SAMSUNG ELECTRONICS

KS88C8016
8-Bit CMOS Microcontroller

Product Specification

OVERVIEW

The KS88C8016 single-chip 8-bit microcontroller is fabricated using a highly advanced CMOS process. Its fast and reliable CPU is based on Zilog's Super8® architecture. Important features include two 8-bit timer/ counters, 16-bit and 20-bit timer/counter arrays with data capture and compare functions, two-channel PWM, A/Dconverter, and a serial I/O module. The KS88C8016 is a powerful and flexible design solution for a wide range of general-purpose applications.

FEATURES

CPU
- SAM8 CPU core

Memory
- 336-byte internal register file
- 16-Kbyte program memory

Instruction Set
- 79 instructions
- IDLE and STOP instructions added for power-down modes

Instruction Execution Time
- 500 ns at 12 MHz f_{OSC} (min.)

Interrupts
- 30 interrupt sources
- 16 interrupt vectors
- Eight interrupt levels
- Fast interrupt processing for one level
- Non-maskable interrupt

Clock Oscillation Circuit
- Maximum 12-MHz CPU clock

I/O Ports
- Five I/O ports (total 40 pins):
- Three nibble-programmable ports
- Two bit-programmable ports for external interrupts

Timer/Counters
- Two 8-bit timer/counters
- 16-bit timer/counter array (two modules):

  16-bit capture/compare module

  Pattern generator module
- 20-bit timer/counter array with six modules:

  Three 16-bit capture modules

  Two16-bit capture/compare modules

  16-bit pattern generator

Serial I/O Interface
- One pin for serial data I/O
- One pin for serial clock I/O
- Selectable transmit and receive rates

A/D Converter
- Eight pins for analog or normal input
- 8-bit digital converter resolution
- 24-µs conversion speed with 8-MHz CPU clock

Pulse Width Modulation
- Two pins for PWM output
- Frequency: 11.72-46.88 kHz at 12 MHz
- 14-bit resolution (8-bit frame)
- Push-pull circuit type

Sync Signal Processing
- V-sync separation from C-sync input
- Pseudo H-sync pattern generator output

Backup Timer
- 16-bit clock update timer

Operating Temperature Range
- – 20°C to + 85°C

Operating Voltage Range
- 4.5 V to 6.0 V

Package Type
- 80-pin QFP

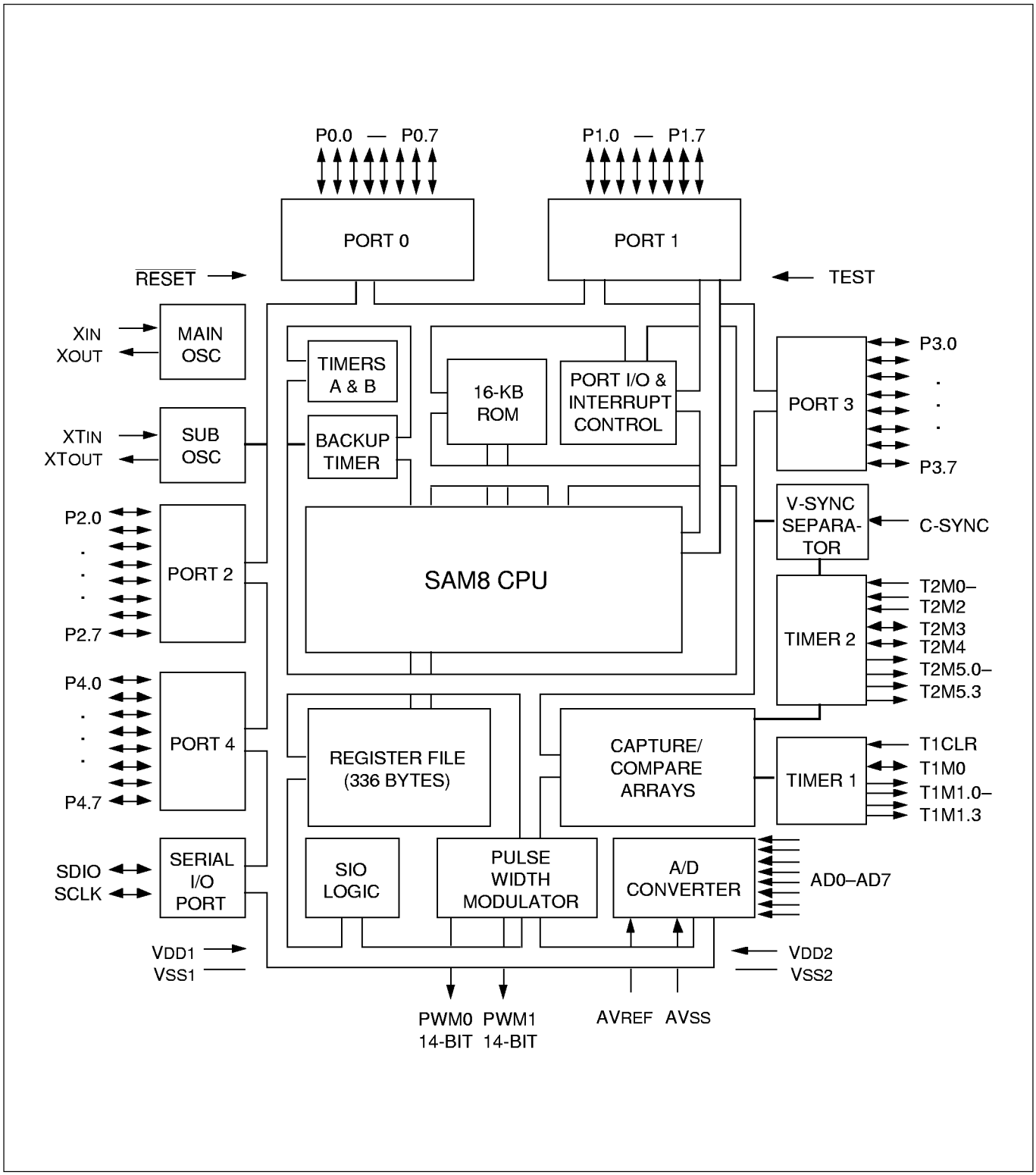


Figure 1. KS88C8016 Block Diagram

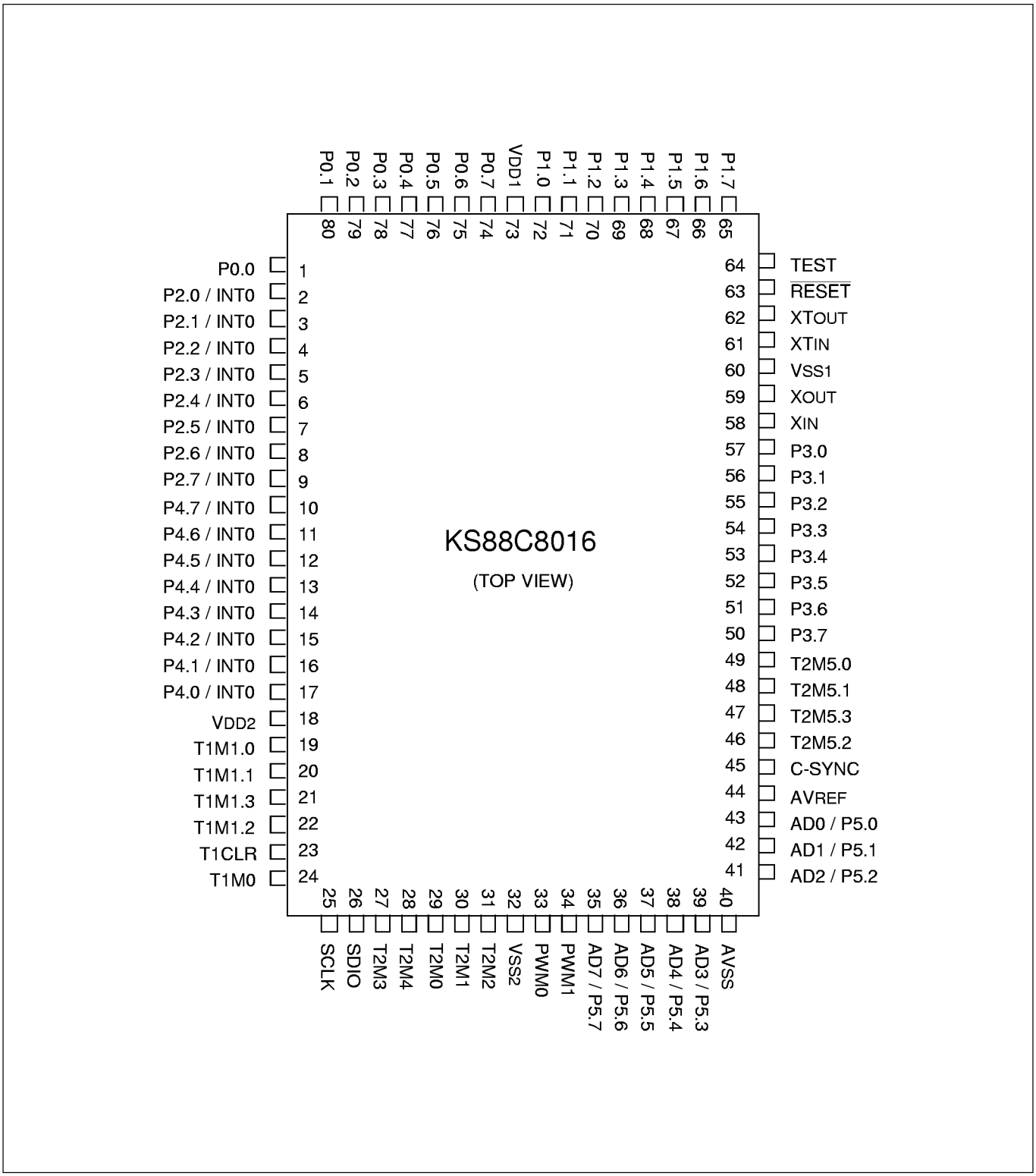


Figure 2. KS88C8016 Pin Assignments (80-Pin QFP)

Table 1. KS88C8016 Pin Descriptions

Pin Name	Pin Type	Pin Description	Circuit Type	Pin Number	Share Pins
P0.0–P0.7	I/O	General I/O port with nibble-programmable pins; input or output mode and pull-ups are software-assignable; output circuit type for normal current only	3	1, 80–74	–
P1.0–P1.7	I/O	General I/O port; same as port 0, but with up to 8 V high current drive capability	3'	72–65	–
P2.0–P2.7 P4.0–P4.7	I/O	General I/O port; input or output mode and pull-ups are software-assignable; alternately, bit-assignable (pin-by-pin) for external interrupts; interrupt enable and pending capability; noise filters	4	2–9, 17–10	–
P3.0–P3.7	I/O	Same as port 0	3	57–50	–
P5.0–P5.7	I	General input port; alternately, for analog data input to the A/D converter	6	43–41, 39–35	AD0–AD2, AD3–AD7
T1M1.0– T1M1.3	O	Timer1 module1 (pattern generator) compare output pins; alternately, for normal 4-bit data output	5	19–22	–
T1CLR	I	External clear signal input to timer1	1	23	–
T1M0	I/O	Data capture input or compare output for timer 1 module 0 (T1M0)	4	24	–
T2M0–T2M2	I	Data capture input pin for T2M0–T2M2	1	29–31	–
T2M3–T2M4	I/O	Data capture input or compare output for timer 2 module 3 and timer 2 module 4	4	27, 28	–
T2M5.0– T2M5.3	O	Timer 2 module 5 compare output pins; alternately, for normal 4-bit data output	5	49–46	–
PWM0, PWM1	O	14-bit pulse width modulator output pins for the PWM0 and PWM1 signals	5	33, 34	–
C-SYNC	I	C-sync input for V-sync signal separation	1	45	–
AD0–AD7	I	Analog input pin for A/D converter; alternately, for use as general input port 5	6	43–41, 39–35	P5.0–P5.2, P5.3–P5.7
AV_{REF}, AV_{SS}	–	Reference voltage inputs for A/D converter	–	44, 40	–
SCLK	I/O	Bi-directional serial data clock pin	4	25	–
SDIO	I/O	Bi-directional serial data pin	4	26	–
X_{IN}, X_{OUT}	–	System clock input and output pins	–	58, 59	–
XT_{IN}, XT_{OUT}	–	Suboscillator clock pins for backup timer	–	61, 62	–
V_{DD2}, V_{SS2}	–	Power input pins for port output (external)	–	18, 32	–
V_{DD1}, V_{SS1}	–	Power input pins for CPU (internal)	–	73, 60	–
$\overline{RESET}$	I	System reset pin (pull-up resistor: 220 kΩ)	2	63	–
TEST	I	Test signal input (connect to V_{SS})	–	64	–

Table 2. Pin Circuit Assignments for the KS88C8016

Circuit Number	Circuit Type	KS88C8016 Assignments
1	Input	T1CLR, T2M0–T2M2, C-SYNC
2	Input	$\overline{\text{RESET}}$
3	I/O	Port 0 (P0.0–P0.7), port 3 (P3.0–P3.7)
3'	I/O	Port 1 (P1.0–P1.7); high current drive capability
4	I/O	Port 2 (P2.0–P2.7), port 4 (P4.0–P4.7)
5	Output	T1M1.0–T1M1.3, T2M5.0–T2M5.3, PWM0, PWM1
6	Input	P5.0–P5.7, AD0–AD7 (share pins)

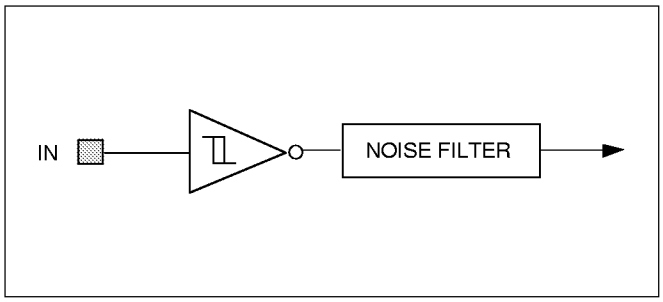


Figure 3. Pin Circuit Type 1 (T1CLR, T2M0–T2M2, C-Sync)

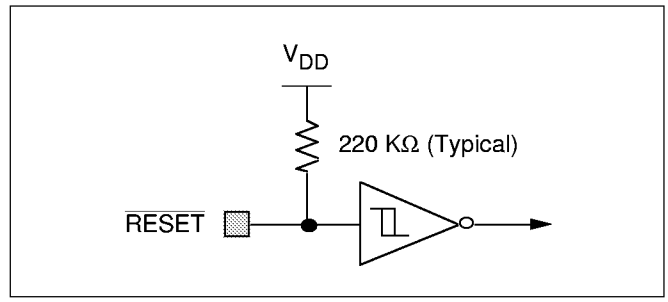


Figure 4. Pin Circuit Type 2 ($\overline{\text{RESET}}$ Pin)

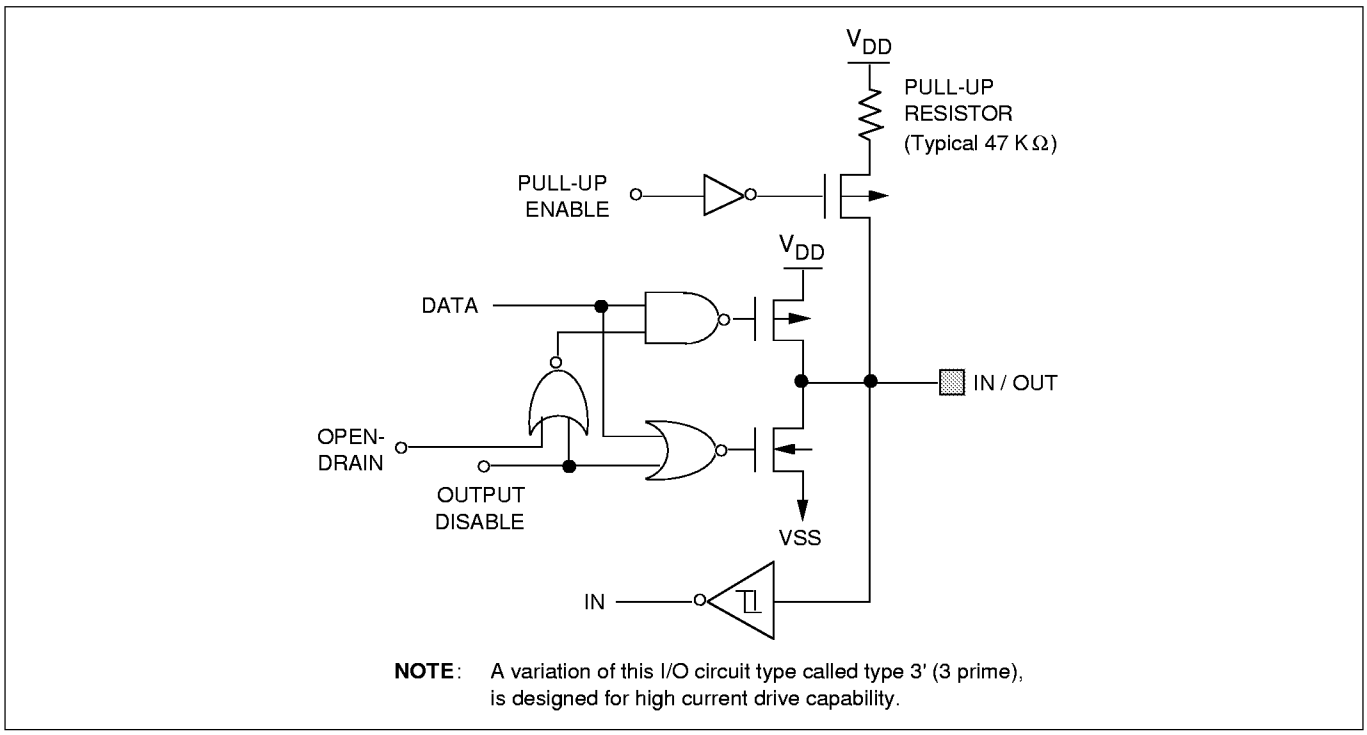


Figure 5. Pin Circuit Type 3 (Type 3: Ports 0 and 3; Type 3': Port 1)

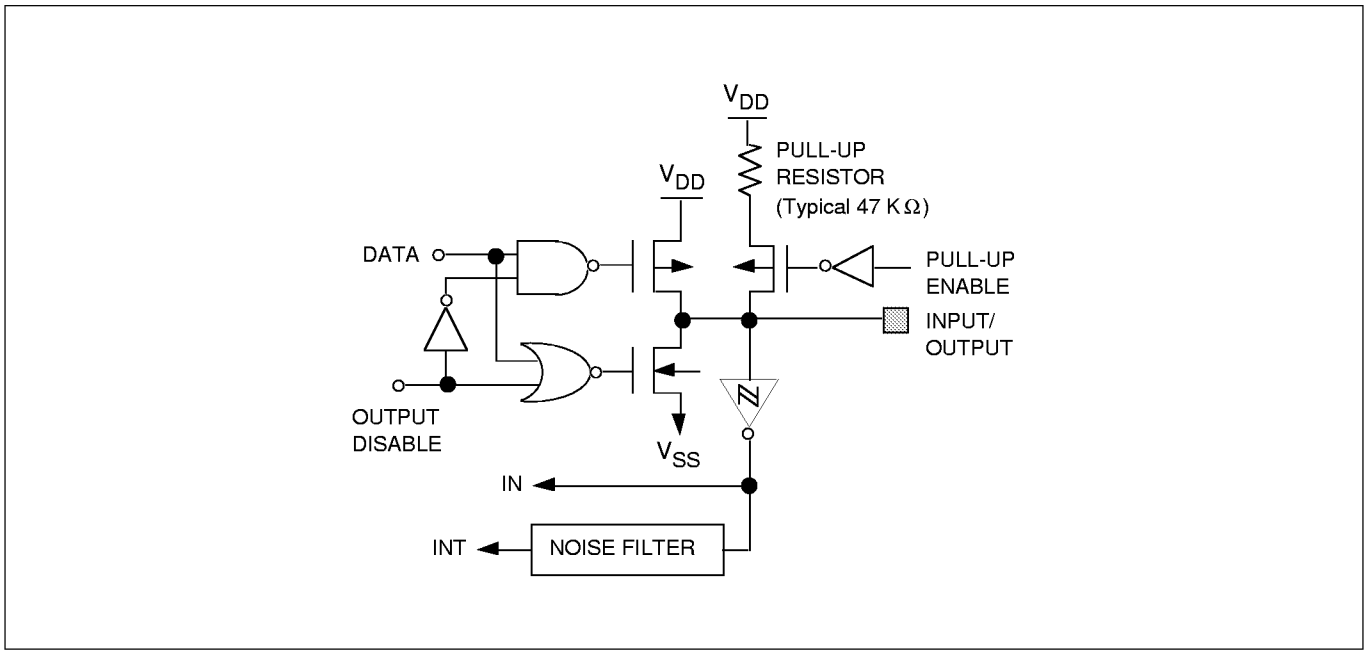


Figure 6. Pin Circuit Type 4 (Ports 2 and 4)

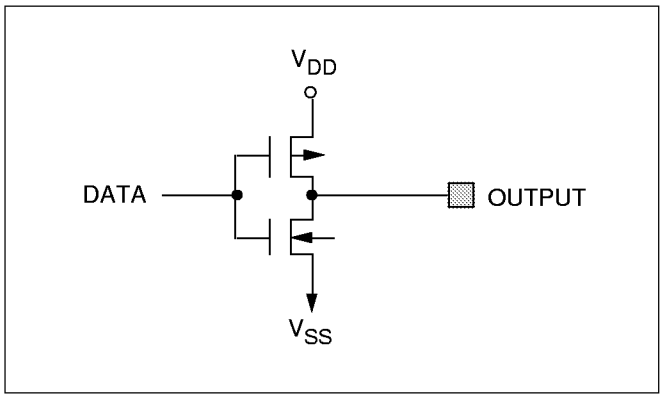


Figure 7. Pin Circuit Type 5 (T1M1.0–T1M1.3, T2M5.0–T2M5.3, PWM0, PWM1)

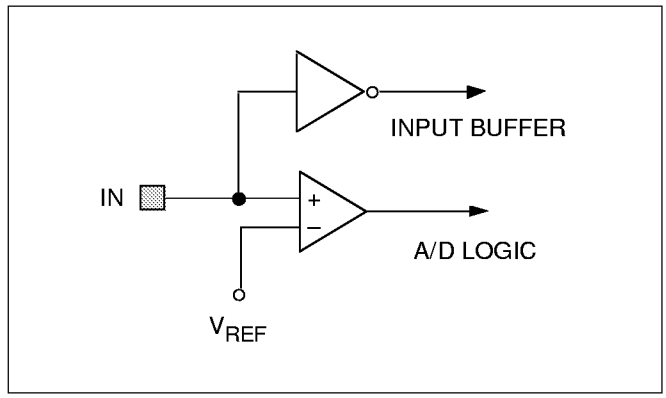


Figure 8. Pin Circuit Type 6 (AD0–AD7, Port 5)

ADDRESS SPACES

OVERVIEW

The KS88C8016 microcontroller has two kinds of address space:

— Registers (internal register file)

— Program memory (ROM)

A 16-bit address bus and an 8-bit data bus supports program memory and data memory operations. A separate 8-bit address bus and the 8-bit data bus carryaddresses and data between the CPU and the register file.

Internal Register File

Thirteen bytes in the internal register file are for system control registers, and 73 bytes for peripheral control and data registers. There are 336 general-purpose registers, including the 16-bit working register area.

Program Memory (ROM)

The KS88C8016 has a 16-Kbyte mask-programmable ROM (0H–3FFFH). Program memory (ROM) stores program code and table data. Instructions can be fetched, or data read, from ROM locations.

The SAM8 interrupt structure supports up to 127 vector addresses. The KS88C8016 interrupt structure uses only 16 vectors. The first 256 bytes of the ROM (0H–FFH) are used for this maximum number of vectors. Unused locations in this address range can be used as normal program memory. The reset address is 0020H.

If you store program code in the vector address area, be careful to avoid overwriting vector addresses stored in this area of the program memory.

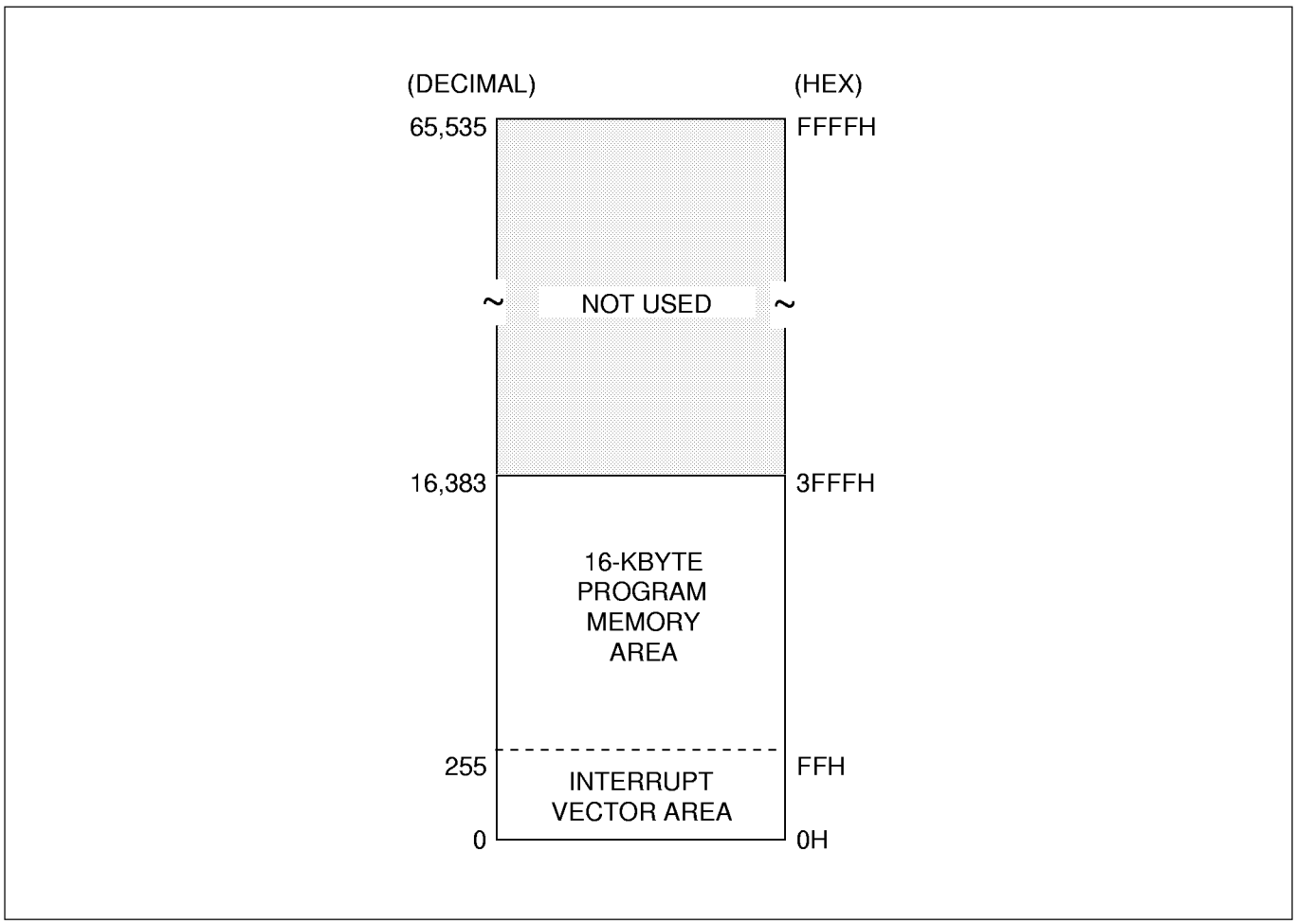


Figure 9. KS88C8016 Program Memory Map

REGISTER ARCHITECTURE

Internal Register File

The 256-byte internal register file is logically extended to duplicate the upper 64-byte area of the register file, as well as the lower 192-byte area, by a factor of two.

This extension into separately addressable register sets, banks, and pages is supported by set bank instructions, the register page pointer, and addressing mode restrictions.

The total addressable register space is expanded from 256 bytes to 1120 bytes. The KS88C8016 can access 422 8-bit registers in this maximum 1120-byte space.

Register Page Pointer (PP)

The register page pointer (PP, DFH) controls page addressing. Bit setting 'x00B' selects page 0 and 'x01B' selects page 1. (Pages 2 and 3 are not used for the KS88C8016.)

Reset clears the page pointer to '000B', selecting page 0.

When you set the page pointer value to select page 1, the entire 256-byte address range (00H–FFH) is swapped. Register addresses on the previously selected page are replaced by those of the new page.

Stack operations in page 0 are handled independently of the page pointer.

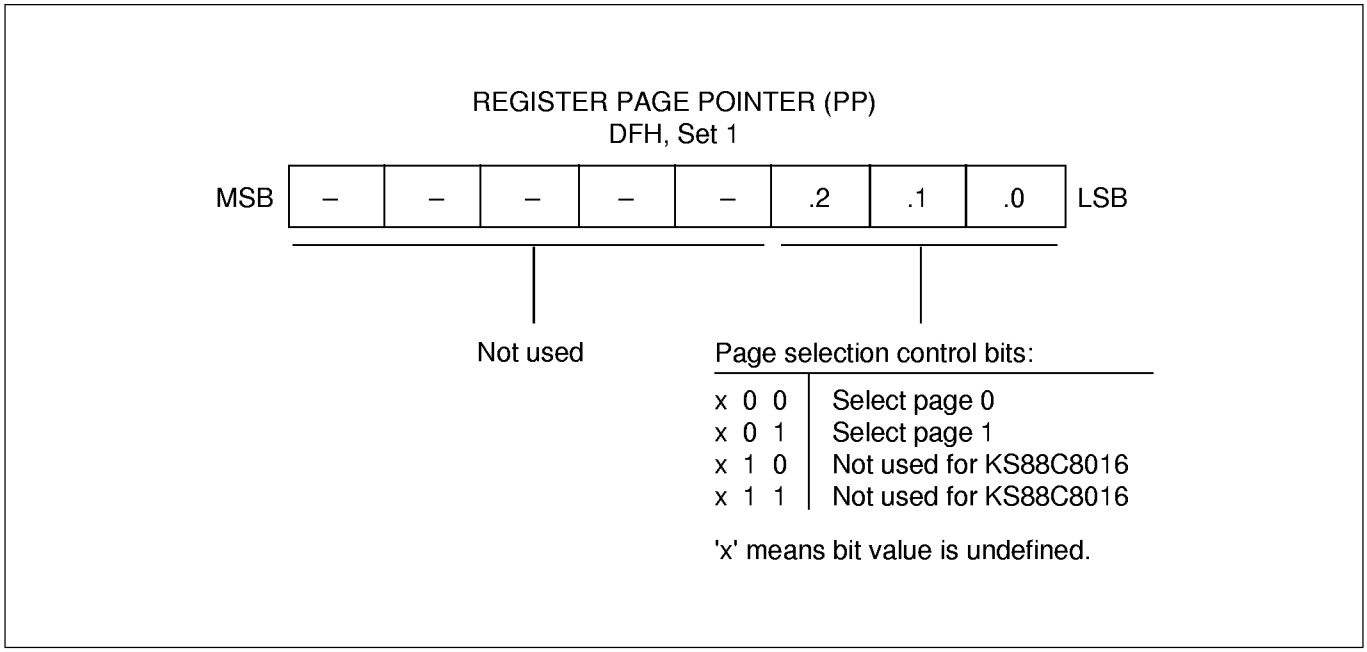


Figure 10. Register Page Pointer (PP)

Register Set 1

The term *set 1* refers to the upper 64-byte space of the register file, locations C0H–FFH. The upper 32-byte space within this 64-byte space is divided into two register banks.

The upper 32-byte area of set 1 (E0H–FFH) is divided into two 32-byte register banks called *bank 0* and *bank 1*. Banks 0 and 1 are addressable using the set register bank instructions, SB0 and SB1. Whenever a reset occurs, bank 0 addressing is automatically selected.

The lower 32-byte area of set 1 is not banked. This non-banked area is called *common area* because it can always be accessed, regardless of which page is pointed to by the register page pointer (PP).

The non-banked 32-byte area in register set 1 contains 16 bytes for mapped system registers (D0H–DFH) and a 16-byte general-purpose common area (C0H–CFH).

Locations in the 16-byte working register area can be used as temporary buffers to enable register data to be transferred between different pages.

Registers in the banked and non-banked areas of set 1 are always directly accessible using the Register addressing mode, regardless of the current page selection. However, the 16-byte working register area can only be accessed using working register addressing.

Register Set 2

The same 64-byte physical space that is used for set 1 register locations C0H–FFH is logically duplicated to add 64 bytes of register space. This expanded area of the register file is called *set 2*.

Addressing mode limitations maintain the division of set 1 and set 2: Set 1 can be accessed using Register addressing mode only, and set 2 can be accessed only using the Register Indirect addressing mode and the Indexed addressing mode.

For the KS88C8016, the set 2 register area is located in page 0. There is no set 2 area in page1 because only locations 00H–7FH are mapped.

192-Byte Prime Register Space

The lower 192 bytes of each 256-byte page (00H–BFH) is called the *prime register area* because registers in these locations can be accessed using all addressing modes, both direct and indirect.

In the KS88C8016 register file, locations 00H–3FH in the prime area of page 0 are used for mapped peripheral control and data registers. The remaining page 0 locations (40H–BFH), and 00H–7FH of the page 1 prime register area, are for general-purpose use.

