
MHz)                    LD                  CLKCON,#00011000B       ;       CPU clock source
select (non-divided)
            LD          SP,#0C0H            ;   KS86C4004 Stack pointer initial
                        •
                        •
                        •
                        •
             EI                             ;   enable interrupt

;----------<< Main loop >>

MAIN
                        •
                        •
                        •
            LD          BTCON,#02H          ;   enable watchdog function
                                                ;       basic Timer Counter (BTCNT) clear
                        •
                        •
                        •
            JP          T,MAIN              ;   for main loop
                        •
                        •
                        •
```

TIMER 0

TIMER 0 CONTROL REGISTERS (T0CONH and T0CONL)

The timer 0 control register low byte, T0CONL, is used to select the timer 0 operating mode (interval timer, capture mode, or PWM mode) and input clock frequency, to clear the timer 0 counter, and to enable the T0 overflow interrupt and T0 match/capture interrupt. It also contains a pending bit for T0 match/capture interrupts.

Timer 0 control register high byte, T0CONH, contains a pending bit for T0 overflow interrupt. Only one bit in T0CONH register is used, T0CONH.0.

A reset clears T0CONL to "00H". This sets timer 0 to normal interval timer mode, selects an input clock frequency of f_{OSC} /4096, and disables the T0 overflow interrupt and match/capture interrupts. The T0 counter can be cleared at any time during normal operation by writing a "1" to T0CONL.3.

The T0 overflow interrupt, T0OVF, is IRQ0 with vector 00H. When a T0 overflow interrupt occurs and is serviced by the CPU, the pending condition is cleared manually set by writing “0” to T0CONH.0. To enable the T0 match/capture interrupt (T0INT, IRQ0, vector 00H), you must set T0CONL.1 to "1". The interrupt service routine must clear the pending condition by writing a "0" to the T0 interrupt pending bit, T0CONL.0.

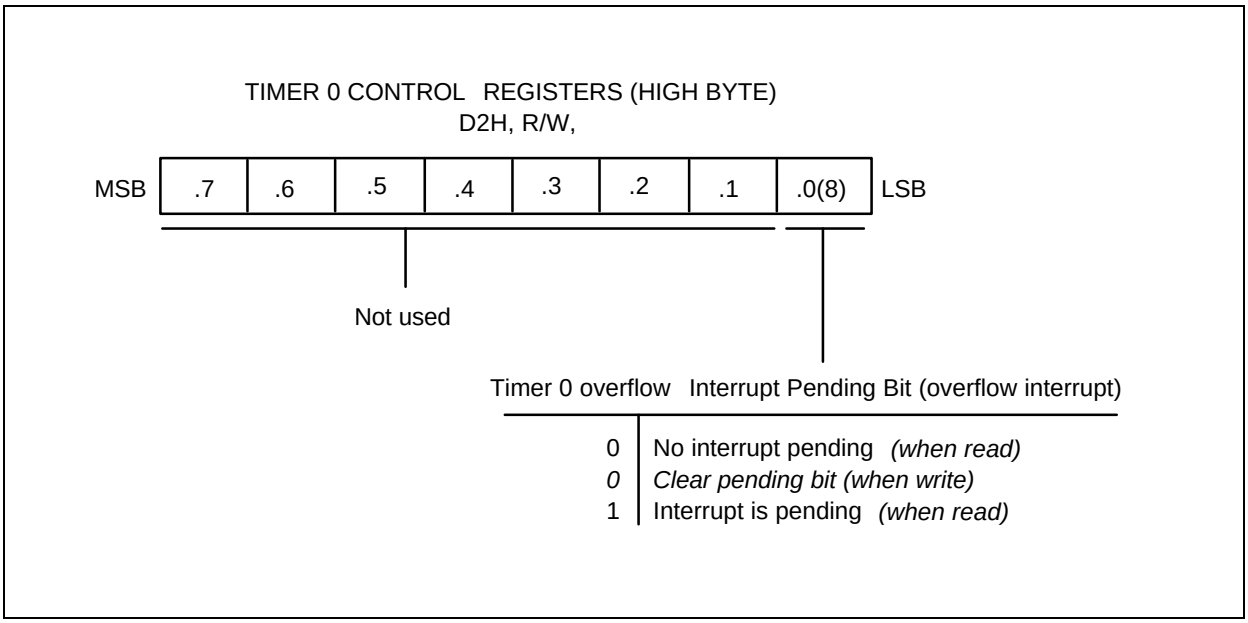


Figure 10-4. Timer 0 Control Registers (T0CONH)

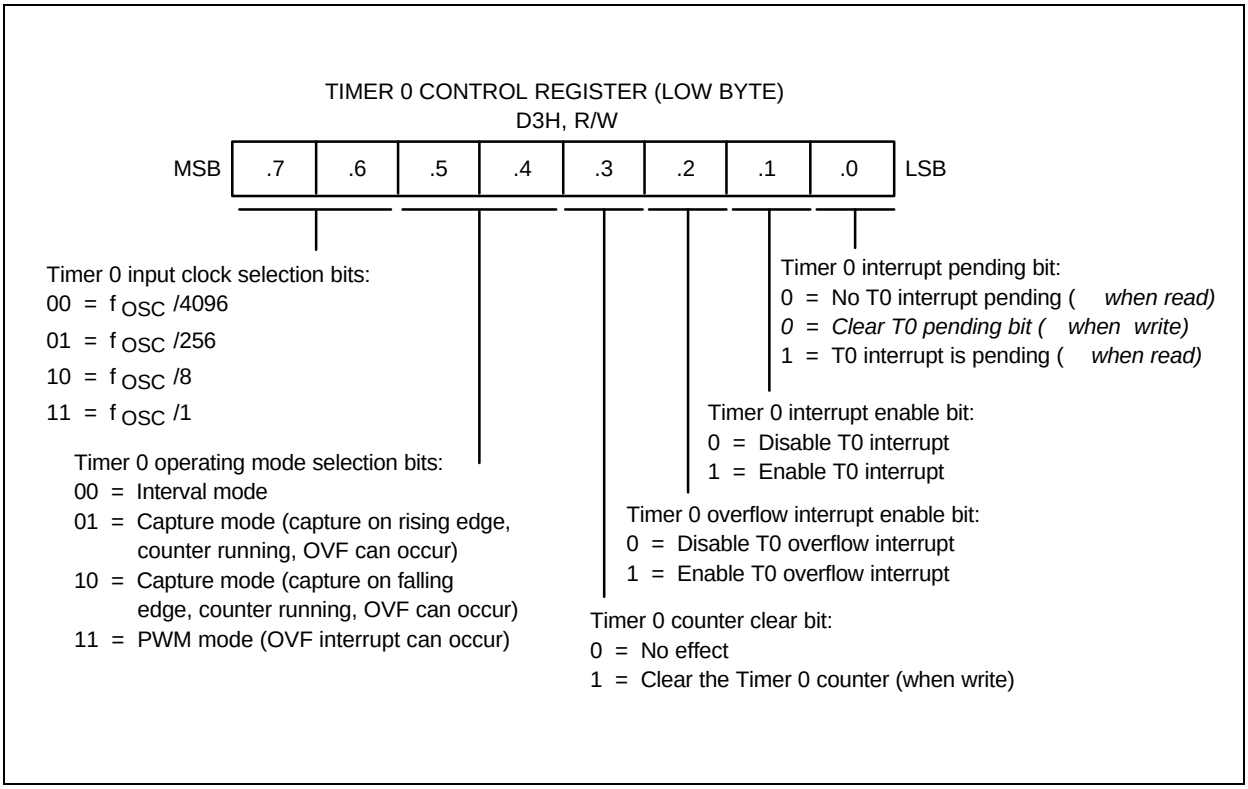


Figure 10-5. Timer 0 Control Registers (T0CONL)

TIMER 0 FUNCTION DESCRIPTION

Timer 0 Interrupts (IRQ0, Vectors 00H)

The Timer 0 module can generate two interrupts; the timer 0 overflow interrupt (T0OVF), and the timer 0 match/capture interrupt (T0INT). T0OVF is interrupt level IRQ0, vector 00H; T0INT is also level IRQ0, vector 00H. The T0OVF interrupt pending condition is cleared by setting the T0CONH.0 pending bit to “0”. The T0INT pending condition must be cleared by software by writing a "0" to the T0CONL.0 pending bit.

INTERVAL TIMER MODE

In interval timer mode, a match signal is generated when the counter value is identical to the value written to the Timer 0 reference data register, T0DATA. The match signal generates a Timer 0 match interrupt (T0INT, vector 00H) and then clears the counter. If, for example, you write the value "10H" to T0DATA, the counter will increment until it reaches "10H". At this point, the Timer 0 interrupt request is generated, the counter value is reset and counting resumes. With each match, the level of the signal at the Timer 0 output pin is inverted.

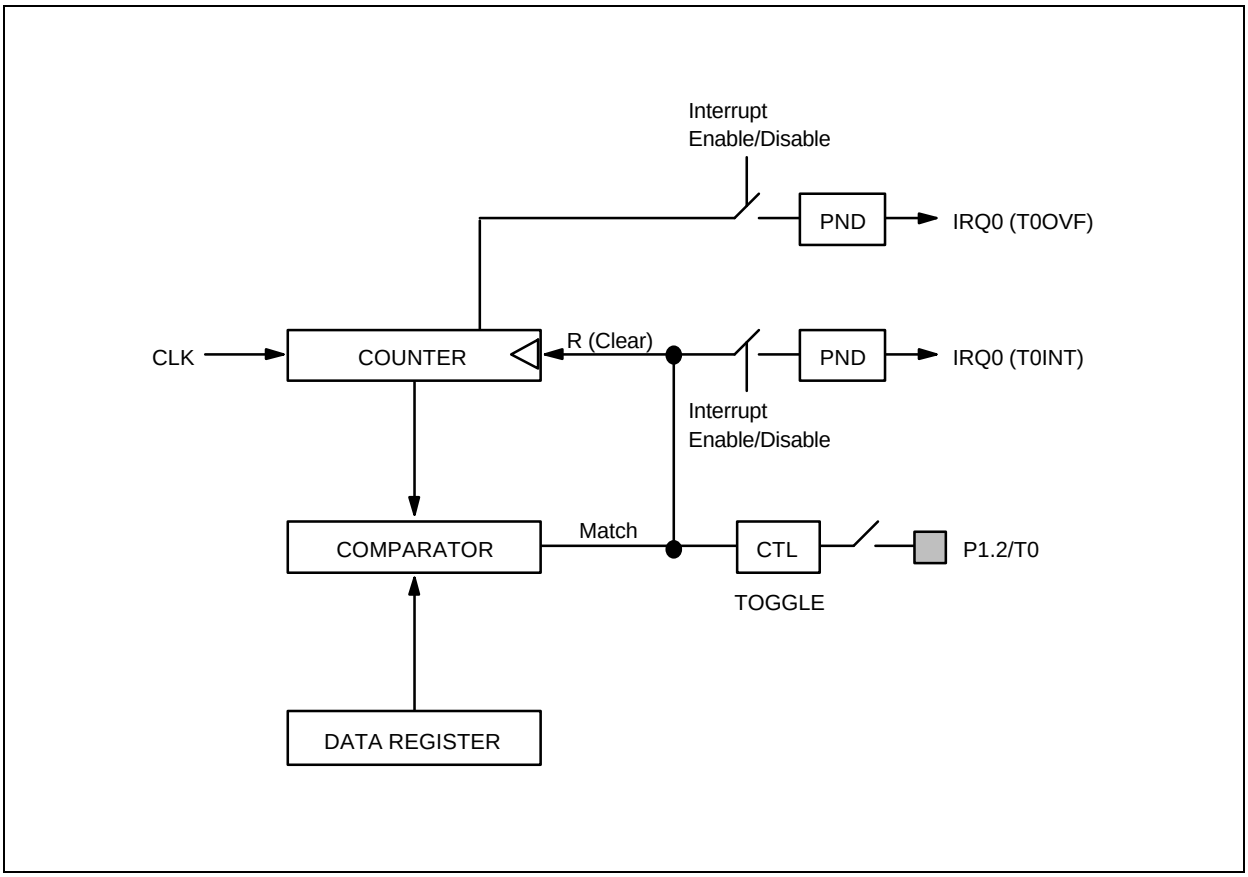
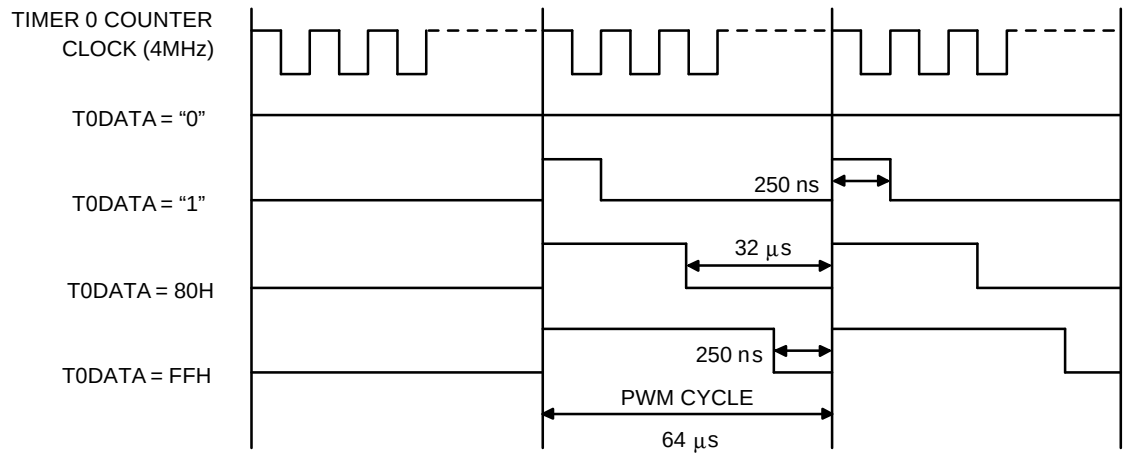


Figure 10-6. Simplified Timer 0 Function Diagram (Interval Timer Mode)

PULSE WIDTH MODULATION MODE

Pulse width modulation (PWM) mode lets you program the width (duration) of the pulse that is output at the T0 pin. As in interval timer mode, a match signal is generated when the counter value is identical to the value written to the T0 data register (T0DATA). In PWM mode, however, the match signal does not clear the counter (it runs continuously, overflowing at "FFH", and continues incrementing from "00H").

Although it is possible to use the match signal to generate a T0INT interrupt, an interrupt is typically not used in PWM-type applications. Instead, the pulse at the T0 pin is held to High level as long as the data register (T0DATA) value is *greater than* the counter (T0CNT) value for 8-bit PWM operation. For 2-bit extension control logic operation (10-bit PWM) the pulse is held to High level when the data value is *equal to* the counter value. One frame width is equal to $t_{CLK} \times 1024$. (See Figure 10-9.)



NOTES

1. A system clock frequency of 4MHz is assumed.
2. The input clock of timer 0 count (T0CNT) is assumed in non-divided (f_{OSC}/1).

Figure 10-7. Simplified Timer 0 Function Diagram (PWM Mode)

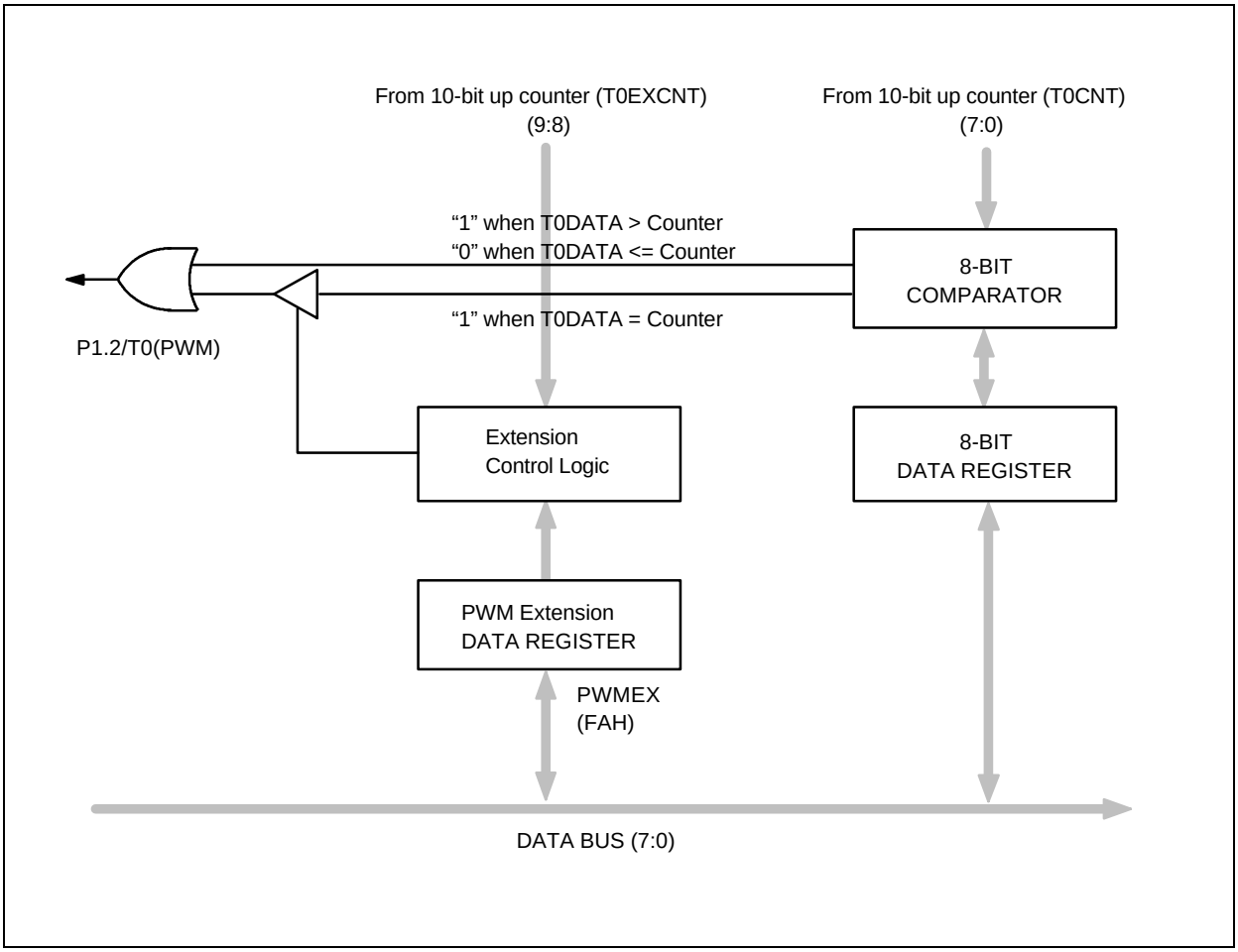


Figure 10-8. PWM Block Function Diagram

Table 10-1. PWM Output "Stretch" Values for Extension Register PWMEX

PWMEX	Cycle Number That is "Stretched"
00H	Not Stretched
01H	2
02H	1, 3
03H	1, 2, 3

NOTE: Bits 0 and 1 of the PWM extension register PWMEX are used only.

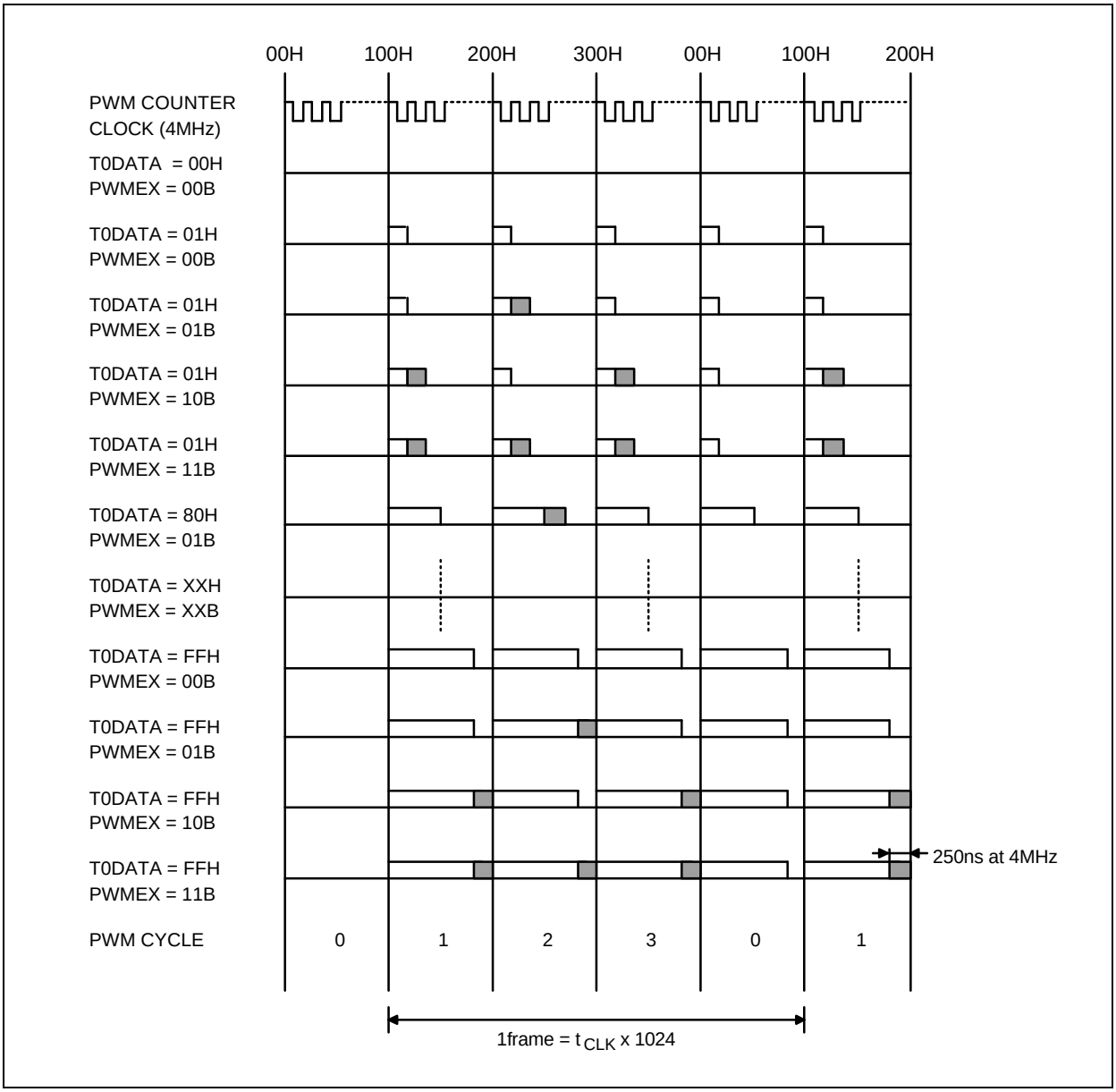


Figure 10-9. 10-Bit PWM Waveforms with Various T0DATA and PWMEX

CAPTURE MODE

In capture mode, a signal edge that is detected at the T0 pin opens a gate and loads the current counter value into the T0 data register. Rising edges or falling edges can be selected to trigger this operation. Both kinds of T0 interrupts can be used in capture mode: T0OVF is generated when a counter overflow occurs, and T0INT is generated when the counter value is loaded into the data register. By reading the captured data value in T0DATA, and assuming a specific value for t_{CLK}, you can determine the pulse width (duration) of the signal being input at the T0 pin. (See Figure 10-10.)

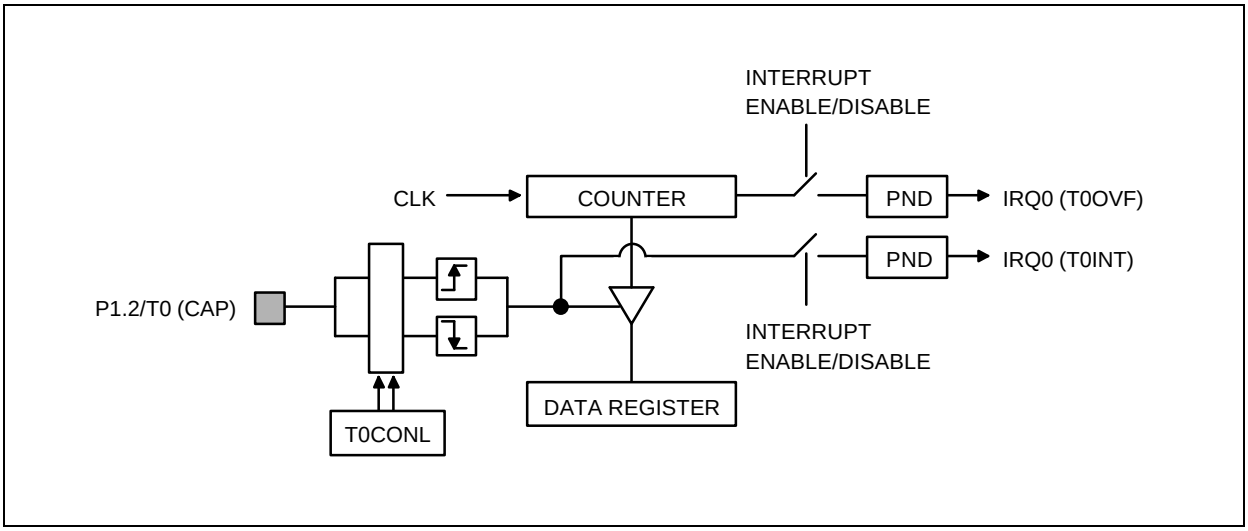
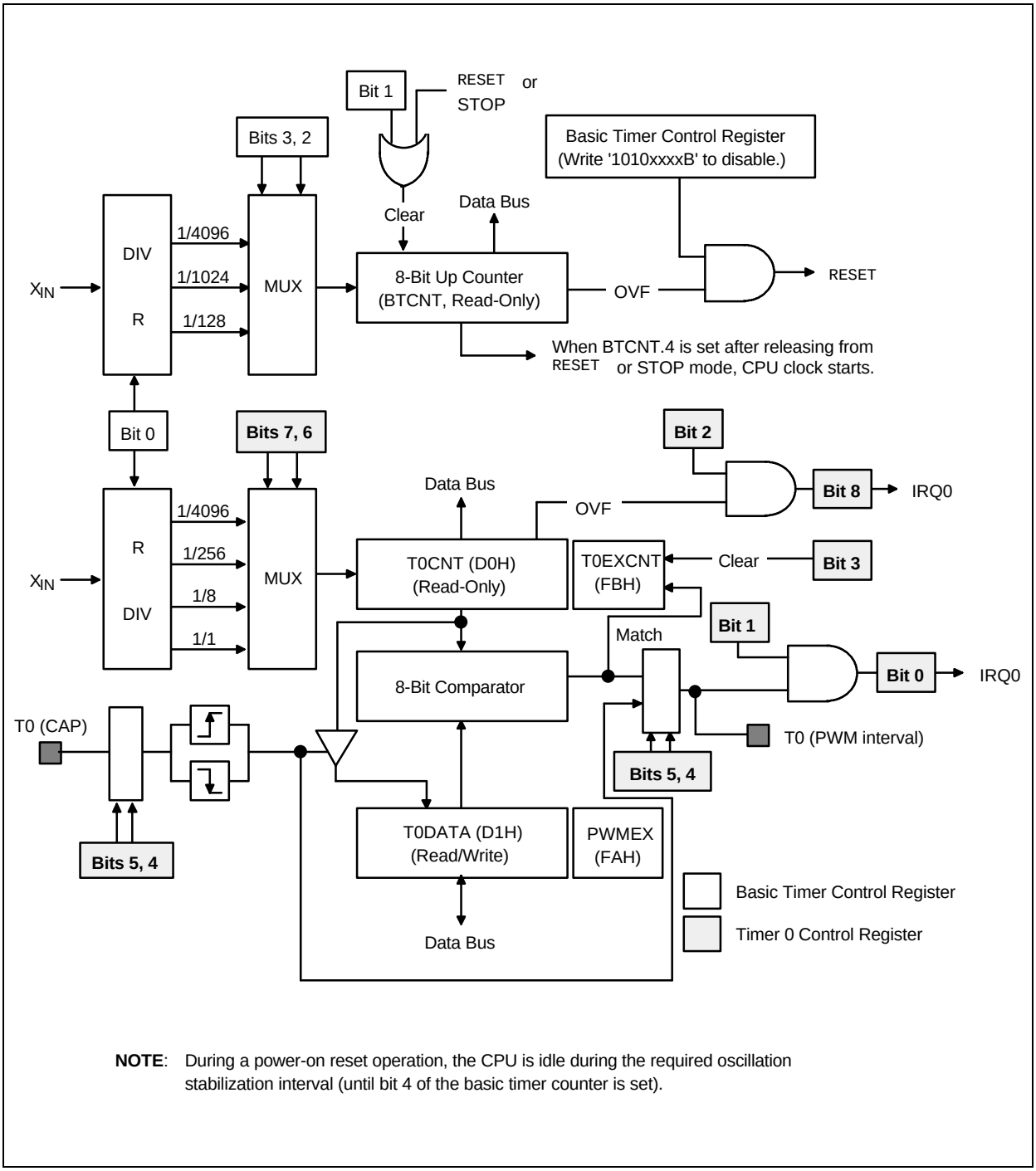


Figure 10-10. Simplified Timer 0 Function Diagram (Capture Mode)



NOTE: During a power-on reset operation, the CPU is idle during the required oscillation stabilization interval (until bit 4 of the basic timer counter is set).

Figure 10-11. Basic Timer and Timer 0 Block Diagram

☞ PROGRAMMING TIP1 -- Configuring Timer 0 (Interval Mode)

The following sample program sets Timer 0 to interval timer mode.