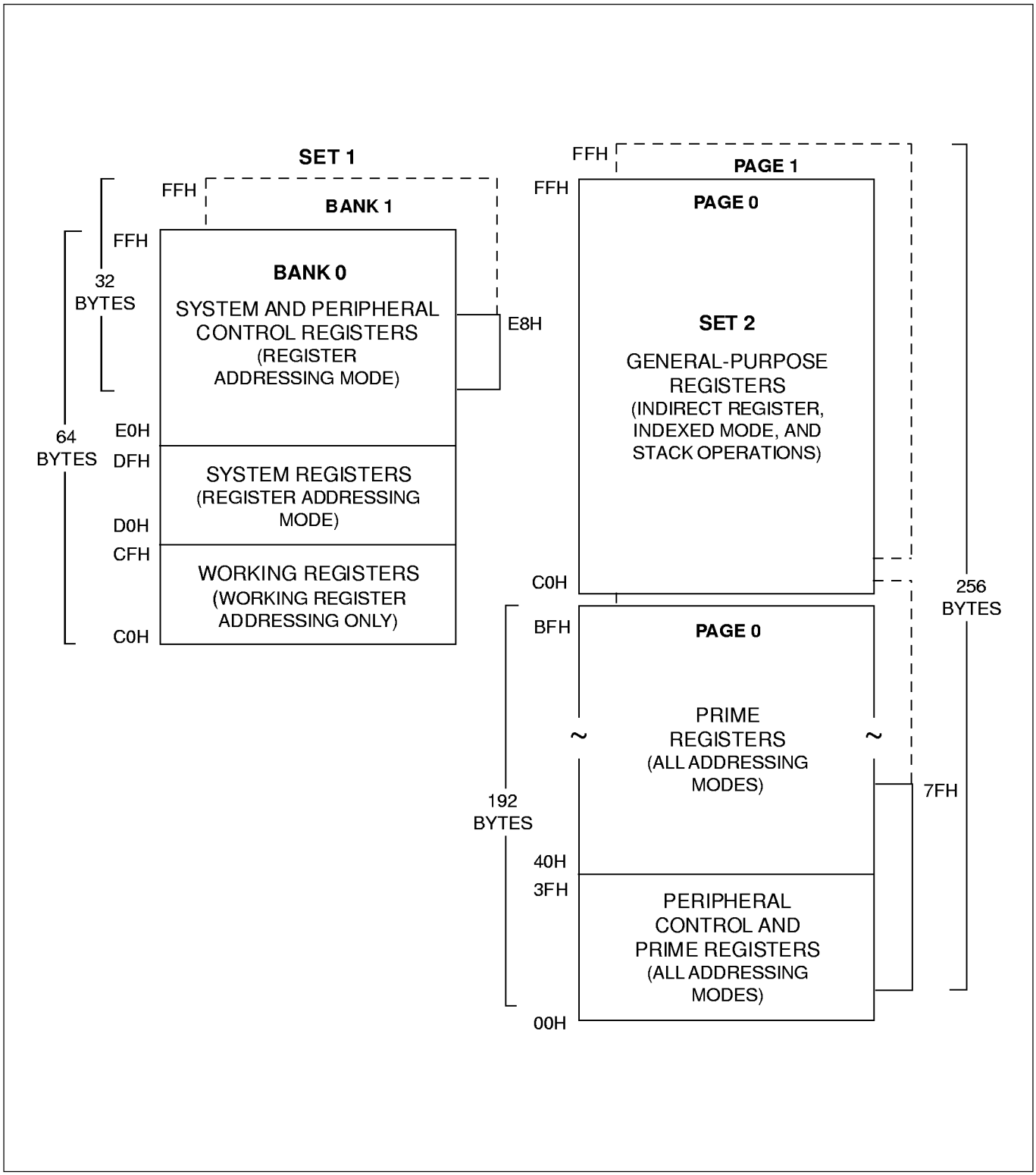


Figure 11. KS88C8016 Register File

CONTROL REGISTERS

Table 3. KS88C8016 Set 1 Registers

Register Name	Mnemonic	Decimal	Hex	Location	R/W
System Flags Register	FLAGS	213	D5H	Set 1	R/W
Register Pointer 0	RP0	214	D6H	Set 1	R/W
Register Pointer 1	RP1	215	D7H	Set 1	R/W
Stack Pointer (High Byte)	SPH	216	D8H	Set 1	R/W
Stack Pointer (Low Byte)	SPL	217	D9H	Set 1	R/W
Instruction Pointer (High Byte)	IPH	218	DAH	Set 1	R/W
Instruction Pointer (Low Byte)	IPL	219	DBH	Set 1	R/W
Interrupt Request Register	IRQ	220	DCH	Set 1	R
Interrupt Mask Register	IMR	221	DDH	Set 1	R/W
System Mode Register	SYM	222	DEH	Set 1	R/W
Register Page Pointer	PP	223	DFH	Set 1	R/W

Table 4. KS88C8016 Set 1, Bank 0 Registers

Register Name	Mnemonic	Decimal	Hex	Location	R/W
Port 0 Data Register	P0	224	E0H	Set 1, Bank 0	R/W
Port 1 Data Register	P1	225	E1H	Set 1, Bank 0	R/W
Port 2 Data Register	P2	226	E2H	Set 1, Bank 0	R/W
Port 0 Control Register	P0CON	227	E3H	Set 1, Bank 0	R/W
Port 1 Control Register	P1CON	228	E4H	Set 1, Bank 0	R/W
Port 2 Control Register (High Byte)	P2CONH	229	E5H	Set 1, Bank 0	R/W
Port 2 Control Register (Low Byte)	P2CONL	230	E6H	Set 1, Bank 0	R/W
Port 2 Interrupt Enable Register	P2INT	231	E7H	Set 1, Bank 0	R/W
Port 2 Interrupt Pending Register	P2PND	232	E8H	Set 1, Bank 0	R/W
Serial I/O Shift Register	SIO	233	E9H	Set 1, Bank 0	R/W
Serial I/O Control Register	SIOCON	234	EAH	Set 1, Bank 0	W
Serial I/O Prescaler	SIOPS	235	EBH	Set 1, Bank 0	W
Timer A Data Register	TADATA	236	ECH	Set 1, Bank 0	W
Timer B Data Register	TBDATA	237	EDH	Set 1, Bank 0	W
Timer 0 (A and B) Control Register	T0CON	238	EEH	Set 1, Bank 0	W*
Timer B Interrupt Enable Register	TBINT	239	EFH	Set 1, Bank 0	W
Backup Timer Control Register	BTCON	240	F0H	Set 1, Bank 0	R/W
External Memory Timing Register	EMT	254	FEH	Set 1, Bank 0	R/W
Interrupt Priority Register	IPR	255	FFH	Set 1, Bank 0	R/W

* Bit 1 of the T0C0N register can also be read.

Table 5. KS88C8016 Set 1, Bank 1, Registers

Register Name	Mnemonic	Decimal	Hex	Location	R/W
Port 3 Data Register	P3	224	E0H	Set 1, Bank 1	R/W
Port 4 Data Register	P4	226	E2H	Set 1, Bank 1	R/W
Port 3 Control Register	P3CON	227	E3H	Set 1, Bank 1	R/W
Port 4 Control Register (High Byte)	P4CONH	229	E5H	Set 1, Bank 1	R/W
Port 4 Control Register (Low Byte)	P4CONL	230	E6H	Set 1, Bank 1	R/W
Port 4 Interrupt Enable Register	P4INT	231	E7H	Set 1, Bank 1	R/W
Port 4 Interrupt Pending Register	P4PND	232	E8H	Set 1, Bank 1	R/W

Table 6. KS88C8016 Page 0 Registers

Register Name	Mnemonic	Decimal	Hex	R/W
Timer 2 Module 4 Prescaler	T2M4PS	0	00H	W
Timer 2 Module 5 Control Register	T2M5CON	1	01H	W
Timer 2 Module 5 Compare Register (High Byte)	T2M5H	2	02H	W
Timer 2 Module 5 Compare Register (Low Byte)	T2M5L	3	03H	W
Timer 2 Module 5 Output Register	T2M5OUT	4	04H	W
Timer 2 Module 3 Prescaler	T2M3PS	5	05H	W
Timer 2 Module 5 Pattern Generator Register	T2M5PG	6	06H	W
Timer 1 Control Register	T1CON	7	07H	W
Timer 1 Prescaler	T1PS	8	08H	W
Timer 1 Module 0 Control Register	T1M0CON	9	09H	W
Timer 1 Module 0 Com/Cap Register (High Byte)	T1M0H	10	0AH	R/W
Timer 1 Module 0 Com/Cap Register (Low Byte)	T1M0L	11	0BH	R/W
Timer 1 Module 0 Toggle Register (LSB)	T1M0TGL	12	0CH	W
Timer 1 Module 1 Output Register	T1M1OUT	13	0DH	W
Timer 1 Module 1 Control Register	T1M1CON	14	0EH	W
Timer 1 Module 1 Compare Register (High Byte)	T1M1H	15	0FH	W
Timer 1 Module 1 Compare Register (Low Byte)	T1M1L	16	10H	W
Timer 1 Module 1 Pattern Generator Register	T1M1PG	17	11H	W
Timer 2 4-Bit Extension Counter	T2EX	18	12H	R
Timer 2 Control Register	T2CON	19	13H	W
Timer 2 Module 0 Prescaler	T2M0PS	20	14H	W
Timer 2 Module 1 Prescaler	T2M1PS	21	15H	W
Timer 2 Module 2 Prescaler	T2M2PS	22	16H	W
Timer 2 Module 0 Control Register	T2M0CON	23	17H	W
Timer 2 Module 1 Control Register	T2M1CON	24	18H	W
Timer 2 Module 2 Control Register	T2M2CON	25	19H	W
Timer 2 Module 0 Capture Register (High Byte)	T2M0H	26	1AH	R
Timer 2 Module 0 Capture Register (Low Byte)	T2M0L	27	1BH	R
Timer 2 Module 1 Capture Register (High Byte)	T2M1H	28	1CH	R
Timer 2 Module 1 Capture Register (Low Byte)	T2M1L	29	1DH	R

Table 6. KS88C8016 Page 0 Registers (Continued)

Register Name	Mnemonic	Decimal	Hex	R/W
Timer 2 Module 2 Capture Register (High Byte)	T2M2H	30	1EH	R
Timer 2 Module 2 Capture Register (Low Byte)	T2M2L	31	1FH	R
Timer 2 Module 3 Control Register	T2M3CON	32	20H	W
Timer 2 Module 3 Com/Cap Register (High Byte)	T2M3H	33	21H	R/W
Timer 2 Module 3 Com/Cap Register (Low Byte)	T2M3L	34	22H	R/W
Timer 2 Module 3 Toggle Register (LSB)	T2M3TGL	35	23H	W
Timer 2 Module 4 Control Register	T2M4CON	36	24H	W
Timer 2 Module 4 Com/Cap Register (High Byte)	T2M4H	37	25H	R/W
Timer 2 Module 4 Com/Cap Register (Low Byte)	T2M4L	38	26H	R/W
Timer 1 Module 0 Prescaler	T1M0PS	39	27H	W
A/D Converter Control Register	ADCON	40	28H	R/W
A/D Digital Input Register (Port 5)	P5 (ADIN)	41	29H	R
A/D Conversion Data Output Register	ADDATA	42	2AH	R
PWM0 Data Register	PWM0	44	2CH	W
PWM0 Data Register (Extension Byte)	PWM0EX	45	2DH	W
PWM1 Data Register	PWM1	46	2EH	W
PWM1 Data Register (Extension Byte)	PWM1EX	47	2FH	W
PWM Control Register	PWMCON	60	3CH	W

☞ PROGRAMMING TIP — Using Load Instructions for Read-Only and Write-Only Registers

The instructions OR (Logical OR), AND (Logical AND), CP (Compare), and LDB (Load Bit) should not be used to access write-only or read-only registers. Use load (LD) instructions instead (except for LDB). Here are some examples:

Example 1:

```
        OR      T0CON,#04H      ;  Invalid use of logical OR instruction!
```

Use a LD instruction instead to manipulate the T0CON register:

```
        BITS    ST0CON.2        ;  ST0CON is a shadow register for T0CON
        LD      T0CON,ST0CON    ;  Set bit 2 in the T0CON register
```

Example 2:

```
        CP      PWMCON,#3CH     ;  Invalid use of the CP instruction!
        JP      EQ,AAA
        •
        •
        •
AAA     NOP
```

Use a shadow register instead to manipulate the PWMCON register:

```
        CP      SPWMCON,#3CH    ;  SPWMCON is a shadow register for PWMCON
        JP      EQ,AAA
        •
        •
        •
AAA     NOP
```

Example 3:

```
        LDB     SIOCON.0,R0     ;  Invalid use of the LDB instruction!
```

Use a shadow register instead to load the value into the SIOCON register:

```
        LDB     SSIOCON.0,R0    ;  SSIOCON is a shadow register for SIOCON
        LD      SIOCON,SSIOCON
```

INTERRUPT STRUCTURE

OVERVIEW

The KS88C8016 microcontroller has thirty peripheral interrupt sources. Sixteen vectors support these sources and the interrupt structure uses all eight interrupt levels (IRQ0–IRQ7). The non-maskable interrupt generated by timer B always has highest priority.

When multiple interrupt levels are active, the interrupt priority register (IPR) determines the order in which contending interrupts are to be serviced. If multiple interrupts occur on the same interrupt level, the interrupt with the lowest vector address is processed first.

An interrupt machine cycle begins when an interrupt request is granted. This disables all subsequent interrupts, saves the program counter and status flags, and branches to the program memory vector location reserved for that interrupt.

INTERRUPT VECTOR ADDRESSES

The KS88C8016 interrupt vector addresses are stored in the first 256 bytes of the ROM. The reset address is 0020H.

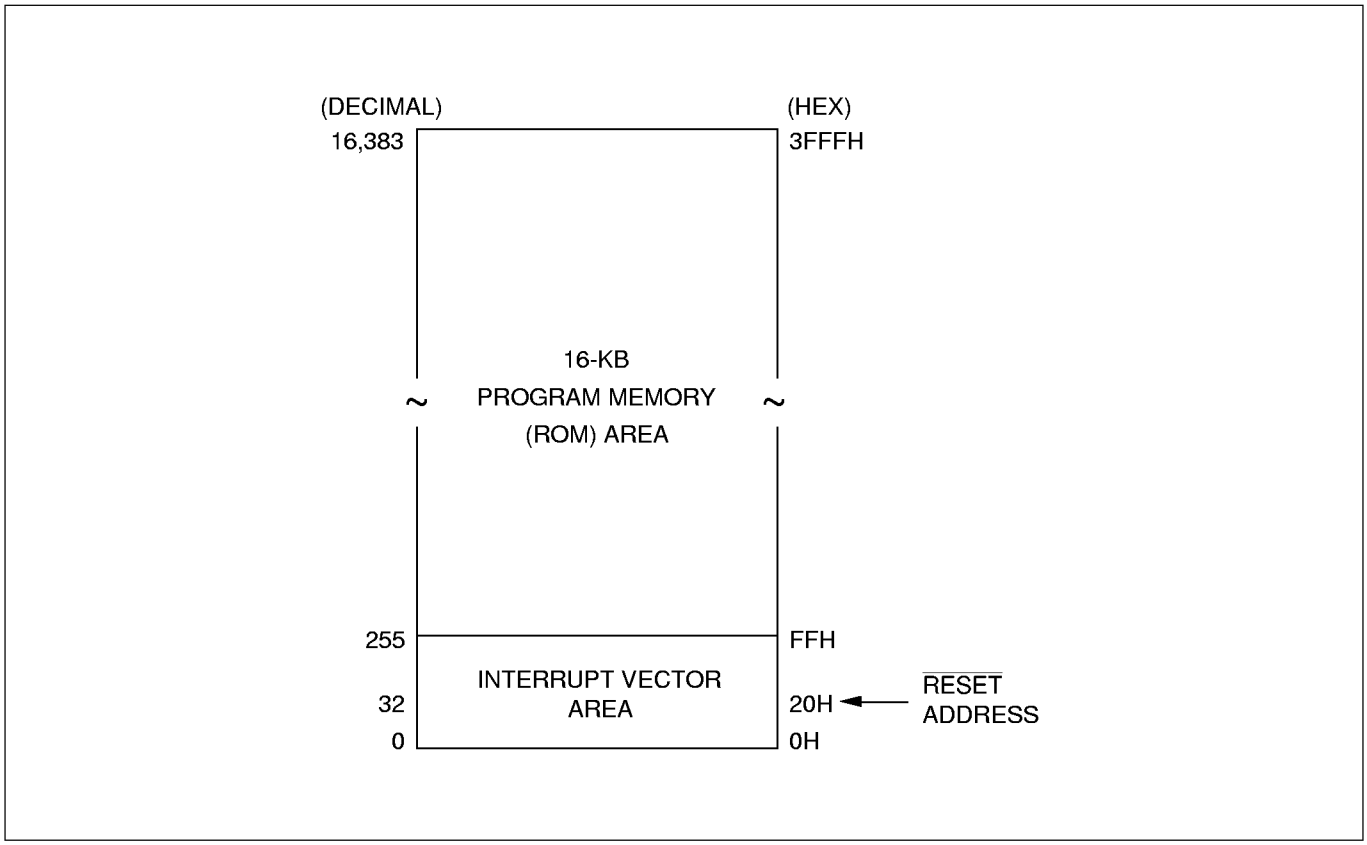


Figure 12. ROM Vector Address Area

Table 7. KS88C8016 Interrupt Vectors

Vector Address		Interrupt Source	Request		Reset/Clear	
Decimal Value	Hex Value		Interrupt Level	Priority in Level	H/W	S/W
254	FEH	Timer B interrupt	NMI	Highest	–	–
252	FCH	Timer 2 counter overflow	IRQ4	–	√	
250	FAH	Timer 2 module 5	IRQ3	–	√	
248	F8H	Timer 2 module 0	IRQ2	–	√	
246	F6H	Timer 2 module 1	IRQ1	1	√	
244	F4H	Timer 2 module 3	IRQ1	0		
242	F2H	Timer 2 module 4	IRQ0	1	√	
240	F0H	Timer 2 module 2	IRQ0	0		
232–239	E8H–EFH	*Reserved*	–	–	–	–
230	E6H	Timer 1 counter overflow	IRQ5	3	√	
228	E4H	Timer 1 module 1	IRQ5	2		
226	E2H	Timer 1 module 0	IRQ5	1		
224	E0H	Timer 1 counter clear	IRQ5	0		
196–223	C4H–DFH	*Reserved*	–	–	–	–
192	C0H	External interrupt	IRQ7	–		√
190	BEH	Timer A interrupt	IRQ6	1		√
186–189	BAH–BDH	*Reserved*	–	–		
184	B8H	Serial I/O interrupt	IRQ6	0	√	
34–183	22H–B7H	*Reserved*	–	–	–	–
32	20H	Power-on, $\overline{\text{RESET}}$	–	–	–	–
0–31	00H–1FH	*Reserved*	–	–	–	–

NOTES:
1. Interrupt priorities are identified in inverse order: '0' is highest priority, '1' is the next highest, and so on.
2. If two or more interrupts within the same level contend, the interrupt with the lowest vector address has priority over one with a higher vector address. These priorities within levels are preset at the factory. For example, in interrupt level IRQ6 the highest-priority interrupt vector is the SIO interrupt, B8H; the lowest-priority interrupt within the level is the timer A interrupt, interrupt vector BEH.
3. You can use the non-maskable interrupt (NMI) simultaneously with other interrupts whose pending bits are cleared by hardware. You cannot use the NMI with the external interrupt (IRQ7) and the timer A interrupt (IRQ6), whose pending bits must be cleared by software.

ENABLE/DISABLE INTERRUPT INSTRUCTIONS (EI, DI)

The Enable Interrupts (EI) instruction globally enables the KS88 interrupt structure. This allows any interrupts to be serviced when they occur, according to their assigned priority.

If an interrupt pending bit was previously enabled by its source, its interrupt will be serviced when the EI is executed.

An EI instruction must always be executed as part of the system initialization routine that follows a reset. The DI (Disable Interrupt) instruction can be executed at any time to globally disable interrupt processing.

During normal operation, the LSB of the system mode register (SYM.0) can be manipulated to dynamically enable and disable interrupt processing.

SYSTEM-LEVEL INTERRUPT CONTROL REGISTERS

Four system-level control registers, together with peripheral source control registers, regulate interrupt processing:

— Each interrupt level, except for the NMI, is enabled or disabled (masked) according to the bit settings in the interrupt mask register (IMR).

— The interrupt priority register (IPR) controls the relative priorities of interrupt request levels.

— The interrupt request register (IRQ) contains an interrupt pending status bit for each level.

— The system mode register (SYM) dynamically enables or disables globally interrupt processing, enables fast interrupt processing for select interrupt levels, and enables or disables the tri-state external memory interface. The external interface is not used for the KS88C8016 microcontroller.

INSTRUCTION POINTER (IP)

The instruction pointer (IP) is a system register that is used to control optional high-speed interrupt processing called *fast interrupts*. The KS88C8016 supports fast interrupt processing for level 7 interrupts (IRQ7) only.

The IP consists of register pair DAH and DBH. DAH contains the most significant (high) byte of a memory address and DBH the least significant (low) byte.

The IP registers are called IPH (high byte, IP15–IP8) and IPL (low byte, IP7–IP0).

Table 8. Interrupt Control Register Overview

Control Register	ID	R/W	Function Description
System mode register	SYM	R/W	Dynamic global interrupt enable/disable and fast interrupt processing (the tri-state external memory interface control bit is not used).
Interrupt mask register	IMR	R/W	Bit settings in the IMR register enable or disable interrupt processing for each interrupt level.
Interrupt priority register	IPR	R/W	Controls the relative high-level processing priorities of the interrupt levels. The eight levels are organized into three groups: A, B, and C. Group A is IRQ0 and IRQ1, group B is IRQ2–IRQ4, and group C is IRQ5–IRQ7.
Interrupt request register	IRQ	R	This register contains a request pending bit for each interrupt level (IRQ0–IRQ7).

SYSTEM MODE REGISTER (SYM)

The system mode register, SYM (DEH, set 1) enables or disables interrupt processing at the CPU level and controls fast interrupt processing.

SYM.0 is the global enable bit for interrupt processing. SYM.1–SYM.4 control fast interrupt processing: SYM.1 is the enable bit and SYM.2–SYM.4 are the fast interrupt level selection bits. SYM.5–SYM.6 are not mapped.

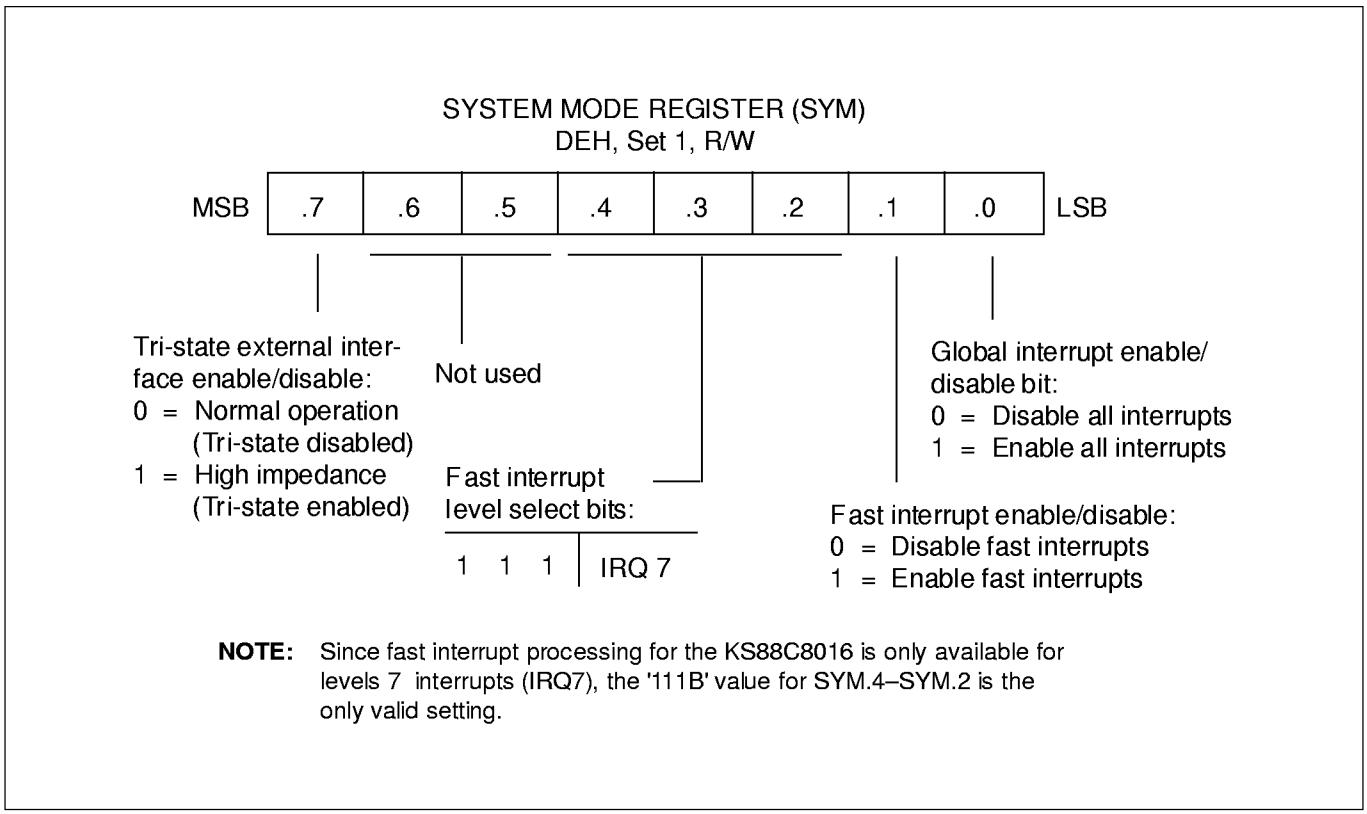
SYM.7 is the enable bit for the tri-state external memory interface. Because the interface is not used for the KS88C8016 implementation, SYM.7 must always be "0" (interface disabled).

Following reset, SYM.0, SYM.1, and SYM.7 are cleared to "0" and other bit values are undetermined.

A program's initialization routine must first execute an Enable Interrupt (EI) instruction to enable global interrupt processing. SYM.0 can then be manipulated during normal operation to enable or disable interrupts.

Fast interrupt processing can only be used level 7 (IRQ7) interrupts. If, however, the SIO interrupt is not enabled, you can then use the timer A interrupt (IRQ6) as a fast interrupt.)

Remember that a fast interrupt and the timer B non-maskable interrupt cannot be active concurrently; you must select either fast interrupt processing for IRQ7 or the timer B interrupt.



NOTE: Since fast interrupt processing for the KS88C8016 is only available for levels 7 interrupts (IRQ7), the '111B' value for SYM.4–SYM.2 is the only valid setting.

Figure 13. System Mode Register (SYM)

INTERRUPT MASK REGISTER (IMR)

The interrupt mask register (IMR) enables or disables interrupt processing for each interrupt level. Each IMR bit corresponds to a specific interrupt level. The IMR register is at register location DDH in set 1.

When the IMR bit of an interrupt level is "0", interrupt processing for that level is disabled (masked). When the IMR bit for a level is "1", interrupt processing for the level is enabled (not masked).

Before the IMR register can be written, interrupt processing must be globally disabled by clearing SYM.0. Following reset, all IMR register values are undetermined.

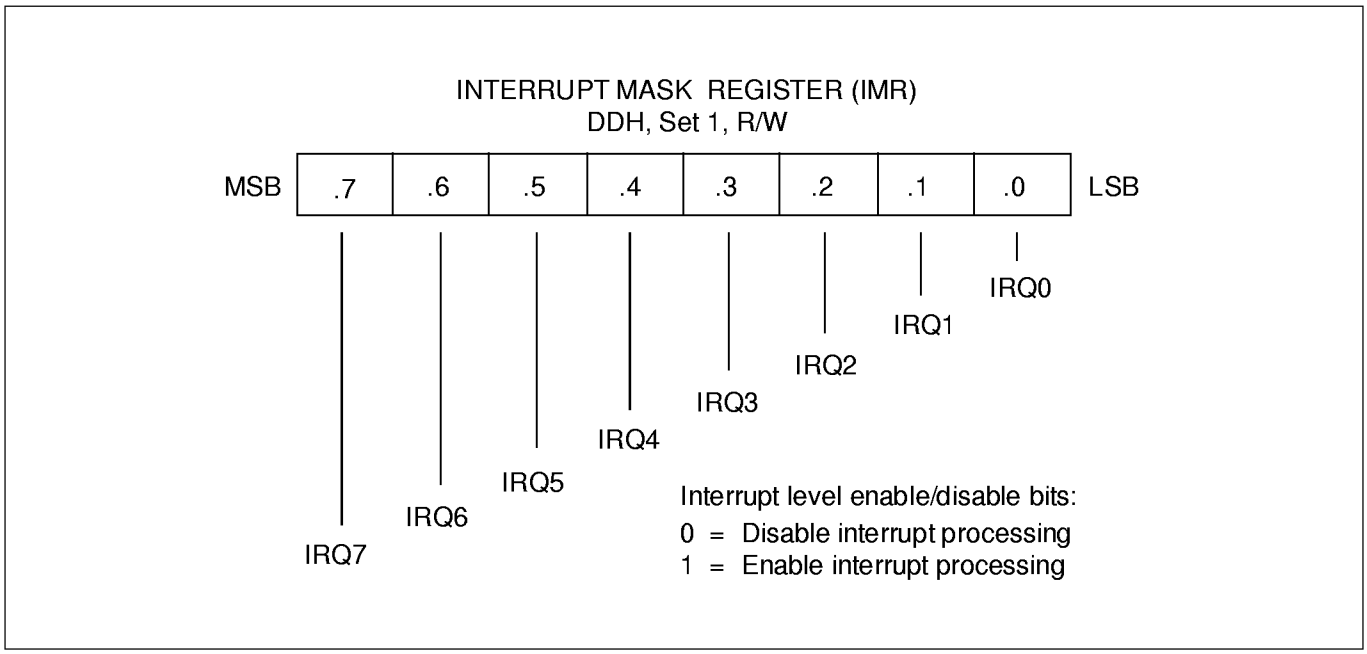


Figure 14. Interrupt Mask Register (IMR)

INTERRUPT PRIORITY REGISTER (IPR)

The interrupt priority register (IPR) sets the relative priorities of the eight interrupt levels. To define these priorities, interrupt levels are organized into groups and subgroups by the interrupt logic. (These groups and subgroups are used solely for the IPR register priority definitions.)

The IPR logic defines three interrupt groups:

Group A IRQ0, IRQ1

Group B IRQ2, IRQ3, IRQ4

Group C IRQ5, IRQ6, IRQ7

The IPR register is mapped to register location FFH in set 1. Before the IPR register can be written, interrupt processing must first be globally disabled by clearing SYM.0. Following a reset operation, IPR register values are undetermined.

IPR bit 0 controls the relative priority of IRQ0 and IRQ1 interrupts. Either IRQ0 > IRQ1 or IRQ1 > IRQ0. IPR bit 2 controls interrupt group B. The possible relationships are expressed as IRQ2 > (IRQ3, IRQ4) or (IRQ3, IRQ4) > IRQ2.

Bit 5 of the IPR register controls the relative priorities of group C interrupts: IRQ5 > (IRQ6, IRQ7) or (IRQ6, IRQ7) > IRQ5. Interrupt groups B and C have subgroups to provide an additional priority relationship between interrupt levels. For subgroup B, IPR bit 3 defines these relationships as either IRQ3 > IRQ4 or as IRQ4 > IRQ3. Bit 6 of the IPR register controls subgroup C relationships: IRQ6 > IRQ7 or IRQ7 > IRQ6.

Bits 7, 4, and 1 of the IPR register provide additional priority logic. This three-bit setting controls the relative priority of interrupt groups A, B, and C. For example, bit setting '001B' selects the group relationship B > C > A, while the setting '101B' selects C > B > A.

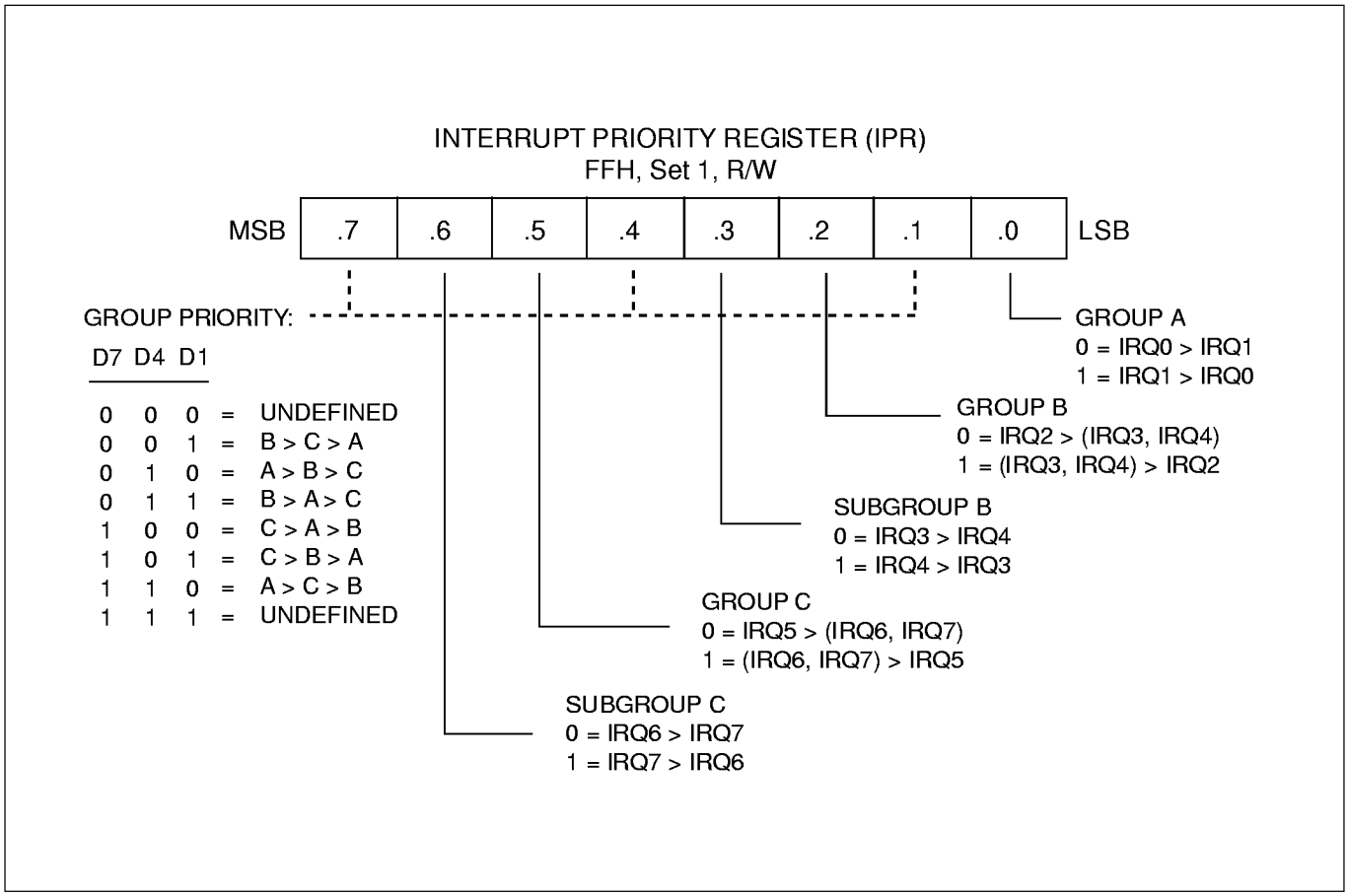


Figure 15. Interrupt Priority Register (IPR)

INTERRUPT REQUEST REGISTER (IRQ)

The interrupt request register (IRQ) contains an interrupt pending bit for each interrupt level. It is mapped to register location DCH in set 1. A "0" value means that no interrupt request is pending; a "1" means that an interrupt request for that level is pending.

To determine the current request status of an interrupt level, the IRQ register can be read (tested) at any time. Reset clears the IRQ register to '00H'.

Software can use the IRQ register to poll interrupt levels that are not controlled by hardware or that have been temporarily masked by IMR register settings. When polling the interrupt request bits, application software must clear the pending bit when the source's interrupt has been serviced.

Writing the corresponding IRQ bit has no effect because the interrupt request must be acknowledged at the source. For the KS88C8016, the interrupt source may be timer A interrupt or an external interrupt at port 2 or port 4.

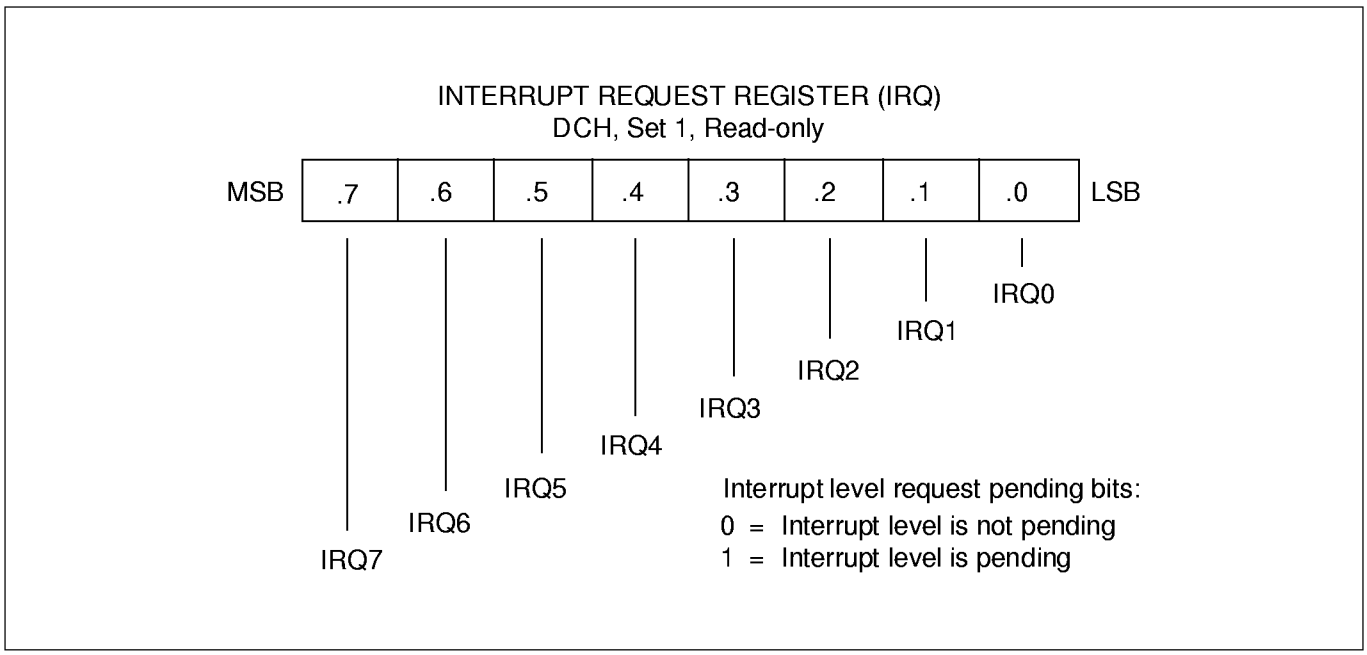


Figure 16. Interrupt Request Register (IRQ)

FAST INTERRUPT PROCESSING

The feature called *fast interrupt processing* completes specific interrupt service routines in a minimum time of six clock cycles instead of the usual 22 cycles.

Two other system control registers support fast interrupts:

— The instruction pointer (IP) holds the starting address of the service routine and saves the PC values, and

— A dedicated register, FLAGS', saves the contents of the FLAGS register when a fast interrupt occurs.

☞ PROGRAMMING TIP — Setting Up the Interrupt Control Structure

This example shows how to enable interrupts for select interrupt sources, disable interrupts for other sources, and to set interrupt priorities for the KS88C8016. The program sequence does the following:

— Enable interrupts for port 4 pins P4.4–P4.7 and timer A, clear timer module 1

— Disable NMI, timer module 1 overflow, backup timer, and P4.0–P4.3 interrupts

— Set interrupt priorities as P4.4–P4.7 > timer A > timer C

```
•
•
•
DI                          ;  Disable interrupts
LD      TBINT,#0AAH         ;  Disable non-maskable interrupt (NMI)
LD      IMR,#85H            ;  IRQ0, IRQ2, IRQ7 are selected
LD      IPR,#80H            ;  IRQ7 > IRQ0 > IRQ2
LD      TADATA,#0FH         ;
LD      T0CON,#86H          ;  Timer A interrupt enable
LD      T1CON,#16H          ;  Start counter 1, enable counter 1 clear interrupt
                            ;  Disable timer module 1 overflow interrupt
SB1                         ;  Select bank 1
LD      P4CONH,#55H         ;  Input, rising edge interrupt selection
LD      P4PND,#0FFH         ;  Reset all port 4 pending registers
LD      P4INT,#0F0H         ;  Enable interrupts for P4.4–P4.7
SB0                         ;  Select bank 0
•
•
•
EI                          ;  Enable interrupts
```

Assuming interrupt sources and priorities are set by the above instruction sequence, it is possible to select interrupt level 7 for fast interrupt processing. (Fast interrupt processing may be selected since the non-maskable interrupt is disabled.) The following instructions enable fast interrupts for level 7 (IRQ7):

```
DI                          ;  Disable interrupts
LDW     IPH,#3000H          ;  Load the service routine address for IRQ7
LD      SYM,#1EH            ;  Enable fast interrupt processing
EI                          ;  Enable interrupts
```

OSCILLATOR CIRCUITS

OVERVIEW

An external crystal generates a maximum 12-MHz CPU clock. The X_{IN} and X_{OUT} pins connect the external oscillation source to the on-chip clock circuit.

MAIN OSCILLATOR CIRCUIT

The main oscillator circuit generates the CPU clock signal. To increase processing speed and to reduce noise levels, the clock circuit has non-divided logic.

SUBOSCILLATOR CIRCUIT FOR BACKUP TIMER

An on-chip suboscillator circuit runs the backup timer during normal operation and in stop mode. If a secondary power source is used, it can run the backup timer during a power outage. It requires a separate external oscillation source with a typical operating frequency of 32768 Hz.

The suboscillator circuit provides a high-resolution (accurate to the second) clock signal to the backup timer module. During a power outage, the backup timer circuit continues operating until power is restored.

An application can be written for real-time timing/counting during a power failure. When the outage occurs, the program clears the backup timer and counting starts as the CPU enters Stop mode.

When power is restored, the CPU "wakes up" and checks the backup timer to see how long it has been "asleep." The backup timer 'time save' value is then added to the stopped main clock value to give the current time.

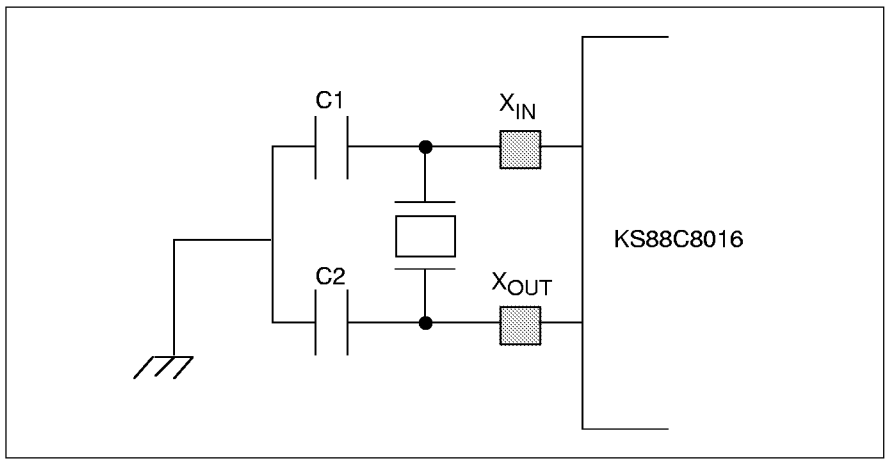


Figure 17. Main Oscillator Circuit (With External Crystal or Resonator)

CLOCK STATUS DURING POWER-DOWN MODES

The two power-down modes, Stop and Idle, affect system clock oscillation as follows: During Stop mode, the main oscillator stops, but contents of the internal register file and special function registers are retained. Stop mode can be released by a reset operation only.

In Idle mode, the internal clock signal is gated off to the CPU, but not to the interrupts, timers, and serial port functions. The current CPU status — including stack pointer, program counter, and flags, and data contained in the internal register file — is retained. Idle mode can be released by an interrupt or by a reset.

$\overline{\text{RESET}}$

A reset operation overrides all other operating conditions and puts the KS88C8016 into a known state. For a reset, the signal level at the $\overline{\text{RESET}}$ pin is held to low level for at least 22 CPU clocks.

The $\overline{\text{RESET}}$ signal is input through a Schmitt trigger circuit and is then synchronized with the CPU clock.

The following events occur during a reset operation:

— All interrupts are disabled.

— Ports 0–4 are set to input mode.

— Peripheral control and data registers are disabled and reset to their initial control values.

— The program counter is loaded with the ROM's reset address, 0020H.

— Eight clocks after $\overline{\text{RESET}}$ goes high, the instruction is fetched from ROM location 0020H and executed.

For power-up, the $\overline{\text{RESET}}$ input must be held low for about 30 milliseconds after the power supply comes within tolerance to allow sufficient time for oscillation stabilization.

Table 9. Set 1 Register Values After RESET

Register Name	Mnemonic	Address		Bit Values After RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
System Flags Register	FLAGS	213	D5H	x	x	x	x	x	x	0	0
Register Pointer 0	RP0	214	D6H	1	1	0	0	0	–	–	–
Register Pointer 1	RP1	215	D7H	1	1	0	0	1	–	–	–
Stack Pointer (High Byte)	SPH	216	D8H	x	x	x	x	x	x	x	x
Stack Pointer (Low Byte)	SPL	217	D9H	x	x	x	x	x	x	x	x
Instruction Pointer (High Byte)	IPH	218	DAH	x	x	x	x	x	x	x	x
Instruction Pointer (Low Byte)	IPL	219	DBH	x	x	x	x	x	x	x	x
Interrupt Request Register	IRQ	220	DCH	0	0	0	0	0	0	0	0
Interrupt Mask Register	IMR	221	DDH	x	x	x	x	x	x	x	x
System Mode Register	SYM	222	DEH	0	–	–	x	x	x	0	0
Register Page Pointer	PP	223	DFH	–	–	–	–	–	0	0	0

Table 10. Set 1, Bank 0 Register Values After RESET

Register Name	Mnemonic	Address		Bit Values After RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
Port 0 Data Register	P0	224	E0H	0	0	0	0	0	0	0	0
Port 1 Data Register	P1	225	E1H	0	0	0	0	0	0	0	0
Port 2 Data Register	P2	226	E2H	0	0	0	0	0	0	0	0
Port 0 Control Register	P0CON	227	E3H	0	0	0	0	0	0	0	0
Port 1 Control Register	P1CON	228	E4H	0	0	0	0	0	0	0	0
Port 2 Control Register (High Byte)	P2CONH	229	E5H	0	0	0	0	0	0	0	0
Port 2 Control Register (Low Byte)	P2CONL	230	E6H	0	0	0	0	0	0	0	0
Port 2 Interrupt Enable Register	P2INT	231	E7H	0	0	0	0	0	0	0	0
Port 2 Interrupt Pending Register	P2PND	232	E8H	0	0	0	0	0	0	0	0
Serial I/O Shift Register	SIO	233	E9H	x	x	x	x	x	x	x	x
Serial I/O Control Register	SIOCON	234	EAH	–	–	–	0	0	0	0	0
Serial I/O Prescaler	SIOPS	235	EBH	–	0	0	0	0	0	0	0
Timer A Data Register	TADATA	236	ECH	0	0	0	0	0	0	0	0
Timer B Data Register	TBDATA	237	EDH	0	0	0	0	0	0	0	0
Timer 0 (A and B) Control Register	T0CON	238	EEH	0	0	0	0	0	0	0	0
Timer B Interrupt Enable Register	TBINT	239	EFH	1	0	1	0	1	0	1	0
Backup Timer Control Register	BTCON	240	F0H	x	x	x	x	x	x	x	0
External Memory Timing Register	EMT	254	FEH	0	1	1	1	1	1	0	–
Interrupt Priority Register	IPR	255	FFH	x	x	x	x	x	x	x	x

Table 11. Set 1, Bank 1 Register Values After RESET

Register Name	Mnemonic	Address		Bit Values After RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
Port 3 Data Register	P3	224	E0H	0	0	0	0	0	0	0	0
Port 4 Data Register	P4	226	E2H	0	0	0	0	0	0	0	0
Port 3 Control Register	P3CON	227	E3H	0	0	0	0	0	0	0	0
Port 4 Control Register (High Byte)	P4CONH	229	E5H	0	0	0	0	0	0	0	0
Port 4 Control Register (Low Byte)	P4CONL	230	E6H	0	0	0	0	0	0	0	0
Port 4 Interrupt Enable Register	P4INT	231	E7H	0	0	0	0	0	0	0	0
Port 4 Interrupt Pending Register	P4PND	232	E8H	0	0	0	0	0	0	0	0

Table 12. Page 0 Register Values After RESET

Register Name	Mnemonic	Address		Bit Values After RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
Timer 2 Module 4 Prescaler	T2M4PS	0	00H	0	0	0	0	0	0	0	0
Timer 2 Module 5 Control Register	T2M5CON	1	01H	0	0	0	0	0	0	0	0
Timer 2 Module 5 Compare Register (High)	T2M5H	2	02H	0	0	0	0	0	0	0	0
Timer 2 Module 5 Compare Register (Low)	T2M5L	3	03H	0	0	0	0	0	0	0	0
Timer 2 Module 5 Output	T2M5OUT	4	04H	–	–	–	–	0	0	0	0
Timer 2 Module 3 Prescaler	T2M3PS	5	05H	0	0	0	0	0	0	0	0
Timer 2 Module 5 Pattern Generator	T2M5PG	6	06H	–	–	–	–	0	0	0	0
Timer 1 Control Register (Counter 1)	T1CON	7	07H	–	–	–	0	0	0	0	0
Timer 1 Prescaler	T1PS	8	08H	0	0	0	0	0	0	0	0
Timer 1 Module 0 Control Register	T1M0CON	9	09H	–	–	–	0	–	0	0	0
Timer 1 Module 0 Com/Cap (High)	T1M0H	10	0AH	x	x	x	x	x	x	x	x
Timer 1 Module 0 Com/Cap (Low)	T1M0L	11	0BH	x	x	x	x	x	x	x	x
Timer 1 Module 0 Toggle Register (LSB)	T1M0TGL	12	0CH	–	–	–	–	–	–	–	0
Timer 1 Module 1 Output	T1M1OUT	13	0DH	–	–	–	–	0	0	0	0
Timer 1 Module 1 Control Register	T1M1CON	14	0EH	–	–	0	0	0	0	0	0
Timer 1 Module 1 Compare Register (High)	T1M1H	15	0FH	0	0	0	0	0	0	0	0
Timer 1 Module 1 Compare Register (Low)	T1M1L	16	10H	0	0	0	0	0	0	0	0
Timer 1 Module 1 Pattern Generator	T1M1PG	17	11H	–	–	–	–	0	0	0	0
Timer 2 (Counter 2) 4-Bit Extension	T2EX	18	12H	–	–	–	–	0	0	0	0
Timer 2 Control Register	T2CON	19	13H	–	–	–	–	0	0	0	0
Timer 2 Module 0 Prescaler	T2M0PS	20	14H	0	0	0	0	0	0	0	0
Timer 2 Module 1 Prescaler	T2M1PS	21	15H	0	0	0	0	0	0	0	0
Timer 2 Module 2 Prescaler	T2M2PS	22	16H	0	0	0	0	0	0	0	0
Timer 2 Module 0 Control Register	T2M0CON	23	17H	–	–	–	–	–	0	0	0
Timer 2 Module 1 Control Register	T2M1CON	24	18H	–	–	–	–	–	0	0	0
Timer 2 Module 2 Control Register	T2M2CON	25	19H	–	–	–	–	–	0	0	0

Table 12. Page 0 Register Values After RESET (Continued)

Register Name	Mnemonic	Address		Bit Values After RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
Timer 2 Module 0 Capture Register (High)	T2M0H	26	1AH	x	x	x	x	x	x	x	x
Timer 2 Module 0 Capture Register (Low)	T2M0L	27	1BH	x	x	x	x	x	x	x	x
Timer 2 Module 1 Capture Register (High)	T2M1H	28	1CH	x	x	x	x	x	x	x	x
Timer 2 Module 1 Capture Register (Low)	T2M1L	29	1DH	x	x	x	x	x	x	x	x
Timer 2 Module 2 Capture Register (High)	T2M2H	30	1EH	x	x	x	x	x	x	x	x
Timer 2 Module 2 Capture Register (Low)	T2M2L	31	1FH	x	x	x	x	x	x	x	x
Timer 2 Module 3 Control Register	T2M3CON	32	20H	–	–	–	0	–	0	0	0
Timer 2 Module 3 Com/Cap (High)	T2M3H	33	21H	x	x	x	x	x	x	x	x
Timer 2 Module 3 Com/Cap (Low)	T2M3L	34	22H	x	x	x	x	x	x	x	x
Timer 2 Module 3 Toggle Register (LSB)	T2M3TGL	35	23H	–	–	–	–	–	–	–	0
Timer 2 Module 4 Control Register	T2M4CON	36	24H	–	–	0	0	0	0	0	0
Timer 2 Module 4 Com/Cap (High)	T2M4H	37	25H	x	x	x	x	x	x	x	x
Timer 2 Module 4 Com/Cap (Low)	T2M4L	38	26H	x	x	x	x	x	x	x	x
Timer 1 Module 0 Prescaler	T1M0PS	39	27H	0	0	0	0	0	0	0	0
A/D Converter Control Register	ADCON	40	28H	0	0	0	0	1	–	–	–
A/D Digital Input Register (Port 5)	P5 (ADIN)	41	29H	0	0	0	0	0	0	0	0
A/D Conversion Data Output Register	ADDATA	42	2AH	x	x	x	x	x	x	x	x
PWM0 Data Register	PWM0	44	2CH	0	0	0	0	0	0	0	0
PWM0 Data Register (Extension Byte)	PWM0EX	45	2DH	–	–	0	0	0	0	0	0
PWM1 Data Register	PWM1	46	2EH	0	0	0	0	0	0	0	0
PWM1 Data Register (Extension Byte)	PWM1EX	47	2FH	–	–	0	0	0	0	0	0
PWM Control Register	PWMCON	60	3CH	0	0	0	–	–	–	–	–

POWER-DOWN MODES

IDLE MODE

Idle mode is invoked by the instruction IDLE (opcode 6FH).

In Idle mode, the CPU goes inactive while select peripherals (timer 0, and I/O ports 2 and 4) remain active. Port pins retain the mode (input or output) they had at the time Idle mode was entered. During Idle mode, the contents of CPU and system registers, and other control and data registers. are retained.

There are two ways to release Idle mode:

1. Activate any enabled interrupt, causing Idle mode to be released. The interrupt is then serviced. Following the IRET from the service routine, the instruction immediately following the one that initiated Idle mode will be executed.

   You can use an externally generated interrupt, the timer B non-maskable interrupt, or a fast interrupt to release Idle mode.

2. Execute a reset operation. Because the clock oscillator continues to operate, the reset is completed by hardware. If interrupts are masked, a reset is the only way to release Idle mode.

STOP MODE

The instruction STOP (opcode 7FH) invokes Stop mode. In Stop mode, both the CPU and its peripherals "go to sleep." In other words, the main oscillator stops and all system functions are halted. Data stored in the internal register file and in peripheral control and data registers are retained, however.

The only way to release Stop mode is by a reset. $\overline{\text{RESET}}$ must be held to low level for at least 22clock cycles. All system and peripheral registers are reset to their default values. Data in the register file remains unchanged.

When the signal level at the $\overline{\text{RESET}}$ pin goes high, the CPU executes the program starting from ROM vector address 0020H.

If the backup timer module is connected to a separate suboscillator, the backup timer will remain active during Stop mode.

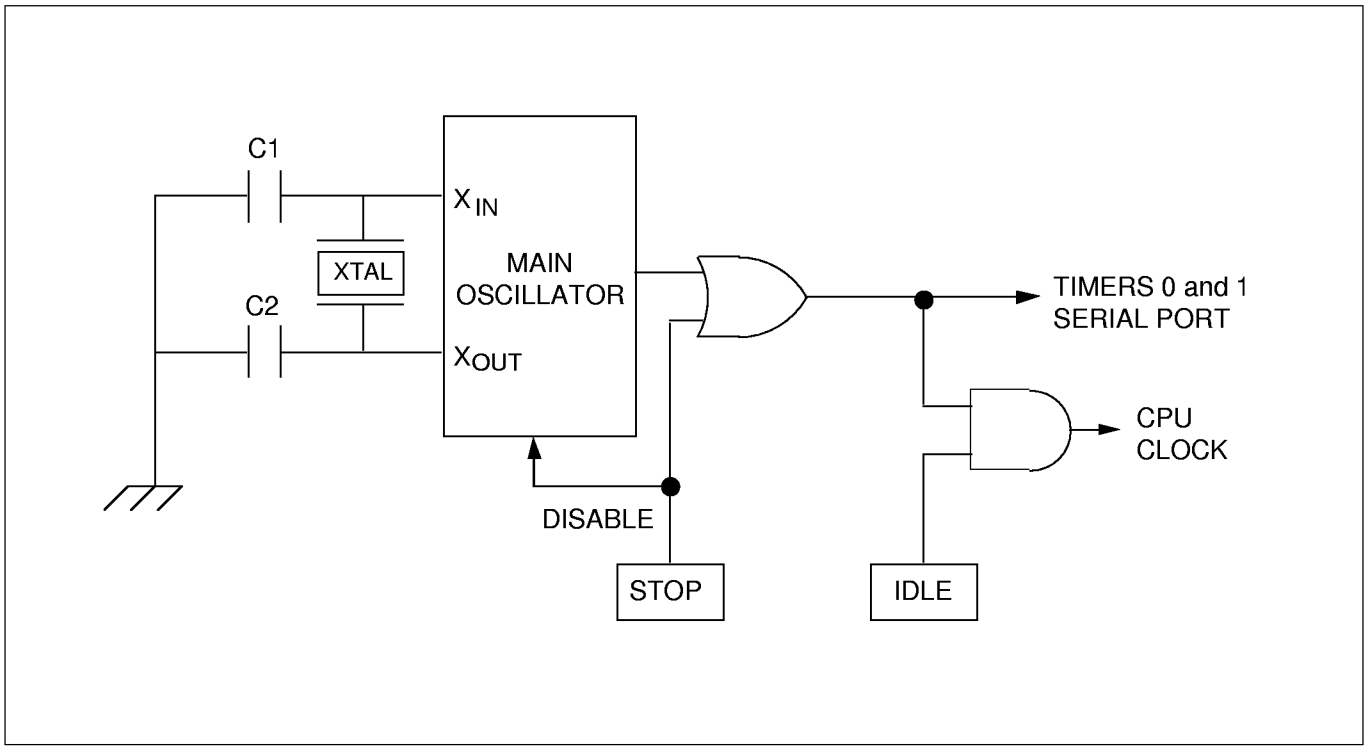


Figure 18. Stop and Idle Mode Function Diagram

☞ PROGRAMMING TIP — Sample KS88C8016 Initialization Routine

The following program is an example of how to make the initial program settings for the KS88C8016 address space, interrupt vectors, and peripheral functions:

```
;<< Chip Directive >>

          CHIP      C: \ DEF \ 88C8016.DEF

;<< Reset Vector >>
          ORG       0020H
          JP        t,START

;<< Interrupt Vector Addresses >>

;0022H–00B7H: Reserved

          ORG       00B8H
          VECTOR    SIOINT              ;  IRQ6, SIO interrupt

;00BAH–00BDH: Reserved

          ORG       00BEH
          VECTOR    TAINT               ;  IRQ6, timer A interrupt
          VECTOR    EXTINT              ;  IRQ7, external interrupt

;00C4H–00DFH: Reserved

;         ORG       00E0H
          VECTOR    C1CLRINT            ;  IRQ5, counter 1 clear interrupt
          VECTOR    T1M0INT             ;  IRQ5, timer 1 module 0 interrupt
          VECTOR    T1M1INT             ;  IRQ5, timer 1 module 1 interrupt
          VECTOR    C1OVINT             ;  IRQ5, counter 1 overflow interrupt

;00E8H–00EFH: Reserved

;         ORG       00F0H
          VECTOR    T2M2INT             ;  IRQ0, timer 2 module 2 interrupt
          VECTOR    T2M4INT             ;  IRQ0, timer 2 module 4 interrupt
          VECTOR    T2M3INT             ;  IRQ1, timer 2 module 3 interrupt
          VECTOR    T2M1INT             ;  IRQ1, timer 2 module 1 interrupt
          VECTOR    T2M0INT             ;  IRQ2, timer 2 module 0 interrupt
          VECTOR    T2M5INT             ;  IRQ3, timer 2 module 5 interrupt
          VECTOR    C2OVINT             ;  IRQ4, counter 2 overflow interrupt
          VECTOR    NMIINT              ;  Non-maskable interrupt

          ORG       0100H

START     DI                            ;       Interrupt disable
          SB0                           ;       Select bank 0
```

(Continued on next page)

☞ **PROGRAMMING TIP — Sample KS88C8016 Initialization Routine (Continued)**

```
;< System Register Settings >

        LD      TBINT,#0AAH     ;  Non-maskable interrupt disable
        CLR     PP              ;  Select page 0
        CLR     EMT             ;  No wait and internal stack area selected
        CLR     SPH             ;  Stack pointer high byte to zero
        CLR     SPL             ;  Stack pointer low byte to zero
        CLR     SYM             ;  Global and fast interrupt disable

;<Interrupt Settings>

        LD      IPR,#10H        ;  Interrupt priorities
                                ;  IRQ0 > 1 > 2 > 3 > 4 > 5 > 6 > 7

        LD      IMR,#11000000B  ;  IRQ levels 6 and 7 enable
                                ;  Level 6  =  SIOINT, TAINT
                                ;  Level 7  =  EXTINT

;< Peripheral Registers and I/O Port Settings >

        LD      TADATA,#0FH     ;  If CPU clock = 8 MHz, interrupt interval
                                ;  is 1 / (8 MHz /1000) × 16 = 2 ms
        LD      T0CON,#06H      ;  Interval mode, interrupt enable
        LD      T1CON,#15H      ;  Start (resume) counter 1
                                ;  Select CPU clock
                                ;  Enable counter 1 clear by external signal
                                ;  Enable counter 1 clear interrupt, but
                                ;  the IMR setting disables the interrupt!
        LD      T2CON,#0CH      ;  Select internal clock (CPU clock /6)
                                ;  Select the overflow from T2H
                                ;  Disable counter 2 overflow interrupt
        LD      SIOCON,#19H     ;  Use falling edge of shift clock
                                ;  Push-pull type
                                ;  Input (receive) mode
                                ;  Interrupt enable
        LD      SIOPS,#00H      ;  Select SCLK as shift clock
                                ;  Prescaler = 0

;< Port 0 Setting >

        LD      P0CON,#11H      ;  Output, push-pull

;< Port 1 Setting >

        LD      P1CON,#11H      ;  Output mode, push-pull
```

(Continued on next page)

☞ PROGRAMMING TIP — Sample KS88C8016 Initialization Routine (Continued)

```
;<Port 2 Setting>

            LD      P2CONL,#65H     ;  P2.0 input mode
                                    ;  P2.1 input mode
                                    ;  P2.2 input mode, pull-up
                                    ;  P2.3 input mode
            LD      P2CONH,#69H     ;  P2.4 input mode
                                    ;  P2.5 input mode, pull-up
                                    ;  P2.6 input mode, pull-up
                                    ;  P2.7 input mode
            LD      P2PND,#0FFH     ;  Clear interrupt source
            LD      P2INT,#04H      ;  P2.0, 1, 3, 4, 5, 6, 7 interrupt disable
                                    ;  P2.2 interrupt enable

;<Port 3 Setting>

            SB1                     ;  Select bank 1
            LD      P3CON,#43H      ;  P3.0–P3.3 output mode, open-drain
                                    ;  P3.4–P3.7 input mode, pull-up

;<Port 4 Setting>

            LD      P4CONL,#00H     ;  P4.0–P4.3 input mode
            LD      P4CONH,#0FFH    ;  P4.4–P4.7 output mode, push-pull
            LD      P4PND,#0FFH     ;  Clear interrupt source
            LD      P4INT,#00H      ;  Disable all P4 interrupts
            SB0                     ;  Select bank 0

;<< Register Initialization >>

            SRP     #0C0H           ;  Working register pointer setting
                                    ;  R0 = C0, R1 = C1, R2 = C2, ... R0F = CF

            LD      R0,#0FFH
RAMCLR0     CLR     @R0             ;  Page 0, 01H–FFH ← 00
            DJNZ    R0,RAMCLR0
            CLR     @R0
;
            LD      PP,#01H         ;  Select page 1
            LD      R0,#7FH
RAMCLR1     CLR     @R0             ;  Page 1, 01H–7FH ← 00
            DJNZ    R0,RAMCLR1
            CLR     @R0
            CLR     PP              ;  Select page 0

            EI                      ;  Interrupt enable
```

(Continued on next page)

☞ **PROGRAMMING TIP — Sample KS88C8016 Initialization Routine (Continued)**

;<< Main Routine >>

```
            CALL    KEYCHECK
                    •
                    •
                    •
            CALL    JOB1
                    •
                    •
                    •
            CALL    JOB2
                    •
                    •
                    •
            CALL    JOB3
                    •
                    •
                    •
            JR      T,MAIN

;<Subroutine 1>

KEYCHECK:
            •
            •
            •
            RET

;<Subroutine 2>

JOB1:
            •
            •
            •
            RET

;<Subroutine 3>

JOB2
            •
            •
            •
            RET

;<Subroutine 4>

JOB3:
            •
            •
            •
            RET
```

(Continued on next page)

☞ **PROGRAMMING TIP — Sample KS88C8016 Initialization Routine (Continued)**

;<< Interrupt Service Routine >>

```
TAINT:  PUSH    RP0
        PUSH    RP1
        PUSH    PP
        CLR     PP
        SRP     #40H            ;  Working register pointer setting
                                ;  R0 = 40, R1 = 41, R2 = 42 ...
        SB0                     ;  Select bank 0
                •
                •
                •
        LD      T0CON,#06H      ;  Pending bit clear
        POP     PP
        POP     RP1
        POP     RP0
        IRET

EXTINT: PUSH    PP
        PUSH    RP0
        PUSH    RP1
        CLR     PP
        SB1
        LD      P4PND,#0FFH     ;  Interrupt pending bit clear
        SB0
        LD      P2PND,#0FFH     ;  Interrupt pending bit clear
        •
        •
        •
        POP     RP1
        POP     RP0
        POP     PP
        IRET

SIOINT: PUSH    RP0
        PUSH    RP1
        PUSH    PP
        CLR     PP
        SRP     #50H            ;  Working register pointer setting
                                ;  R0 = 50, R1 = 51, R2 = 52 ...
        SB0                     ;  Select bank 0
        •
        •
        •
        LD      R0,SIO          ;  R0 (50) ← SIO input data
        •
        •
        •
```

(Continued on next page)

☞ **PROGRAMMING TIP — Sample KS88C8016 Initialization Routine (Concluded)**

```
                •
                •
                •
                POP     PP
                POP     RP1
                POP     RP0
                IRET

;< Unused Interrupt Vectors >

C20VINT:

T2M0INT:

T2M1INT:

T2M2INT:

T2M3INT:

T2M4INT:

T2M5INT:

C1OVINT:

T1M1INT:

T1M0INT:

C1CLRINT:

NMIINT:

IRET
;
END
```

I/O PORTS

OVERVIEW

The KS88C8016 has five 8-bit I/O ports, ports 0–4. The CPU accesses these ports by directly writing to or reading from respective port register addresses. Each port can be flexibly configured to meet various system design requirements.

Port 2 and port 4 pins can be configured selectively to either input or output mode. They also can be used as input pins for external interrupt signals.

A/D converter input pins AD0–AD7 can be alternately used as an 8-bit input port 5 (P5.0–P5.7). In addition, the four output pins for timer 1 module 1 (T1M1.0–T1M1.3), and for timer 2 module 5 (T2M5.0–T2M5.3), can alternately serve as normal 4-bit output ports.

PORTS 0, 1, AND 3

Ports 0, 1, and 3 are configured by software on a nibble basis to input or output mode. The architecture of port 1 differs from ports 0 and 3 only in that port 1 supports high-current loads (15mA typical value).

Ports 0, 1, and 3 have identical 8-bit control registers. P0CON and P1CON are at addresses E3H and E4H in set 1, bank 0; P3CON is at address E3H in set 1, bank 1.

Bits 7–4 of each register control the upper-nibble configuration; bits 3–0 control the lower-nibble configuration.

I/O PORTS 0, 1, AND 3 CONTROL REGISTERS
P0CON, E3H, Set 1, Bank 0, R/W
P1CON, E4H, Set 1, Bank 0, R/W
P3CON, E3H, Set 1, Bank 1, R/W

MSB	.7	.6	.5	.4	.3	.2	.1	.0	LSB

Upper nibble port configuration Lower nibble port configuration

7 (3)	6 (2)	5 (1)	4 (0)	Port Selection
x	0	x	0	Input, Schmitt trigger
x	1	x	0	Input, Schmitt trigger, pull-up
x	0	0	1	Output, push-pull
x	0	1	1	Output, open-drain
x	1	1	1	Output, open-drain, pull-up

('x' means don't care.)

Figure 19. Port 0, 1, and 3 Control Registers (P0CON, P1CON, P3CON)

PORTS 2 AND 4

Port 2 and port 4 pins can be configured selectively to either input or output mode. They can also be used as input pins for external interrupt signals. The control register structure for ports 2 and 4 is identical. Each port has two 8-bit control registers for configuring port pins.

P2CONH configures the upper nibble pins (P2.7–P2.4) and P2CONL configures the lower nibble pins (P2.3–P2.0).

P4CONH configures the upper nibble pins (P4.7–P4.4) and P4CONL configures the lower nibble pins (P4.3–P4.0)

Paired bit values in the P2CON and P4CON registers provide the necessary range of configuration options for each pin.