```
;----------<< Interrupt vector address >>

                ORG     0000H
                VECTOR  00H, COMMON_INT         ; IRQ0/Interrupt vector address

;----------<< Initialize system and peripherals >>

                ORG     0100H                   ; Reset start address
RESET           DI                              ; disable interrupt
                LD      BTCON,#00000010B        ; enable watchdog function
                                                ; clock source: fosc/4096 (104 ms overflow at 10 MHz)
                LD      CLKCON,#00011000B       ; CPU clock source select (non-divided)
                LD      SP,#0C0H                ; KS86C4004 Stack pointer initial
                •
                •
                •
                •
                LD      T0DATA,#26H             ; 1 ms interval at 10 MHz system clock
                                                ; 100 ns x 256 x 39 = 998.4 µs

                LD      T0CONH,#00H             ; timer 0 overflow interrupt disable
                LD      T0CONL,#01001010B       ; timer 0 match interrupt enable
                                                ; clock source: fosc/256
                EI                              ; enable interrupt

;----------<< Main loop >>

MAIN            •
                •
                •
                LD      BTCON,#02H              ; enable watchdog function
                                                ; basic counter (BTCNT) clear
                JP      T,MAIN                  ; for main loop
                •
                •

;----------<< Interrupt service routine >>

COMMON_INT
                TM      T0CONL,#00000001B
                JP      NZ,TIMER0_INT
                •
                •
                •
                •
                IRET
```

```
TIMER0_INT  AND   T0CONL,#11110110B      ; Timer0 pending bit clear
            INC   TIMER_MODE
            CP    TIMER_MODE,#5
            JR    ULT,TMODE_JP
            CLR   TIMER_MODE

TMODE_JP    LD    R9,TIMER_MODE
            RCF
            RLC   R9                     ; TIMER_MODE x 2
            CLR   R8
            LDC   R10,#TBL_TMODE[RR8]
            LDC   R11,#TBL_TMODE+1[RR8]
            CALL  @RR10                  ; MULTI CALL
            •
            •
            IRET

TBL_TMODE                                ; MULTI CALL ADDRESS
            DW    DSP_7SEGMENT           ; TA_MODE = 0
            •
            •
            DW    KEY_SCAN               ; TA_MODE = 4

DSP_7SEGMENT                             ; display
            •
            •
            RET

KEY_SCAN                                 ; key scanning
            •
            RET
            •
            •
            •
            END
```

☞ PROGRAMMING TIP2 -- Configuring Timer 0 (PWM Mode)

The following sample program sets Timer 0 to 10-bit PWM mode.

```
                •
                •
                •
                ORG     0100H               ; reset start address
RESET           DI                          ; disable interrupt
                LD      BTCON,#10100010B    ; disable watchdog function
                LD      CLKCON,#00011000B   ; CPU clock source select (non-divided)
                LD      SP,#0C0H            ; KS86C4004 Stack pointer initial
                •
                •
                LD      T0CONH,#00H         ; timer0 overflow interrupt disable
                LD      P1CON,#10111010B    ; P1.2 PWM output mode
                •
                •