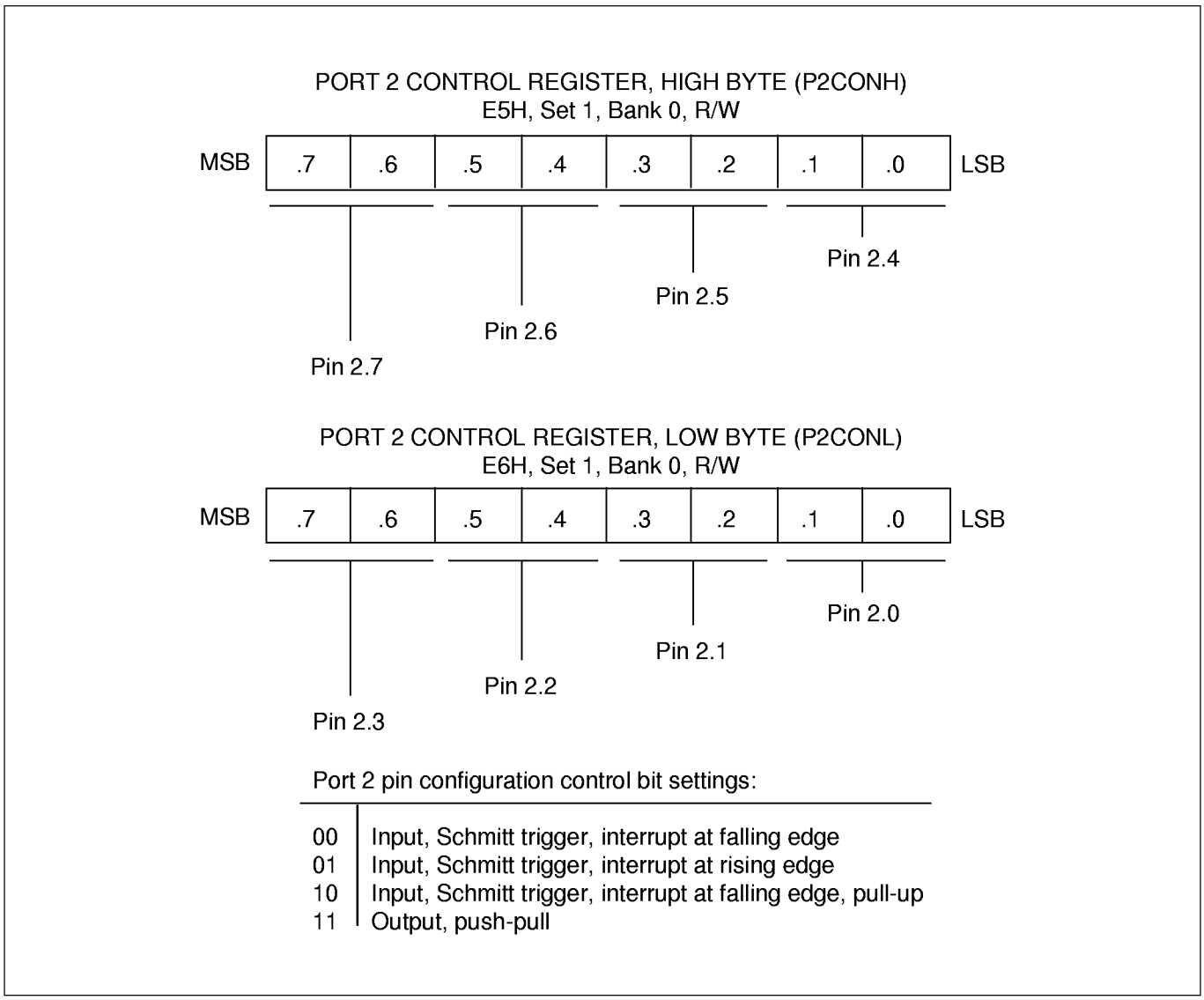


Figure 20. Port 2 Control Registers (P2CONH, P2CONL)

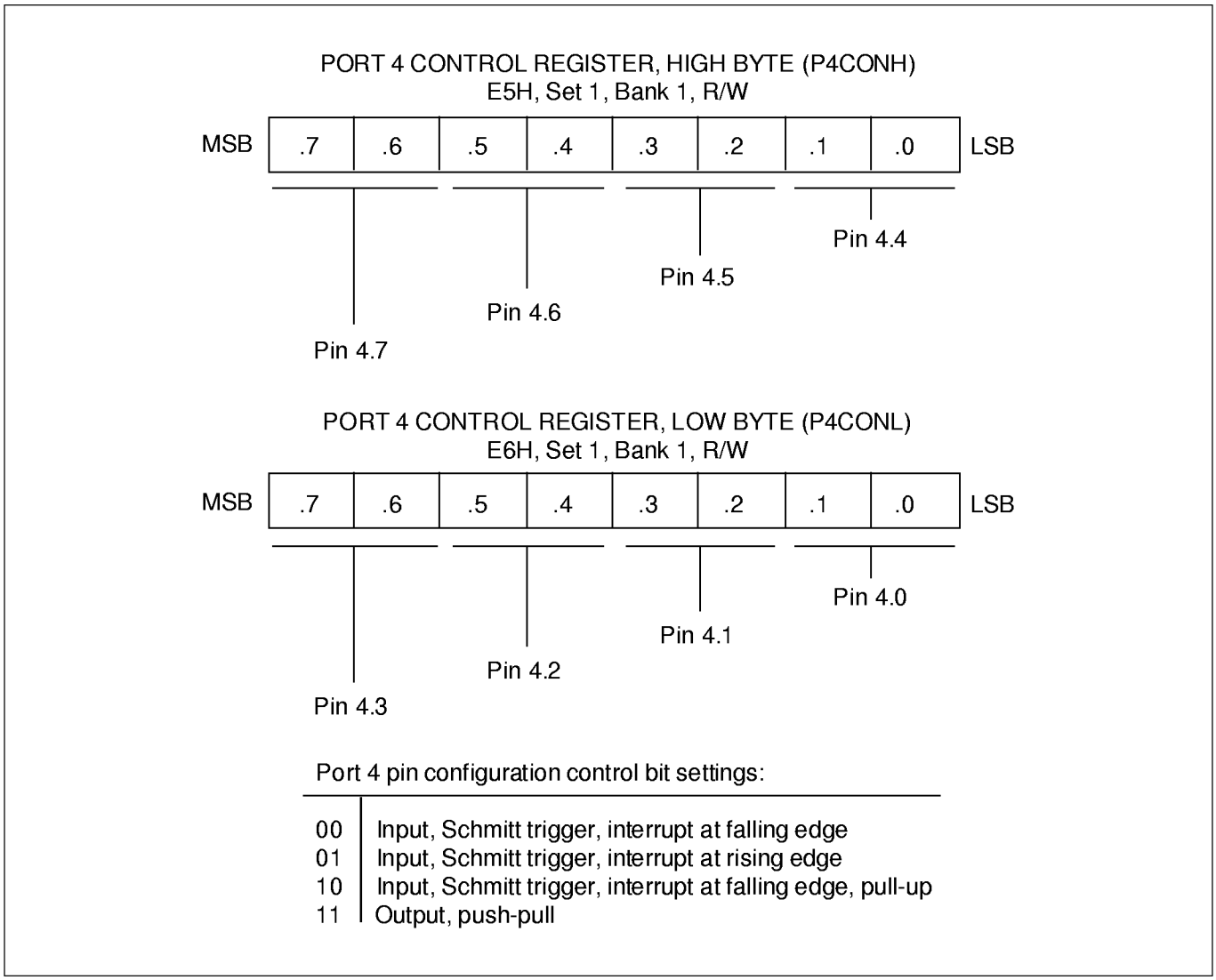


Figure 21. Port 4 Control Registers (P4CONH, P4CONL)

PORT 2 AND 4 INTERRUPT ENABLE/DISABLE FUNCTION

By setting bits in the port 2 and port 4 interrupt enable registers (P2INT, P4INT), pins can be configured individually to generate interrupt requests when specific signal edges are detected.

When the interrupt enable bit of any port 2 or port 4 pin is "1", a rising or falling signal edge at that pin will generate an interrupt request.

Remember that the pin also must be set to input mode by the paired bit values in the corresponding P2CON or P4CON register.

The signal edge then sets the corresponding bit in the port's interrupt pending register (P2PND, P4PND) to "1", and the IRQ pulse goes low. This signals the CPU that an interrupt request is waiting.

When the CPU has serviced the interrupt request, the application program must clear the appropriate interrupt pending register bit. To do this, write a "1" to the appropriate bit in the P2PND or P4PND register.

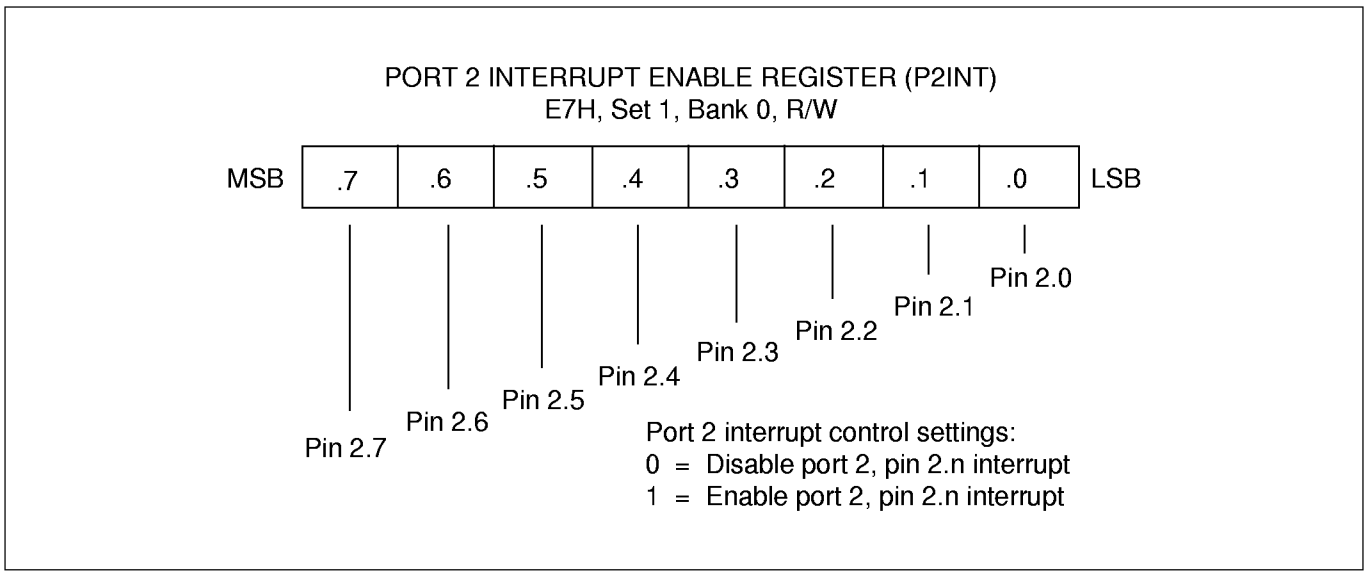


Figure 22. Port 2 Interrupt Enable Register (P2INT)

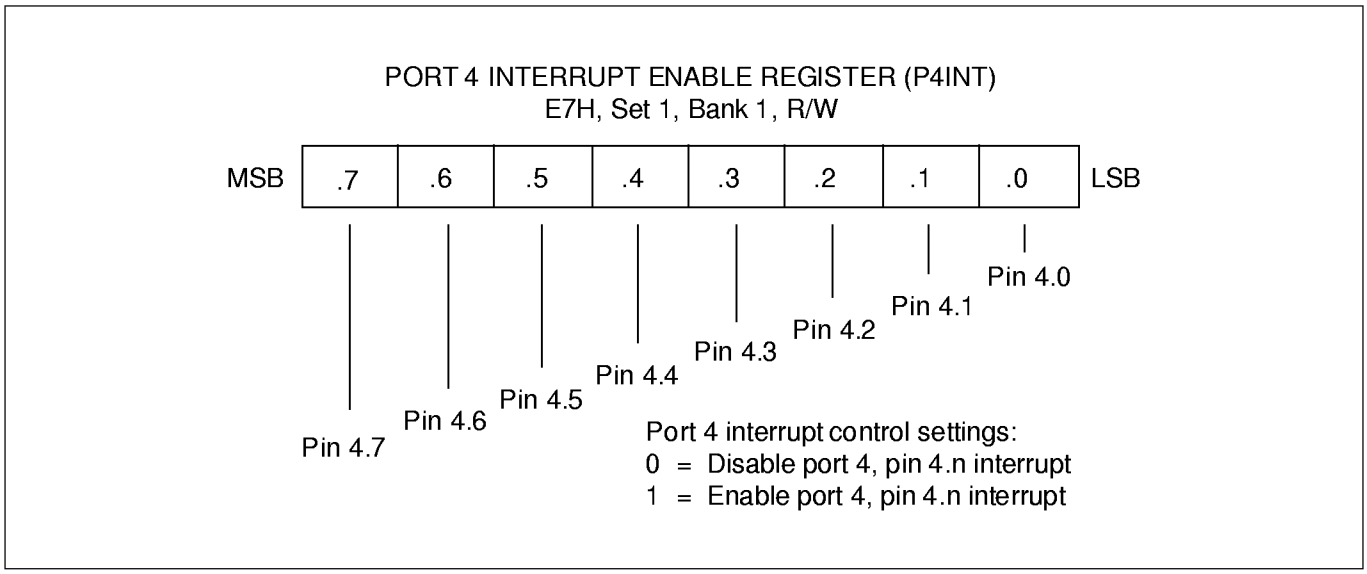


Figure 23. Port 4 Interrupt Enable Register (P4INT)

PORT INTERRUPT PENDING REGISTER TYPES

There are two types of port pending registers: One type is cleared by hardware after the interrupt request is acknowledged. The other type (like the P2PND and P4PND registers) must be cleared by software after the interrupt is acknowledged.

The port 2 and port 4 interrupt pending registers P2PND and P4PND are read and write addressable by software. When the interrupt service routine has executed, the application program clears the appropriate pending bit by writing a *"1"* to the appropriate bit (instead of a "0" as might be expected).

We also recommend that you use Load instructions (except for LDB) to manipulate interrupt pending registers.

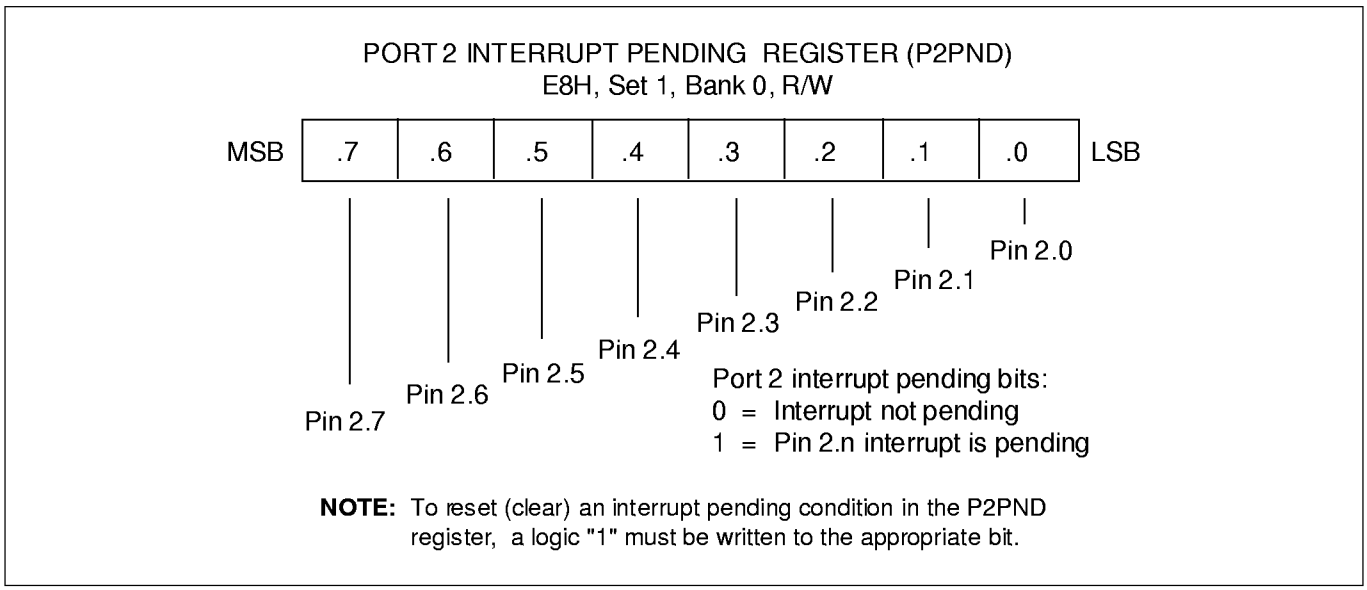


Figure 24. Port 2 Interrupt Pending Register (P2PND)

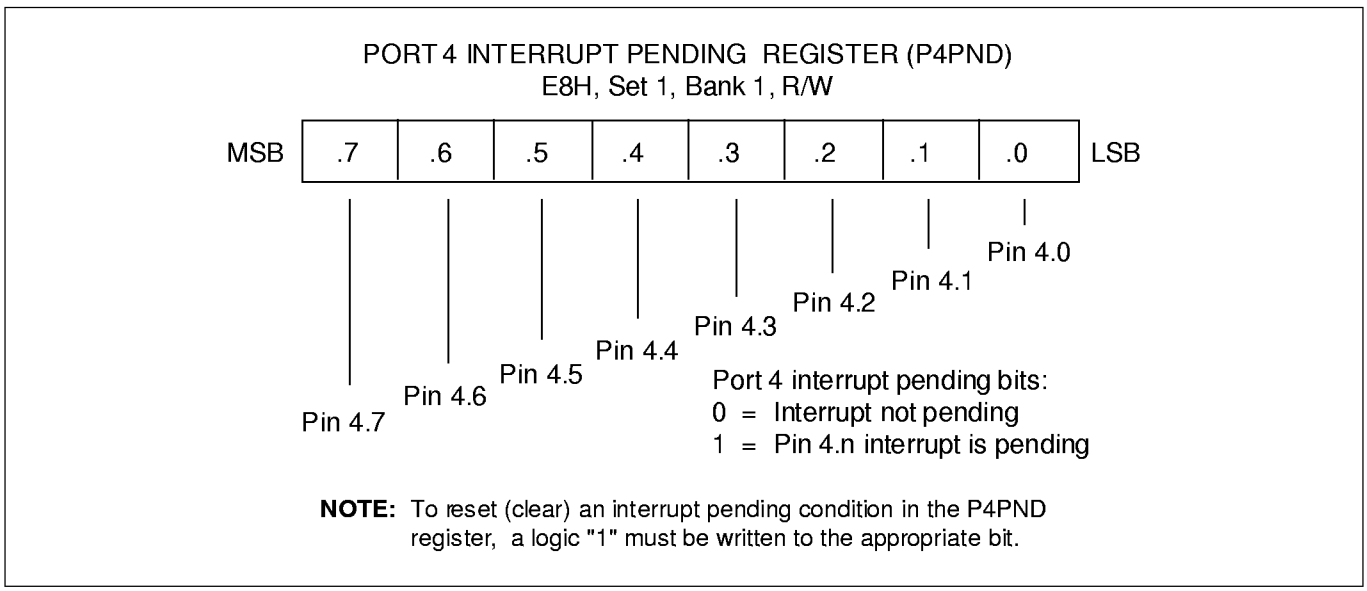


Figure 25. Port 4 Interrupt Pending Register (P4PND)

TIMER MODULE 0

OVERVIEW

The timer/counter module T0 consists of two 8-bit timer/ counters called timer A and timerB. Each timer/counter has an 8-bit counter with 4-bit prescaler, an 8-bit data register, and an 8-bit comparator.

The control registers T0CON and TBINT control module 0 operation.

Timers A and B run continuously; the corresponding counters cannot be reset or accessed directly.

Timer A and B Operating Modes

Timers A and B are designed to operate in interval mode or pulse width modulation (PWM) mode. However, because the KS88C8016 does not have output pins for PWM output, timer A and timer B are restricted to interval mode operation only.

The LSB of the T0CON register controls operating mode for timerA. The LSB of the TBINT register controls the timer B operating mode. Because PWM mode is not available, the EAPWM and EBPWM control bits must always be cleared to "0" to allow interval timer operation.

Timer Clock Input

The same clock input drives both timers A and timer B. There is only one input clock option — the CPU clock frequency divided by 1000. When the CPU clock input is disabled, the T0 module also is disabled.

When the TCS bit (bit 3) in the T0CON register is "1", the T0 module is disabled; when it is "0", T0 runs on the divided-by-1000 CPU clock.

The CPU clock frequency is divided by a 4-bit prescaler located in bits 4–7 of the T0CON register (TPS3–TPS0).

Timer A Interrupts

In interval mode, both timers generate a match signal when the count value and the referenced data value in the TADATA or TBDATA register is the same. When the match is detected, the count register is cleared, an interrupt request is generated, and counting resumes.

For timer A, the interrupt is enabled by the ETAI bit in the T0CON register. Timer A's interrupt pending bit (bit 1 in the T0CON register) should be cleared by software after the interrupt is serviced.

The timer A interrupt is controlled by an enable bit and a pending bit in T0CON.

Timer A Pending Bit (TAIP)

Bit 1 (TAIP) in the timer module 0 control register T0CON is the interrupt pending bit for timer A. It is the only readable bit in the T0CON register.

When a TAIP read shows a logic zero value, no interrupt is pending. When TAIP is logic one, a timer A interrupt is pending. To clear the pending condition, write a "1" to the TAIP bit.

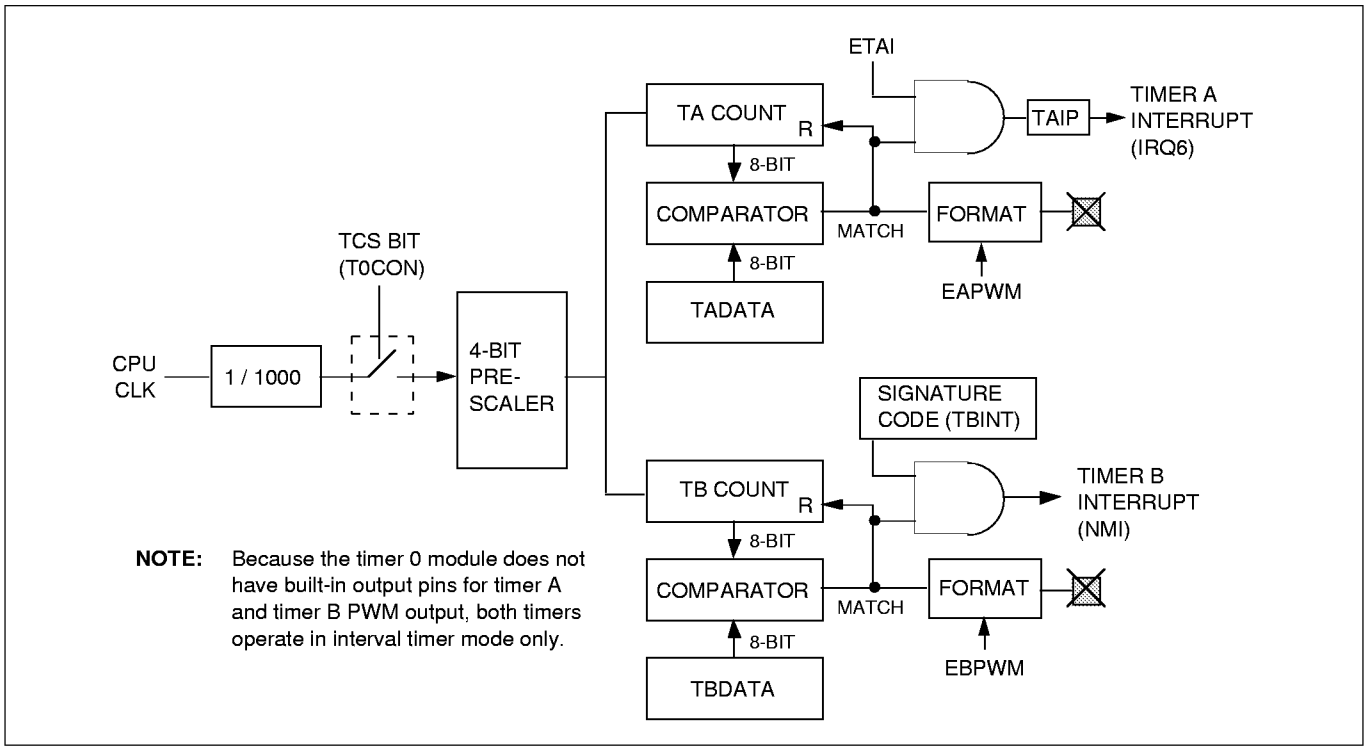


Figure 26. Timer 0 Function Block Diagram

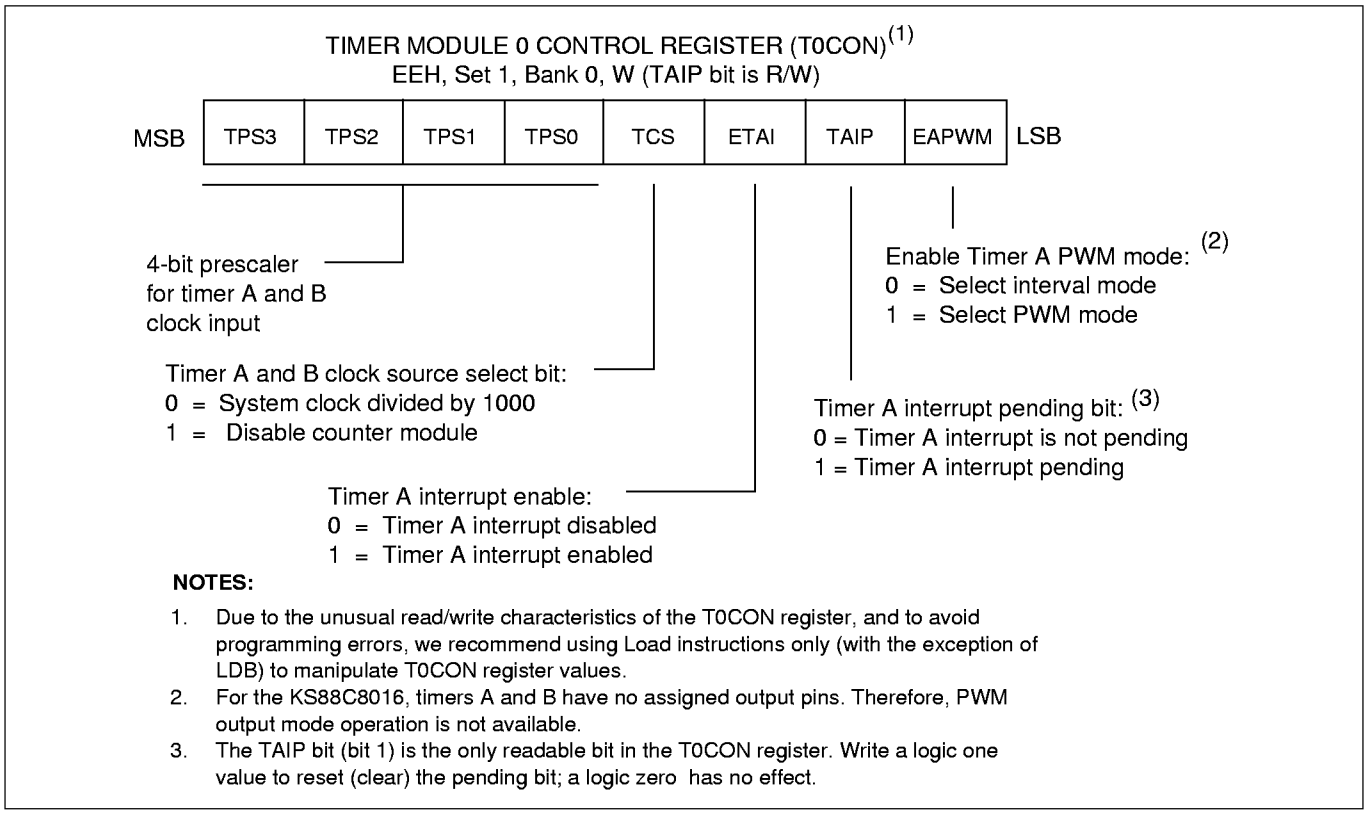


Figure 27. Timer A and B Control Register (T0CON)

INTERVAL MODE FUNCTION DESCRIPTION

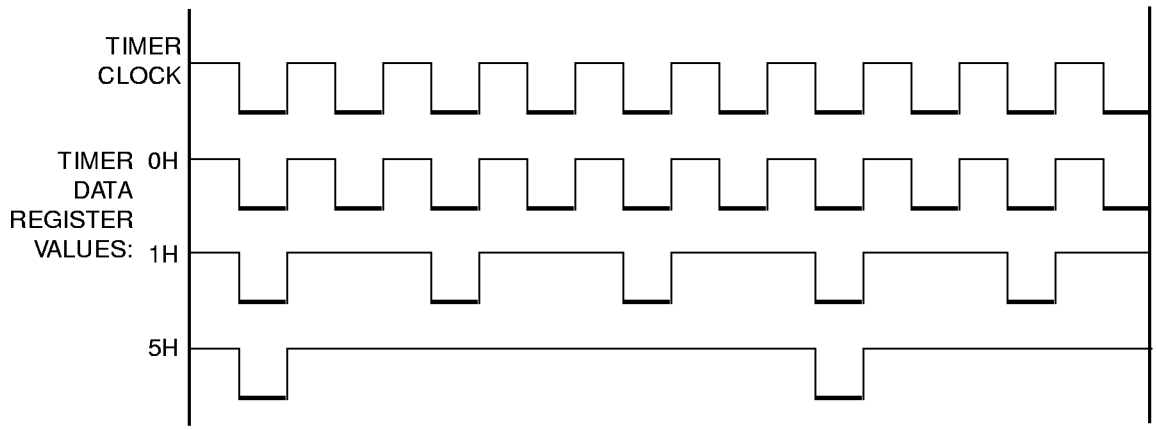
Timer B operates in interval mode exactly like timer A, except that it has its own data register and is controlled by the EBPWM bit in the TBINT register. Timer B also generates a non-maskable interrupt.

Although timers A and B are designed to toggle the output pin values TA_{OUT} and TB_{OUT}, these output pins are not included in the KS88C8016.

In interval mode, the timer output signal toggles low during each clock interval on every match of the timer counter and its corresponding timer data register. (The timer output signal does *not* have a 50% duty cycle.)

When a match occurs, the timer is reset to zero. The pulse interval is determined by the combination of the timer frequency and the value written into the timer A and timerB data registers. TADATA and TBDATA.

Figure 28 shows the waveforms generated during timer A and timer B interval mode, assuming a 4-MHz CPU clock.



NOTE: A prescaler value of 00H is assumed.

Figure 28. Timer A and B Waveforms in Interval Mode

TIMER B INTERRUPT ENABLE REGISTER (TBINT)

Timer B can generate a non-maskable interrupt (NMI) that has the highest level priority in the global interrupt structure. Using the NMI feature, timer B can serve as a watchdog timer. The timer B interrupt enable register, TBINT, has two functions:

— Enable/disable the timer B non-maskable interrupt

— Select timer B operating mode (interval or PWM)

Bits 7–1 of the TBINT register are reserved for a 7-bit signature code. Whenever the 7-bit value is 1010101xB, the NMI is disabled. This value is automatically written to the TBINT bits following a reset operation.

To enable the timer B NMI, write any value *other than* the disable interrupt signature code to TBINT.7–TBINT.1. The enable timer B PWM mode control bit (EBPWM) is the LSB of the TBINT register. When it is "0", timer B operates in normal interval timer mode; when it is "1", timer B operates in PWM mode.

Because the KS88C8016 does not have PWM output pins, bit0 (EBPWM) should always be "0". If you inadvertently enable PWM mode operation by setting the EBPWM bit "1", timer B can still generate the non-maskable interrupt, however.

NOTE

You can use the NMI simultaneously with other interrupts whose pending bits are cleared by hardware. You cannot, however, use the NMI with the external interrupt (IRQ7) and the timer A interrupt (IRQ6), whose pending bits must be cleared by software.

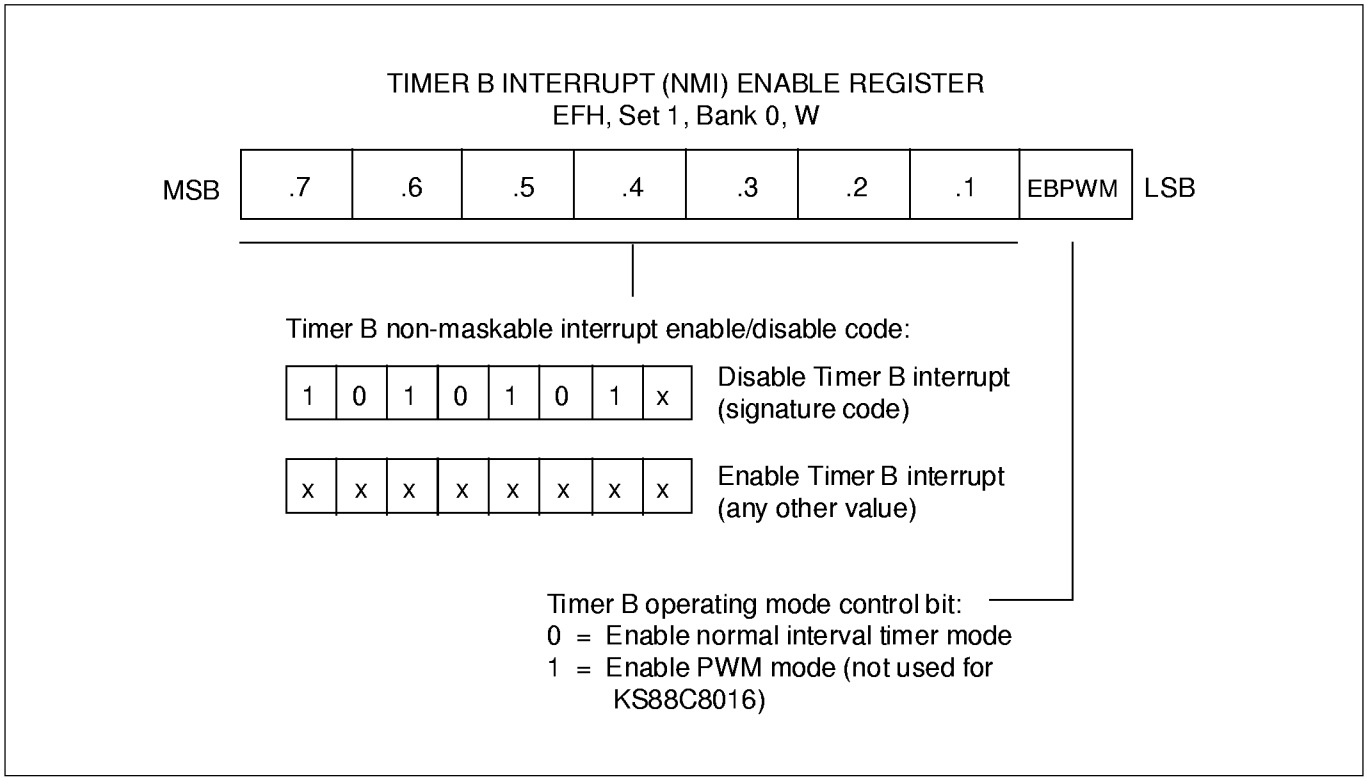


Figure 29. Timer B NMI Interrupt Enable Register

☞ PROGRAMMING TIP — Configuring Timer A and Timer B

This example sets timer A to normal interval mode, disables timer B, sets the frequency of the timer clock, and determines the sequence of instructions that follows a timer A interrupt. The program parameters are as follows:

— Timer A is used in interval mode; the timer interval is set to 2 ms; timer B is disabled.

— Oscillation frequency = 6 MHz

— General register 90H (page 1) ← 90H + 91H + 92H + 93H + 94H (page 1) is executed after a timer A interrupt.

```
        ORG     0020H           ; Reset address
        JP      T,Start
        ORG     00BEH           ; Timer A interrupt vector
        VECTOR  TAINT
        ORG     00FEH           ; Timer B interrupt vector
        VECTOR  TBINT
        ORG     0100H
        •
        •
        •
START   DI
        •
        •
        •
        LD      TBINT,#0AAH     ; Disable NMI (TBINT)
        LD      T0CON,#56H      ; PS ← 5 (for divide-by-six)
                                ; CPU clock /1000 is selected for timer 0 (A and B)
                                ; Enable timer A interrupt, reset timer A pending register
                                ; Select interval mode for timer A
                                ;
        LD      TADATA,#01H     ; TADATA ← 01 (divided by two)
                                ; (6 MHz /1000) ÷ 6 ÷ 2 = 0.5 kHz (= 2 ms)
                                ;
        EI                      ; Enable interrupts
        •
        •
        •
TAINT   PUSH    PP              ; Save page pointer to the stack
        PUSH    RP0             ; Save RP0 to stack
        SRP0    #90H            ; RP0 ← 90H
        LD      PP,#01H         ; Page pointer ← 01H (select page 1)
        ADD     R0,R1           ; R0 ← R0 + R1
        ADC     R0,R2           ; R0 ← R0 + R2
        ADC     R0,R3           ; R0 ← R0 + R3
        ADC     R0,R4           ; R0 ← R0 + R4
                                ; (R0 ← R0 + R1 + R2 + R3 + R4)
        LD      T0CON,#56H      ; Reset timer A pending register
                                ;
        POP     RP0             ; Restore register pointer 0 value
        POP     PP              ; Restore page pointer value
TBINT   IRET                    ; Return from interrupt service routine
        •
        •
        •
```

TIMER MODULES 1 AND 2

TIMER 1 (T1)

OVERVIEW

The timer 1 array (T1) has the following components:

— 16-bit counter (counter 1) with 8-bit prescaler
— 16-bit compare/capture function block (timer 1, module 0)
— Pattern generator function block (timer 1, module 1)
— Timer 1 control register (T1CON)
— Module control registers (T1M0CON, T1M1CON)
— External timer clear signal input pin (T1CLR)
— One I/O pin for T1M0; four output pins for T1M1

The 16-bit compare/capture module (timer 1 module 0, or T1M0) can be set to either capture mode or to compare mode. In capture mode, T1M0 register settings control edge selection (rising, falling, or both).

The pattern generator module (timer 1 module 1, or T1M1) has three operating modes: direct output mode, toggle mode, and pattern generation mode.

The T1M1 output pins, T1M1.0–T1M1.3, can be programmed to operate in the same mode, or individually in different modes.

COUNTER 1

Counter 1 is a 16-bit free running incrementing counter that gives time base value for both timer 1 modules.

To start (or resume) counter operation, you set bit 4 in the timer 1 control register T1CON (the EC1 bit) to "1". When the counter stops, it maintains its current count value, and then resumes counting from this value when the EC1 bit is reset to "1".