;-------<< MAIN LOOP >>

MAIN            •
                •
                •
                LD      BTCON,#02H          ; enable watchdog function
                                            ; basic counter (BTCNT) clear
                JP      T,MAIN              ; for main loop
                •
                •
PWM_OUT         •
                LD      T0DATA,#34          ; duty = 34/256 (High width)
                LD      PWMEX,#02H          ; total duty = 138/1024 (High width)
                LD      T0CONL,#11111000B   ; timer0 PWM mode/non-divided
                •
                •
                LD      T0DATA,#80H         ; duty = 128/256 (High width)
                LD      PWMEX,#00H          ; total duty = 512/1024 (High width)
                •
                •
```

TIMER 1

TIMER 1 CONTROL REGISTER (T1CON)

The timer 1 control register,T1CON, located at F3H operates in interval timer mode. By setting the appropriate bits in T1CON you can select the input clock frequency and enable the Timer 1 interrupt. T1CON also contains a pending bit for Timer 1 interrupt.

A reset clears T1CON to "00H". This sets timer 1 to normal interval mode and selects an input clock frequency of f_{osc}/512 and disables the Timer 1 interrupt.

You may clear the timer 1 counter by either setting T1CON.2 to “1” or enable ZCD clear signal to clear the timer 1 counter by setting T1CON.3 to “1”.

To enable Timer 1 match interrupt (IRQ0, vector 00H) you must set T1CON.1 to “1”. The interrupt service routine must clear the pending condition by writing a “0” to the Timer 1 interrupt pending bit, T1CON.0.

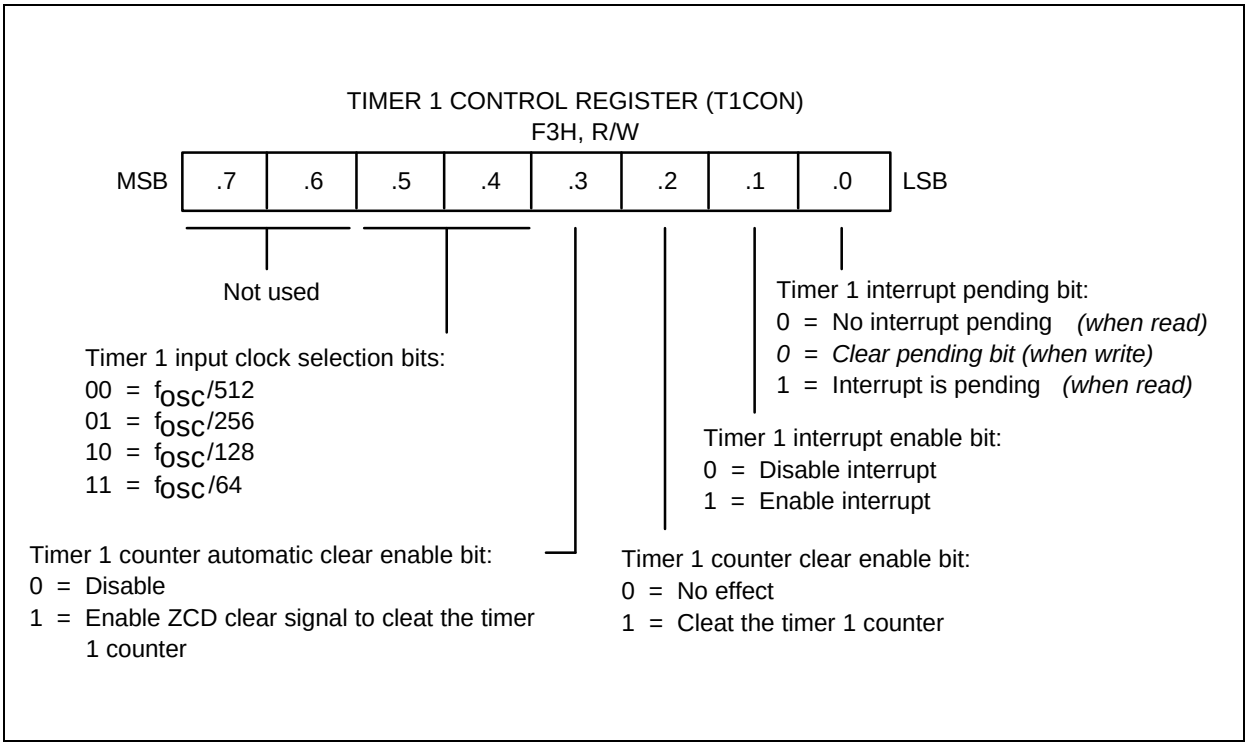
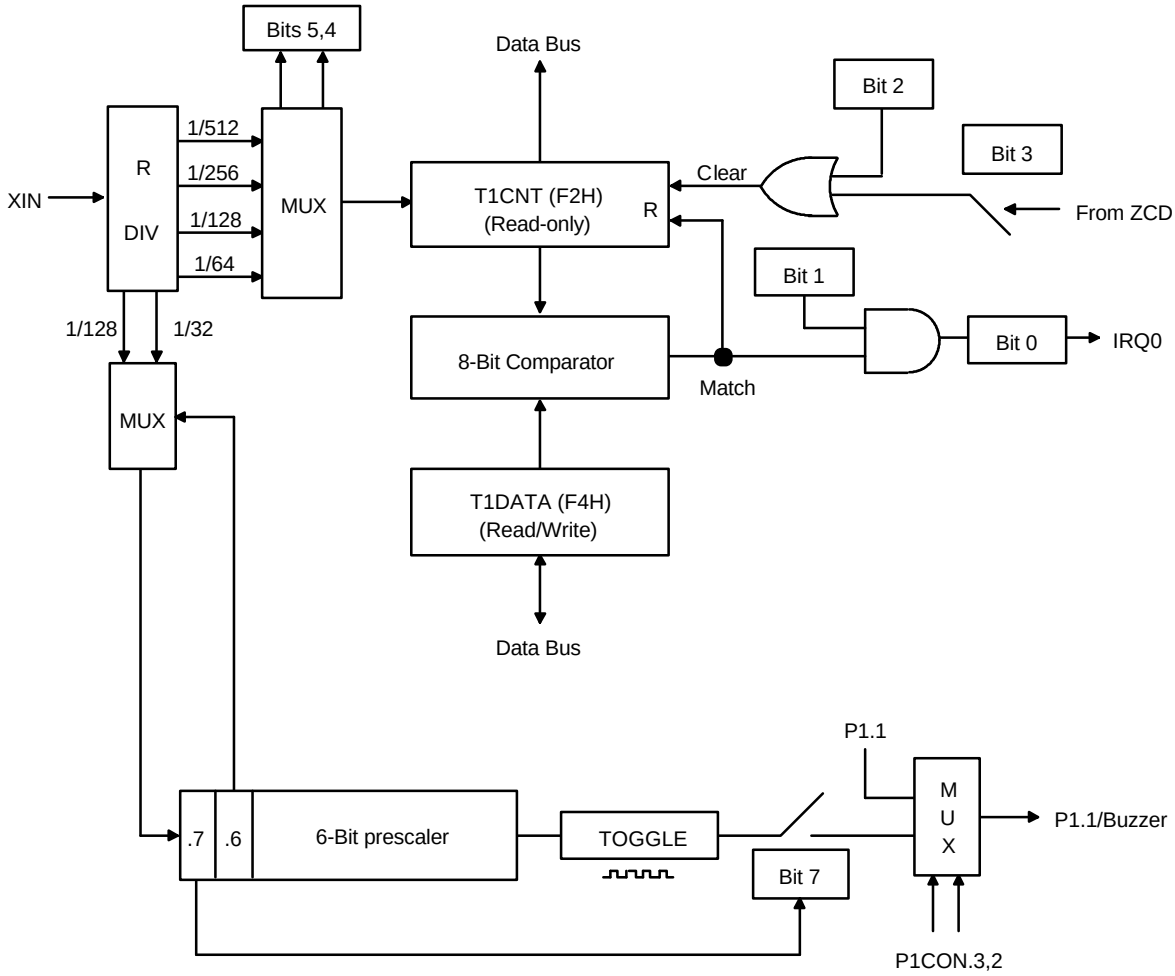


Figure 10-12. Timer 1 Control Register (T1CON)



NOTE: When P1.1/Buzzer is used as Buzzer output pin, the initial value is “0” (Low Level).
(When the bit7 of BUZPS is set to “0”, the output of P1.1 is Low.)

Figure 10-13. Timer 1 Block Diagram

BUZZER OUTPUT CONTROL REGISTER (BUZPS)

Buzzer output control register is used to select the frequency from 200 Hz to 20 KHz. And these various frequency can be used to generate the melody signal. By selecting the clock source (bit of BUZPS) and the value of prescaler, the desire frequency can be obtained. The BUZPS.7 can be used to control the buzzer output when P1.1 is set to buzzer output mode (configure P1CON).

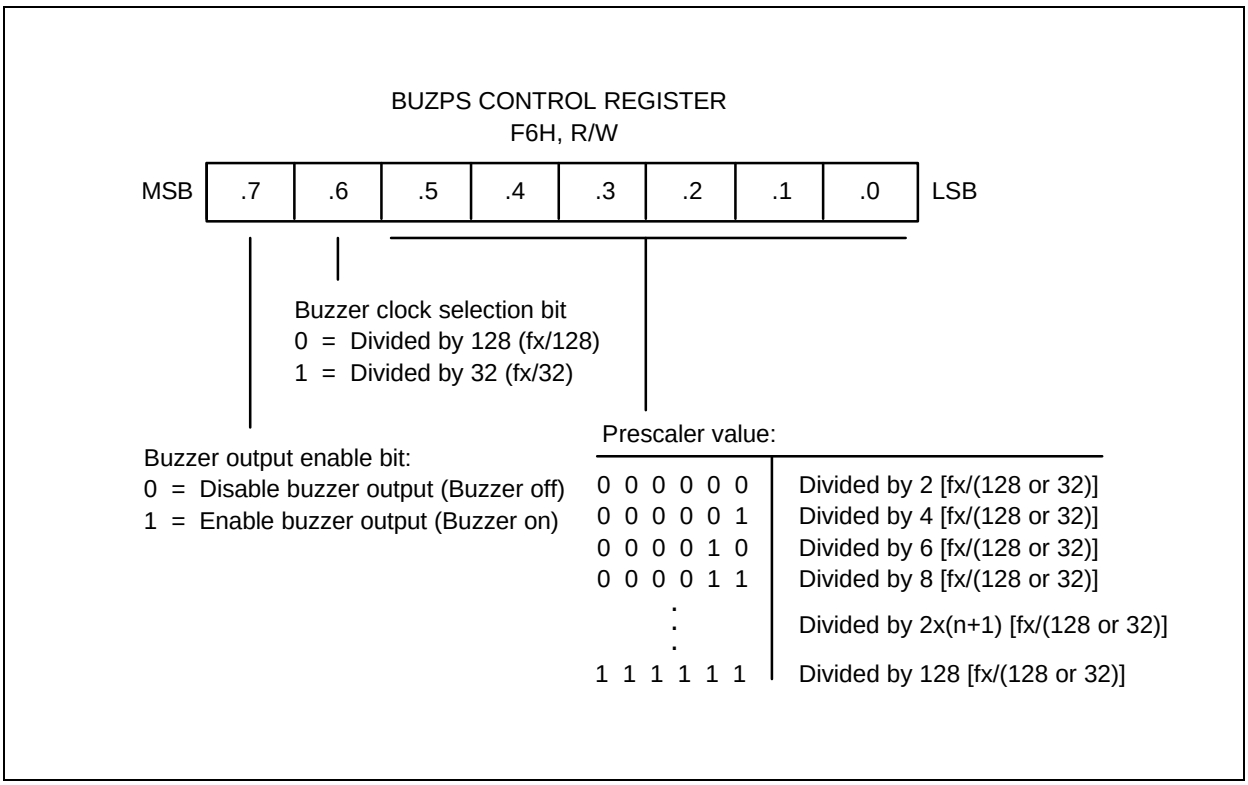


Figure 10-14. Buzzer output control register (BUZPS)

☞ PROGRAMMING TIP – Configuring Timer 1

The following sample program sets Timer 1 to interval timer mode and generates the melody signal by using buzzer.

```
                .INCLUDE    "C:\SMDS2P\INCLUDE\REG\86C4004.REG"

;-----------------<< Interrupt vector address >>

                .ORG      0000H
                .VECTOR   00H, COMMON_INT          ; IRQ0/Interrupt vector address

;-----------------<< Initialize system and peripherals >>

                .ORG      0100H                    ; reset start address
RESET:          DI                                 ; disable interrupt
                LD        BTCON,#00000010B         ; enable watchdog function
                                                   ; clock source: fosc/4096 (262 ms overflow
                                                   ; at 4 MHz)
                LD        CLKCON,#00011000B        ; CPU clock source select (non-divided)
                LD        SP,#0C0H                 ; KS86C4004 Stack pointer initial
                LD        P1CON,#00001100B         ; P1.1 buzzer output mode
                LD        T1DATA,#4DH              ; 10 ms interval at 4 MHz system clock
                                                   ; 250 ns x 512 x 78 = 9.984 ms
                LD        T1CON,#00000110B         ; timer1 match interrupt enable
                                                   ; clock source: fosc/512
                •
                •

                EI                                 ; enable interrupt

;-----------------<< Main loop >>

MAIN:           •
                •
                •
                LD        BTCON,#02H               ; enable watchdog function
                                                   ; basic counter (BTCNT) clear
                JP        T,MAIN                   ; for main loop

MELODY_CHK:
                OR        TIMER_CHK,#00000100B
                OR        MELODY_FLAG,#00000001B   ; melody signal enable
                LD        MUSIC_INTERVAL,#1
                CLR       BEEP_POINTER
                RET

;-----------------<< Interrupt service routine >>

COMMON_INT:
```

```
                TM      T1CON,#00000001B        ; timer 1 interrupt pending check
                JP      NZ,TIMER1_INT
                •
                •
                •
                IRET

TIMER1_INT:
                AND     T1CON,#11111010B        ; timer 1 pending bit clear
                TM      MELODY_FLAG,#00000001B          ; MELODY_FLAG.bit0 == 0 melody disable
                JP      NZ,MELODY_ENABLE        ;                       == 1 melody enable
                IRET

MELODY_ENABLE:
                DEC     MUSIC_INTERVAL
                CP      MUSIC_INTERVAL,#0       ; note duration check
                JP      EQ,MELODY_LOAD          ; next DB data load?
                IRET

MELODY_LOAD:                                    ; melody interval reload
                LD      R15,BEEP_POINTER
                RL      R15                     ; BEEP_POINTER x 2
                CLR     R14
                LDC     R8,#MELODY_DB[RR14]     ; R8 ← Note duration
                LDC     R9,#MELODY_DB+1[RR14]   ; R9 ← tone
                LD      MUSIC_INTERVAL,R8
                LD      BUZPS,R9                ; P1.1 buzzer signal on, clock selection
                INC     BEEP_POINTER
                CP      MUSIC_INTERVAL,#0FFH    ; Special code check
                JP      EQ,MELODY_OFF
                IRET

MELODY_OFF:
                AND     MELODY_FLAG,#11111110B          ; melody signal disable
                LD      BUZPS,#0                ; Buzzer off
                IRET