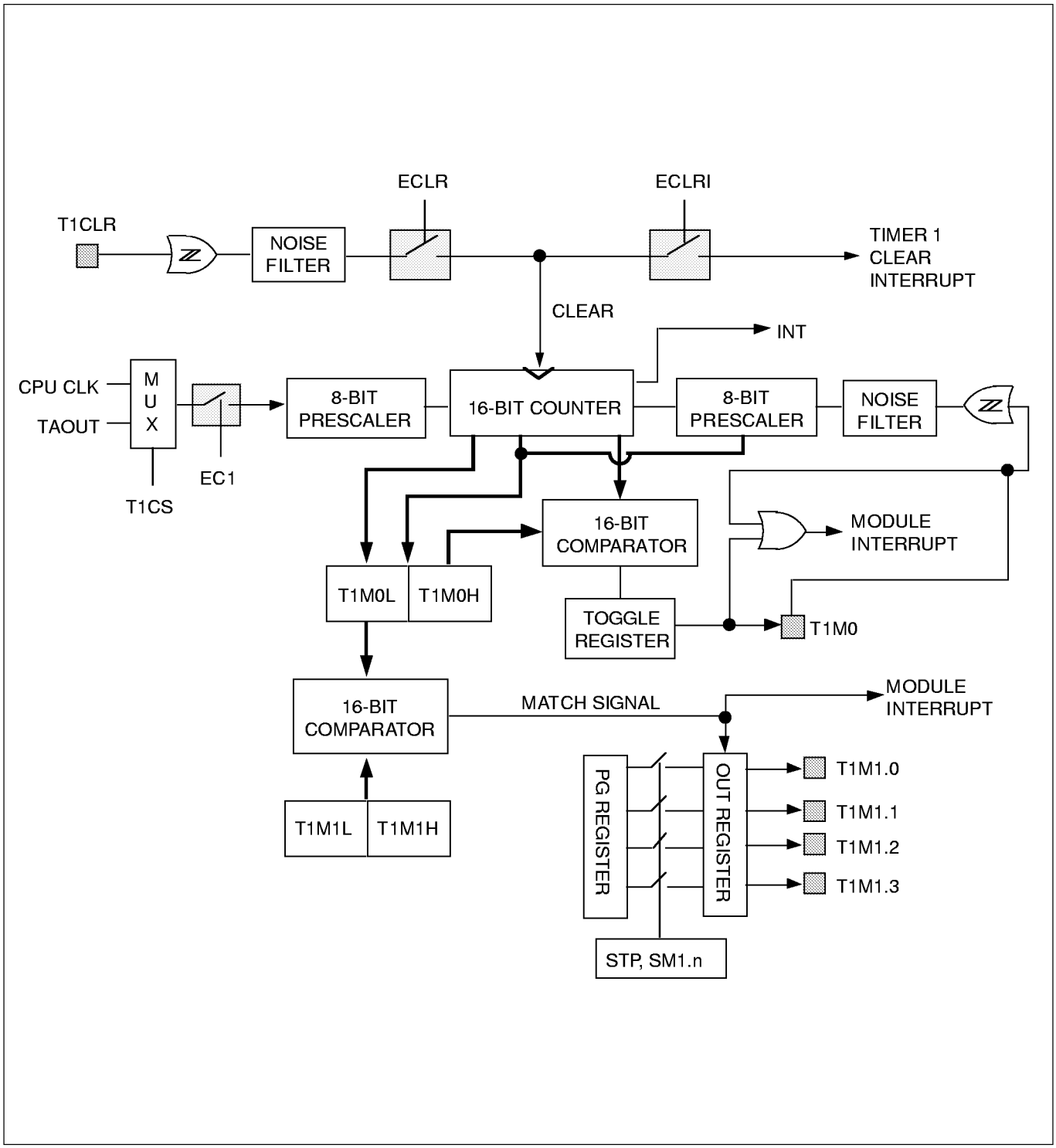


Figure 30. Timer 1 Function Diagram

TIMER 1 CONTROL REGISTER (T1CON)

Bit 3 in the timer 1 control register T1CON —the T1CS bit — selects the clock input for the timer 1 counter. There are two options: the internal CPU clock signal, or the timer A counter overflow (match) signal. The selected counter clock is sent to the 8-bit prescaler. The final counter 1 clock rate is therefore based on the programmed prescaler value.

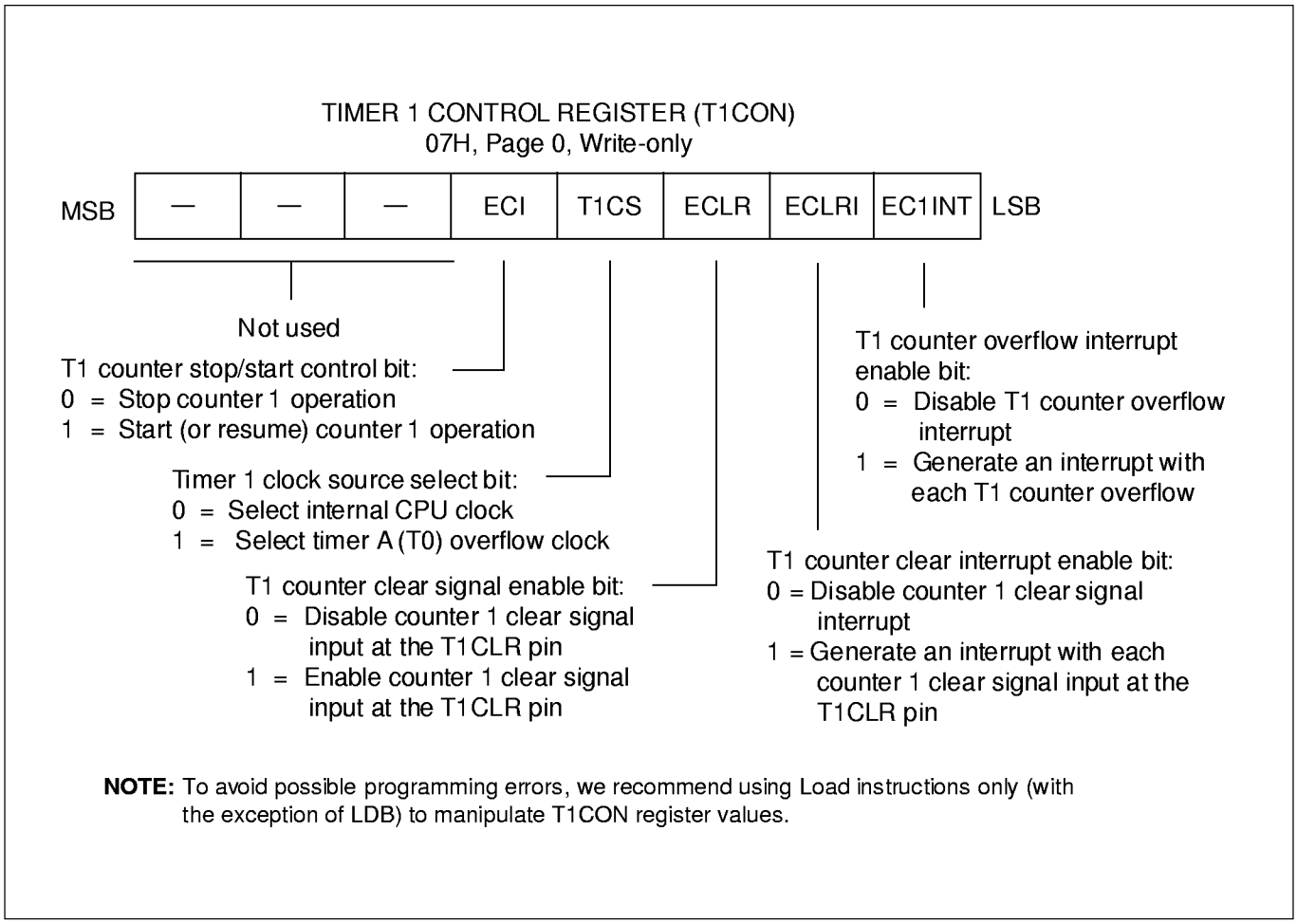
The counter 1 value is incremented with each clock signal until it reaches FFFFH. It then generates an overflow signal and resumes counting from zero. By setting bit 0 in the T1CON register (EC1INT) to "1", you can use the counter 1 overflow signal to generate a counter 1 overflow interrupt (vector E6H).

An external clear signal, input at the T1CLR pin, resets the value of the free running counter 1 to zero. T1CON bit 2 (ECLR) enables or disables this external clear signal at the T1CLR pin.

When the ECLR bit is "1", a rising edge at T1CLR clears counter 1 to zero. The counter continues incrementing. (The state of the T1CLR pin is ignored unless the ECLR bit is set.)

The external clear signal input at T1CLR also can be used to generate the T1CLR interrupt (vector E0H). To enable this function, you must set the T1CON bit 1 (ECLRI).

When ECLRI is "1", an interrupt will be generated each time a counter 1 clear signal is input at the T1CLR pin.



NOTE: To avoid possible programming errors, we recommend using Load instructions only (with the exception of LDB) to manipulate T1CON register values.

Figure 31. Timer 1 Control Register (T1CON)

TIMER 1 MODULE 0 (T1M0)

Timer 1 module 0 has a capture/compare module, an 8-bit prescaler to control the input clock, and the T1M0 input pin (pin24).

The T1M0 control register, T1M0CON, selects capture or compare mode.

Three special-purpose registers support T1M0 capture/compare operations:

- Cap/com register (high byte), T1M0H, 0AH, page 0
- Cap/com register (low byte), T1M0L, 0BH, page 0
- Toggle register, T1M0TGL, 0CH, page 0

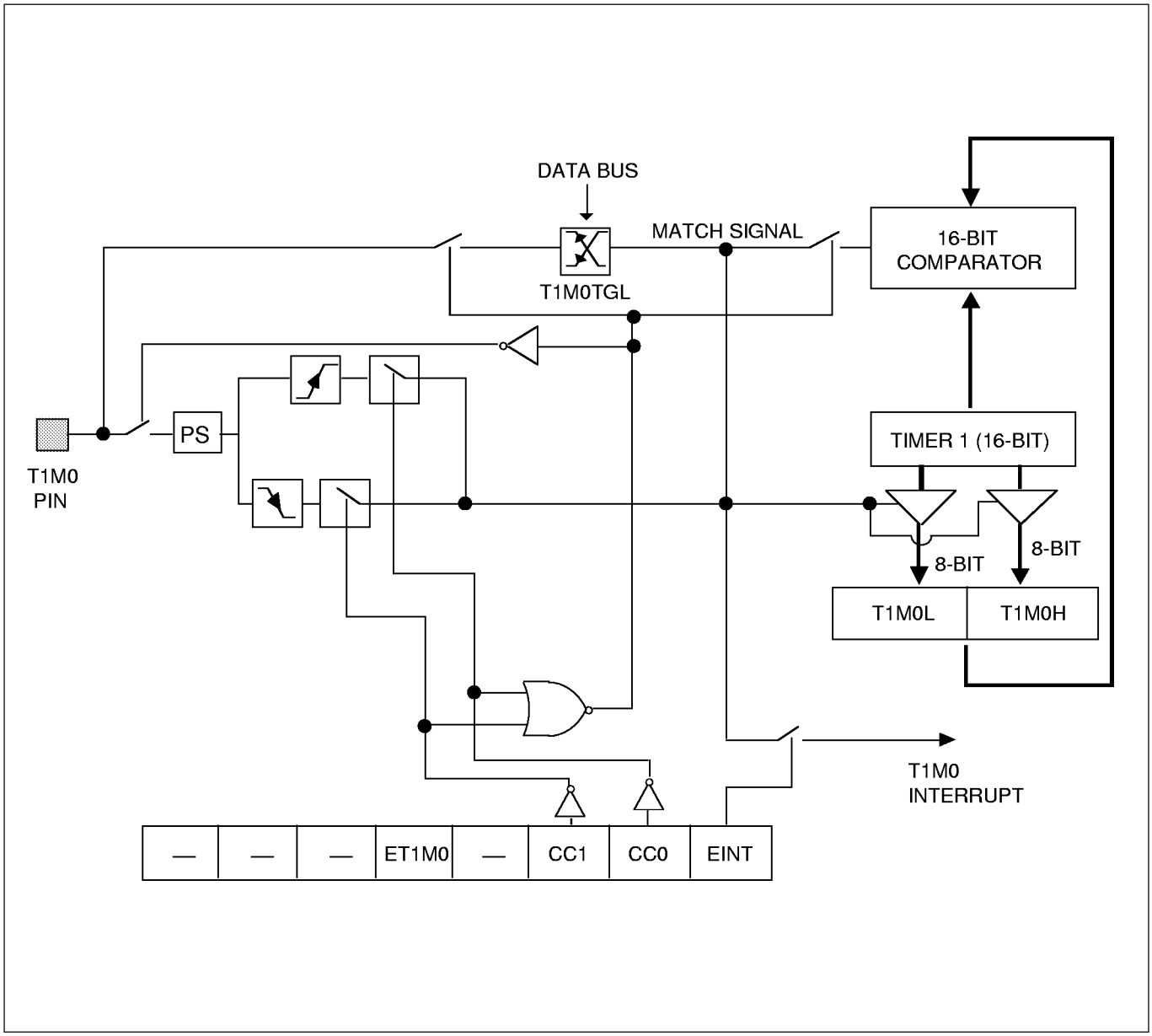


Figure 32. Timer 1 Module 0 (T1M0) Function Diagram

T1M0 CONTROL REGISTER (T1M0CON)

The T1M0 control register T1M0CON (page 0, 09H, write-only) has three functions:

— Enable and disable the module (bit 4)

— Select one of four operating modes (bits 1 and 2)

— Enable or disable the T1M0 interrupt (bit 0)

T1M0CON bits 1 and 2 (CC0 and CC1, respectively) select one of the four operating modes:

— Capture on both edges at the T1M0 pin

— Capture on rising edge at the T1M0 pin

— Capture on falling edge at the T1M0 pin

— Compare mode

To avoid possible programming errors, we recommend using Load instructions (except for LDB) to manipulate T1M0CON register values.

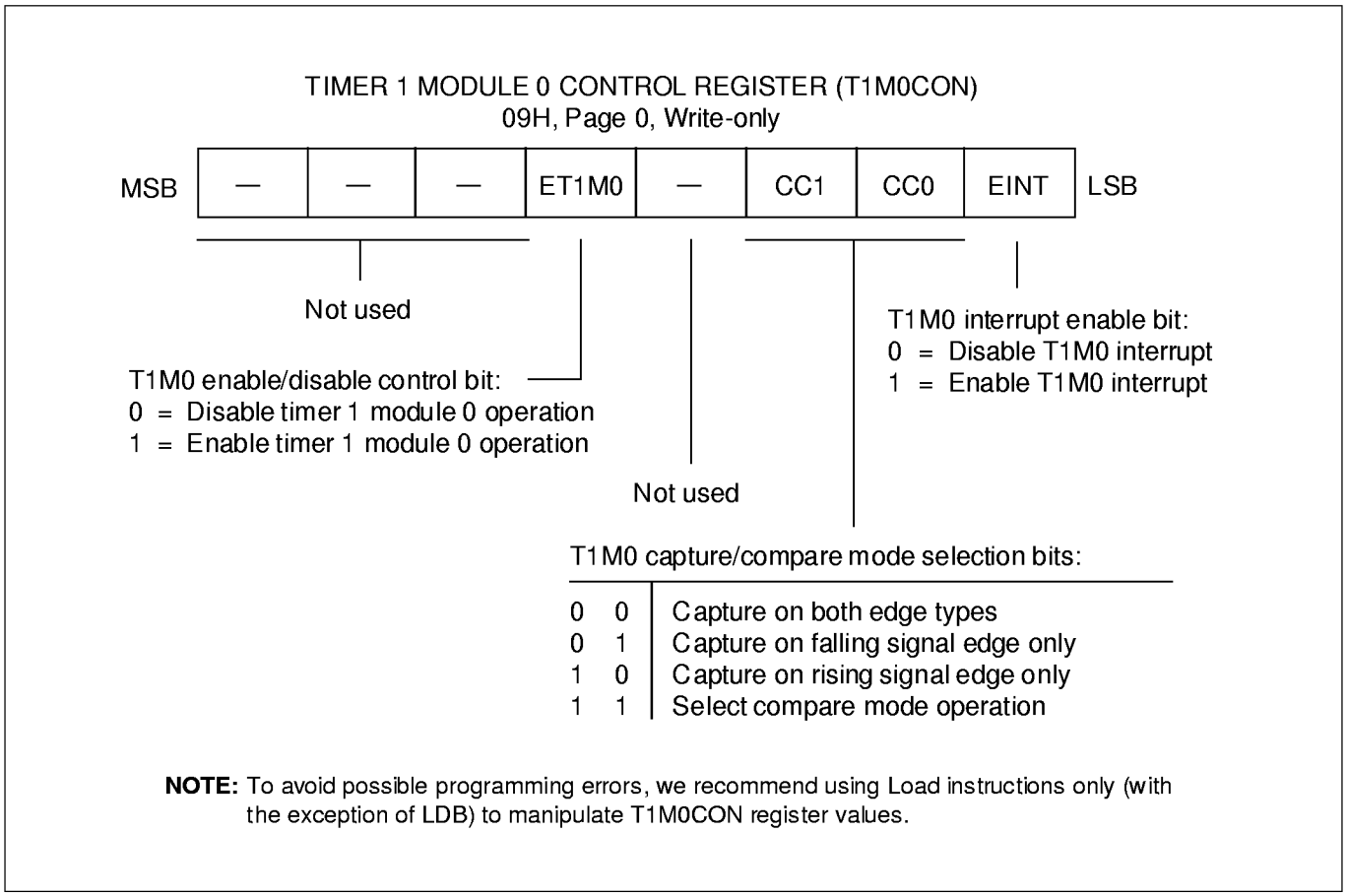


Figure 33. Timer 1 Module 0 Control Register (T1M0CON)

T1M0 Capture Mode Operation

In capture mode, the counter 1 value is latched into the 16-bit capture and compare register (T1M0H and T1M0L) whenever a signal edge is detected at the T1M0 pin.

The triggering edge can be rising, falling, or both edge types, as selected by bit settings in the T1M0CON register.

The CPU can then read the captured value from the high-byte and low-byte registers (0AH and 0BH, page0).

When the LSB of the T1M0CON register (EINT) is set to "1", a triggering signal edge at T1M0 also generates a T1M0 interrupt (vector E2H).

T1M0 Compare Mode Operation

In compare mode, the counter 1 value is compared with the contents of the 16-bit capture and compare register (T1M0H and T1M0L).

The capture/compare register contains either the data previously captured from the counter, or data that was written to it by the CPU.

When the counter value is equal to the value of the capture/ compare register, a match signal is generated.

The match signal toggles the state of the toggle register. (The toggle register is the LSB of the 8-bit register at address 0CH; the CPU writes the toggle value directly to this bit.)

If the EINT bit (T1M0CON.0) is "1", the match signal generates a T1M0 interrupt.

8-BIT PRESCALER OPERATION

Eight-bit prescalers modify the clock frequencies of timer 1 modules 0 and 1. The prescalers divide the frequency of the input clock by the value in the corresponding prescaler register, plus 1 (see Table 13):

Table 13. T1M0 and T1M1 8-Bit Prescaler Settings

Clock Input to Prescaler	Prescaler Register Setting	Resulting Output Clock
1 MHz	00000000H	1 MHz
	00000001H	500 kHz
	00001001H	100 kHz
	11111111H	3.91 kHz
	etc.	etc.

TIMER 1 MODULE 1 (T1M1)

Timer 1 module 1 is a pattern generator with the following functional components:

- Control register (T1M1CON)
- 16-bit compare register (T1M1H and T1M1L)
- Pattern generator register (T1M1PG)
- 4-bit output register (T1M1OUT)
- Output pins (T1M1.0–T1M1.3)

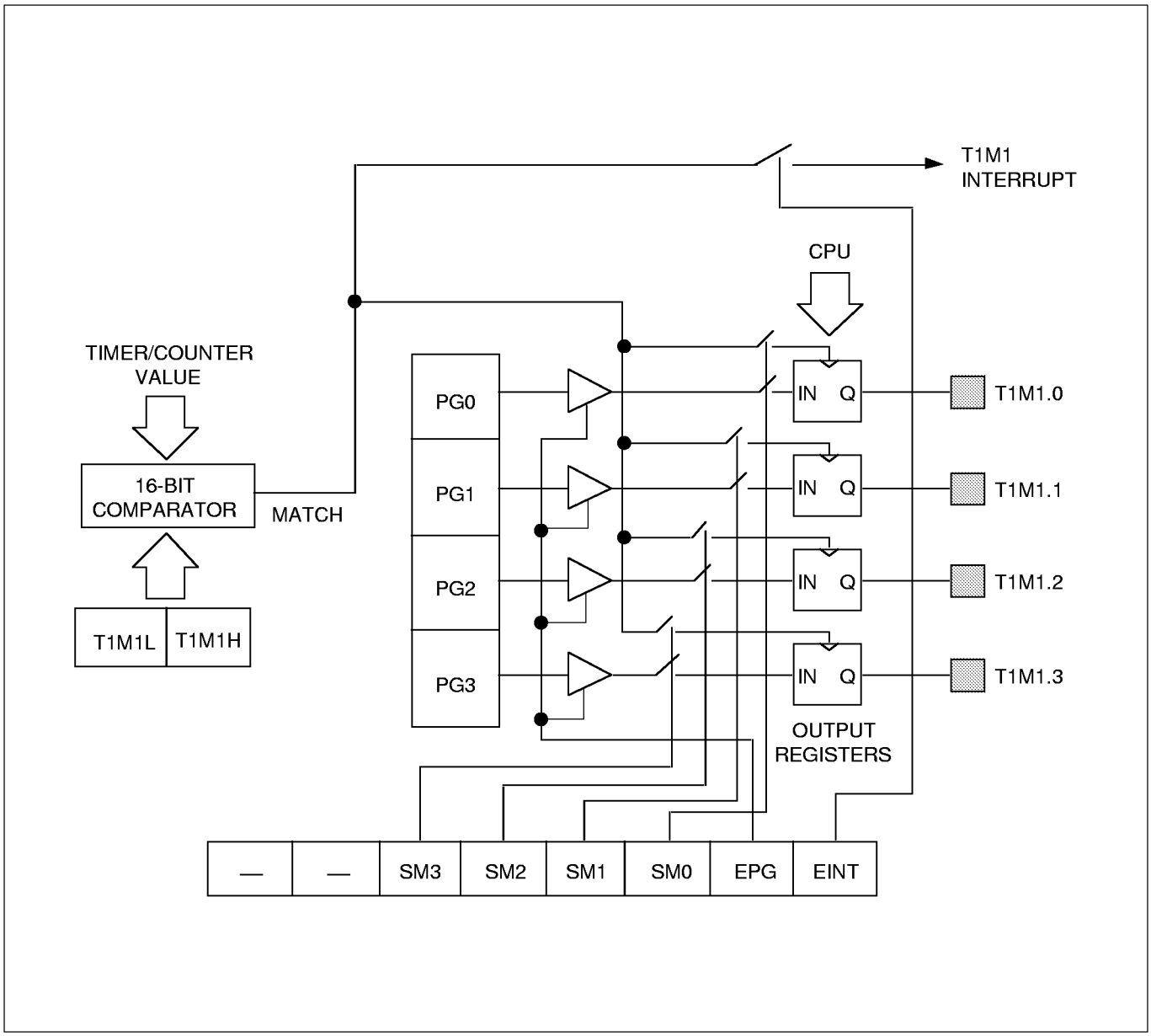


Figure 34. Timer 1 Module 1 Function Diagram

TIMER 1 MODULE 1 CONTROL REGISTER (T1M1CON)

The T1M1 control register T1M1CON (page 0, 0EH, write-only) has three functions:

— Enable/disable individual T1M1 output pins (bits 2–5)

— Enable/disable pattern generator output to each T1M1 output pin (bit 1)

— Enable/disable the T1M1 interrupt (bit 0)

The four select pin (SP) bits in the T1M1CON register (bits 2–5) correspond to the four T1M1 output pins, T1M1.0–T1M1.3. The pin's SP bit value, with the value of the EPG bit (Enable Pattern Generator), selects one of three operating modes for that pin:

— Normal mode (direct output)

— Toggle mode

— Pattern generator mode (next-state programming mode)

A reset clears the T1M1CON register to 00H, configuring all T1M1 pins to normal (direct output) operating mode.

An SP value of "1" selects an alternative operating mode:

— If SPn = "1" and EPG = "0", toggle mode is selected for that pin;

— If SPn = "1" and EPG = "1", pattern generator output is enabled to the output pin, causing it to operate in pattern generation mode.

Direct Output Mode

In direct output mode, data in the 4-bit output register are directly latched to the four output pins, T1M1.0–T1M1.3. The CPU writes data directly to the T1M1 pins by writing the output register T1M1OUT (0DH, page 0).

Toggle Mode

In toggle mode, when the counter1 value equals the 16-bit T1M1 compare register value, a match signal is generated that toggles the corresponding pin's value in the output register.

If the EINT bit (T1M1CON.0) is "1", the match signal will also generate a T1M1 interrupt (vector E4H).

Each output pin can be programmed individually to operate in toggle mode. For example, if SP0–SP2 are "1" and SP3 = "0" (and assuming EPG = "0"), then pins T1M1.0–T1M1.2 operate in toggle mode, while T1M1.3 operates in direct output mode. In this configuration, the output at the T1M1.3 pin would therefore determine the module's external operating status.

Pattern Generation Mode

In pattern generation mode, a match condition causes the data in the pattern generator register (T1M1PG, 11H, page 0) to be loaded into the output register, T1M1OUT. (The CPU would have previously written the data to the pattern generator register.) The match signal also generates a T1M1 interrupt if EINT is set.

TIMER 1 MODULE 1 CONTROL REGISTER (T1M1CON)
0EH, Page 0, Write-only

MSB	—	—	SP3	SP2	SP1	SP0	EPG	EINT	LSB

Not used

Output control bits for pins T1M1.3 (SP3) – T1M1.0 (SP0):

Pattern generator enable bit for T1M1.3 (SP3) – T1M1.0 (SP0):

T1M1 interrupt enable bit:
0 = Disable T1M1 interrupt
1 = Enable T1M1 interrupt

SPn	EPG	Effect of SPn and EPG Setting for the T1M1 Pins
0	x	Select direct output mode for pin T1M1.n
1	0	Select toggle mode for T1M1.n
1	1	Select pattern generator mode for T1M1.n

NOTE: To avoid possible programming errors, we recommend using Load instructions only (with the exception of LDB) to manipulate T1M1CON register values.

Figure 35. Timer 1 Module 1 Control Register (T1M1CON)

TIMER 2 (T2)

OVERVIEW

The timer 2 array, commonly used for servo-motor applications, has the following components:

— 16-bit counter (counter 2)
— 4-bit extension counter (T2EX)
— Three 16-bit capture modules (modules 0, 1, 2)
— Two 16-bit capture/compare modules (modules 3 and 4)
— Five 8-bit prescalers for modules 0–4
— 16-bit pattern generator (module 5)
— Timer 2 control register (T2CON)
— Control registers for each module (T2M0CON–T2M5CON)
— One I/O pin each for modules T2M0–T2M4
— Four output pins for pattern generator module T2M5

Modules 0, 1, and 2 are identical capture-only modules with four programmable operating modes. Each module has an input pin connected to an 8-bit prescaler.

Module 3 is a capture/compare module with four modes and high-current drive capability. The T2M3 pin input can be used to generate the T2M3 interrupt.

Timer 2 module 4 is similar to module 3 except that it also has a V-sync separator unit. The input to T2M4 can be one of three sources: 1) C-sync signal, 2) V-sync output of the V-sync separator unit, or 3) external input at the T2M4 pin.

Eight-bit prescalers are directly connected to the input pins of T2M0–T2M4 for use in timing applications.

T2M5 is a 16-bit pattern generator with a 4-bit output register. It operates exactly like the timer 1 module 1 pattern generator: in direct output mode, toggle mode, or pattern generation mode.

T2M5 can be used to generate head switching outputs, control head pulses in "Record" mode, and to generate a V-loc signal for VCR applications.

The T2M5.3 pin also can output an H-sync waveform that conforms to either the NTSC or PAL standard.

COUNTER 2

Counter 2 is a 16-bit free running incrementing counter. It can be extended to 20 bits using the 4-bit extension counter, T2EX (12H, page 0).

You select 16-bit or 20-bit counter operation by setting bit 1 (SOV) in the timer 2 control register, T2CON. Counter 2 provides the time base value for the six timer2 modules.

The timer 2 clock source bits (T2CS0 and T2CS1) in the T2CON register select the input source for counter 2. Counter 2 is disabled when these bits are both "0".

Three other T2CSn bit combinations select a non-divided CPU clock, a divided-by-2 CPU clock, or a divided-by-6 CPU clock as the counter 2 source.

Setting the T2CSn bits to select the clock source automatically enables the counter. When counter 2 is disabled (that is, when T2CS0 and T2CS1 are "0"), counter 2 retains its current count. Then, when the counter is re-activated, it resumes counting from that value.

T2CON bit 1 (SOV) lets you select 16-bit counter or 20-bit counter operation. It does so by determining the value for counter2 overflow signal generation:

When SOV = "0", counter 2 operates as a 16-bit counter with an overflow value of FFH; when SOV = "1", it operates in 20-bit mode with an overflow value of FFFH.

If T2CON bit 0 (EC2INT) is "1", a counter 2 overflow in either 16-bit or 20-bit mode will generate a timer 2 interrupt (vector FCH).

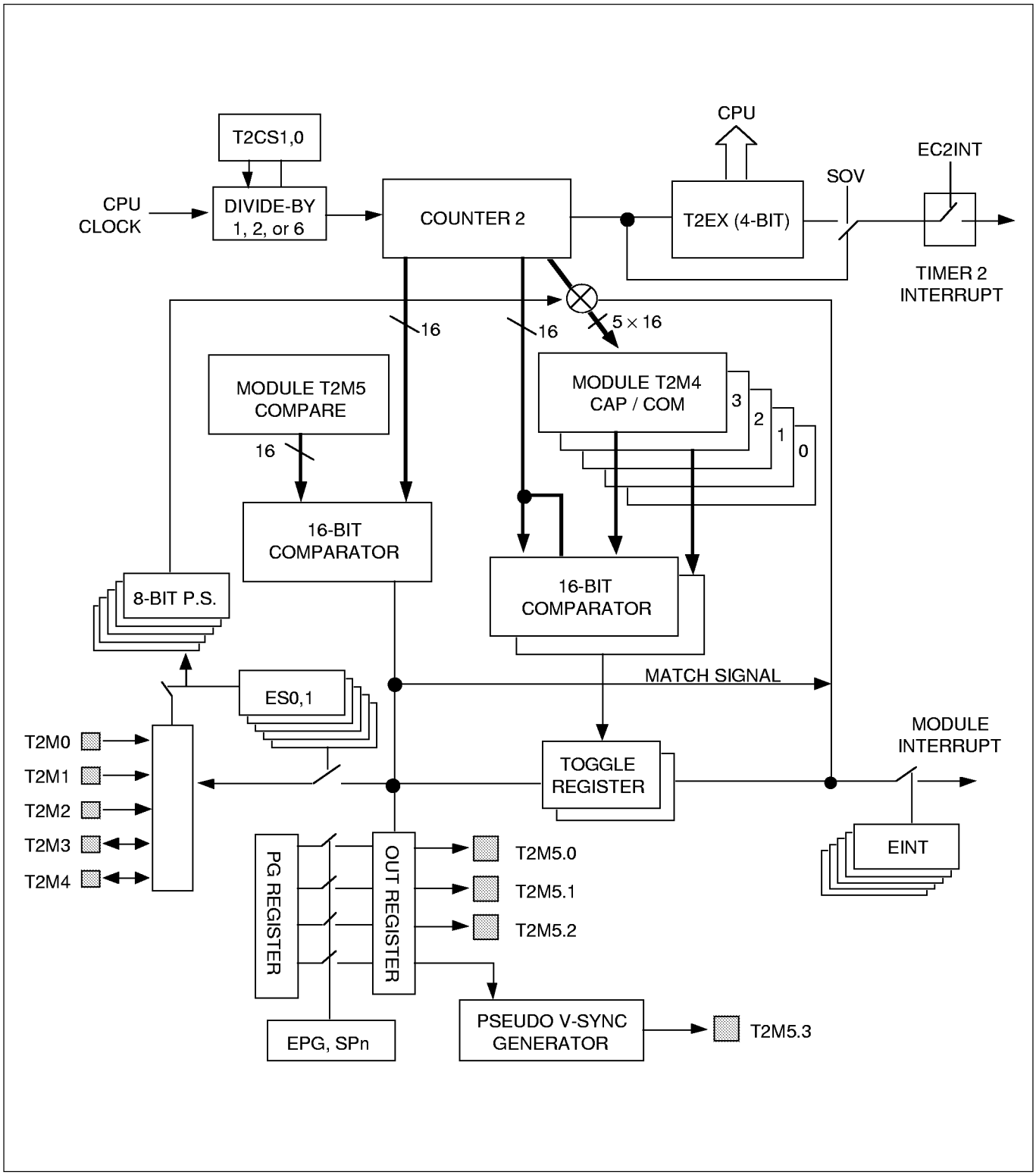


Figure 36. Timer 2 Function Diagram

TIMER 2 CONTROL REGISTER (T2CON)

The timer 2 control register T2CON (13H, page 0) has the following three functions:

— Clock source selection, and enable/disable, for the timer 2 counter (T2CS0 and T2CS1)

— Timer 2 counter 16-bit or 20-bit operating mode selection (SOV)

— Timer 2 counter overflow interrupt enable/disable (EC2INT)

To avoid possible programming errors, we recommend that you use Load instructions only (except for LDB, Load Bit) to manipulate the write-only T2CON register.

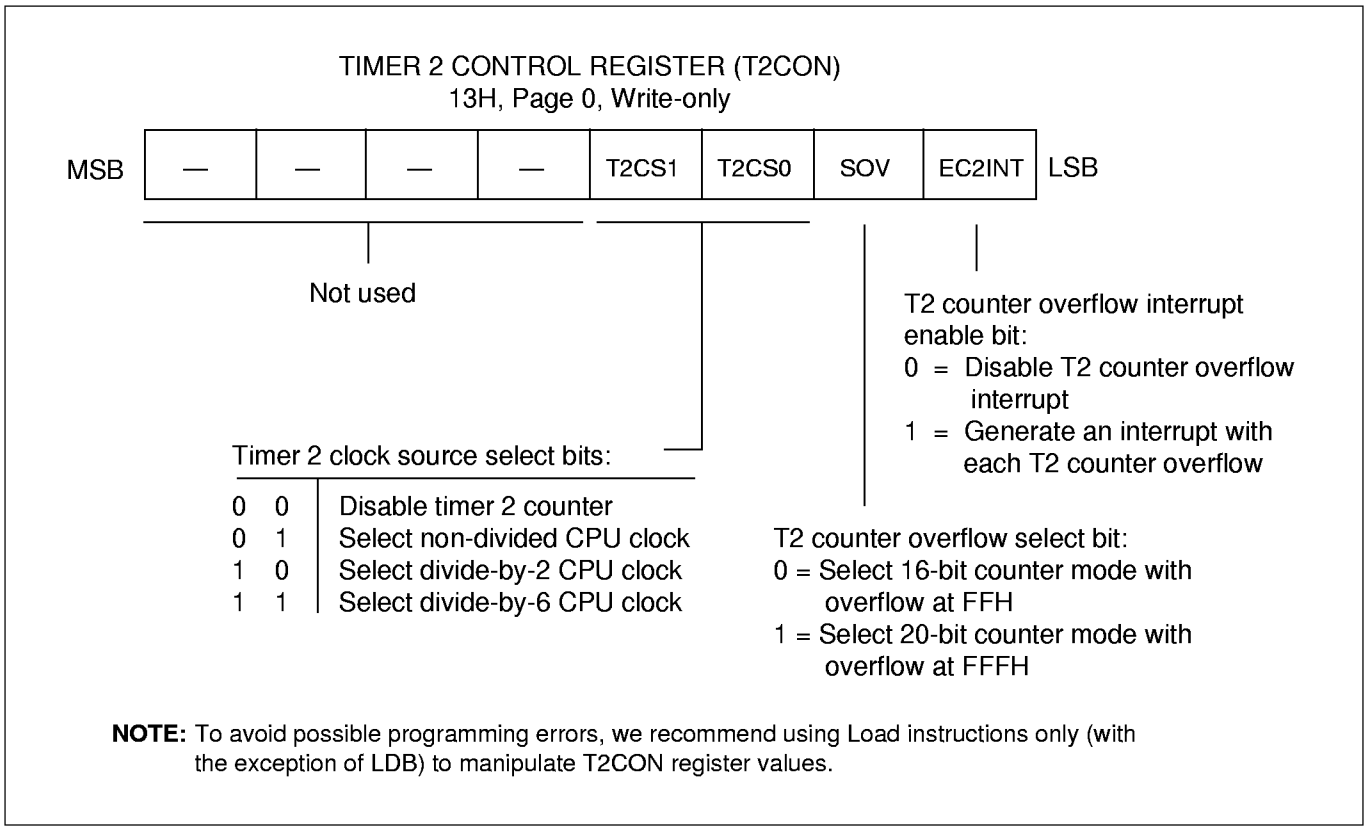


Figure 37. Timer 2 Control Register (T2CON)

TIMER 2 MODULES 0, 1, and 2 (T2M0, T2M1, T2M2)

Modules T2M0, T2M1 and T2M2 are 16-bit capture-only modules. Each module has its own input pin connected to an 8-bit prescaler for frequency selection.

These three modules can be configured by the CC0 and CC1 bits in their control registers to operate in four different modes. When CC0 and CC1 are both "0", the corresponding module is disabled.

To start capture processing, a signal is input at the module's input pin and is directed to the 8-bit prescaler. The counter 2 value is then latched into each module's capture register on the triggering edge (rising, falling, or both edges, as selected by CC0 and CC1 bits). The CPU can then read this value from the capture register.

If the module's EINT bit is set, the triggering signal edge also generates a module interrupt (vector F8H for T2M0, F6H for T2M1, and F0H for T2M2).

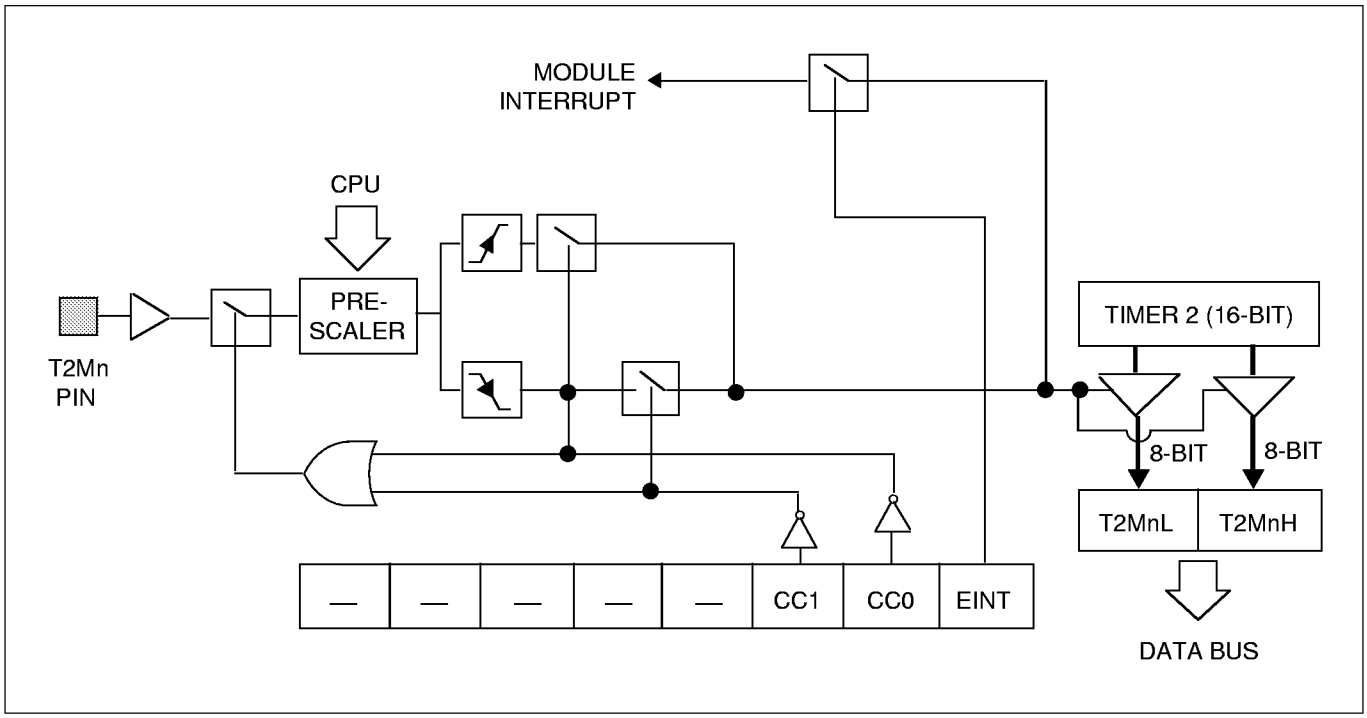


Figure 38. Timer 2 Modules 0, 1, and 2 Function Diagram

TIMER 2 MODULES 0, 1, and 2 CONTROL REGISTERS
T2M0CON: 17H, Page 0, Write-only
T2M1CON: 18H, Page 0, Write-only
T2M2CON: 19H, Page 0, Write-only

MSB									LSB
	—	—	—	—	—	CC1	CC0	EINT	

Not used

T2Mn interrupt enable bit:
0 = Disable T2Mn interrupt
1 = Enable T2Mn interrupt

T2Mn capture mode edge selection bits:

0	0	Disable timer 2 module n operation
0	1	Capture on rising signal edge only
1	0	Capture on falling signal edge only
1	1	Capture on both edge types

NOTES:
1. n = 0, 1, and 2
2. To avoid possible programming errors, we recommend using Load instructions only (with the exception of LDB) to manipulate T2MnCON register values.

Figure 39. Control Registers for Timer 2 Modules 0, 1, and 2

TIMER 2 MODULE 3 (T2M3)

Timer 2 module 3 (T2M3) is a 16-bit compare and capture module. T2M3 is configured by the CC0 and CC1 bits in the control register T2M3CON to operate in one of four modes.

An 8-bit prescaler controls clock input. Timer 2 module 3 is the only timer module that has high current drive capability.

In compare mode, the counter 2 value is compared with the contents of the T2M3 compare/capture register. When these values are equal, a match signal is generated that inverts the toggle register value. (The toggle value is the LSB of the 8-bit read-write register at location 23H in page 0.)

If the EINT bit (bit 0 in of the T2M3CON register) is set, the match signal also generates a T2M3 interrupt (vector F4H).

In capture mode, the counter 2 value is latched into the capture register on every signal edge at the T2M3 pin.

The triggering edge is selected by the CC0 and CC1 bit settings as rising, falling, or both edge types.

The CPU can read the captured value directly from the capture register. If the EINT bit is set, the triggering edge at the T2M3 pin generates a T2M3 interrupt.

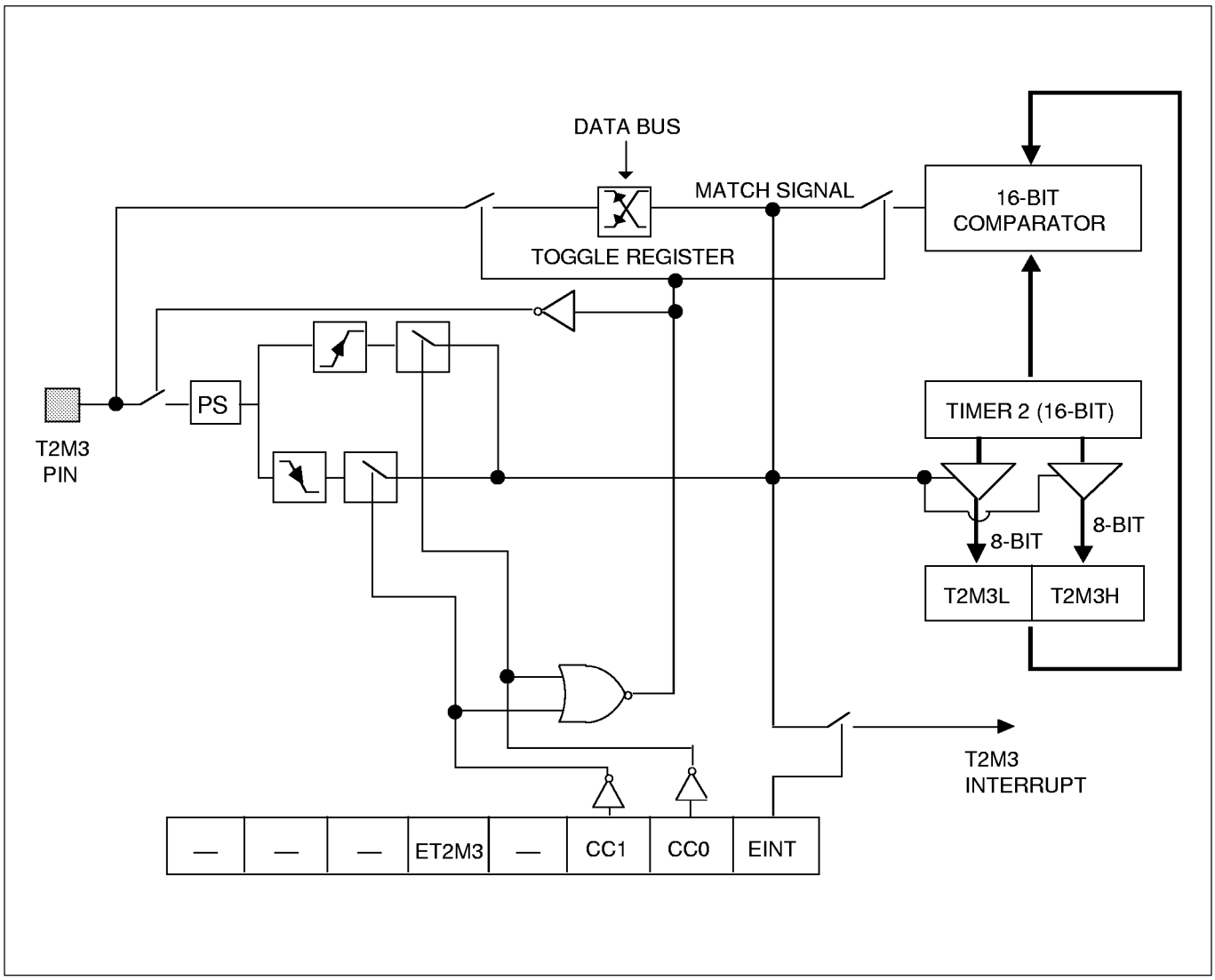


Figure 40. Timer 2 Module 3 Function Diagram

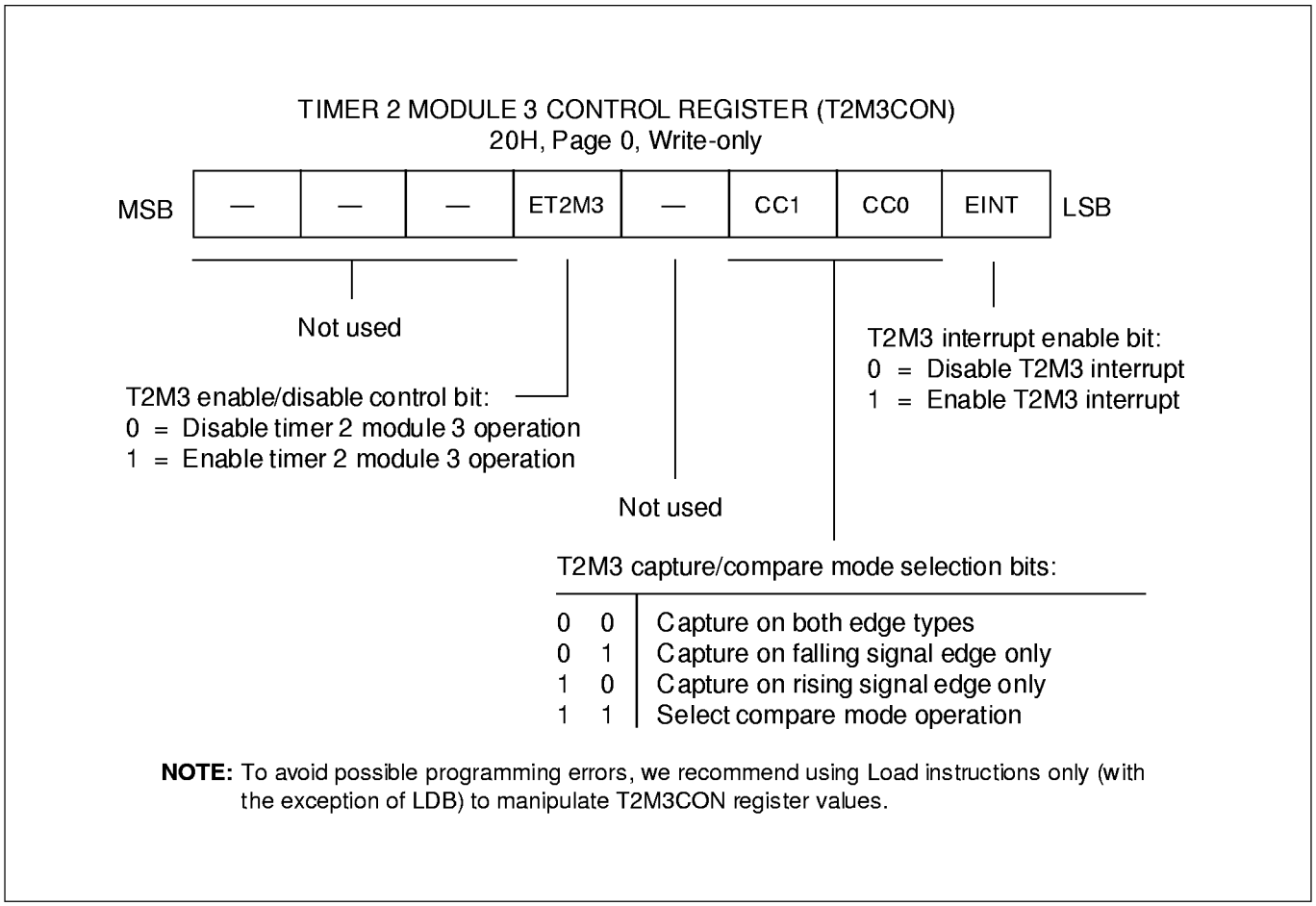


Figure 41. Timer 2 Module 3 Control Register (T2M3CON)

TIMER 2 MODULE 4 (T2M4)

Timer 2 module 4 is a 16-bit compare/capture module. The CC0 and CC1 bits in the module control register T2M4CON select one of four operating modes.

Module 4 differs from the other modules in the Timer 1 or Timer 2 arrays in that it interfaces with the V-sync separator unit.

Figure 43 shows how the input to T2M4 can be either the C-sync signal, the V-sync output of the V-sync separator unit, or an external signal at the T2M4 pin.

The SIG0 and SIG1 bits in the T2M4 control register select that of these signals is used as input.

In compare mode, the counter 2 value is compared with the contents of the module's capture/compare register.

When these values are equal, a match signal is generated that inverts the value of the toggle register.

For T2M4, the toggle register is bit 5 of the T2M4CON control register. Bit 5 (TGL) can be read and written by the CPU.

If the EINT bit (T2M4CON.0) is set, the match signal also generates a T2M4 interrupt (vector F2H).

In capture mode, the counter 2 value is latched into the capture register at every triggering edge of the T2M4 pin (rising, falling, or both edge types, as selected by the CC0 and CC1 bits).

The CPU can read this value directly from the capture register. If the EINT bit is set, this signal edge also generates a T2M4 interrupt.

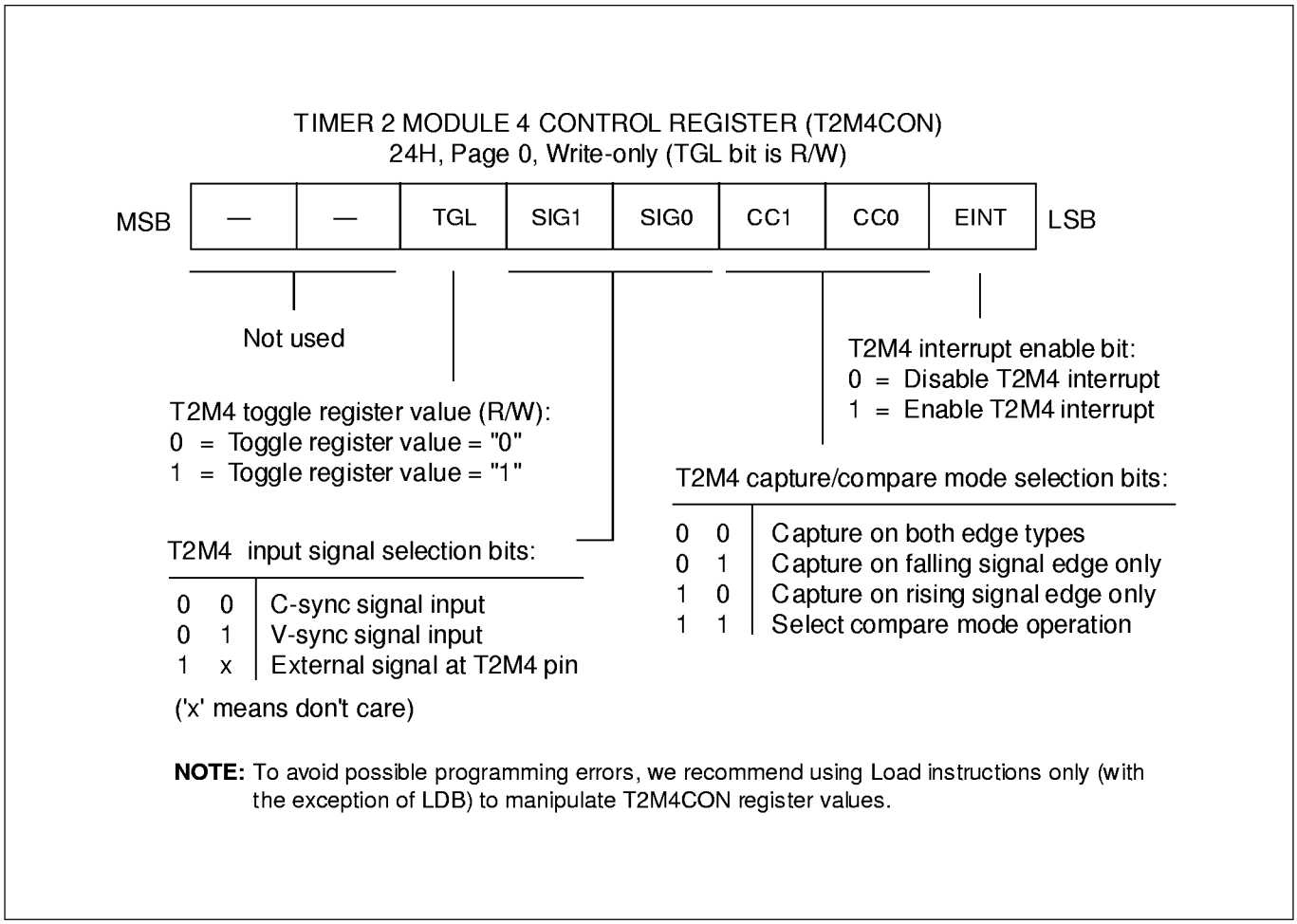


Figure 42. Timer 2 Module 4 Control Register (T2M4CON)

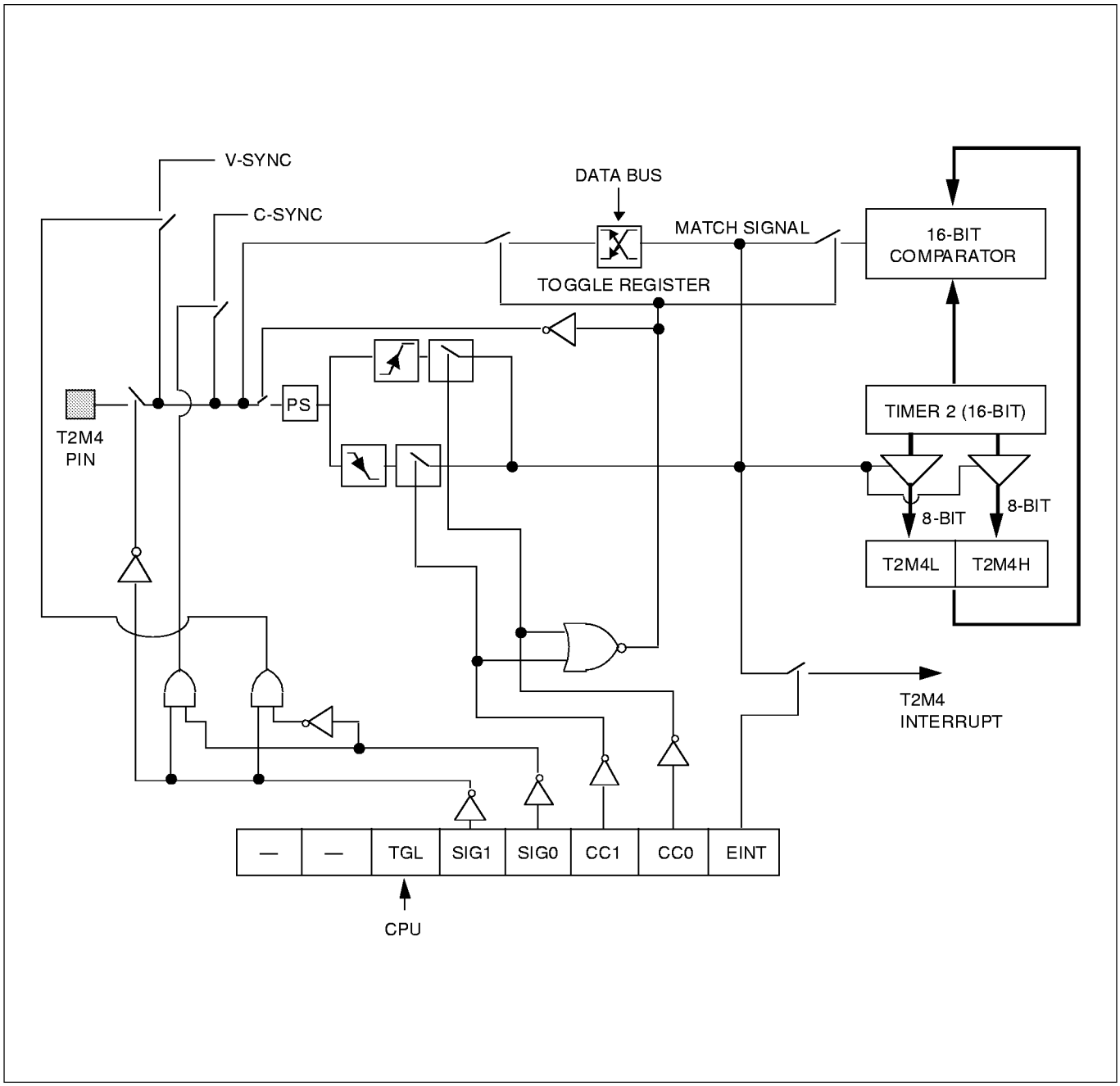


Figure 43. Timer 2 Module 4 Function Diagram

V-SYNC SIGNAL SEPARATOR

The V-sync separator module separates the vertical synchronization (V-sync) signal from the composite synchronization (C-sync) signal.

The C-sync signal consists of two signals: a horizontal synchronization (H-sync) signal and the V-sync signal.

V-sync and H-sync signals are sometimes referred to as 'vertical retrace' and 'horizontal retrace' signals, respectively.

The H-sync signal occurs at just over 15 kHz (NTSC standard) and appears on an oscilloscope as a series of narrow pulses. The H-sync pulse width is approximately 4.8 μs.

The V-sync signal's duration is approximately three horizontal scans (3H or < 200 μs). It therefore occurs 60times per second (NTSC standard).

During the vertical retrace interval, the phase of the H-sync pulses is inverted. (The inverted H-sync pulse is not completely identical with the V-sync pulse.) The V-sync separator can then detect the inverted pulses by measuring pulse widths.

The V-sync portion of a C-sync signal has a pulse width greater than 27 μs. When this occurs, the state of the V-sync output pin goes high after approximately 8.75 to 9 μs. The separator unit then counts seven falling edges of the C-sync signal, and sends the V-sync pin's level low.

The width of the V-sync signal is three H-sync cycles minus the V-sync delay, or about 128μs. The V-sync signal output can be selected as the input of timer 2 module 4 by setting the SIG1 and SIG0 bits in the T2M4CON register to '01B'.The V-sync signal has the following timing relationship:

$$8.75\ \mu s < t_D < 9\ \mu s$$

t_W (width of separated V-sync signal) = $3\ H - t_D$

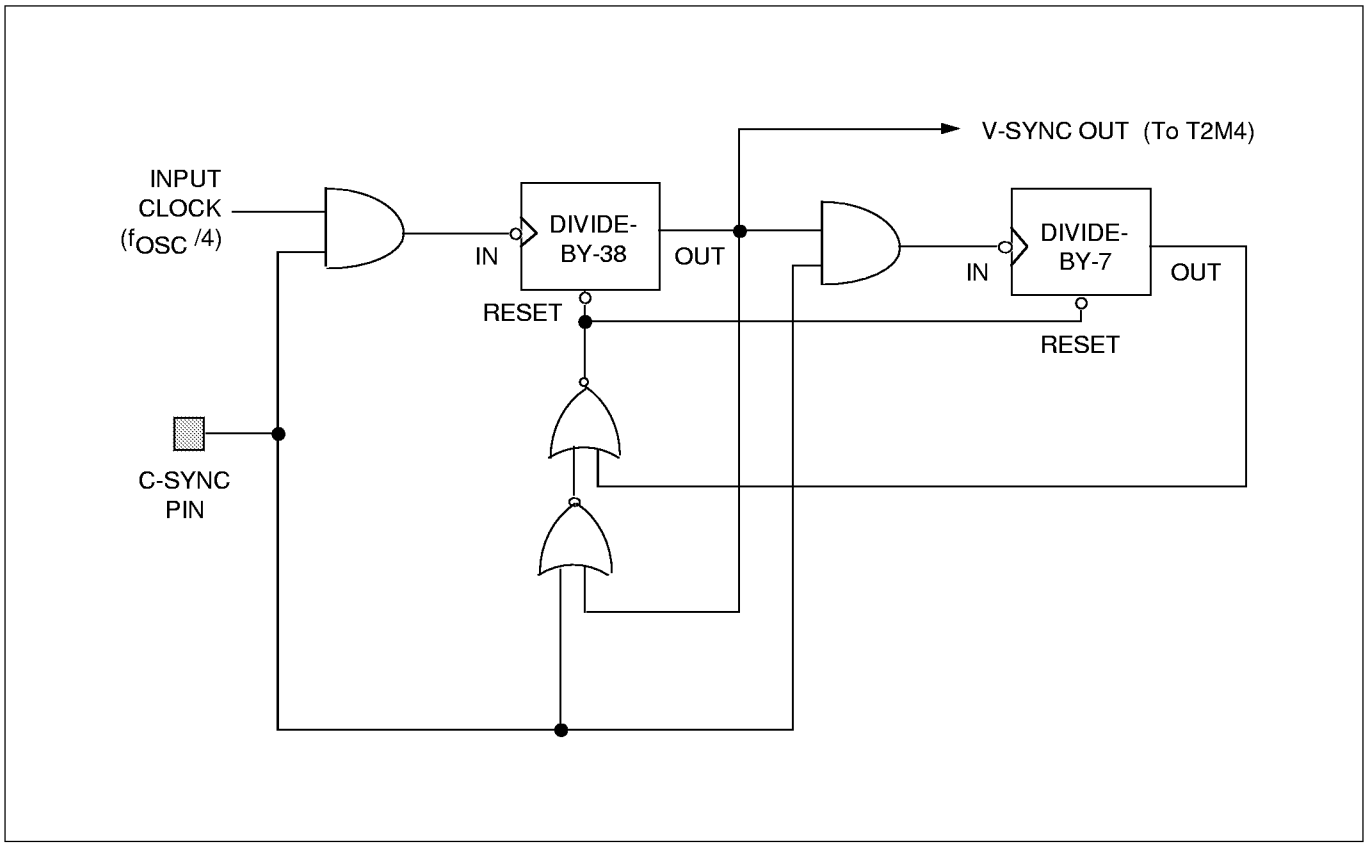
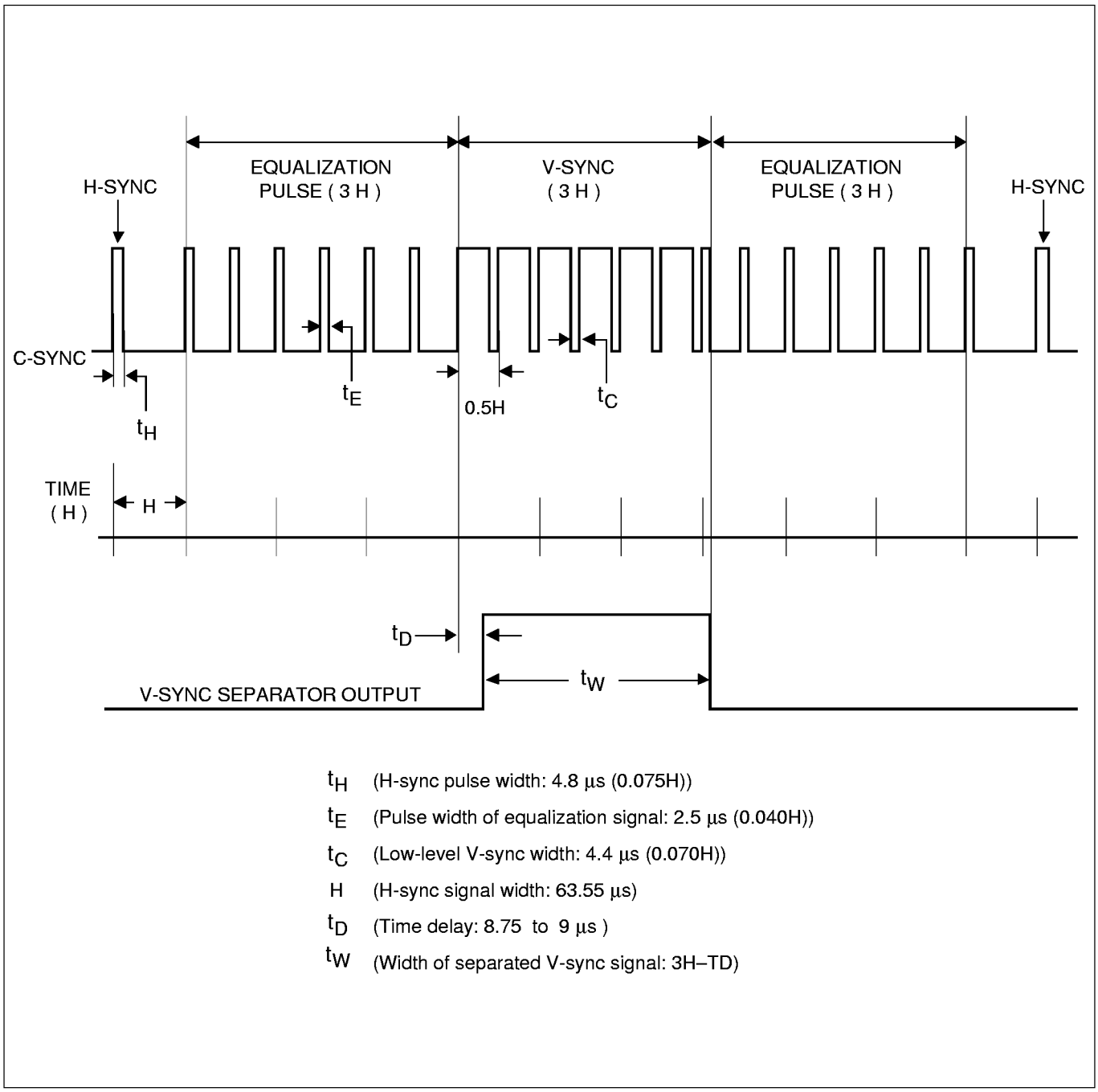


Figure 44. V-Sync Signal Separator Function Diagram

SEPARATING ODD OR EVEN FRAMES OF THE V-SYNC SIGNAL

The V-sync separator can detect odd or even signal frames by software. To separate odd and even V-sync signals (in a video recorder application, for example), the SIG1 and SIG0 bits and the T2M4 control register T2M4CON must first be cleared to logic zero. This selects the C-sync signal as the timer 2 module 4 input. A program can then be written to separate the odd frames (or even frames) using timer 2 module 4 in capture mode.



t_H (H-sync pulse width: 4.8 µs (0.075H))

t_E (Pulse width of equalization signal: 2.5 µs (0.040H))

t_C (Low-level V-sync width: 4.4 µs (0.070H))

H (H-sync signal width: 63.55 µs)

t_D (Time delay: 8.75 to 9 µs)

t_W (Width of separated V-sync signal: 3H–TD)

Figure 45. C-Sync Signal Timing Diagram

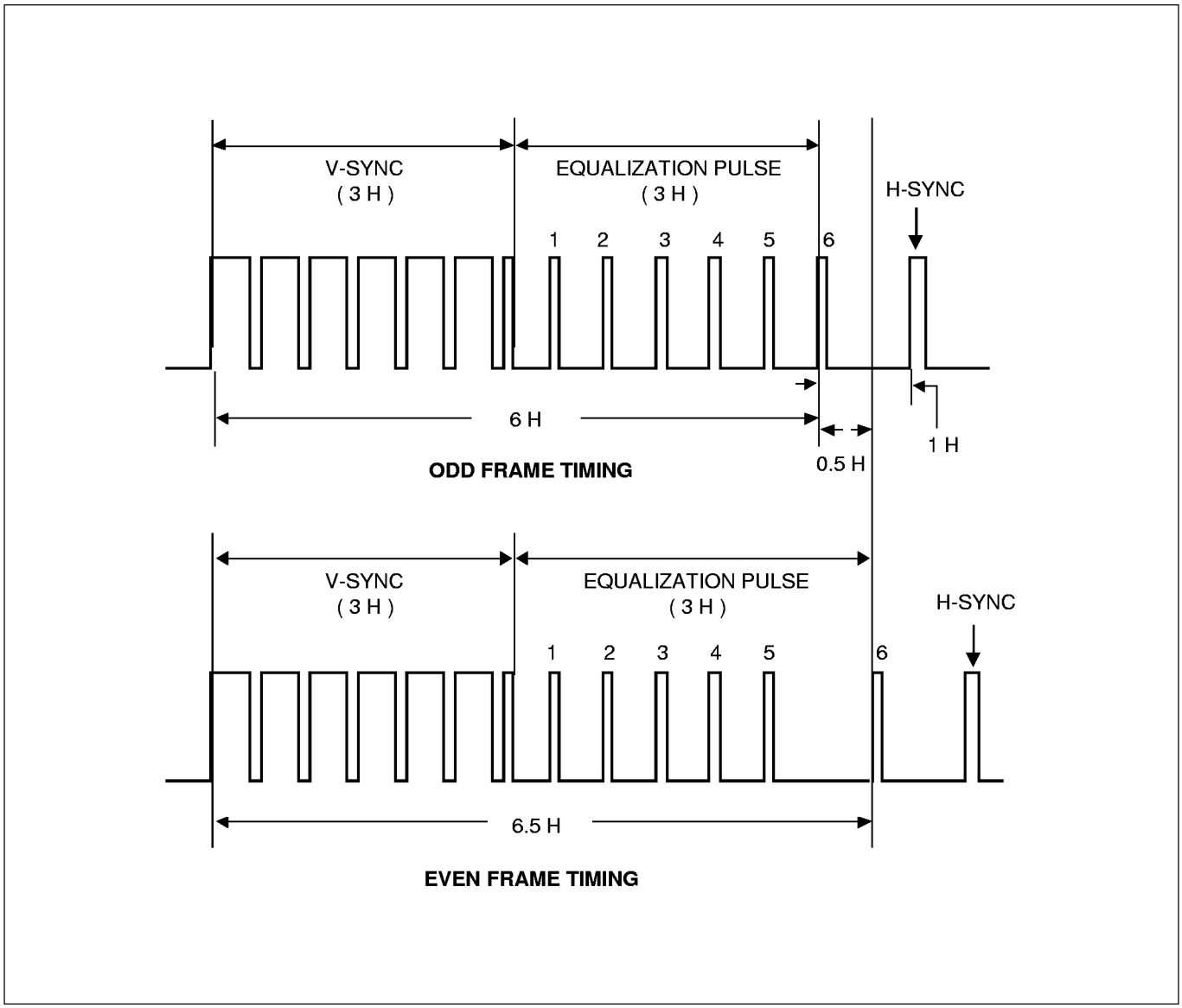


Figure 46. V-Sync Odd/Even Frame Timing Diagram

TIMER 2 MODULE 5 (T2M5)

Timer 2 module 5 (T2M5) is a pattern generator with four output pins (T2M5.0–T2M5.3). It is functionally similar to the T1M1 module. Output pins T2M5.0–3 are connected to the output register T2M5OUT. In addition, T2M5 also has a pseudo H-sync output function for pin T2M5.3.