MELODY_DB:                                      ; DW wxyzH (wx: Note duration
                                                        ; yz:Frequency,Tone)
                                                ; 4 MHz OSC Clock
                DW      32BCH,32B6H,32AFH,32ADH         ; 500 ms interval
                DW      32A8H,32A3H,32A0H,329EH ; C4–C5
                DW      3200H
                DW      649EH,649BH,6498H,6496H ; 1000 ms interval
                DW      6494H,6492H,64FFH,64FBH ; C5–C6
                DW      3200H
                DW      32FBH,32F5H,32F0H,32ECH         ; 500 ms interval
                DW      32E8H,32E4H,32E0H,32DFH; C6–C7
                DW      0FF00H                  ; special code (melody end)
                •
                •
```

11 A/D CONVERTER

OVERVIEW

The 10-bit A/D converter (ADC) module uses successive approximation logic to convert analog levels entering at one of the eight input channels to equivalent 10-bit digital values. The analog input level must lie between the AV_{REF} and AV_{SS} values. The A/D converter has the following components:

— Analog comparator with successive approximation logic
— D/A converter logic (resistor string type)
— ADC control register (ADCON)
— Eight multiplexed analog data input pins (ADC0–ADC7)
— 10-bit A/D conversion data output register (ADDATAH/L): KS86C4104
— 8-bit A/D conversion data output register (ADDATAH): KS86C4004
— AV_{REF} and AV_{SS} pins

To initiate an analog-to-digital conversion procedure, you write the channel selection data in the A/D converter control register ADCON to select one of the eight analog input pins (ADCn, n = 0–7) and set the conversion start or enable bit, ADCON.0. The read-write ADCON register is located at address F7H.

During a normal conversion, A/D C logic initially sets the successive approximation register to 200H (the approximate half-way point of an 10-bit register). This register is then updated automatically during each conversion step. The successive approximation block performs 10-bit conversions for one input channel at a time. You can dynamically select different channels by manipulating the channel selection bit value (ADCON.6–4) in the ADCON register. To start the A/D conversion, you should set a the enable bit, ADCON.0. When a conversion is completed, ACON.3, the end-of-conversion (EOC) bit is automatically set to 1 and the result is dumped into the ADDATA register where it can be read. The A/D converter ten enters an idle state. Remember to read the contents of ADDATA before another conversion starts. Otherwise, the previous result will be overwritten by the next conversion result.

NOTE

Because the ADC does not use sample-and-hold circuitry, it is important that any fluctuations in the analog level at the ADC0–ADC7 input pins during a conversion procedure be kept to an absolute minimum. Any change in the input level, perhaps due to circuit noise, will invalidate the result.

USING A/D PINS FOR STANDARD DIGITAL INPUT

The ADC module's input pins are alternatively used as digital input in port 3 and port 2. The ADC0–ADC5 share pin names are P3.0–P3.5 and ADC6–ADC7 share pin names are P2.2–P2.3, respectively

A/D CONVERTER CONTROL REGISTER (ADCON)

The A/D converter control register, ADCON, is located at address F7H. Only bits 6–3 and 0 are used in the KS86C4004/4104 implementation. ADCON has three functions:

— Bits 6–4 select an analog input pin (ADC0–ADC7).

— Bit 3 indicates the status of the A/D conversion.

— Bit 0 starts the A/D conversion.

Only one analog input channel can be selected at a time. You can dynamically select any one of the eight analog input pins (ADC0–ADC7) by manipulating the 3-bit value for ADCON.6–ADCON.4

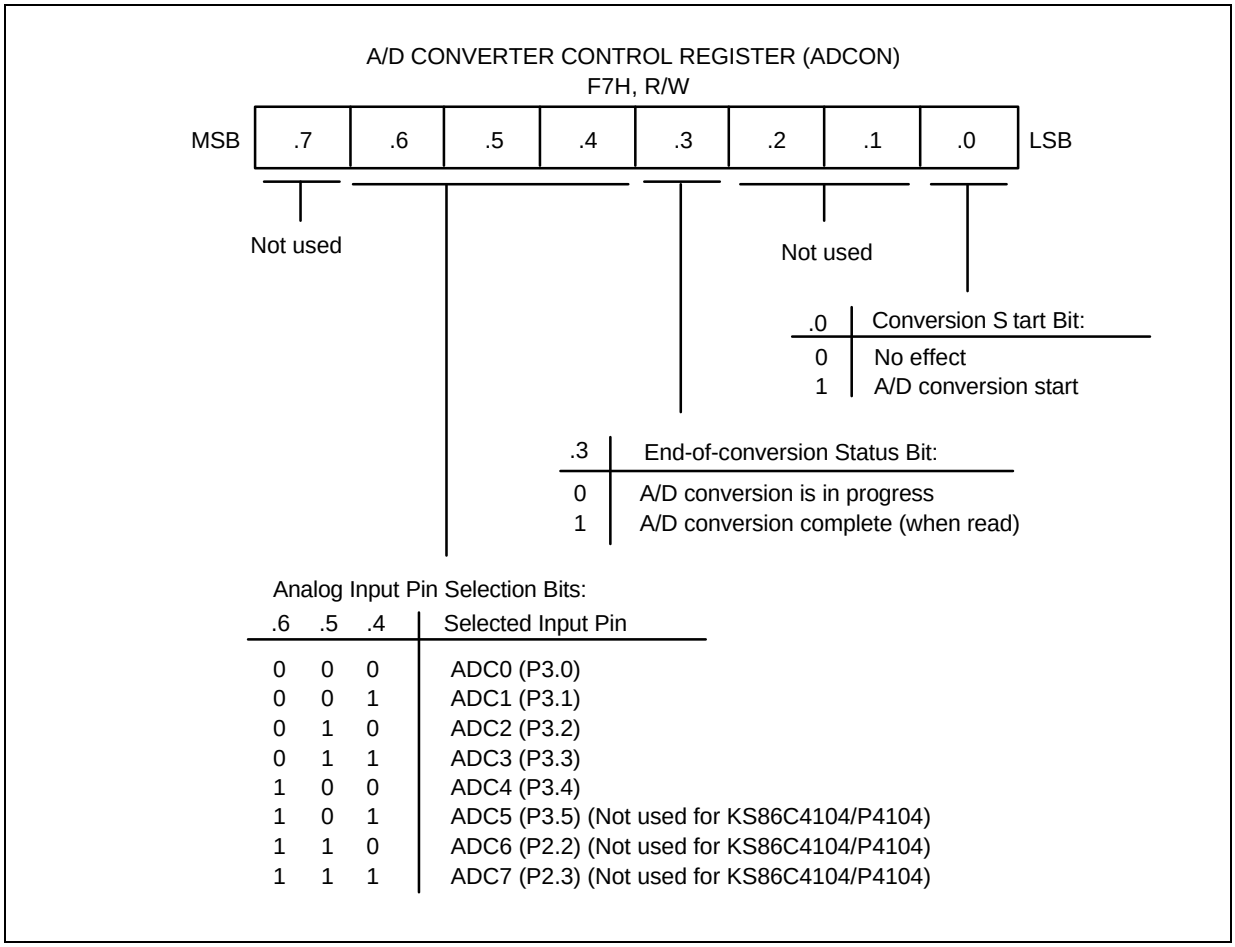


Figure 11-1. A/D Converter Control Register (ADCON)

INTERNAL REFERENCE VOLTAGE LEVELS

In the ADC function block, the analog input voltage level is compared to the reference voltage. The analog input level must remain within the range AV_{SS} to AV_{REF} (usually, AV_{REF} = V_{DD}).

Different reference voltage levels are generated internally along the resistor tree during the analog conversion process for each conversion step. The reference voltage level for the first bit conversion is always 1/2 AV_{REF}.

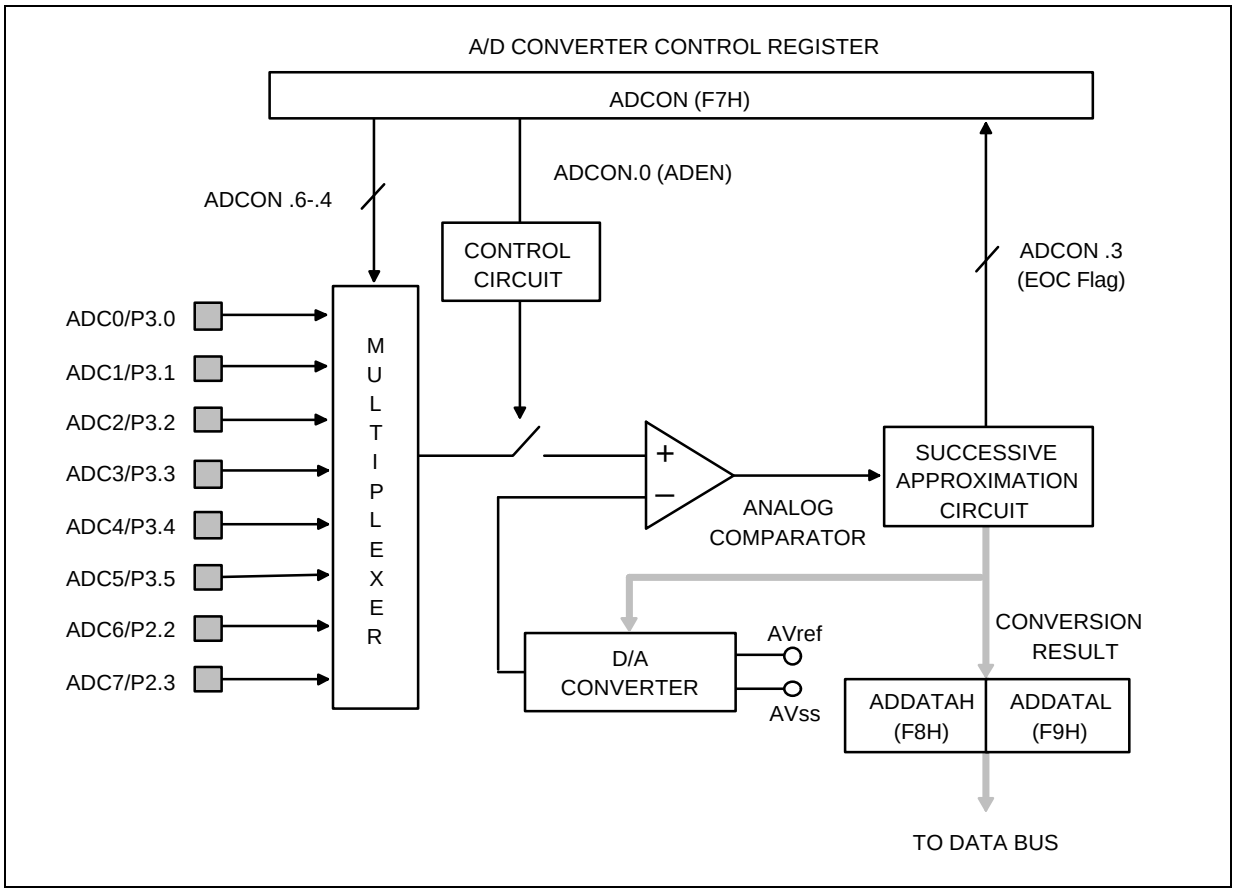


Figure 11-2. A/D Converter Circuit Diagram

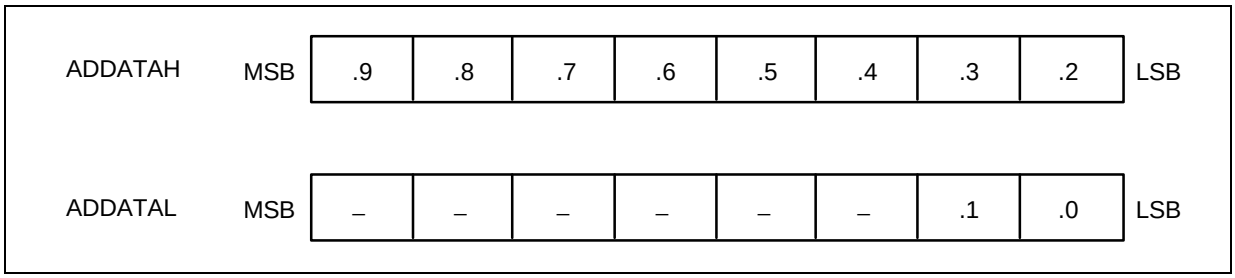


Figure 11-3. A/D Converter Data Register (ADDATAH/L)

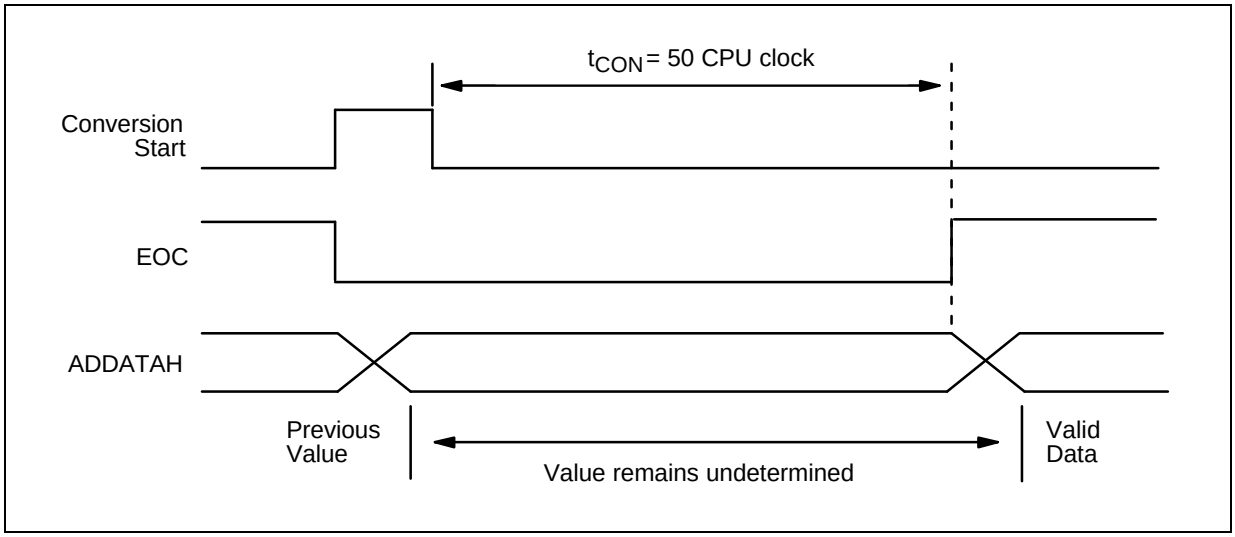


Figure 11-4. KS86C4004 A/D Converter Timing Diagram

CONVERSION TIMING (KS86C4004)

The A/D conversion process requires 4 steps (4 clock edges) to convert each bit and 18 clocks to step-up A/D conversion. Therefore, total of 50 clocks are required to complete an 8-bit conversion: With an 10 MHz CPU clock frequency, one clock cycle is 100 ns. If each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

4 clocks/bit x 8-bits + step-up time (18 clock) = 50 clocks
50 clock x 100 ns = 5 µs at 10 MHz, 1 clock time = CPU clock

INTERNAL A/D CONVERSION PROCEDURE

1. Analog input must remain between the voltage range of AV_{SS} and AV_{REF}.
2. Configure the analog input pins to input mode by making the appropriate settings in P3CONH, P3CONL and P2CON registers.
3. Before the conversion operation starts, you must first select one of the eight input pins (ADC0–ADC7) by writing the appropriate value to the ADCON register.
4. When conversion has been completed, (50 CPU clocks have elapsed), the EOC flag is set to “1”, so that a check can be made to verify that the conversion was successful.
5. The converted digital value is loaded to the output register, ADDATAH, than the ADC module enters an idle state.
6. The digital conversion result can now be read from the ADDATAH register.

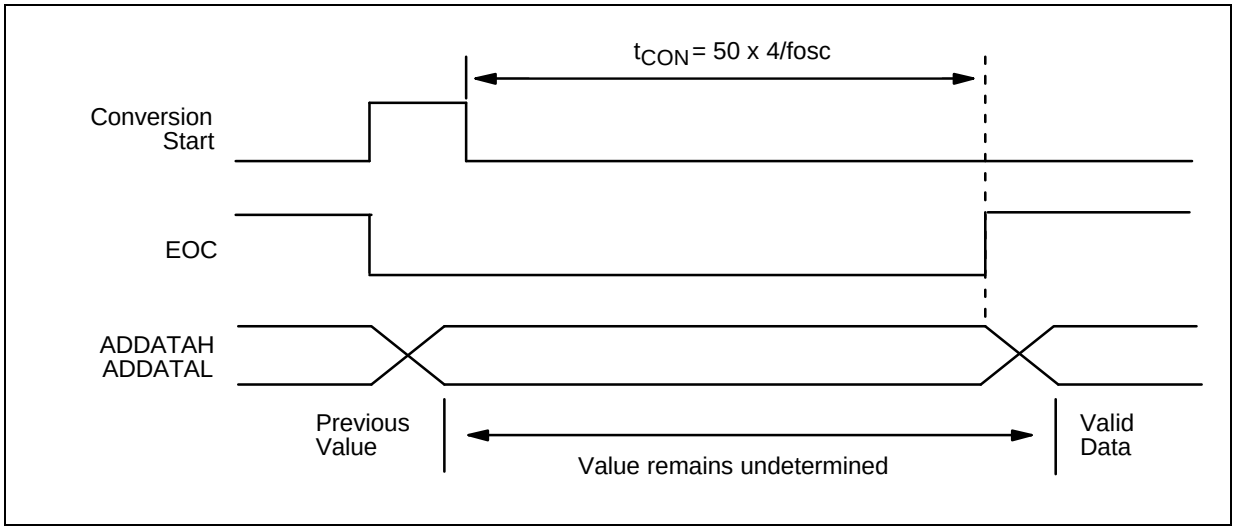


Figure 11-5. KS86C4104 A/D Converter Timing Diagram

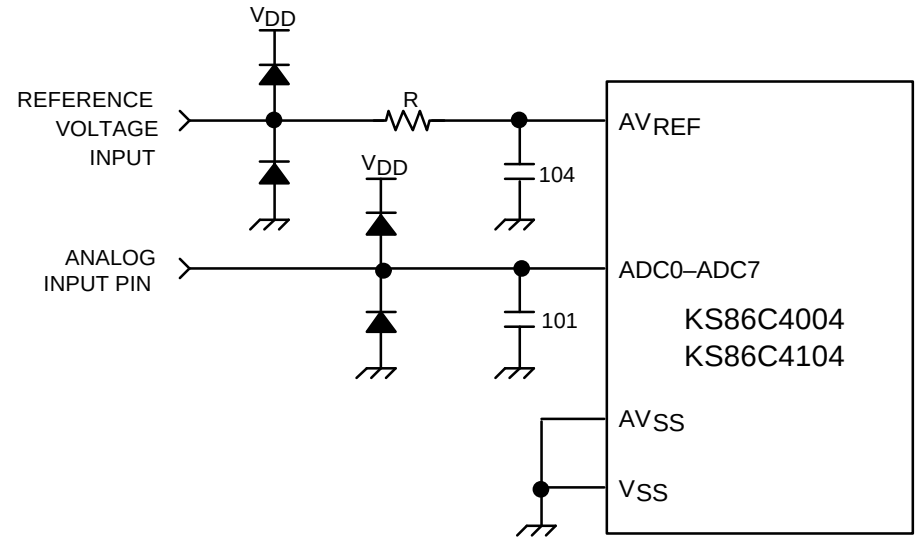
CONVERSION TIMING (KS86C4104)

The A/D conversion process requires 4 steps (4 clock edges) to convert each bit and 10 clocks to step-up A/D conversion. Therefore, total of 50 clocks are required to complete an 10-bit conversion: With an 10 MHz CPU clock frequency, one clock cycle is 400 ns (4/fosc). If each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

4 clocks/bit x 10-bits + step-up time (10 clock) = 50 clocks
50 clock x 400 ns = 20 μs at 10 MHz, 1 clock time = 4/fosc

INTERNAL A/D CONVERSION PROCEDURE

1. Analog input must remain between the voltage range of AV_{SS} and AV_{REF}.
2. Configure the analog input pins to input mode by making the appropriate settings in P3CONH and P3CONL registers.
3. Before the conversion operation starts, you must first select one of the five input pins (ADC0–ADC4) by writing the appropriate value to the ADCON register.
4. When conversion has been completed, (50 clocks have elapsed), the EOC flag is set to “1”, so that a check can be made to verify that the conversion was successful.
5. The converted digital value is loaded to the output register, ADDATAH (8-bit) and ADDATAL (2-bit), than the ADC module enters an idle state.
6. The digital conversion result can now be read from the ADDATAH and ADDATAL register.



NOTE: The symbol "R" signifies an offset resistor with a value of from 50 to 100 Ω.

Figure 11-6. Recommended A/D Converter Circuit for Highest Absolute Accuracy

☞ **PROGRAMMING TIP PROGRAMMING TIP – Configuring A/D Converter**

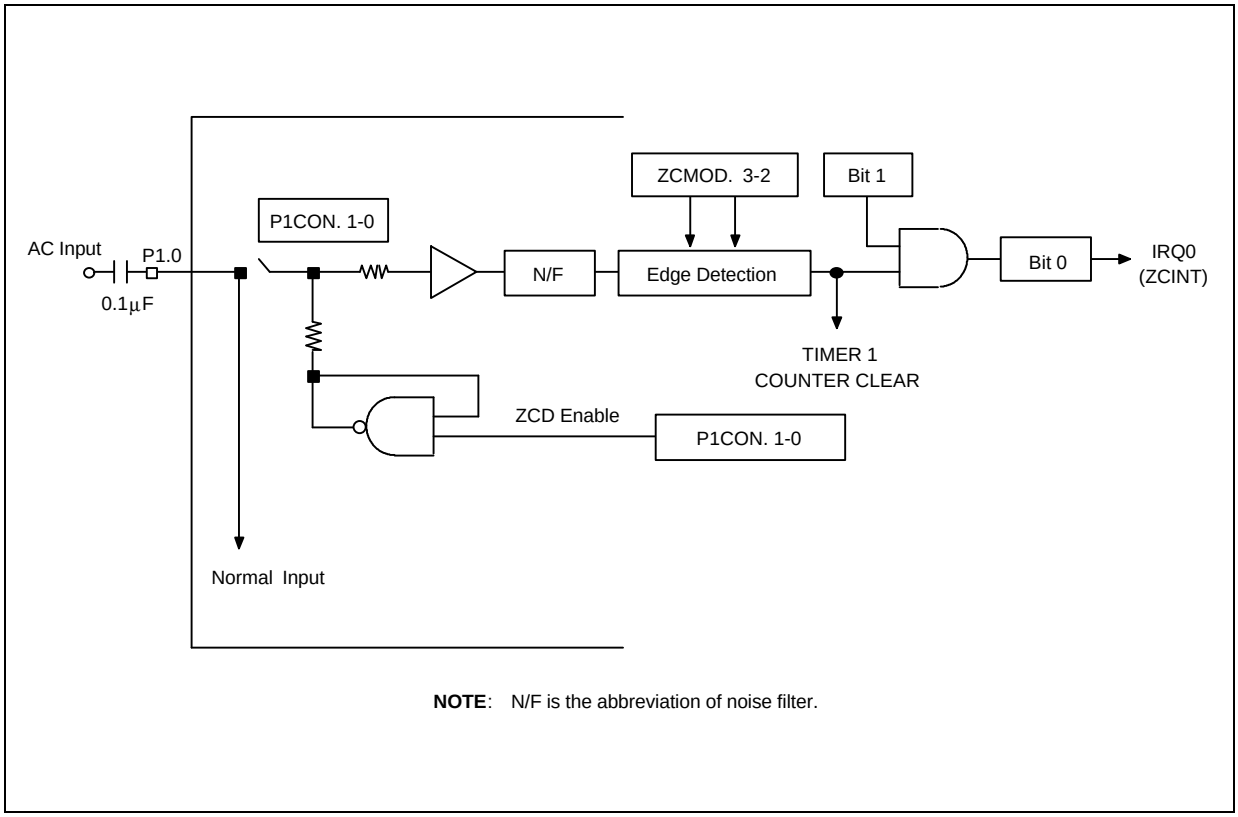
```
                •
                •
                •
                LD      P3CONL,#00001111B       ; P3.1-0 A/D Input MODE
                LD      P2CON,#11110000B        ; P2.3-2 A/D Input MODE
                LD      P2DPUR,#00000000B       ; P2 PULL-UP Disable
                •
                •
                •
                LD      ADCON,#00000001B        ; channel ADC0: P3.0/conversion start
AD0_CHK:        TM      ADCON,#00001000B        ; A/D conversion end ? → EOC check
                JR      Z,AD0_CHK               ; no
                LD      AD0BUFH,ADDATAH         ; Conversion data
                LD      AD0BUFL, ADDATAL
                •
                •
                LD      ADCON,#01100001B        ; channel ADC6: P2.2/Conversion start
AD6_CHK:        TM      ADCON,#00001000B        ; A/D conversion end ? → EOC check
                JR      Z,AD6_CHK               ; no
                LD      AD6BUH,ADDATAH          ; Conversion data
                LD      AD6BUFL,ADDATAL
                •
                •
```

12 ZERO-CROSSING DETECTION CIRCUIT

OVERVIEW

Zero-crossing detection circuit in Samsung's KS86C4004/C4104, generates a digital signal in synchronism with an AC signal input. It provides the timing signal for operations which are synchronized with the AC line. The zero crossing detection circuit digitizes the AC signal it receives from the power supply.

By setting bits 1 and 0 in port 1 control register (P1CON), you can enable zero-crossing detection. Zero-crossing detector is shown in Figure 12-1.



NOTE: N/F is the abbreviation of noise filter.

Figure 12-1. Zero-Crossing Detector Diagram

ZERO-CROSSING DETECTOR CONTROL REGISTER

The zero crossing detector control register, ZCMOD, is used to select interrupt mode (interrupt on falling edge, rising edge or both).

Reset clears ZCMOD to '00H', and configures interrupt selection mode to falling edge and disables ZCD interrupt. The interrupt pending bit must be cleared by writing "0" to ZCMOD.0

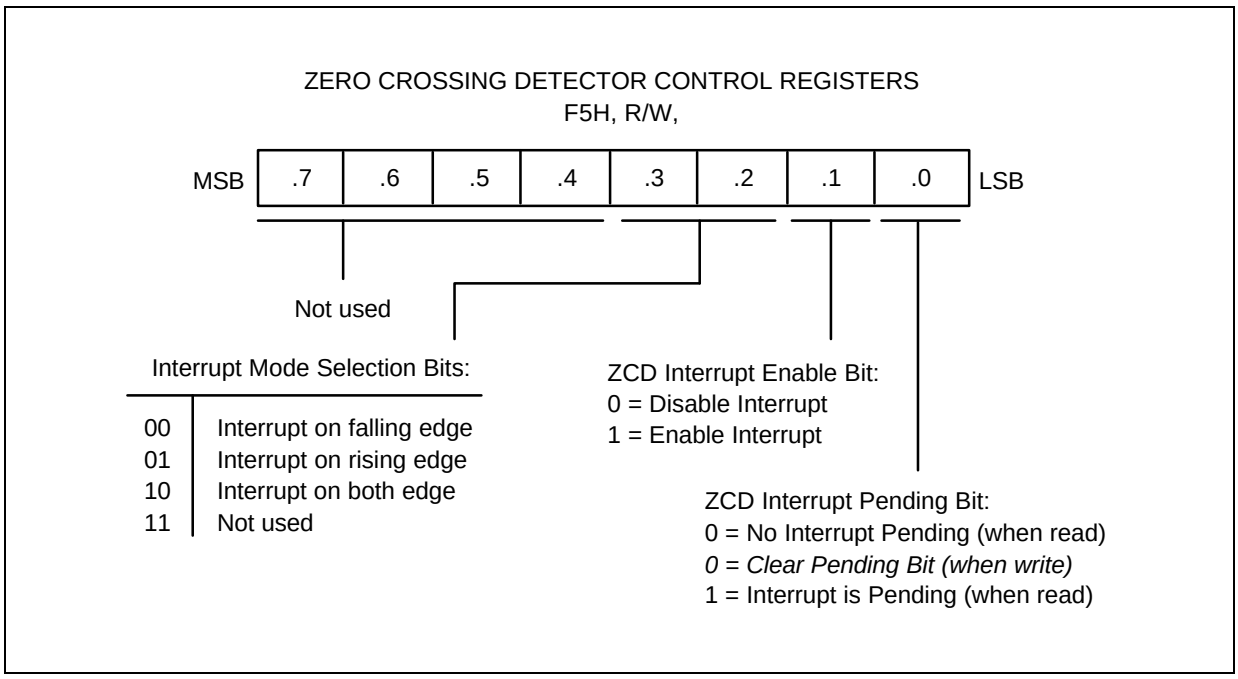


Figure 12-2. Zero-Crossing Detector Control Register (ZCMOD)

ZERO CROSS DETECTOR

ZCD circuit detects the zero-cross point of the AC waveform. Three types of detection can be selected, the point from positive to negative, the point from negative to positive, and both.

The zero cross detection circuit has the noise filter circuit in it. The detected zero cross point can be used to clear the timer 1 counter (T1CON.3 = 1).

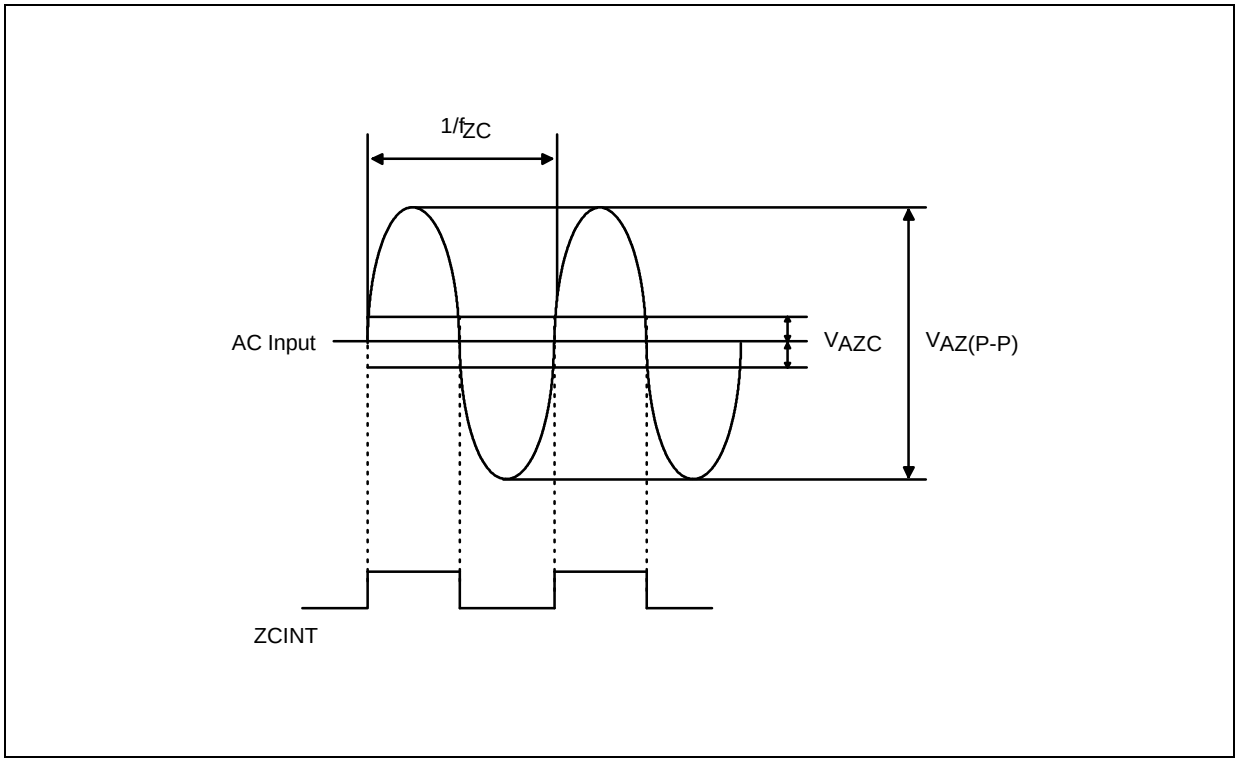


Figure 12-3. Zero-Crossing Waveform Diagram

☞ PROGRAMMING TIP – Configuring ZCD and Timer1

Programming procedure

1. Configure port 1 input pin to alternative function mode by making the appropriate bit setting to P1CON.
2. Select the correct clock source for timer 1 and set bit 3 (T1CON.3 = 1) to enable the ZCD signal to clear the timer 1. Load the proper data to timer 1 data register.
3. Enable the timer 1 interrupt in ZCD interrupt.
4. Set the relay on signal in timer 1 interrupt and disable the timer 1 interrupt.