T2M5 has the following functional components:

— Control register (T2M5CON)
— 16-bit compare register (T2M5H and T2M5L
— Pattern generator register (T2M5PG)
— 4-bit output register (T5M5OUT)
— Output pins (T2M5.0–T2M5.3)

The T2M5.2 output pin has high current drive capability. The T2M5.3 output pin also can be configured as a pseudo H-sync output in either NTSC or PAL standard format.

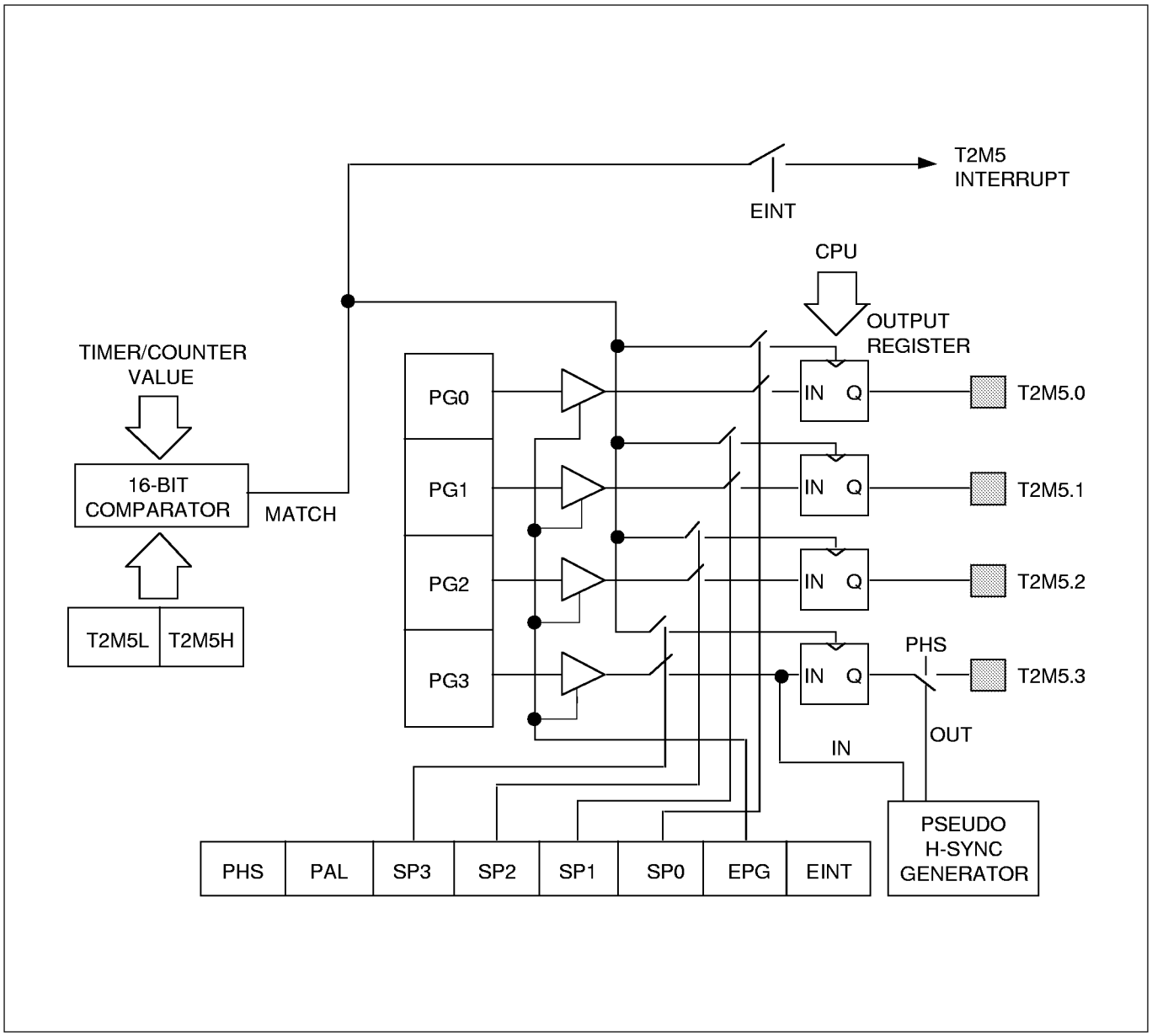


Figure 47. Timer 2 Module 5 Function Diagram

TIMER 2 MODULE 5 CONTROL REGISTER (T2M5CON)

The T2M5 control register T2M5CON (01H, page 0, write-only) has five functions:

— Enable/disable pattern generator output to each T2M5 output pin (bit 1)

— Enable/disable the T2M5 interrupt (bit 0)

— Select pseudo H-sync output mode for pin T2M5.3

— Configure pseudo H-sync waveform output to either NTSC or PAL standard

The four select pin (SP) bits in the T2M5CON register, bits 2–5, correspond directly to the four T2M5 output pins T2M5.0–T2M5.3. The pin's SP bit value, with the value of the EPG (Enable Pattern Generator) bit , T2M5CON.1, selects one of the operating modes for that pin:

— Normal mode (direct output)

— Toggle mode

— Pattern generator mode (next-state programming mode)

A reset clears the T2M5CON register to 00H, configuring all T2M5 pins to normal (direct output) operating mode. An SP value of "1" selects an alternative operating mode:

— If SPn = "1" and EPG = "0" then toggle mode is selected for that pin;

— If SPn = "1" and EPG = "1" the pattern generator output is enabled to the output pin, causing it to operate in pattern generation mode.

In addition, if T2M5CON bit 7 (PHS) is set to "1", the T2M5.3 pin no longer operates according to the corresponding SP3 and EPG bit setting, but outputs a pseudo H-sync waveform instead.

If bit 6 (PAL) is "0", the pseudo H-sync waveform conforms to NTSC standard. Otherwise, it conforms to the PAL standard.

To avoid possible programming errors, we recommend that you use Load instructions only (except for LDB, Load Bit) when manipulating T2M5CON register values.

Direct Output Mode

In direct output mode, data in the 4-bit output register are latched to the four output pins, T2M5.0–T2M5.3. In this way, the CPU writes data directly to the T2M5 pins by writing the output register T2M5OUT (04H, page 0).

Toggle Mode

In toggle mode, when the counter2 value matches the 16-bit T2M5 compare register value, a match signal is issued to toggle the corresponding pin's value in the output register. If the EINT bit (T2M5CON.0) is set, the match signal generates a T2M5 interrupt (vector E4H).

Each output pin can be programmed individually to operate in toggle mode. For example, if SP0–SP2 are "1" and SP3 = "0" (and EPG = "0"), then pins T2M5.0–T2M5.2 operate in toggle mode; T2M5.3 operates in direct output mode, assuming PHS = "0".

Pattern Generation Mode

In pattern generation mode, a match condition causes the data in the pattern generator register (T2M5PG, 06H, page 0) to be loaded into the output register, T2M5OUT. (The CPU would have previously written the data to the pattern generator register.)

The match signal generates a T2M5 interrupt if the EINT bit is "1".

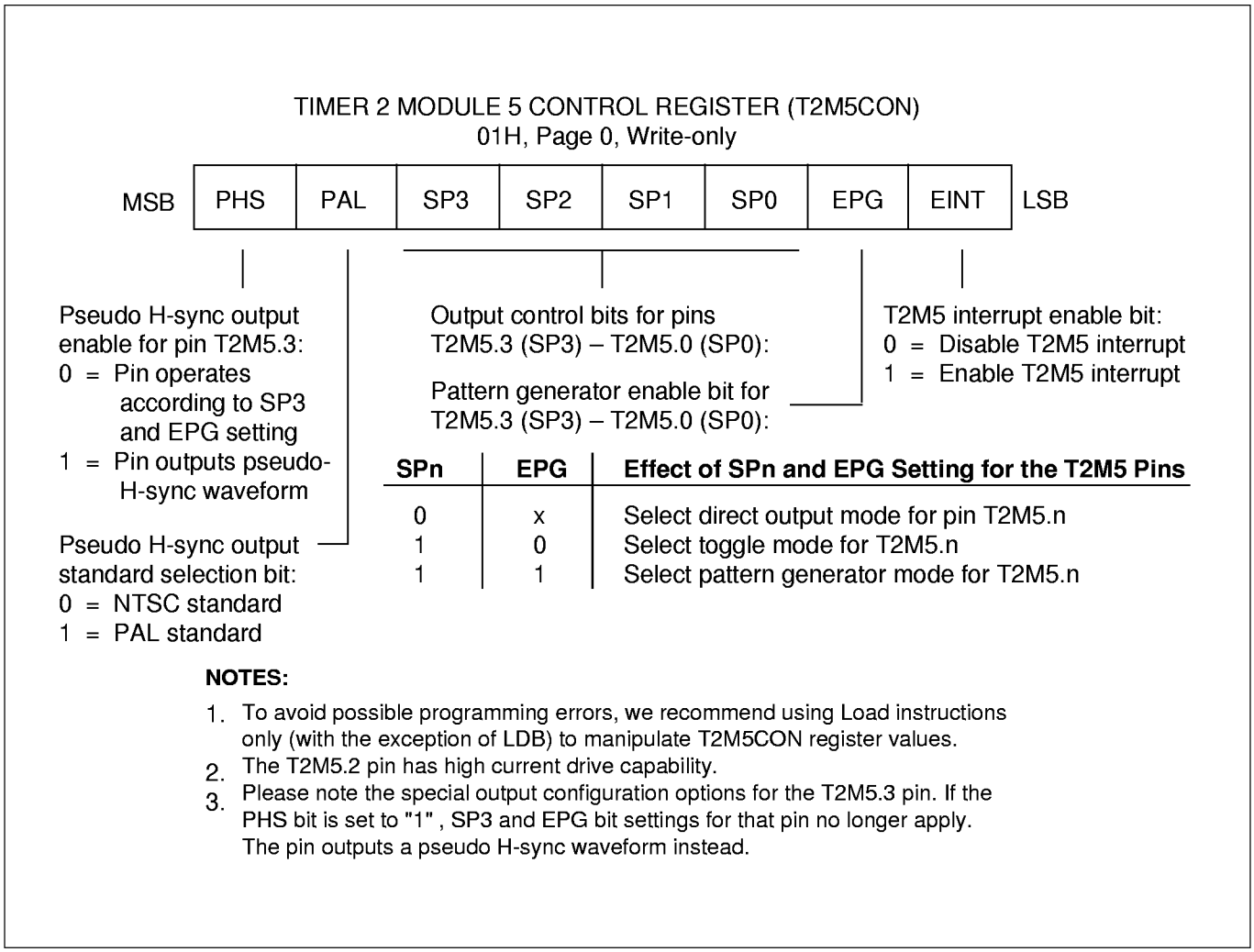


Figure 48. Timer 2 Module 5 Control Register (T2M5CON)

PSEUDO H-SYNC GENERATOR

Pseudo H-sync signals can be used for special playback features in VCR applications such as 'freeze picture' or search mode, or by an OSD device. The H-sync generator outputs a horizontal sync signal at the T2M5.3 pin when the PHS bit in the T2M5CON register is "1". Otherwise, the T2M5.3 pin operates in the same way as the other three T2M5 pins: in pattern generation mode, toggle mode, or direct output mode.

When the pseudo H-sync output is selected, the waveform that is output at the T2M5.3 pin appears on an oscilloscope as a series of narrow pulses.

The PAL bit setting determines whether the H-sync waveform conforms to the PAL or NTSC standard:

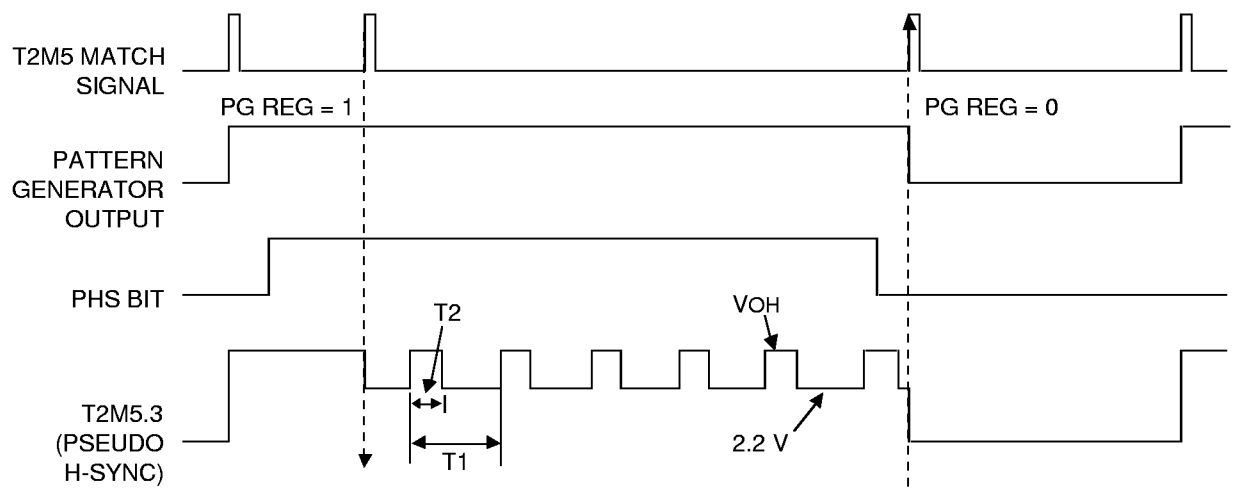
— When PAL = "1", the H-sync pulse frequency is 510 clocks (63.75 μs with an 8-MHz CPU clock)

— When PAL = "0," the H-sync pulse frequency is 508 clocks (63.50 μs with an 8-MHz CPU clock)

— The width of an H-sync pulse is always 41 clocks (5.125 μs)

The voltage swing of the pseudo H-sync waveform is from V_{OH} down to about 2.2 V. The T2M5.3 pin is designed to source or sink 20 mA.

Depending on the pin's load, the offset can be up to 0.8 V, giving a level of 3 V. We recommended using a 3-KΩ pull-down resistor for most applications.



If PAL = "1" (PAL standard output)
T1 = 510 clock cycles (63.75 μs)
T2 = 41 clock cycles (5.125 μs)

If PAL = "0" (NTSC standard output)
T1 = 508 clock cycles (63.5 μs)
T2 = 41 clock cycles (5.125 μs)

NOTE: Clock measurements assume an 8 MHz CPU clock rate.

Figure 49. Pseudo H-Sync Generator Timing Diagram

PWM MODULES

OVERVIEW

The KS88C8016 pulse width modulation (PWM) function block has two PWM modules, PWM0 and PWM1. Each module has the following components:

— 8-bit counter with 6-bit extension counter for 14-bit output resolution
— 8-bit reference data register
— 8-bit comparator
— 8-bit extension data register (6 bits are used)
— PWM output pin

Both PWM modules use a single control register, PWMCON. It contains a 2-bit prescaler to permit modification of the CPU clock frequency. By comparing the extension counter value with the extension register value, the duty cycle of the PWM output is "stretched" at specific intervals.

PWM output can be used, for example, in voltage synthetic tuning (VST) tasks in color televisions, and to control on/off timing for external circuits that require high base frequencies.

PWM CLOCK RATE

The timing characteristics of both output channels are identical, and are based on the maximum 12-MHz CPU clock frequency. The 2-bit prescaler value in the PWMCON register determines the frequency of the counter clock: Bit-pair 6/7 (PS0/ PS1) can be set to divide the CPU clock frequency by 1 (non-divided), or by 2, 3, or 4. Because the maximum CPU clock rate for the KS88C8016 is 12 MHz, the base PWM frequency is 46.87 kHz (12MHz divided by 256). This assumes a non-divided CPU clock.

PWM CONTROL REGISTER (PWMCON)

The control register for the PWM0 and PWM1 modules, PWMCON, is at register address 3CH in page0. Bit settings in this write - only register control two PWM functions:

— PWM counter stop/start (resume) operation (ECTR bit)
— 2-bit prescaler value for PWM counter clock input (PS1 and PS0 bits)

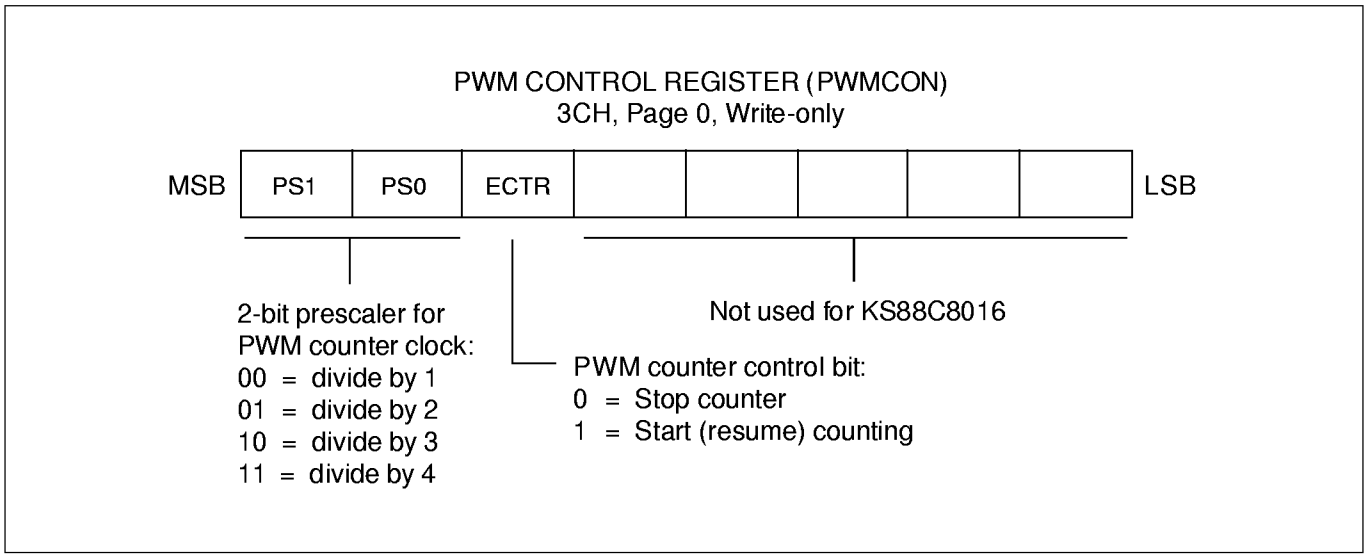


Figure 50. PWM Control Register (PWMCON)

PWM DATA AND EXTENSION REGISTERS

Two special-purpose write-only registers, located in page 0 of the internal register file, determine the PWM output value generated by each module:

— 8-bit data registers PWM0 (2CH) and PWM1 (2EH)

— 8-bit extension registers PWM0EX (2DH) and PWM1EX (2FH), of which only six bits are used for the extension value

To start the PWM counter, bit 5 (ECTR) in the PWMCON register must be set to "1".Then, to program the desired PWM output, initialization values must be loaded into the 8-bit data registers (PWM0, PWM1) and the 6-bit extension registers (PWM0EX, PWM1EX). See Table 14.

A reset operation disables all PWM output. The current counter value is retained when the counter stops. Then, when the counter starts, counting resumes at the retained value.

Table 14. PWM0 and PWM1 Control and Data Registers

Register Name	Mnemonic	Address (Page 0)	Function
PWM0 Data Registers	PWM0EX	2DH	6-bit extension ("stretch") value
	PWM0	2CH	8-bit PWM0 basic cycle frame value
PWM1 Data Registers	PWM1EX	2FH	6-bit extension ("stretch") value
	PWM1	2EH	8-bit PWM1 basic cycle frame value
PWM Control Register	PWMCON	3CH	PWM0/1 counter stop/start (resume), and 2-bit prescaler for CPU clock

PWM COUNTER

The 16-bit PWM counter has a lower byte counter and an upper byte counter. The lower byte counter is compared to the PWM data register value to determine the PWM module's base operating frequency.

The six most significant bits of the upper byte counter toggle with each overflow of the lower counter.

The 6-bit extended counter value is compared with the 6-bit value written to the module's extension register (PWM0EX or PWM1EX) to control the "stretching" of the PWM output duty cycle at specific intervals.

PWM FUNCTION DESCRIPTION

The PWM output value toggles whenever the lower 8-bit counter matches the reference value stored in the module's data register (PWM0 or PWM1). That is, the overflow value of the lower counter shifts to the upper 6-bit counter. In this way, the reference value written to the data register determines the module's base duty cycle.

The PWM output toggles again based on the module's selected operating frequency (46.87 kHz with 12-MHz CPU clock input). The six most significant bits of the 8-bit upper counter (bits 8–13) toggle on each match of the data register and lower counter, and at a frequency determined by the counter clock.

The value in the 6-bit extension counter is compared with the PWM extension logic and the settings in the 6-bit extension data register (PWM0EX, PWM1EX). This 6-bit extension counter value, together with extension logic and the PWM module's extension register, is then used to "stretch" the duty cycle of the PWM output. The "stretch" value is one extra cycle at specific intervals of the base duty cycle (see Table15).

If, for example, the value in the extension register is 1, the 32nd output pulse will be one cycle longer than the other 63 pulses. If the base duty cycle is 50%, the duty of the 32nd pulse will therefore be "stretched" to approximately 51% duty. If you write 20H to the extension, all odd-numbered pulses will be one cycle longer. If you write 3FH to the extension register, all pulses will be stretched by one cycle except the 64th pulse.

PWM output goes to an output buffer and then to the corresponding PWM0 or PWM1 output pin: PWM0 (pin 33) or PWM1 (pin 34).

In this way, you can get high output resolution at high frequencies. If the PWM frequency remains high enough, the output level of a simple low-pass filter will be sufficient to ensure the smooth operation of a servo-motor.

Table 15. PWM Output "Stretch" Values for Extension Registers PWM0EX and PWM1EX

PWM0/1EX Bit	Cycle Number That Is "Stretched"
7	Not used
6	Not used
5	1, 3, 5, 7, 9, . . . , 55, 57, 59, 61, 63
4	2, 6, 10, 14, . . . , 50, 54, 58, 62
3	4, 12, 20, . . . , 44, 52, 60
2	8, 24, 40, 56
1	16, 48
0	32

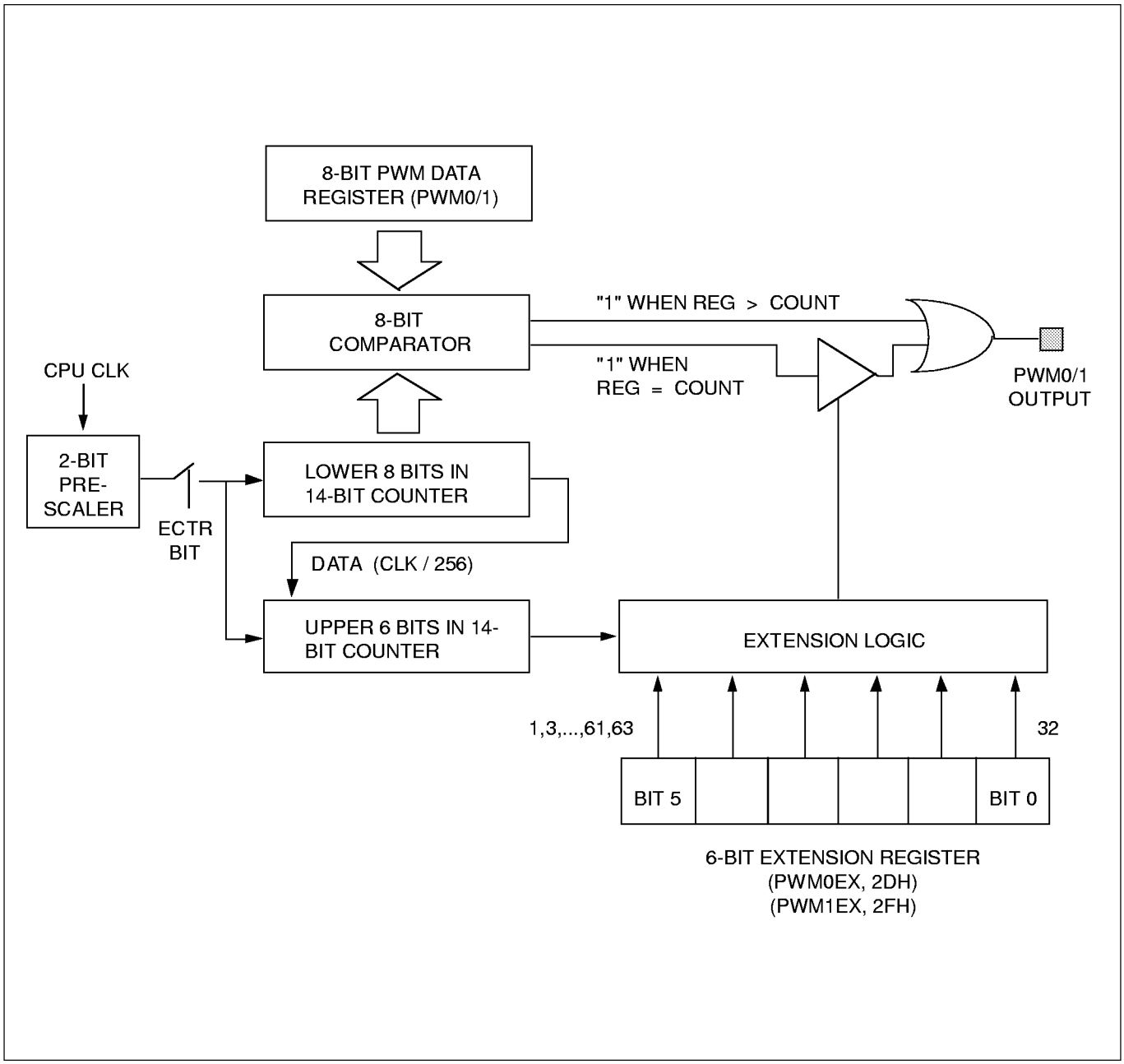


Figure 51. Function Diagram for PWM Modules 0 and 1

☞ PROGRAMMING TIP — Programming the 14-Bit PWM Modules to Sample Specifications

This example shows how to program the 14-bit pulse-width modulation module PWM0. (The procedure for programming the PWM1 module is identical.) The program parameters are as follows:

— The oscillation frequency of the main crystal is 8 MHz

— PWM0 data is in working register R0

— PWM0EX (PWM0 extension value) is in working register R1, bits 0–5

The program should perform the following operations:

1. Set the PWM0 frequency to 31.25 kHz
2. If R3.0 = "1", then PWM ← PWM + 12H
   (If an overflow occurs from R0, then R0 ← 0FFH and R1 ← 3FH.)
3. If R3.0 = "0", then PWM ← PWM – 11H
   (If an underflow occurs from R0, then R0 ← 00H and R1 ← 00H.)

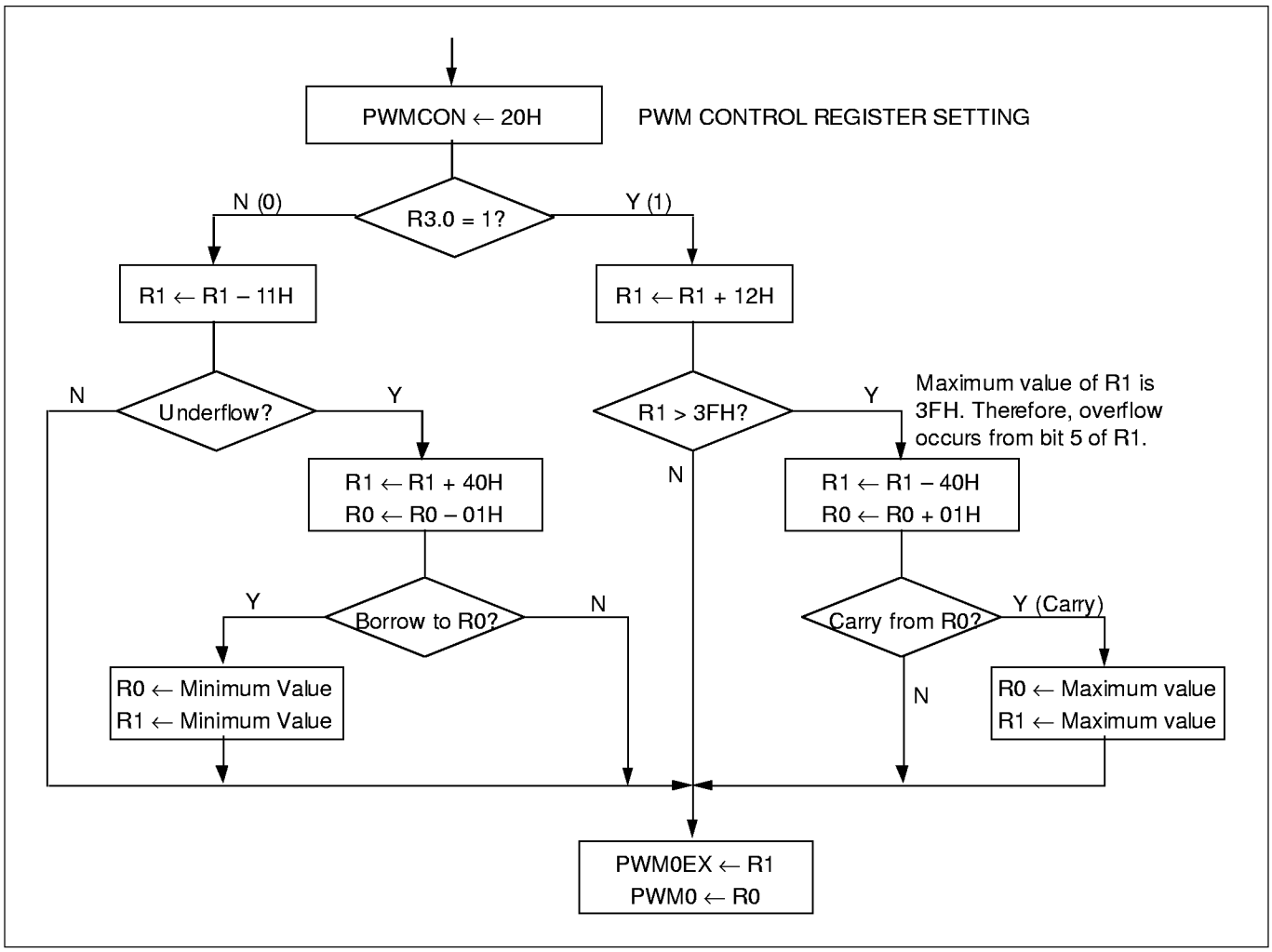


Figure 52. Decision Flowchart for PWM0 Programming Tip

☞ **PROGRAMMING TIP — (Continued)**

```
        •
        •
        •
        LD      PWMCON,#20H     ;  PS ← 0 (Select 31.25-kHz PWM frequency)
                                ;  Enable the PWM counter
        •
        •
        •
        BTJRF   DECR,R3.0       ;  If R3.0 = "0", then jump to DECR
INCR    ADD     R1,#12H         ;  If R3.0 = "1", then add 12H to the PWM data
        CP      R1,#3FH         ;  Since PWM0EX value is 6-bit, maximum value is 3FH
        JP      ULE,AAA         ;  If R1 ≤ 3FH, go to AAA
        SUB     R1,#40H         ;  PWM0EX (= R1) > 3FH, PWM0EX (R1) ← PWM0EX –
40H
        INC     R0              ;  R0 ← R0 + 1H
        JP      NZ,AAA          ;  If no overflow, jump to AAA for update
        LD      R0,#0FFH        ;  If overflow, set 0FFH to R0
        LD      R1,#3FH         ;  Set 3FH to R1
        JR      T,AAA           ;  Jump to AAA unconditionally
DECR    SUB     R1,#11H         ;  R3.0 = "0", so subtract 11H from PWM data
        JP      NC,AAA          ;  If no borrow, jump to AAA for update
        ADD     R1,#40H         ;  PWM0EX (= R1) < 0, PWM0EX (R1) ← PWM0EX + 40H
        SUB     R0,#01H         ;  Decrement R0 (R0 ← R0 – 1)
        JP      NC,AAA          ;  If no borrow, jump to AAA
        CLR     R0              ;  If borrow, set 00H to R0 and R1
        CLR     R1              ;
AAA     LD      PWM0EX,R1       ;  Load new value to PWM0EX (= 6 bits)
        LD      PWM0,R0         ;  Load new value to PWM0
        •
        •
        •
```

BACKUP TIMER

OVERVIEW

The 16-bit backup timer can be used to update the current time for application-specific clock time functions.

A suboscillator clock pulse drives the backup timer. This clock pulse is divided by 32768 (2^{15}) Hz to produce the internal timer clock. (A 32768-Hz crystal generates a slow 1-Hz clock pulse).

To start the backup timer, you write any value to its control register BTCON (E7H). Because the counter value itself cannot be written, this write operation simply clears the counter to ensure that counting will resume from 0000H. The CPU can always read the current counter value.

NOTE

The interrupt generation function controlled by BTCON is not included in the KS88C8016.

The first read that occurs after a write operation reads the *upper* byte of the counter (bits 15–8); the second read will read the *lower* byte (bits 7–0). When the counter reaches a value of 0FFFFH, an overflow signal is generated. With the 32768-Hz crystal, this occurs about 18.2 hours after the initial write operation.

In the event of a power outage, the backup timer can be used to restore the current clock time value when power is restored. This assumes that a secondary power source is available. A secondary power source allows volatile data stored in RAM to be retained during a power outage.

Programs written for real-time counting and timing during power failures generally use this sequence: When power goes out, the backup timer is cleared and real-time counting begins as the CPU enters Stop mode. When power is restored, the CPU "wakes up" and checks the back-up timer to see how long it has been "asleep." It then adds the backup timer count value to the stopped main clock value, giving the correct time.

BACKUP TIMER FUNCTION DESCRIPTION

You activate the backup timer by writing a value to its control register BTCON (address E7H).

Because the counter value cannot be written, this write operation resets the counter to 0000H and starts the counter. The CPU can always read the counter value at address E7H.

The first read operation (and all subsequent odd-numbered reads) after a write operation reads the upper byte of the counter (bits 15–8); the second read operation (and subsequent even-numbered reads) after a write operation reads the lower byte (bits 7–0).

When the counter reaches a value of FFFFH, the timer overflows and automatically resumes counting at 0000H. The reset value of the 16-bit counter is undefined.

BACKUP TIMER CONTROL REGISTER (BTCON)

The backup timer control register, BTCON, serves a dual purpose: to enable the application program to clear the 16-bit time save counter and resume counting, and to make it possible to read the 16-bit time save value when system power is restored.