```
            .ORG        0000H
            .VECTOR     00H, COMMON_INT         ; IRQ0/Interrupt vector address

            .ORG        0100H                   ; Reset start address
RESET:      DI                                  ; Disable interrupt
            LD          BTCON, #00000010B       ; Enable watchdog function
                                                ; clock source:fosc/4096
                                                      ; (104 ms overflow at 10 MHz)
            LD          CLKCON, #00011000B      ; CPU clock source select (non-divided)
            LD          SP, #0C0H               ; KS86C4004 Stack pointer initial
            LD          P1CON, #1010101011B     ; P1.0 ZCD input enable/P1.1-3 push-pull
            LD          ZCMOD, #00001010B       ; Enable both edge interrupt
            LD          T1DATA, #81H
            LD          T1CON, #00001100B       ; ZCD clear enable (fosc/512)
                                                ; Timer1 interrupt disable
            •
            •
            EI                                  ; Enable interrupt
            •
MAIN:       •
            LD          BTCON,#02H              ; Enable watchdog function
                                                ; Basic counter (BTCNT) clear
            •
            •
            JP          T,MAIN                  ; For main loop
            •
            •
            •

COMMON_INT:
            TM          ZCMOD, #00000001B
            JP          NZ, ZCD_INT
            TM          T1CON, #00000001B
            JP          NZ, TIMER1_INT
            •
            •
            •
```

```
TIMER1_INT:
            AND     T1CON, #11111100B       ; timer1 pending bit clear/t1 int disable
            XOR     P1, #00000100B          ; P1.2 toggle
            IRET
ZCD_INT:    AND     ZCMOD, #11111110B       ; pending bit clear
            XOR     P1, #00001000B          ; P1.3 toggle
            LD      T1CON, #00001010B       ; Timer1 interrupt enable (fosc/512)
                                            ; Enable ZCD clear signal to clear the
                                                    ; timer 1 counter
            IRET
            •
            •
```

NOTES

13 ELECTRICAL DATA

OVERVIEW

In this section, the following KS86C4004/C4104 electrical characteristics are presented in tables and graphs:

— Absolute maximum ratings
— D.C. electrical characteristics
— A.C. electrical characteristics
— Oscillator characteristics
— Oscillation stabilization time
— Operating Voltage Range
— Schmitt trigger input characteristics
— Data retention supply voltage in Stop mode
— Stop mode release timing when initiated by a RESET
— A/D converter electrical characteristics
— Zero-crossing detector
— Zero Crossing Waveform Diagram

Table 13-1. Absolute Maximum Ratings

(T_A = 25°C)

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}	–	– 0.3 to + 6.5	V
Input voltage	V_I	All input ports	– 0.3 to V_{DD} + 0.3	V
Output voltage	V_O	All output ports	– 0.3 to V_{DD} + 0.3	V
Output current high	I_{OH}	One I/O pin active	– 18	mA
		All I/O pins active	– 60	
Output current low	I_{OL}	One I/O pin active	+ 30	mA
		Total pin current for ports 1, 2, 3	+ 100	
		Total pin current for ports 0	+ 200	
Operating temperature	T_A	–	– 40 to + 85	°C
Storage temperature	T_{STG}	–	– 65 to + 150	°C

Table 13-2. DC Electrical Characteristics

(T_A = – 40°C to + 85°C, V_{DD} = 2.7 V to 5.5 V)

Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Input high voltage	V_{IH1}	Ports 1,2,3, and RESET	V_{DD}= 2.7 to 5.5 V	0.8 V_{DD}	–	V_{DD}	V
	V_{IH2}	Port 0		0.7 V_{DD}			
	V_{IH3}	X_{IN} and X_{OUT}		V_{DD} –0.1			
Input low voltage	V_{IL1}	Ports 1,2,3, and RESET	V_{DD}= 2.7 to 5.5 V	–	–	0.2 V_{DD}	V
	V_{IL2}	Port 0				0.3 V_{DD}	
	V_{IL3}	X_{IN} and X_{OUT}				0.1	
Output high voltage	V_{OH}	I_{OH} = – 1 mA ports 0, 1, 2, 3	V_{DD}= 4.5 to 5.5 V	V_{DD} – 1.0	–	–	V
Output low voltage	V_{OL1}	I_{OL} = 15 mA port 0	V_{DD}= 4.5 to 5.5 V	–	0.4	2.0	V
	V_{OL2}	I_{OL} = 4 mA port 1,2,3	V_{DD}= 4.5 to 5.5 V		0.4	2.0	

Table 13-2. DC Electrical Characteristics (Continued)

(T_A = – 40°C to + 85°C, V_{DD} = 2.7 V to 5.5 V)

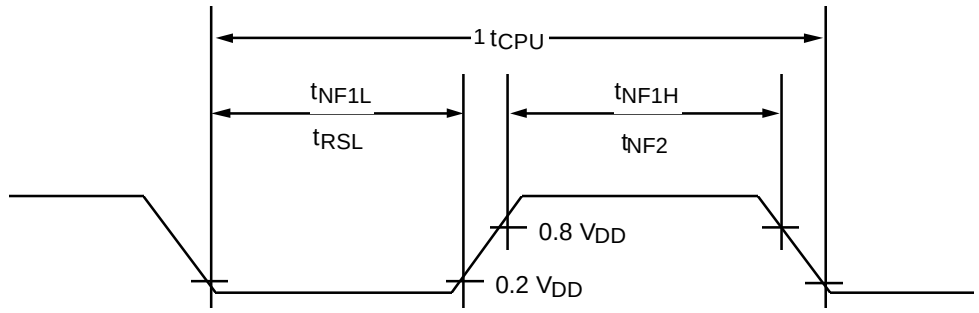
Parameter	Symbol	Conditions		Min	Typ	Max	Unit
Input high leakage current	I_{LIH1}	All inputs except I_{LIH2}	$V_{IN} = V_{DD}$	–	–	1	µA
	I_{LIH2}	X_{IN}, X_{OUT}	$V_{IN} = V_{DD}$			20	
Input low leakage current	I_{LIL1}	All inputs except I_{LIL2} and RESET	V_{IN} = 0 V	–	–	– 1	µA
	I_{LIL2}	X_{IN}, X_{OUT}	V_{IN} = 0 V			– 20	
Output high leakage current	I_{LOH}	All outputs	$V_{OUT} = V_{DD}$	–	–	2	µA
Output low leakage current	I_{LOL}	All outputs	V_{OUT} = 0 V	–	–	– 2	µA
Pull-up resistors	R_P	V_{IN} = 0 V Ports 0–3 and RESET	V_{DD} = 5 V	30	47	70	kΩ
			V_{DD} = 3 V	30	280	350	
Supply current	I_{DD1}	Run mode 10 MHz CPU clock	V_{DD} = 5 V ± 10%	–	7.5	15	mA
		8 MHz CPU clock	V_{DD} = 3 V ± 10%		3	6	
	I_{DD2}	Idle mode 10 MHz CPU clock	V_{DD} = 5 V ± 10%		2	5	
		8 MHz CPU clock	V_{DD} = 3 V ± 10%		0.7	2.5	
	I_{DD3}	Stop mode	V_{DD} = 5 V ± 10%		0.1	5	µA
			V_{DD} = 3 V ± 10%				

NOTE: D.C. electrical values for Supply current (I_{DD1} to I_{DD3}) do not include current drawn through internal pull-up resisters, output port drive current, ZCD and ADC.

Table 13-3. AC Electrical Characteristics

(T_A = –20°C to + 85°C, V_{DD} = 2.7 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Interrupt input high, low width	t_{INTH}, t_{INTL}	Port 2 V_{DD} = 5V ± 10%	–	200	–	ns
RESET input low width ZCD noise filter	t_{RSL} –	Input V_{DD} = 5V ± 10%	–	1	–	µs



NOTE: The unit t_{CPU} means one CPU clock period.

Figure 13-1. Input Timing Measurement Points

Table 13-4. Oscillator Characteristics

(T_A = – 40°C to + 85°C)

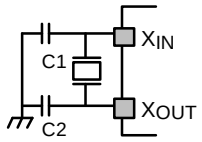
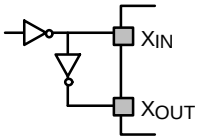
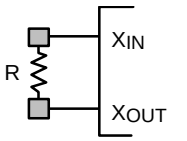
Oscillator	Clock Circuit	Test Condition	Min	Typ	Max	Unit
Main crystal or ceramic		V_{DD} = 4.5 to 5.5 V V_{DD} = 2.7 to 4.5 V	1 1	– –	10 8	MHz
External clock		V_{DD} = 4.5 to 5.5 V V_{DD} = 2.7 to 4.5 V	1 1	– –	10 8	
RC oscillator		V_{DD} = 4.75 to 5.25 V R = 8.2K	–	4 (P1.3/ CLO)	–	

Table 13-5. Oscillation Stabilization Time

(T_A = – 40°C to + 85°C, V_{DD} = 2.7 V to 5.5 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Main crystal	f_{OSC} > 1.0 MHz	–	–	20	ms
Main ceramic	Oscillation stabilization occurs when V_{DD} is equal to the minimum oscillator voltage range.	–	–	10	
External clock (main system)	X_{IN} input high and low width (t_{XH}, t_{XL})	25	–	500	ns
Oscillator stabilization wait time	t_{WAIT} when released by a reset (1)	–	$2^{16}/f_{OSC}$	–	ms
	t_{WAIT} when released by an interrupt (2)	–	–	–	

NOTES:

1. f_{OSC} is the oscillator frequency.
2. The duration of the oscillator stabilization wait time, t_{WAIT}, when it is released by an interrupt is determined by the settings in the basic timer control register, BTCON.

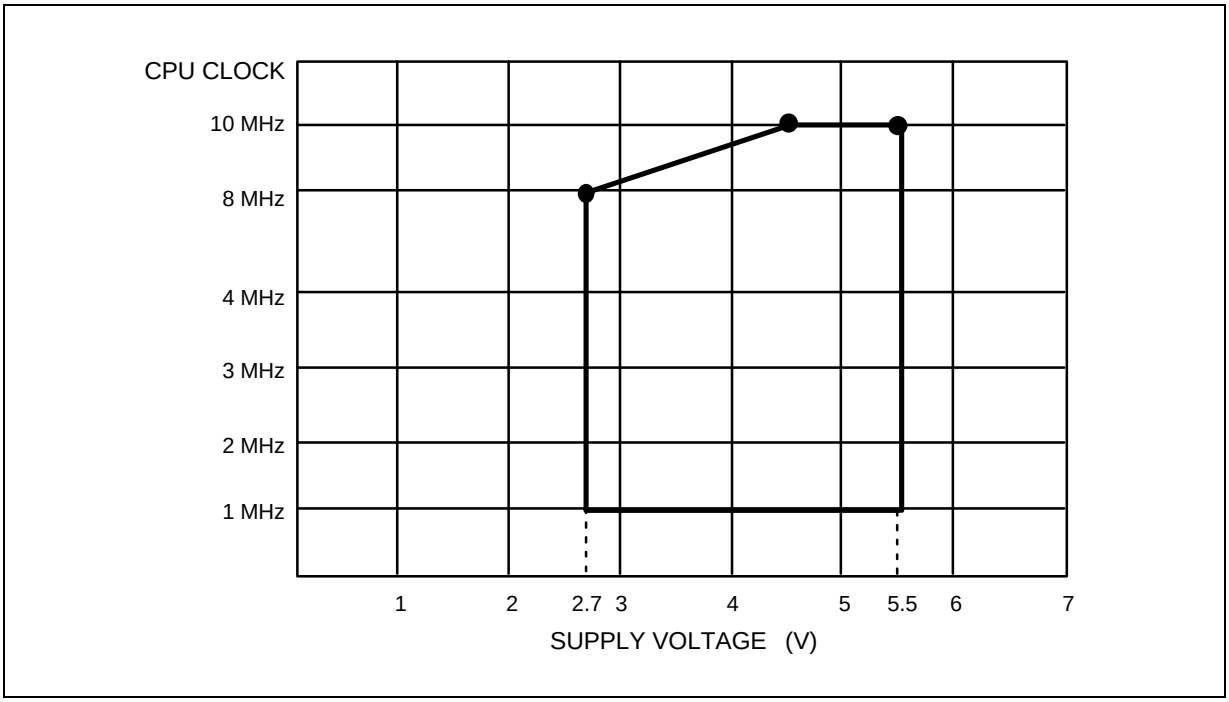


Figure 13-2. Operating Voltage Range

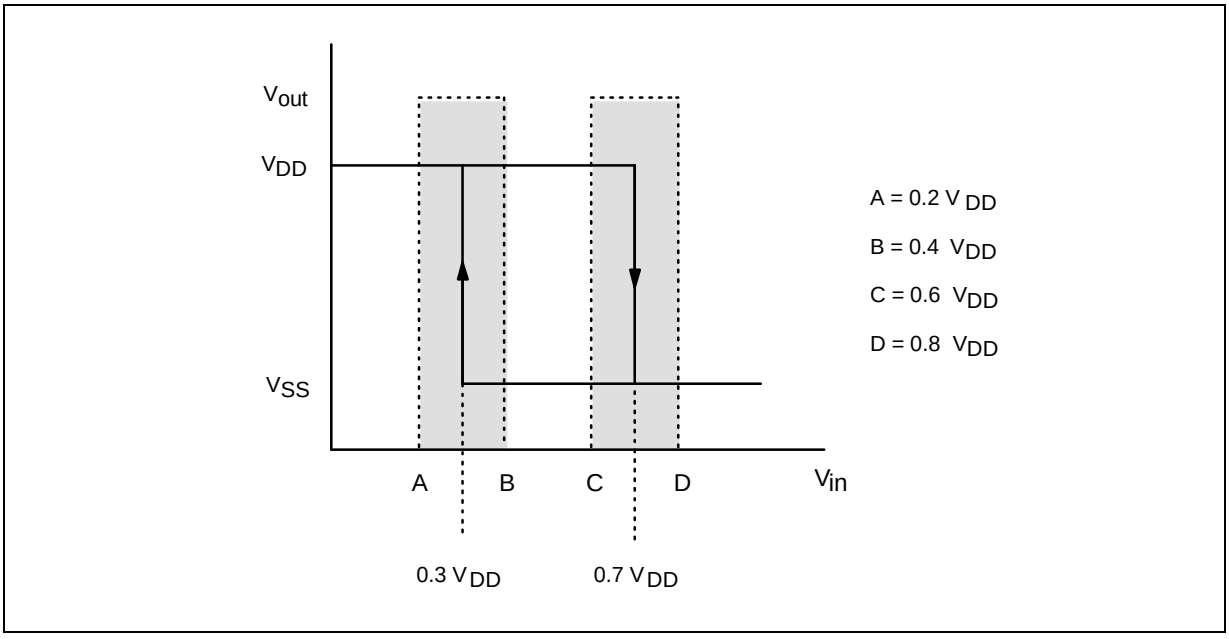


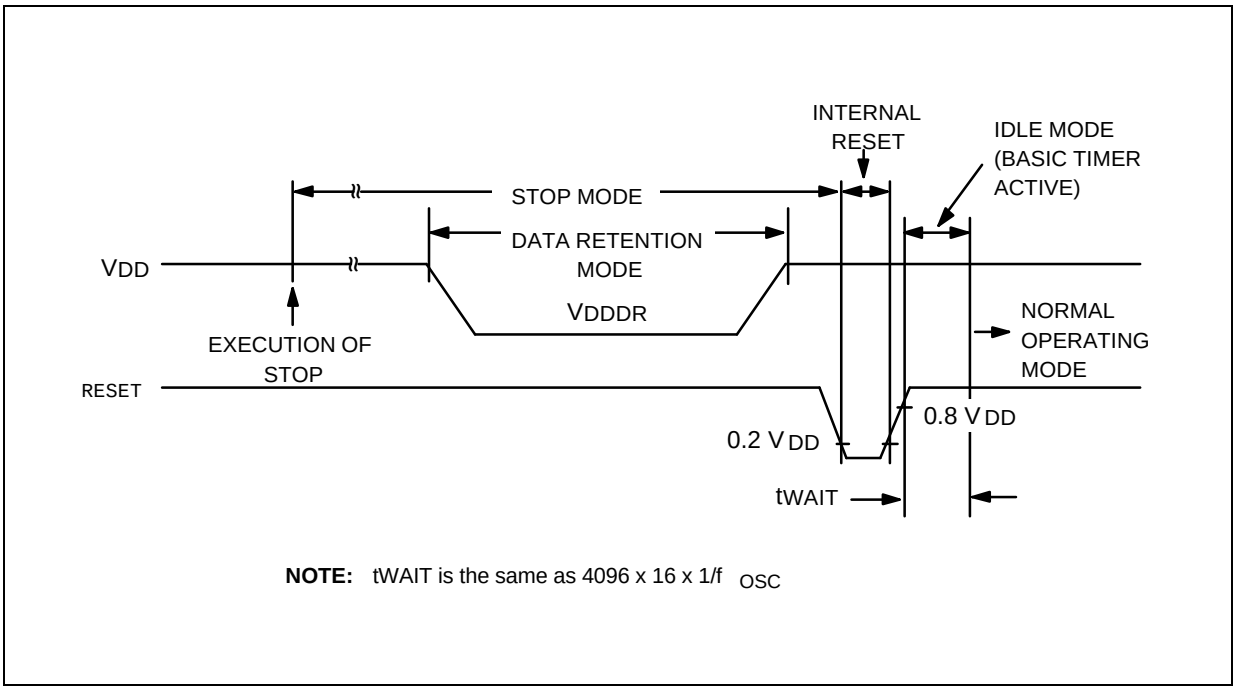
Figure 13-3. Schmitt Trigger Input Characteristics Diagram

Table 13-6. Data Retention Supply Voltage in Stop Mode

(T_A = – 40°C to + 85°C, V_{DD} = 2.7 V to 5.5V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	Stop mode	2.0	–	5.5	V
Data retention supply current	I_{DDDR}	Stop mode; V_{DDDR} = 2.0 V	–	0.1	5	µA

NOTE: Supply current does not include current drawn through internal pull-up resistors or external output current loads.



NOTE: tWAIT is the same as 4096 x 16 x 1/f_{OSC}

Figure 13-4. Stop Mode Release Timing When Initiated by a RESET

Table 13-7. A/D Converter Electrical Characteristics (KS86C4004)

(T_A = – 40°C to + 85°C, V_{DD} = 2.7 V to 5.5 V, V_{SS} = 0 V) KS86C4004: 8-bit ADC

Parameter	Symbol	Test Conditions	Min	Typ	Max	Unit
Total accuracy		V_{DD} = 5.12 V	–	–	± 2	LSB
Integral linearity error	ILE	CPU clock = 10 MHz AV_{REF} = 5.12 V		–	± 1.5	
Differential linearity error	DLE	AV_{SS} = 0 V		–	± 1	
Offset error of top	EOT			– 1	± 2	
Offset error of bottom	EOB			– 1	± 2	
Conversion time(1)	t_{CON}	fcpu = 10 MHz	5	–	–	µs
Analog input voltage	V_{IAN}	–	AV_{SS}	–	AV_{REF}	V
Analog input impedance	R_{AN}	–	2	–	–	MΩ
ADC reference voltage	AV_{REF}	–	2.5	–	V_{DD}	V
ADC reference ground	AV_{SS}	–	V_{SS}	–	V_{SS} + 0.3	V
Analog input current	I_{ADIN}	AV_{REF} = V_{DD} = 5 V conversion time = 5 µs	–	–	10	µA
ADC block current (2)	I_{ADC}	AV_{REF} = V_{DD} = 5 V conversion time = 5 µs	–	1	3	mA
		AV_{REF} = V_{DD} = 3 V conversion time = 5 µs		0.5	1.5	
		AV_{REF} = V_{DD} = 5 V Power down mode	–	100	500	nA

NOTES:

1. "Conversion time" is the time required from the moment a conversion operation starts until it ends.
2. I_{ADC} is operating current during A/D conversion.

Table 13-8. A/D Converter Electrical Characteristics (KS86C4104)

(T_A = – 40°C to + 85°C, V_{DD} = 2.7 V to 5.5 V, V_{SS} = 0 V) KS86C4104: 10-bit ADC

Parameter	Symbol	Test Conditions	Min	Typ	Max	Unit
Resolution			–	10	–	bit
Total accuracy		V_{DD} = 5.12 V	–	–	± 3	LSB
Integral linearity error	ILE	CPU clock = 10 MHz AV_{REF} = 5.12 V		–	± 2	
Differential linearity error	DLE	AV_{SS} = 0 V		–	± 1	
Offset error of top	EOT			± 1	± 3	
Offset error of bottom	EOB			± 0.5	± 2	
Conversion time (1)	t_{CON}	10-bit conversion 50 x 4/ f_{OSC} (3)	20	–	–	µs
Analog input voltage	V_{IAN}	–	AV_{SS}	–	AV_{REF}	V
Analog input impedance	R_{AN}	–	2	–	–	MΩ
Analog reference voltage	AV_{REF}	–	2.5	–	V_{DD}	V
Analog ground	AV_{SS}	–	V_{SS}	–	V_{SS} + 0.3	V
Analog input current	I_{ADIN}	AV_{REF} = V_{DD} = 5 V conversion time = 20 µs	–	–	10	µA
Analog block current (2)	I_{ADC}	AV_{REF} = V_{DD} = 5 V conversion time = 20 µs		1	3	mA
		AV_{REF} = V_{DD} = 3 V conversion time = 20 µs		0.5	1.5	mA
		AV_{REF} = V_{DD} = 5 V when power down mode		100	500	nA

NOTES:

1. "Conversion time" is the time required from the moment a conversion operation starts until it ends.
2. I_{ADC} is operating current during A/D conversion.
3. f_{OSC} is the main oscillator clock.

Table 13-9. Zero Crossing Detector

(T_A = – 40°C to + 85°C, V_{DD} = 4.5 V to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Test Conditions	Min	Typ	Max	Unit
Zero-crossing detection input voltage	V_{ZC}	AC connection c = 0.1 µF	1.0	–	3.0	Vp-p
Zero-crossing detection accuracy	V_{AZC}	f_{ZC} = 60 Hz (sine wave) V_{DD} = 5 V f_{OSC} = 10 MHz	–	–	± 150	mV
Zero-crossing detection input frequency	f_{ZC}	–	40	–	200	Hz

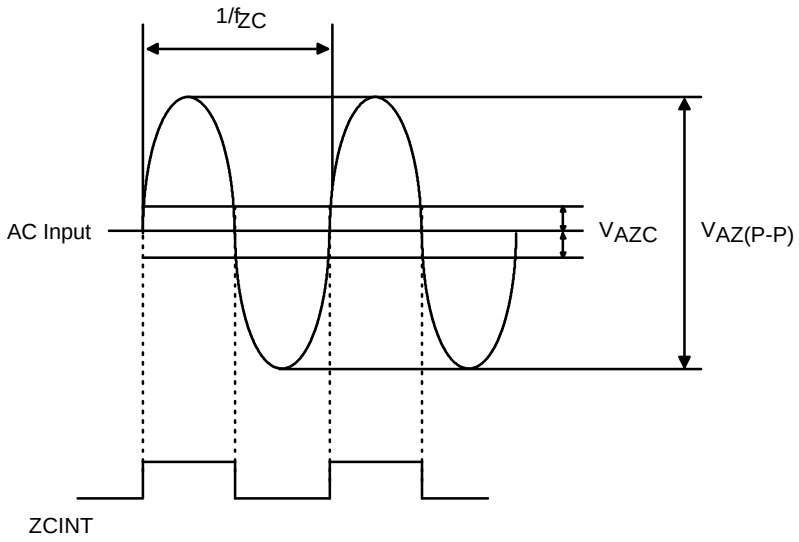


Figure 13-5. Zero Crossing Waveform Diagram

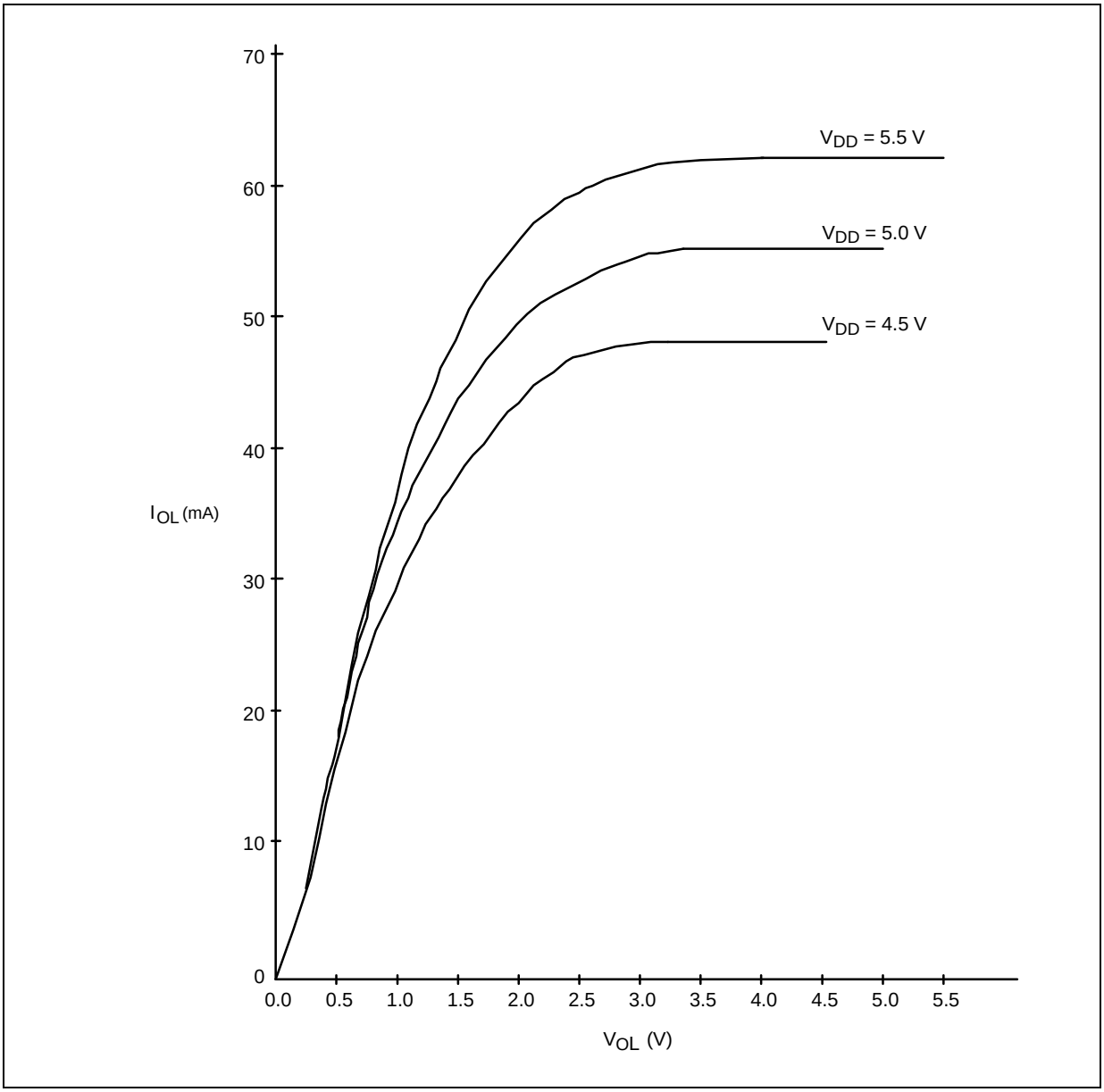


Figure 13-6. I_{OL} vs. V_{OL} (P0, TA = 25 °C)

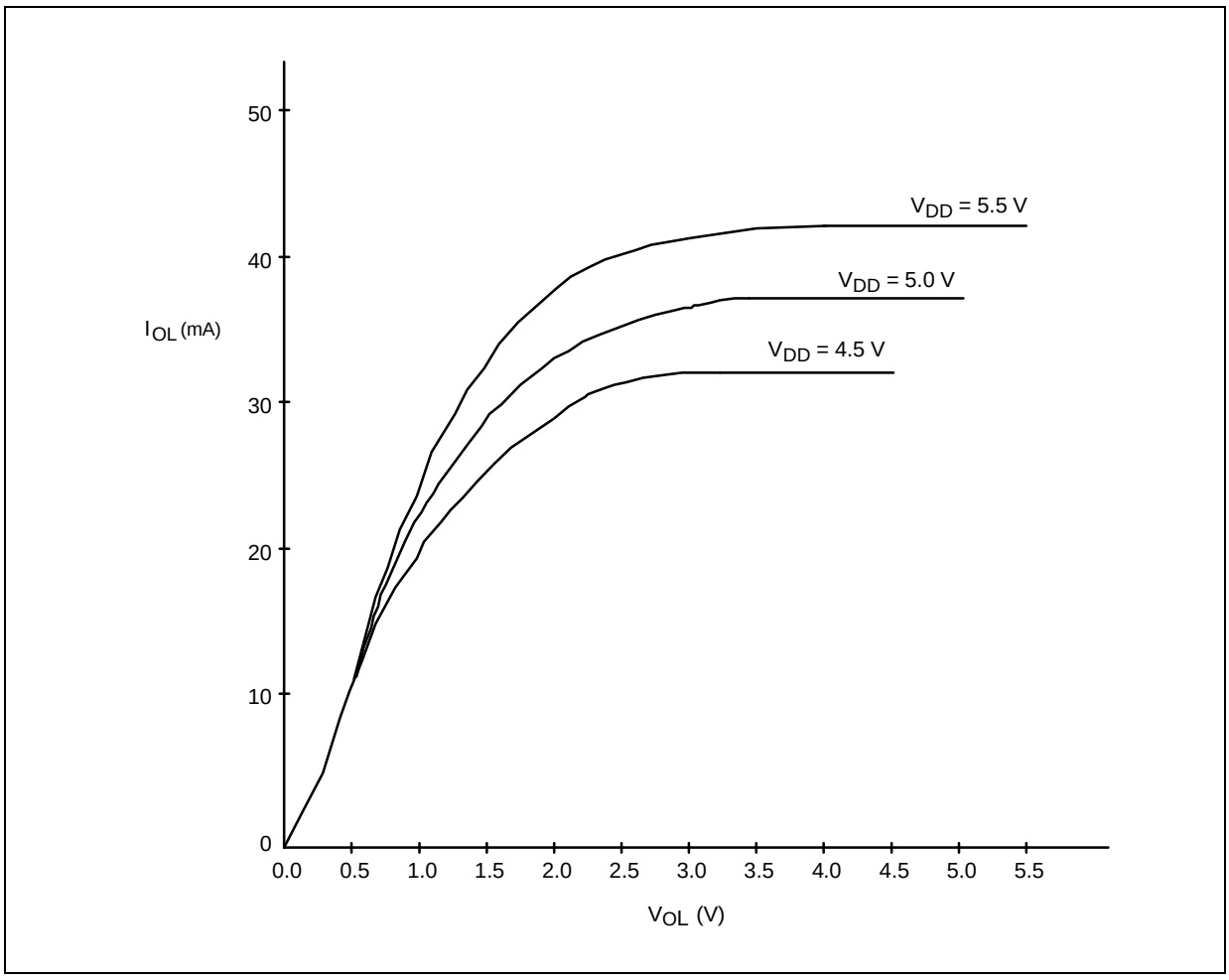


Figure 13-7. I_{OL} vs. V_{OL} (P1–P3, TA = 25 °C)

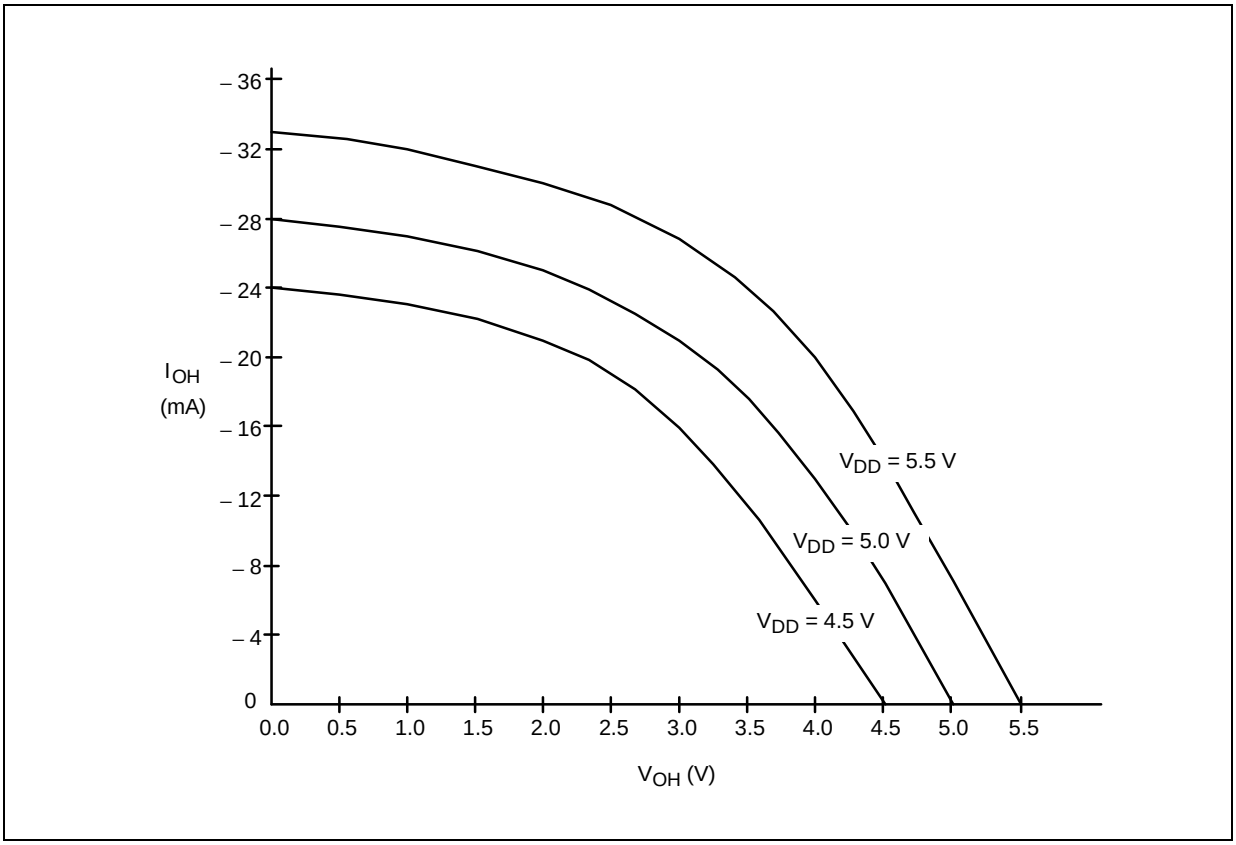


Figure 13-8. I_{OH} vs. V_{OH} (P0, TA = 25 °C)

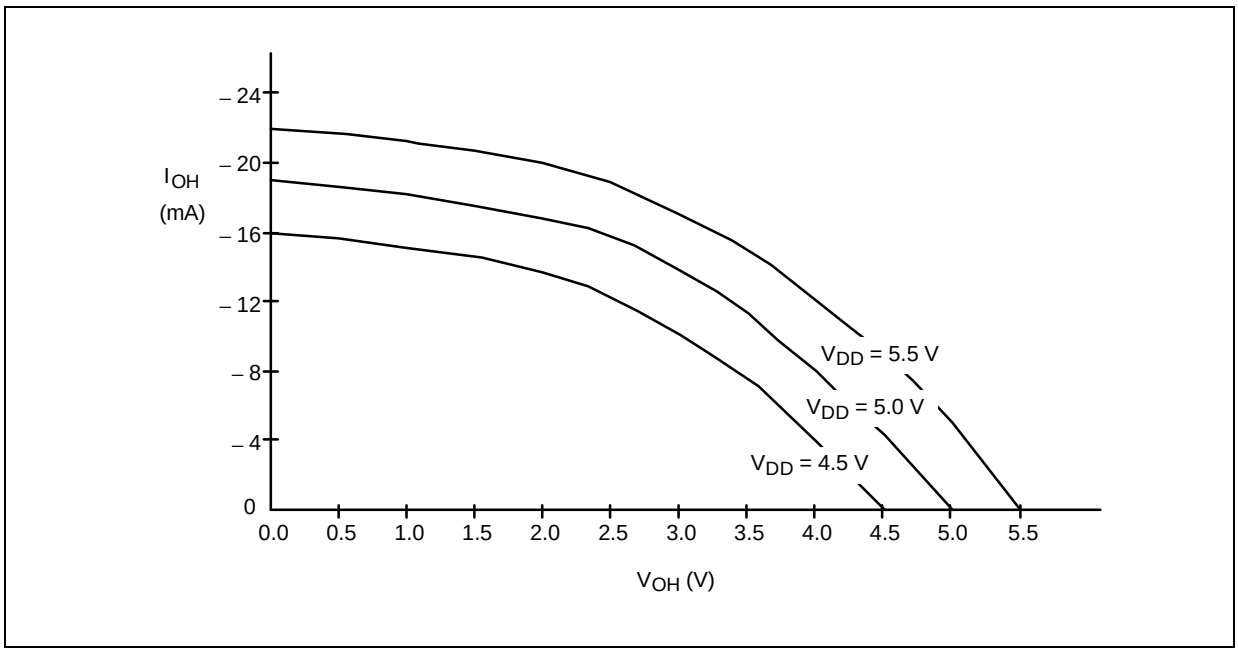
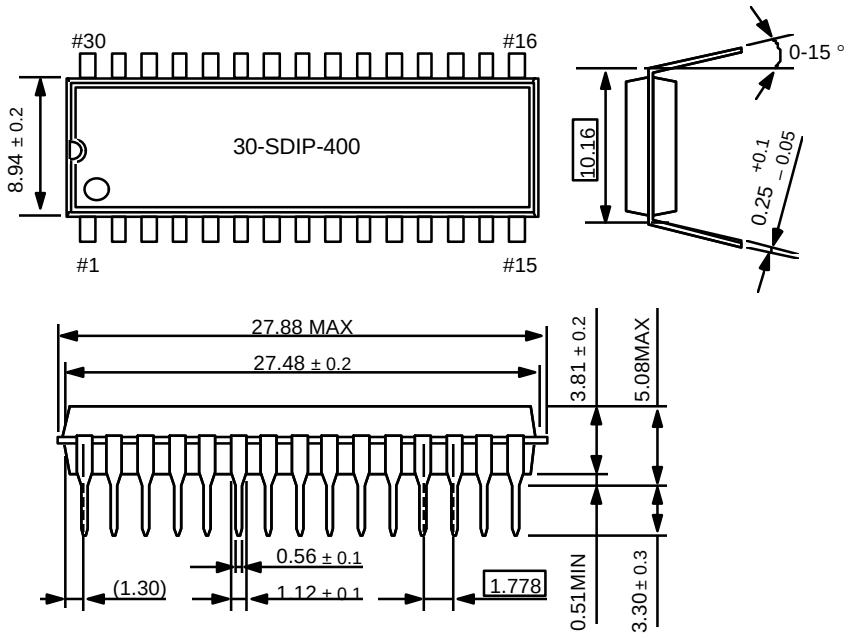


Figure 13-9. I_{OH} vs. V_{OH} (P1–P3, TA = 25 °C)

14 MECHANICAL DATA

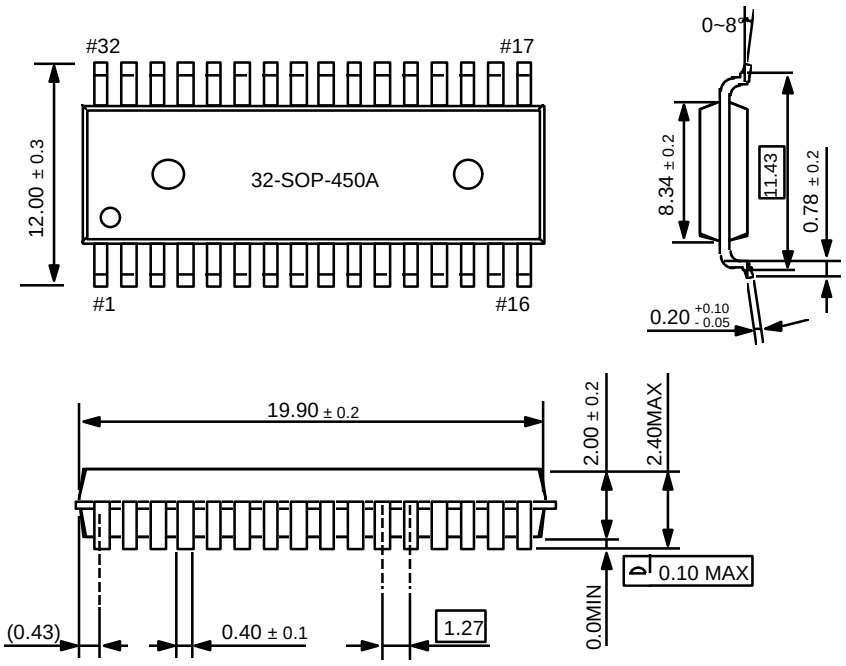
OVERVIEW

The KS86C4004/C4104 is available in a 30-pin SDIP package (Samsung: 30-SDIP-400) and a 32-pin SOP package (32-SOP-450A), a 24-pin SDIP package (24-SDIP-300) and a 24-pin SOP package (24-SOP-375). Package dimensions are shown in Figures 14-1, 14-2, 14-3, and 14-4.



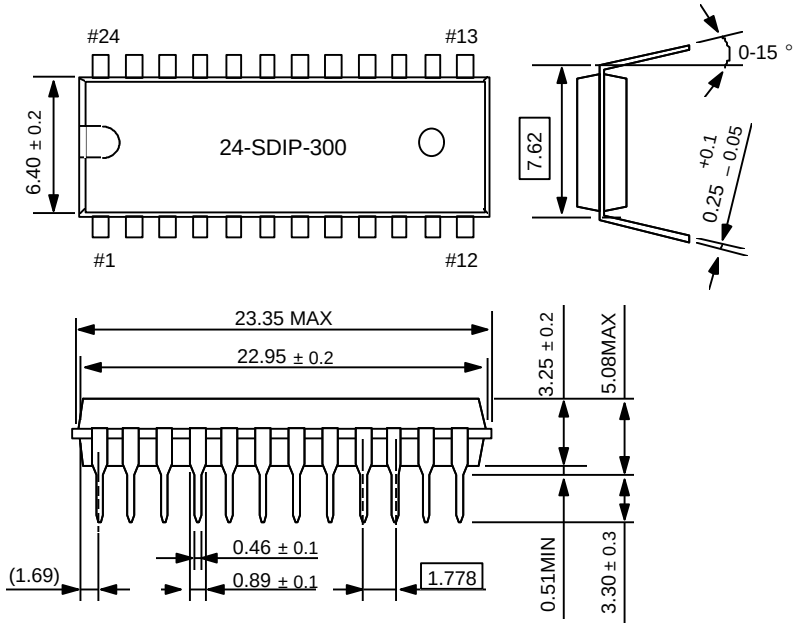
NOTE: Dimensions are in millimeters.

Figure 14-1. 30-Pin SDIP Package Dimensions



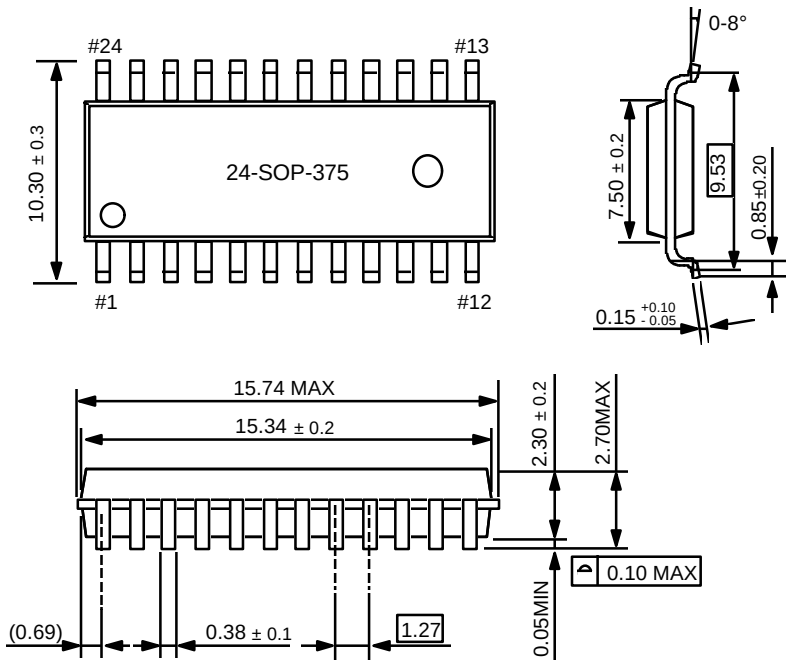
NOTE: Dimensions are in millimeters.

Figure 14-2. 32-SOP-450A Package Dimensions



NOTE: Dimensions are in millimeters.

Figure 14-3. 24-SDIP-300 Package Dimensions



NOTE: Dimensions are in millimeters.

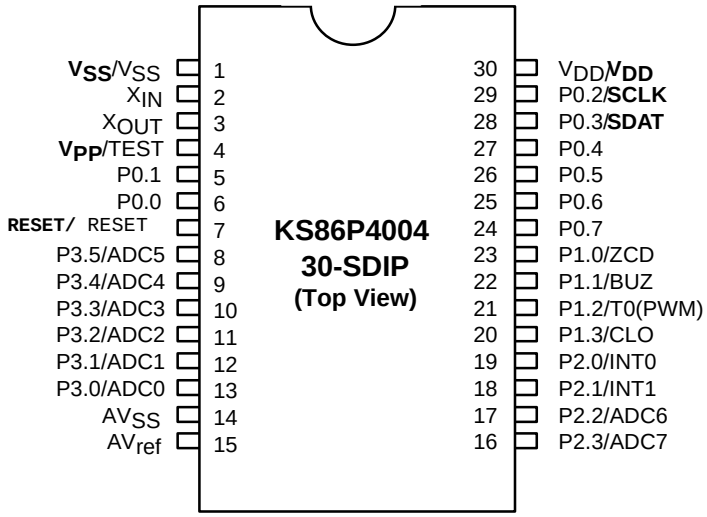
Figure 14-4. 24-SOP-375 Package Dimensions

15 KS86P4004/P4104 OTP

OVERVIEW

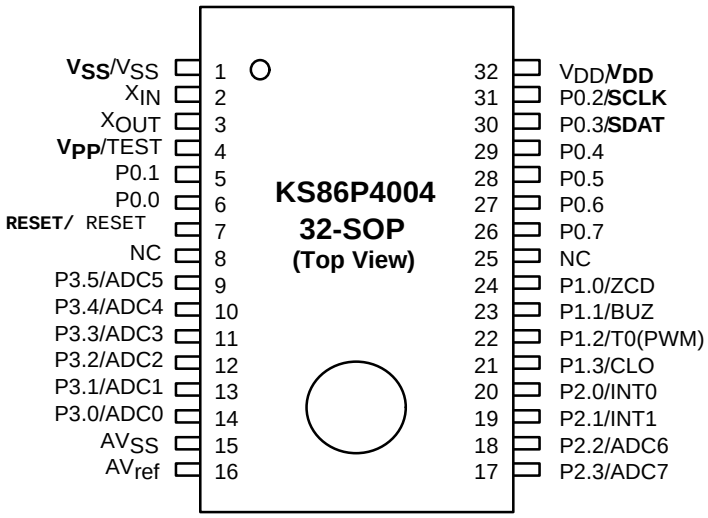
The KS86P4004/P4104 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the KS86C4004/C4104 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The KS86P4004/P4104 is fully compatible with the KS86C4004/C4104 , both in function and in pin configuration. Because of its simple programming requirements, the KS86P4004/P4104 is ideal for use as an evaluation chip for the KS86C4004/C4104 .



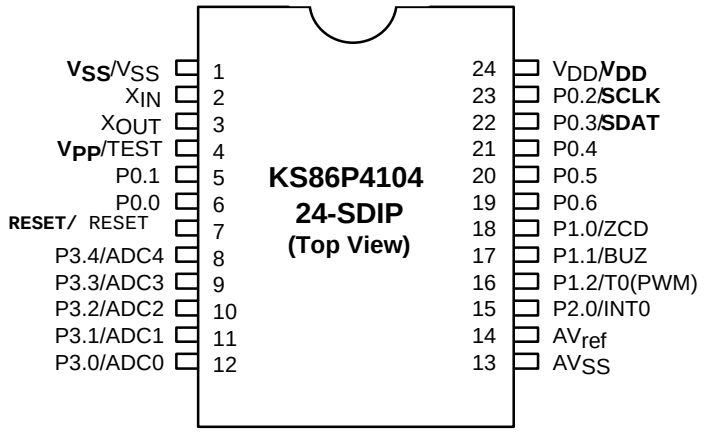
NOTE: The bolds indicate an OTP pin name.

Figure 15-1. Pin Assignment Diagram (30-Pin SDIP Package)



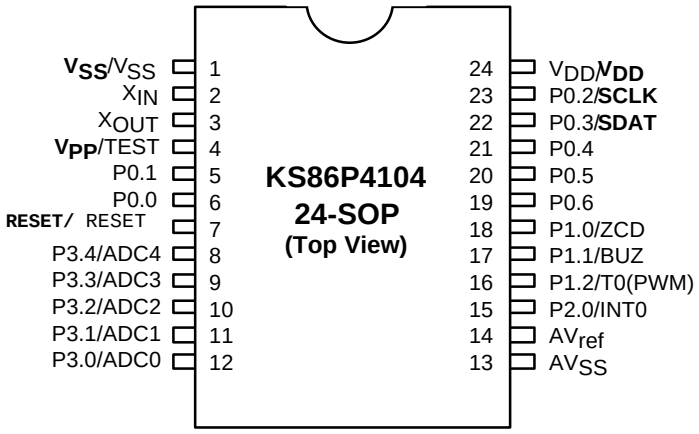
NOTE: The bolds indicate an OTP pin name.

Figure 15-2. Pin Assignment Diagram (32-Pin SOP Package)



NOTE: The bolds indicate an OTP pin name.

Figure 15-3. Pin Assignment Diagram (24-Pin SDIP Package)



NOTE: The bolds indicate an OTP pin name.

Figure 15-4. Pin Assignment Diagram (24-Pin SOP Package)

Table 15-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip	During Programming			
Pin Name	Pin Name	Pin No.	I/O	Function
P0.3	SDAT	KS86P4004: 28 (30) KS86P4104: 22 (22)	I/O	Serial data pin (output when reading, Input when writing) Input and push-pull output port can be assigned
P0.2	SCLK	KS86P4004: 29 (31) KS86P4104: 23 (23)	I/O	Serial clock pin (input only pin)
TEST	V_{PP} (TEST)	4	I	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.5 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode. (Option)
RESET	RESET	7	I	Chip Initialization
V_{DD}/V_{SS}	V_{DD}/V_{SS}	KS86P4004: 30 (32) / 1 KS86P4104: 24 (24) / 1	I	Logic power supply pin.

NOTE: ( ) means the SOP OTP pin number.

Table 15-2. Comparison of KS86P4004/P4104 and KS86C4004/C4104 Features

Characteristic	KS86P4004/P4104	KS86C4004/C4104
Program Memory	4-Kbyte EPROM	4-Kbyte mask ROM
Operating Voltage (V_{DD})	2.7 V to 5.5 V	2.7 V to 5.5 V
OTP Programming Mode	V_{DD} = 5 V, V_{PP} (TEST) = 12.5 V	
Pin Configuration	30 SDIP/32 SOP/24 SDIP/24 SOP	
EPROM Programmability	User Program 1 time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V_{PP} (TEST) pin of the KS86P4004/P4104, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 15-3 below.

Table 15-3. Operating Mode Selection Criteria

V_{DD}	VPP (TEST)	REG/MEM	ADDRESS (A15-A0)	R/W	MODE
5 V	5 V	0	0000H	1	EPROM read
	12.5 V	0	0000H	0	EPROM program
	12.5 V	0	0000H	1	EPROM verify
	12.5 V	1	0E3FH	0	EPROM read protection

NOTE: "0" means Low level; "1" means High level.

NOTES

16 DEVELOPMENT TOOLS

OVERVIEW

Samsung provides a powerful and easy-to-use development support system in turnkey form. The development support system is configured with a host system, debugging tools, and support software. For the host system, any standard computer that operates with MS-DOS as its operating system can be used. One type of debugging tool including hardware and software is provided: the sophisticated and powerful in-circuit emulator, SMDS2+, for KS57, KS86, KS88 families of microcontrollers. The SMDS2+ is a new and improved version of SMDS2. Samsung also offers support software that includes debugger, assembler, and a program for setting options.

SHINE

Samsung Host Interface for in-circuit Emulator, SHINE, is a multi-window based debugger for SMDS2+. SHINE provides pull-down and pop-up menus, mouse support, function/hot keys, and context-sensitive hyper-linked help. It has an advanced, multiple-windowed user interface that emphasizes ease of use. Each window can be sized, moved, scrolled, highlighted, added, or removed completely.

SAMA ASSEMBLER

The Samsung Arrangeable Microcontroller (SAM) Assembler, SAMA, is a universal assembler, and generates object code in standard hexadecimal format. Assembled program code includes the object code that is used for ROM data and required SMDS program control data. To assemble programs, SAMA requires a source file and an auxiliary definition (DEF) file with device specific information.

SASM86

The SASM86 is an relocatable assembler for Samsung's KS86-series microcontrollers. The SASM86 takes a source file containing assembly language statements and translates into a corresponding source code, object code and comments. The SASM86 supports macros and conditional assembly. It runs on the MS-DOS operating system. It produces the relocatable object code only, so the user should link object file. Object files can be linked with other object files and loaded into memory.

HEX2ROM

HEX2ROM file generates ROM code from HEX file which has been produced by assembler. ROM code must be needed to fabricate a microcontroller which has a mask ROM. When generating the ROM code (.OBJ file) by HEX2ROM, the value "FF" is filled into the unused ROM area upto the maximum ROM size of the target device automatically.

TARGET BOARDS

Target boards are available for all KS86-series microcontrollers. All required target system cables and adapters are included with the device-specific target board.

OTPs

One times programmable microcontrollers (OTPs) are under development for KS86C4004/C4104 microcontroller.

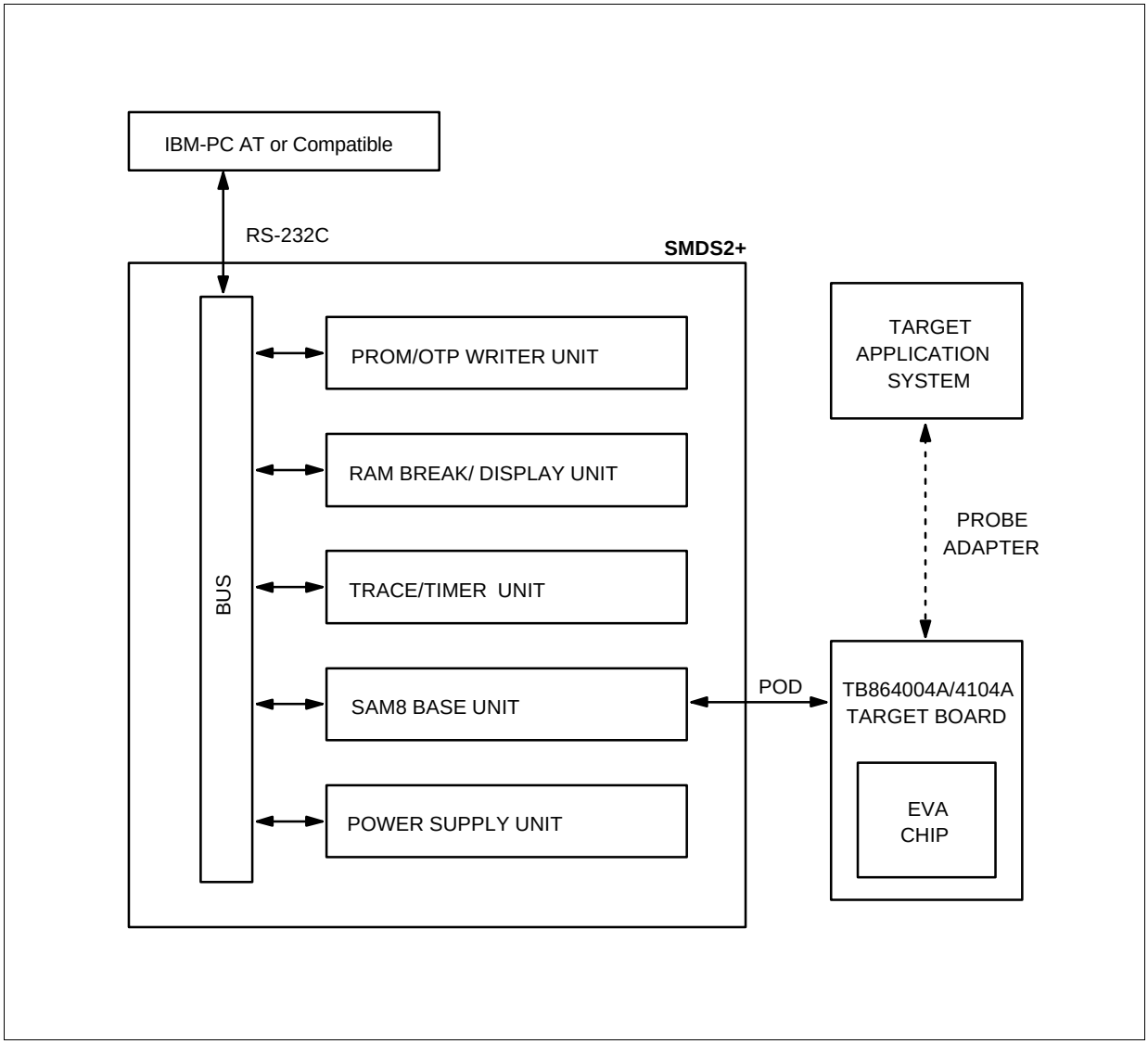


Figure 16-1. SMDS Product Configuration (SMDS2+)

TB864004A/4104A TARGET BOARD

The TB864004A/4104A target board is used for the KS86C4004/C4104 microcontrollers. It is supported by the SMDS2+ development systems. The TB864004A/4104A target board can also be used for KS86C4004/C4104.

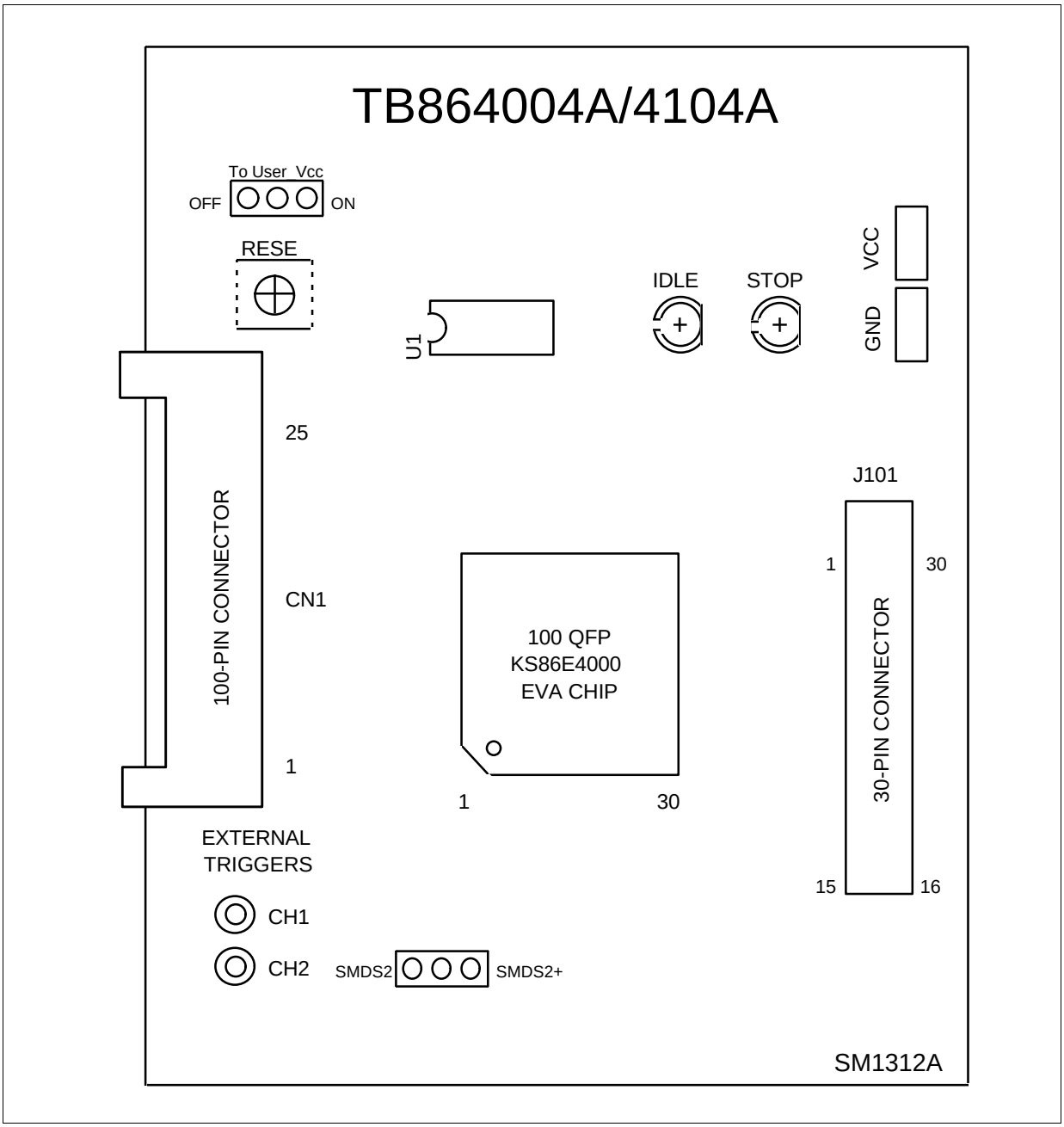
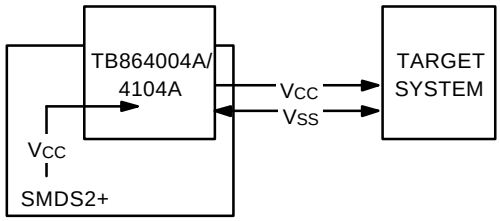
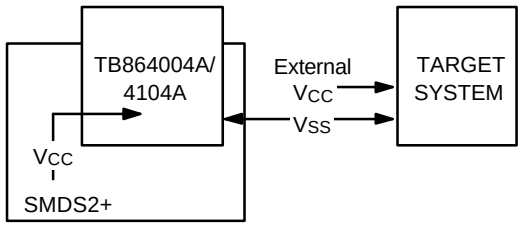


Figure 16-2. TB864004A/4104A Target Board Configuration

Table 16-1. Power Selection Settings for TB864004A/4104A

"To User_Vcc" Settings	Operating Mode	Comments
To User_Vcc OFF ON		The SMDS2+ main board supplies V_{CC} to the target board (evaluation chip) and the target system.
To User_Vcc OFF ON		The SMDS2+ main board supplies V_{CC} only to the target board (evaluation chip). The target system must have its own power supply.

NOTE: The following symbol in the "To User_Vcc" Setting column indicates the electrical short (off) configuration:

SMDS2+ Selection (SAM8)

In order to write data into program memory that is available in SMDS2+, the target board should be selected to be for SMDS2+ through a switch as follows. Otherwise, the program memory writing function is not available.

Table 16-2. The SMDS2+ Tool Selection Setting

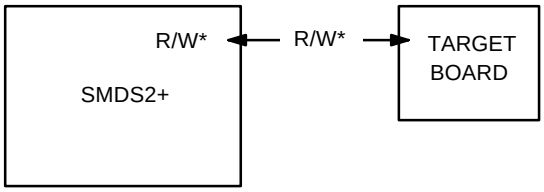
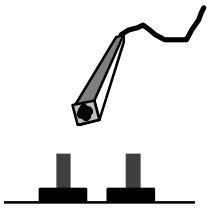
"SW1" Setting	Operating Mode
SMDS2 SMDS2+	

Table 16-3. Using Single Header Pins as the Input Path for External Trigger Sources

Target Board Part	Comments
EXTERNAL TRIGGERS CH1 CH2	 Connector from external trigger sources of the application system You can connect an external trigger source to one of the two external trigger channels (CH1 or CH2) for the SMDS2+ breakpoint and trace functions.

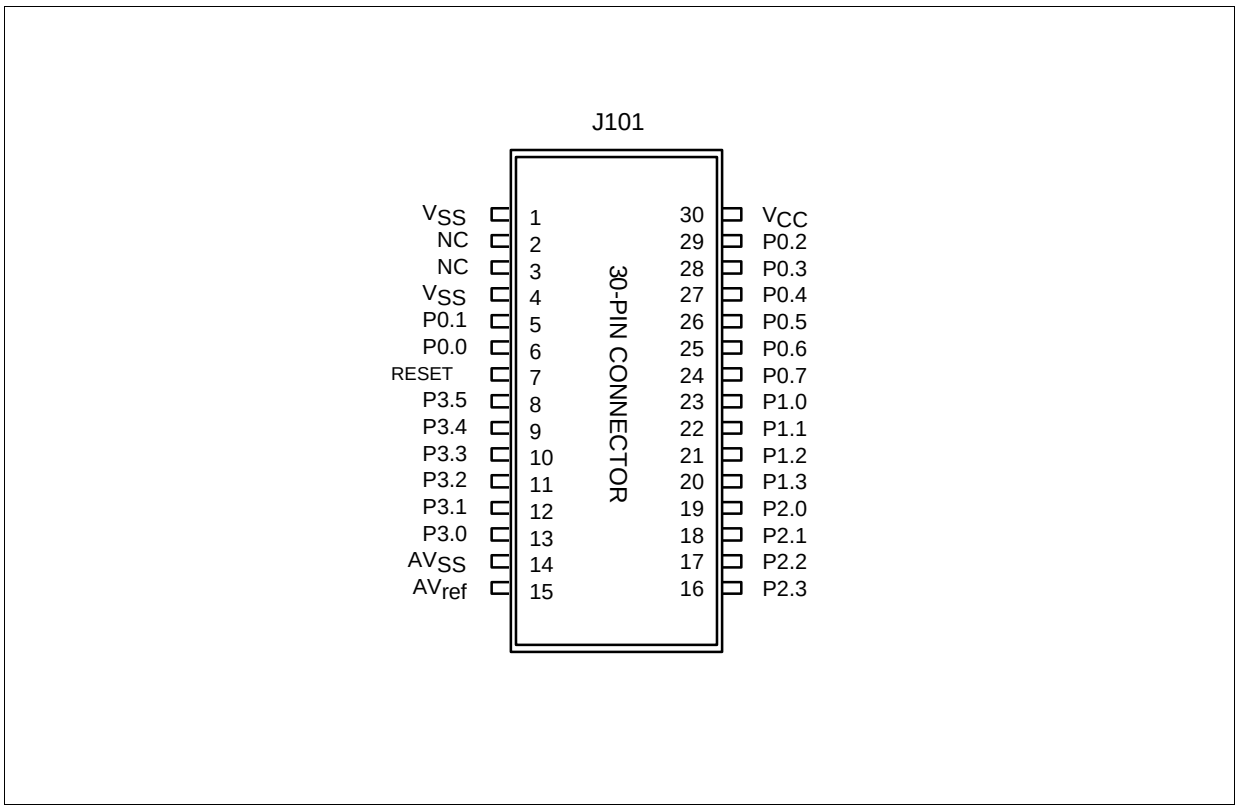


Figure 16-3. 30-Pin Connector for TB864004A/4104A

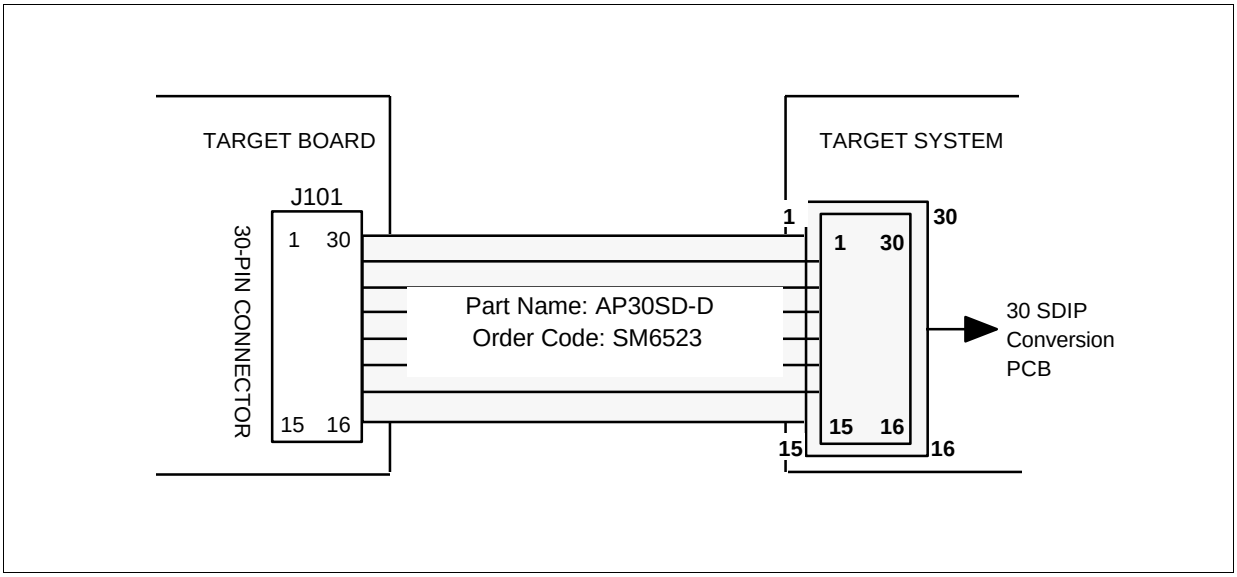


Figure 16-4. KS86C4004 Probe Adapter for 30-SDIP Package

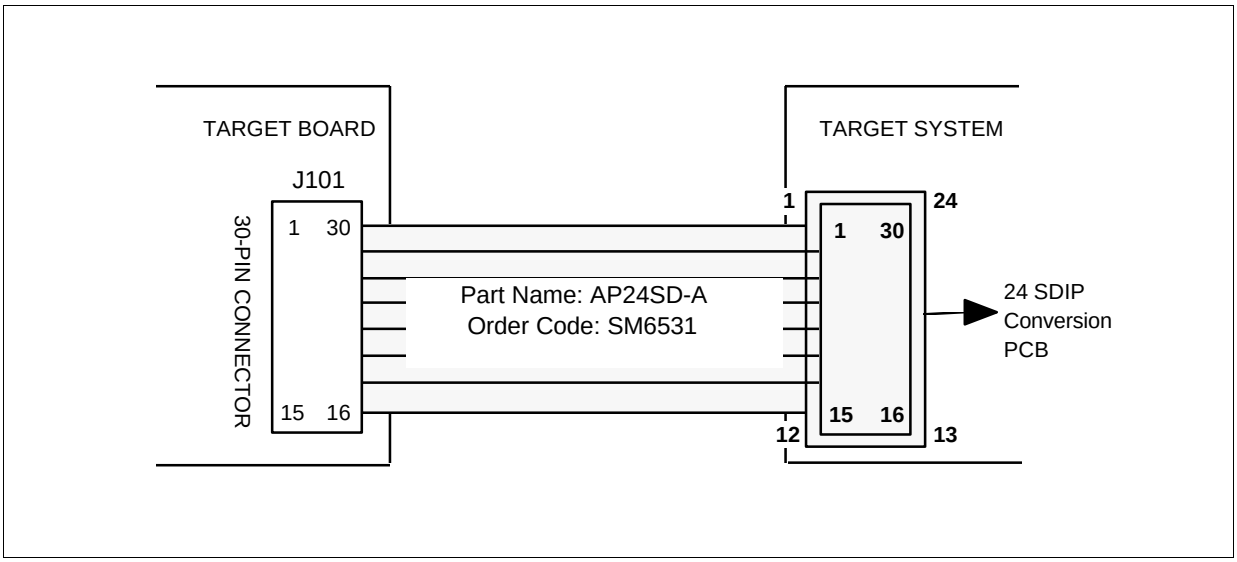


Figure 16-5. KS86C4104 Probe Adapter for 24-SDIP Package

NOTES