To clear the time save counter and resume counting from 0000H, any 8-bit value is written to the BTCON register at address E7H.

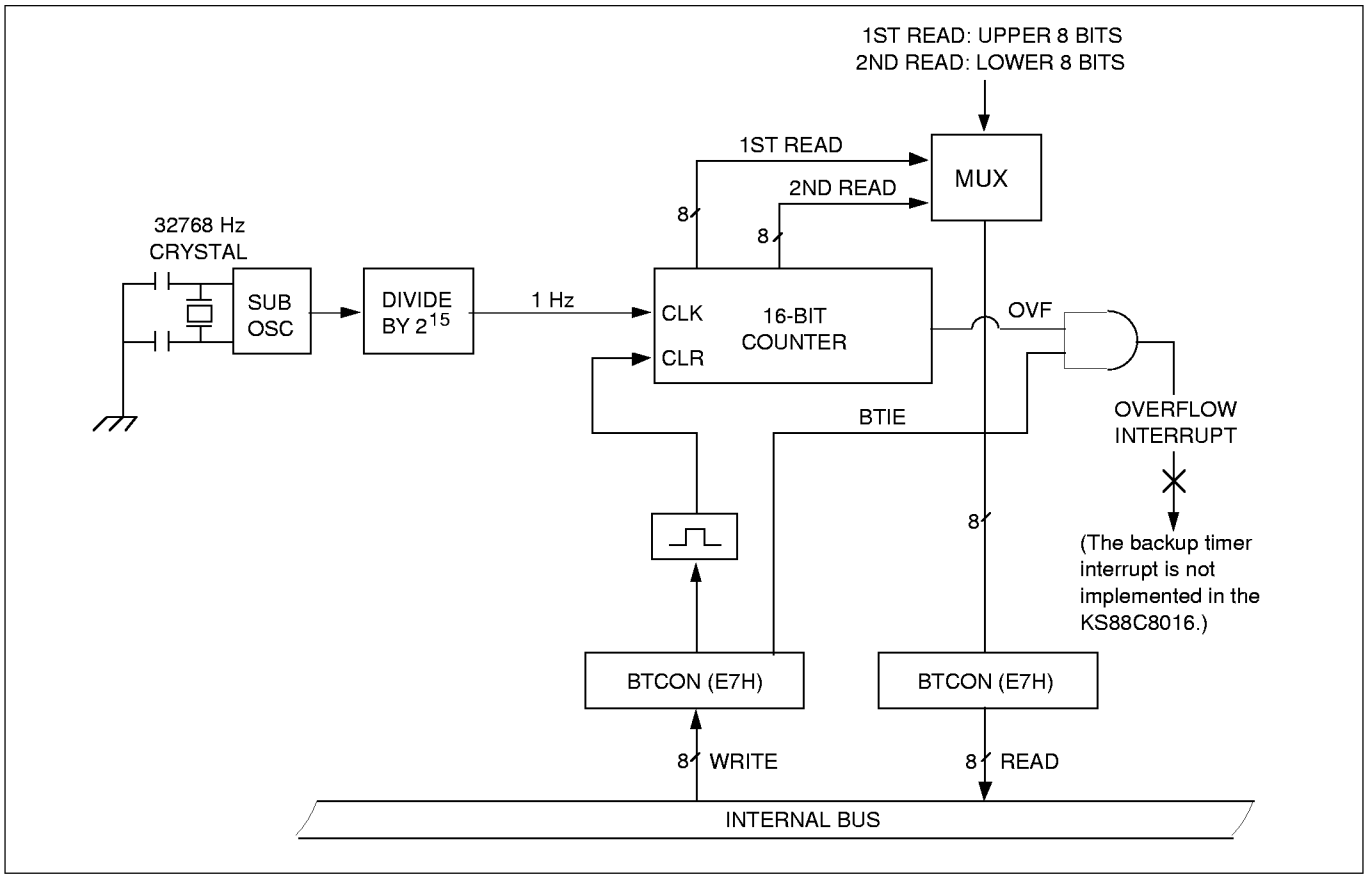


Figure 53. Backup Timer Function Block Diagram

BACKUP TIMER CONTROL REGISTER (BTCON)
E7H, Set 1, R/W

MSB	.7	.6	.5	.4	.3	.2	.1	BTIE	LSB

To clear the backup (time save) timer, simply write any hexadecimal value to the BTCON register.

* The first read operation (and subsequent odd-numbered reads) will read all 8 bits of the upper byte of the 16-bit counter value.

* The second read operation (and subsequent even-numbered reads) will read all 8 bits of the the lower byte of the 16-bit counter value.

Backup timer interrupt control bit:
0 = Disable backup timer interrupt
1 = Invalid setting for KS88C8016

Figure 54. Backup Timer Control Register (BTCON)

☞ PROGRAMMING TIP — Reading Time Save Values from the Backup Timer Control Register

This example shows how to use the backup timer control register BTCON to read upper and lower bytes of the 16-bit time save counter. As you will note, the BTCON register must be read in succession to concatenate the complete value. Given a timer clock frequency of 32768 Hz, the program parameters are as follows:

— Load the lower nibble of the backup timer (low byte) to R0.

— Load the upper nibble of the backup timer (low byte) to R1.

— Load the lower nibble of the backup timer (high byte) to R2.

— Load the upper nibble of the backup timer (high byte) to R3.

```
RESET   NOP
        •
        •
        •
        LD      BTCON,#00H      ; Reset and start the backup timer
        •
        •
        •
MAIN    NOP
        •
        •
        •
        LD      R2,BTCON        ; First (odd time) read value after write to BTCON
                                ; is the upper byte value of the backup timer
        LD      R0,BTCON        ; Second (even time) read value after a write to BTCON
                                ; is the lower byte value of the backup timer
        LD      R1,R0           ; R1 ← R0
        SWAP    R1              ; Upper nibble ↔ lower nibble
        LD      R3,R2           ; R3 ← R2
        SWAP    R3              ; Upper nibble ↔ lower nibble
        AND     R0,#0FH         ; Mask out the upper nibble
        AND     R1,#0FH         ;       "
        AND     R2,#0FH         ;       "
        AND     R3,#0FH         ; Mask out the upper nibble
        •
        •
        •
        JP      T,MAIN          ; For loop
```

A/D CONVERTER

OVERVIEW

The 8-bit A/D converter (ADC) module uses successive approximation logic to convert analog levels at one of the eight input channels to equivalent 8-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
— Successive approximation register
— D/A converter logic (resistor ladder type)
— ADC control register (ADCON)
— Eight multiplexed analog data input pins (AD0–AD7)
— 8-bit A/D conversion data output register (ADOUT)
— 8-bit digital input register (ADIN, port 5)
— AV_{REF} and AV_{SS} input pins

An analog-to-digital conversion procedure is initiated when the CPU writes a value to the ADCON register at address 28H to select the ADn input pin. You select the desired input channel (AD0–AD7) by setting the appropriate SCHn bits in the ADCON register.

During a normal conversion, the successive approximation register is initially set to 80H (the approximate half-way point of an 8-bit register). It is then updated automatically during each conversion step.

The KS88C8016 performs 8-bit conversions for only one input channel at a time. Channels can be selected dynamically by manipulating the SCH bits (SCH0–SCH2) in ADCON.

The A/D conversion process requires 24 CPU clocks to convert each bit and therefore requires 192 clocks to complete an 8-bit conversion: With an 8-MHz CPU clock frequency, one clock cycle is125ns. Therefore, if each bit requires 24 clocks, you calculate the conversion rate as follows:

125 ns × 24 clocks × 8 bits = 192 clocks (24 μs) at 8 MHz

The digital result is then dumped into the output register ADOUT at address 2AH. The A/D converter unit then enters an idle state.

Because the ADC does not generate an interrupt to signal a completed conversion, the contents of ADOUT must first be read out before another conversion initiated. Otherwise, the previous result will be overwritten.

NOTE

Because the ADC does not use sample-and-hold circuitry, it is important that any fluctuations in the analog level at the AD0–AD7 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

You can alternately use the KS88C8016 A/D unit as a standard digital input port, port 5. The AD0–AD7 share pin names are P5.0–P5.7, accordingly.

Incoming port 5 data values are read directly from the 8-bit digital input register ADIN, at location 29H in page 0.

A/D CONVERTER CONTROL REGISTER (ADCON)

The A/D converter control register, ADCON, is at address 28H in page 0. Only bits 4–7 are used in the KS88C8016 implementation. ADCON is read-write addressable using 1-bit or 8-bit instructions. It has two functions:

— Bits 4, 5, and 6 (SCH0–SCH2) select the analog data input pin.
— Bit 7 is a test bit for factory use only.

After a reset, the AD0 pin (pin number 43) is selected as the analog data input pin, and the test bit is turned off. (The test bit should always be "0".)

Only one analog input channel can be selected at a time. Any one of the eight analog input pins (AD0–AD7, pins 43–35) can be selected dynamically by manipulating the SCH bits.

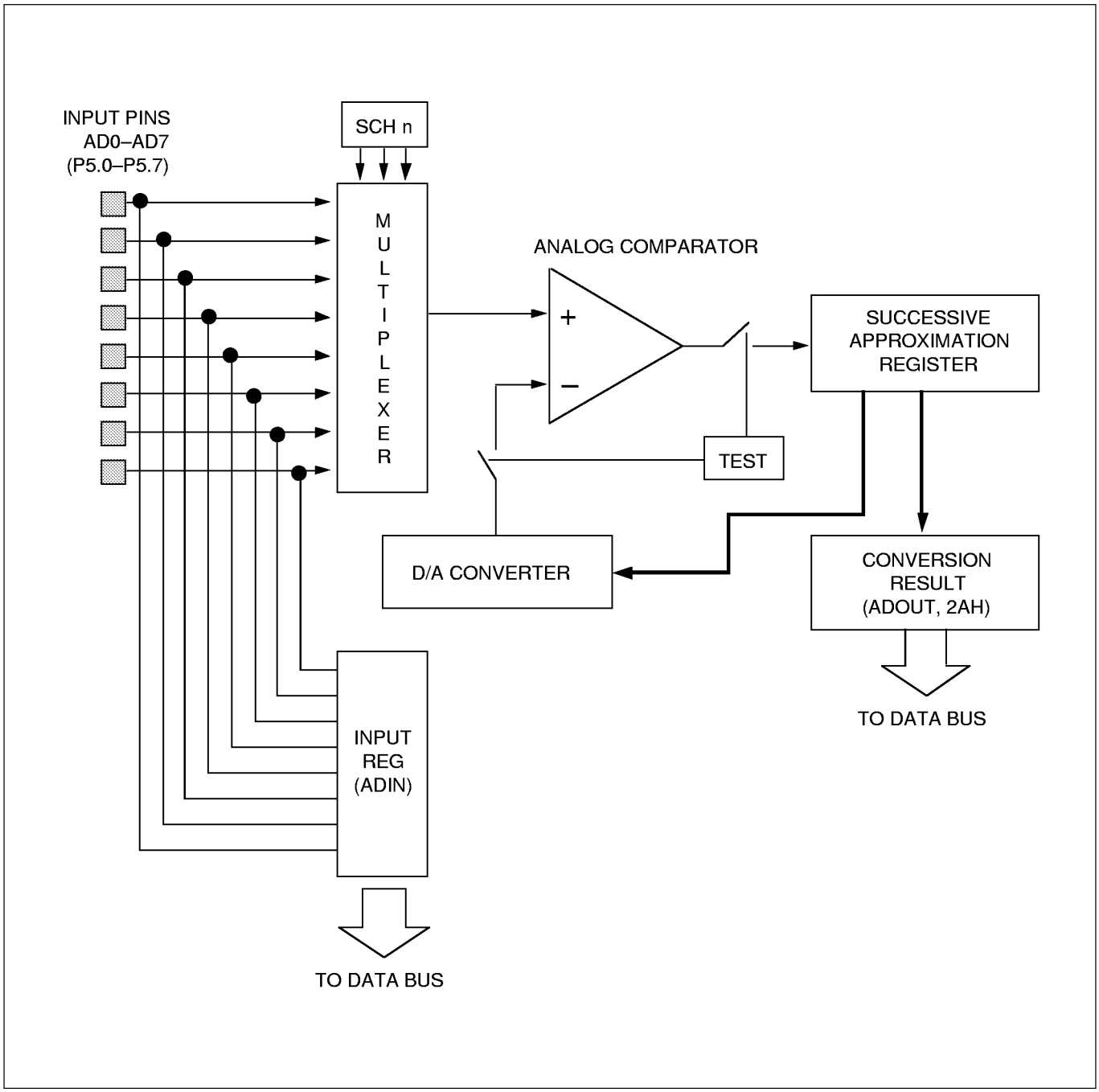


Figure 55. A/D Converter Functional Block Diagram

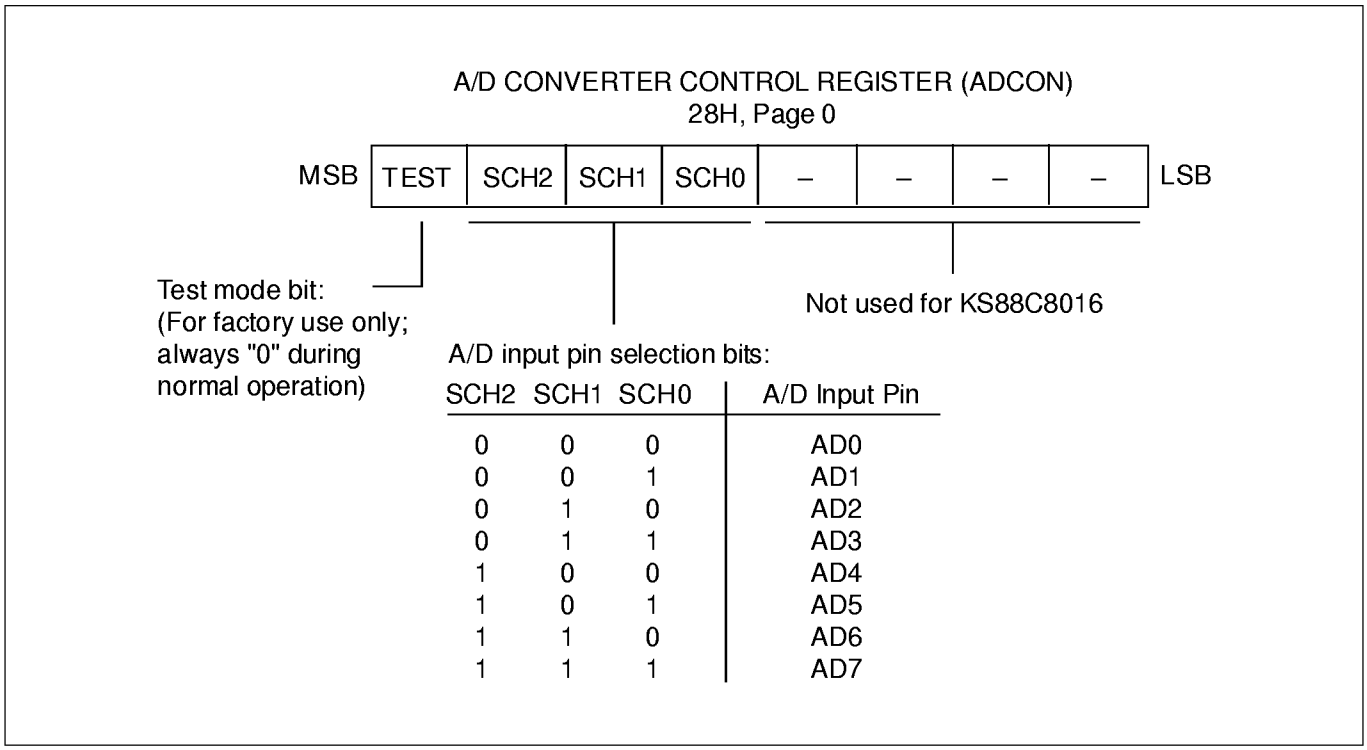


Figure 56. A/D Converter Control Register (ADCON)

INTERNAL A/D CONVERSION PROCEDURE

1. To enable the A/D converter for incoming analog data, you must first select one of the eight input pins (AD0–AD7) by writing the appropriate value to the ADCON register bits SCH0–SCH2.
2. Analog data is then input within the acceptable voltage range, between AV_{SS} and AV_{REF}.
3. If input voltage > first reference voltage (1/2 AV_{REF}), the first conversion output is "1";

   If input voltage < first reference voltage (1/2 AV_{REF}), the first conversion output is "0".
4. The operation described in step 3 is then repeated eight times.
5. After 192 clocks (24 μs with an 8-MHz CPU clock) have elapsed, the converted digital values are loaded to the output buffer ADOUT (page 0, 2AH) and the ADC module goes into an idle state.
6. The converted digital value can now be read from the ADOUT register.
7. To perform another conversion, execute step 1 through step 6 again.

INTERNAL REFERENCE VOLTAGE LEVELS

In the ADC function block, the analog input voltage level is compared to the reference voltage.

For the KS88C8016, the analog input level must be within the range AV_{SS} to AV_{REF}, where $AV_{REF} \leq V_{DD}$.

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}.

☞ PROGRAMMING TIP — A/D Conversion Noise Level and Sampling Frequency Considerations

This example shows how to program the A/D converter module. Two important assumptions for this sample program are that the noise level on the analog channel is minimal, and that the A/D converter sampling frequency is as high as possible. The program performs the following operation:

— Assuming a low noise level and high sampling frequency, load an A/D conversion value to the register R0 and set bit 7 in R1.

```
ADFLAG  EQU     7
        •
        •
        •
        CLR     PP              ;  Select page 0
        •
        •
        •
        LD      ADCON,#00H      ;  Select the AD0 pin and start conversion
        LD      R2,#10          ;  During the conversion time, an approximate wait time
                                ;  of 193 cycles (minimum) are required.
LOOP    NOP
        DJNZ    R2,LOOP
        LD      R0,ADDATA       ;  R0 ← conversion value
        BITS    R1.ADFLAG       ;  Set R1.7
        •
        •
        •
```

In some systems, the noise level may be quite high or the A/D conversion sampling frequency may be low. In such cases, we can suggest the following method to filter out noise: Perform each A/D conversion three times, discard the maximum and minimum values, and take the median value as the result. This sample program uses this method to perform the same operation as the first A/D sample program:

— Assume a high noise level and a low sampling frequency.

— Use median value of three conversions as the result.

— Load an A/D conversion value to the register R0 and set bit 7 in R1.

```
        •
        •
        •
AD0     EQU     0
AD1     EQU     1
AD2     EQU     2
AVE     EQU     0
MIN     EQU     1
MAX     EQU     2
TEMP    EQU     3
ADFLAG  EQU     7
        •
        •
        •
        CLR     PP              ;  Select page 0
        •
        •
        •
```

☞ PROGRAMMING TIP — A/D Conversion (Continued)

```
                •
                •
                •
        LD      RTEMP,#03H      ;  R3 ← 03H
ADLOOP  LD      ADCON,#00H      ;  Select AD0 pin and start conversion
        LD      R4,#10H         ;  Wait minimum 193 cycles for conversion
                                ;  (24 μs at 8 MHz)
LOOP    NOP                     ;
        DJNZ    R4,LOOP         ;
        NOP                     ;
        NOP                     ;
        NOP                     ;
        LD      RAD0,RAD1       ;  Save conversion result to R0–R2
        LD      RAD1,RAD2       ;  (the first conversion result will be in R0, the second
        LD      RAD2,ADDATA     ;  result in R1, and the third result in R2)
                                ;
        DJNZ    RTEMP,ADLOOP    ;  Conversion completed three times? If not, return to loop
        CP      RMAX,RMIN       ;
        JP      UGE,JAD1        ;  If MAX ≥ MIN, then jump to JAD1
        LD      RTEMP,RMIN      ;  If MAX < MIN, exchange the values (MAX ↔ MIN)
        LD      RMIN,RMAX       ;          "
        LD      RMAX,RTEMP      ;          "
                •
                •
                •
JAD1    CP      RMAX,RAVE       ;
        JP      UGE,JAD2        ;  If MAX ≥ AVE, then go to JAD2
        LD      RTEMP,RMAX      ;  If MAX < AVE, then exchange MAX and AVE
        LD      RMAX,RAVE       ;          "
        LD      RAVE,RTEMP      ;          "
JAD2    CP      RMIN,RAVE       ;
        JP      ULE,JAD3        ;  If MIN ≤ AVE, then go to JAD3
        LD      RTEMP,RMIN      ;  If MIN > AVE, then exchange MIN and AVE
        LD      RMIN,RAVE       ;          "
        LD      RAVE,RTEMP      ;          "
JAD3    BITS    R7.ADFLAG       ;  Set ADFLAG
                •
                •
                •
```

SERIAL I/O INTERFACE

OVERVIEW

The serial data I/O (SIO) module can interface with various external devices requiring serial data transfer. The SIO function block has the following components:

— 8-bit shift register
— Programmable 5-bit prescaler
— Bi-directional serial I/O pin (SDIO)
— Input/output pin for serial I/O clock (SCLK)
— SIO control register (SIOCON)
— SIO prescaler control register (SIOPS)

SIO operations start when the CPU writes data to the SIO shift register (address E9H, set 1, bank0).

On the basis of current SIOCON register settings, the SIO unit either transmits or receives eight bits of data, one bit at a time, on each edge of its internal or external shift clock.

When one 8-bit serial transmit or receive operation is completed, the SIO unit enters an idle state until the CPU again writes data to the shift register.

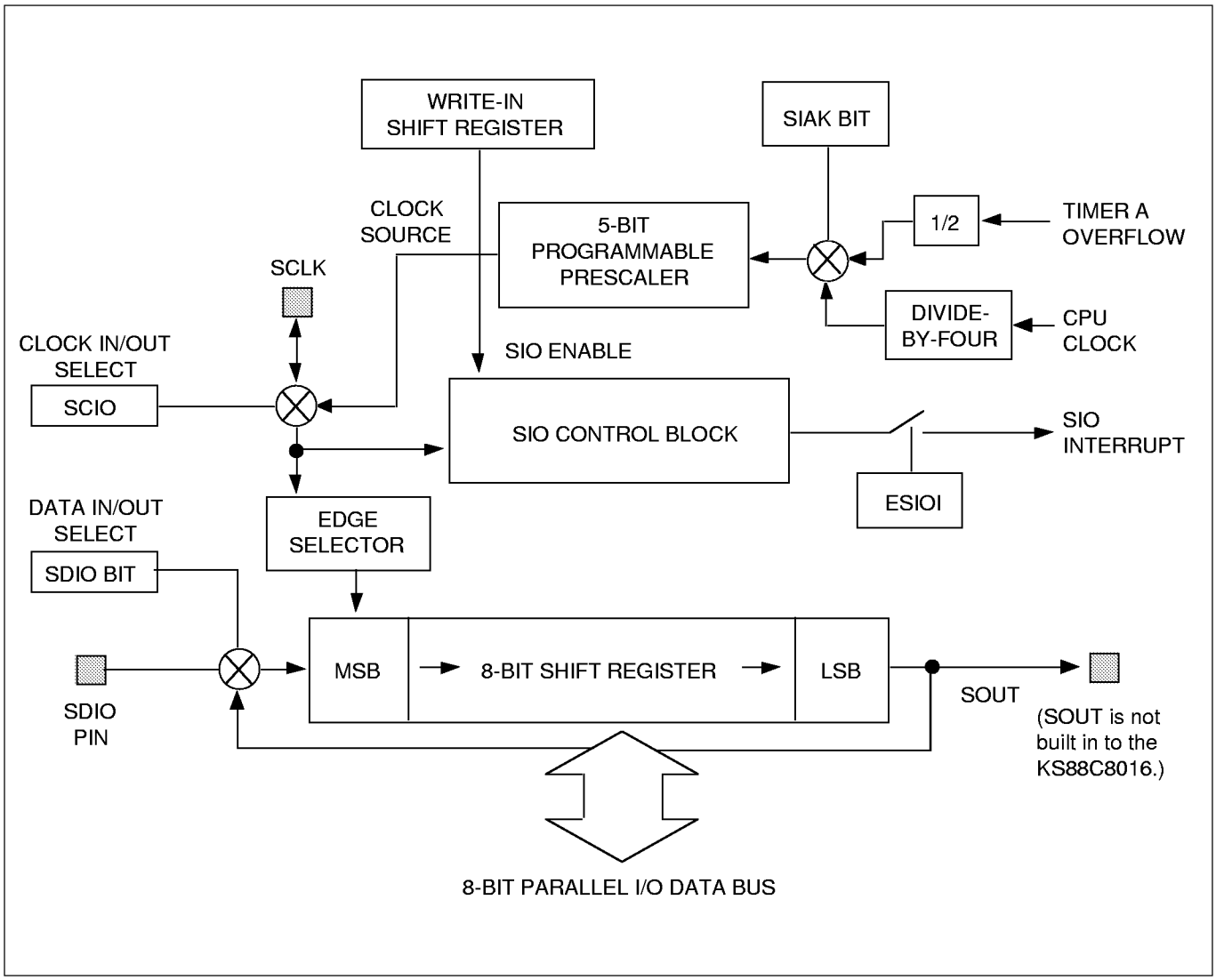


Figure 57. Serial I/O Interface Function Diagram

SIO PRESCALER CONTROL REGISTER (SIOPS)

The serial I/O prescaler control register SIOPS is in set 1, page 0, address EBH. It has three functions:

— Shift clock selection
— SIO clock selection
— 5-bit prescaler value for the shift clock

Shift Clock Selection

The SCIO bit in the SIOPS register selects the shift clock for SIO operations. There are two options: externally generated clock input at the SCLK pin, or the internal prescaler output of the SIO unit.

When the SCIO bit in the SIOPS register is "0", the SCLK pin is configured to input mode. The incoming clock signal can then directly drive the internal shift clock. This lets a slower peripheral device control the speed at which it receives or transmits the data.

To select the internal prescaler output as the shift clock, set SIOPS bit 6 (SCIO) to "1".

The SIAK bit setting (bit 5) selects either the timer A overflow signal (divided by two) or the system clock (divided by four) as the clock input to the 5-bit prescaler.

The prescaler output, as determined by the SCIO and SIAK settings and the prescaler value, drives the internal shift clock.

NOTE

If you use an external clock source for SIO shift operations, the pulse waveform (normally high level) cannot be validated.

Programmable Prescaler Clock

The rate at which the SIO transmits and receives data is determined by the frequency of the selected clock source. The programmer must set or clear the SIAK bit in the SIOPS register to select the prescaler input clock. There are two selections:

— Main system clock frequency divided by four (CPU clock divided by four)
— Timer A overflow signal (divided by two by internal SIO logic)

The 5-bit prescaler can be programmed to divide the selected incoming clock by a value from 1 to 32. To set the prescaler 'divide-by' value, write the appropriate hexadecimal value to PS4–PS0.

The prescaler clock also can be output at the SCLK pin to provide a data transfer clock to peripheral devices. Here, SCLK is held at low level when SIO unit is idle. When a RESET occurs, the SIOPS register is cleared to 00H.

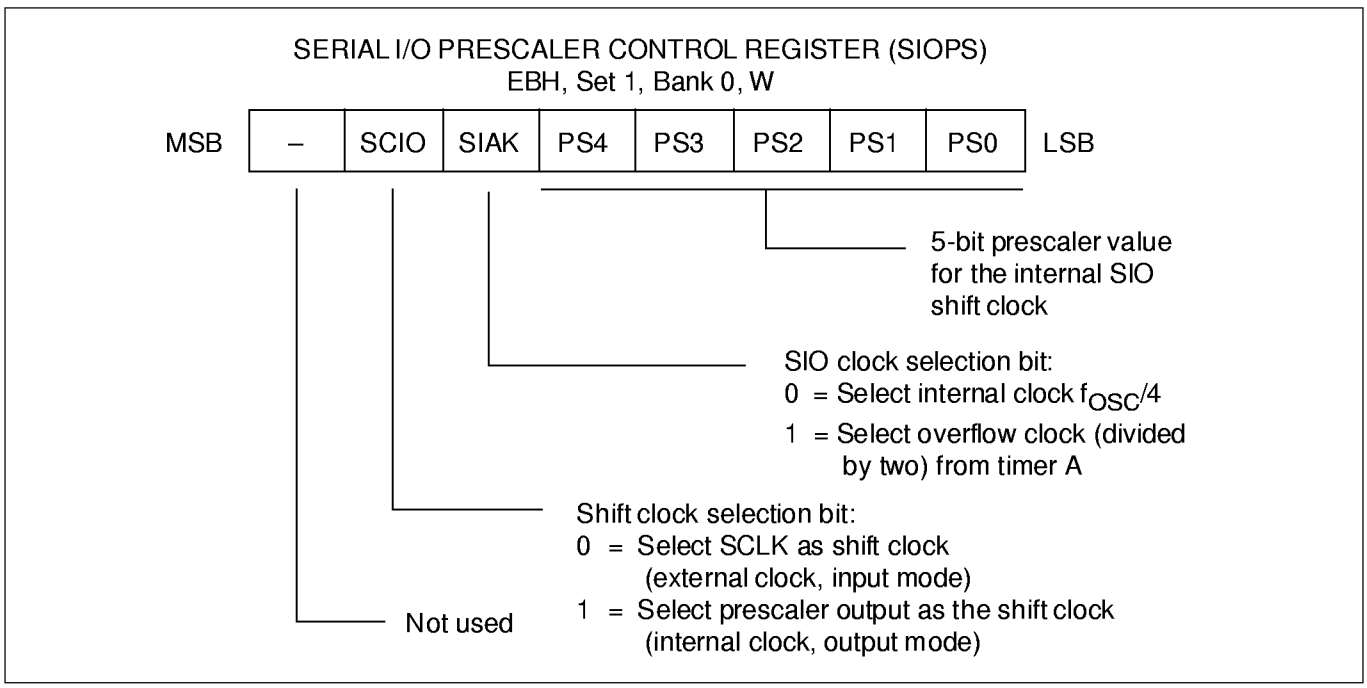


Figure 58. Serial I/O Prescaler Control Register (SIOPS)

SERIAL I/O CONTROL REGISTER (SIOCON)

The serial I/O control register SIOCON is in set 1, bank 0 at address EAH. It has the following functions:

- Select rising or falling shift clock edge (bit 4, EGS)
- Configure the SCLK pin circuit in output mode (bit 2, SOP0, and bit 3, SOP1)
- Select transmit mode or receive mode (bit 1, SDIO)
- Enable or disable SIO interrupt requests (bit 0, ESIOI)

The SIO shift register shifts data on each rising or falling edge of the internal shift clock, as selected by the EGS bit in the SIOCON register. If SDIO = "1", the SDIO pin is set to output and the SIO module enters transmit mode.

On each clock edge, data is shifted one bit in the direction of MSB to LSB. The LSB value is transmitted through the SDIO pin and is then shifted back to the MSB. This method saves the data in the shift register while it is being transmitted.

When eight data bits have been shifted, the unit enters an idle state, and an SIO interrupt (vector B8H) is generated, assuming the ESIOI bit is "1".

When the SDIO bit in the SIOCON register is "0", the SDIO pin set to input and the SIO module enters receive mode. A receive operation starts when the CPU writes data to the SIO shift register. Data is then read at the SDIO pin on each clock edge, and shifted from MSB to LSB until the 8-bit shift register is full. When the shift register is full, the SIO unit enters an idle state. It then issues an interrupt to tell the CPU that data is present and can be read.

Two bits in the SIOCON register, SOP0 and SOP1, configure the SCLK and SDIO pins when they are in output mode. Choices for the output circuit are open-drain with pull-up resistor, open-drain without pull-up, and push-pull. The SCLK pin also has noise filters for inputs.

$\overline{\text{RESET}}$ clears the SIOCON register to 00H. This sets the SIO module to receive mode, open-drain without pull-ups, selects the rising edge of the shift clock, and disables the SIO interrupt.

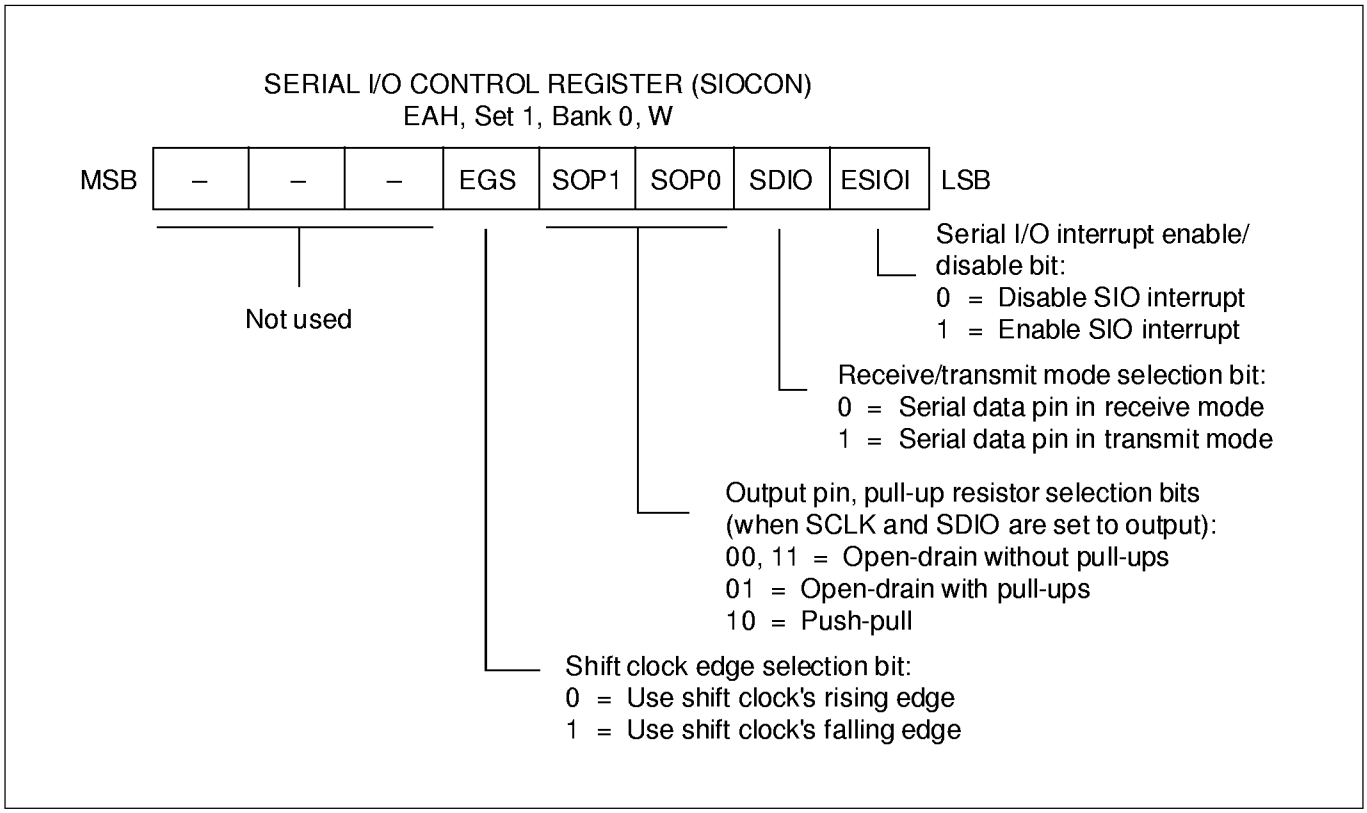


Figure 59. Serial I/O Control Register (SIOCON)

☞ PROGRAMMING TIP — Programming the Serial I/O Interface

This example shows how to program the SIO unit according to sample specifications. Please assume an 8-MHz CPU clock frequency and use the SIO unit's internal clock. The baud rate for transmit mode is 2 MHz; for receive mode, it is 500 kHz. The parameters for the sample program are as follows:

1. If P0.0 is low, then transmit the SIO_DATA_T.
2. If P0.1 is low, then receive data from an external device.
3. Write a subroutine for this operation called SIO_T_R.

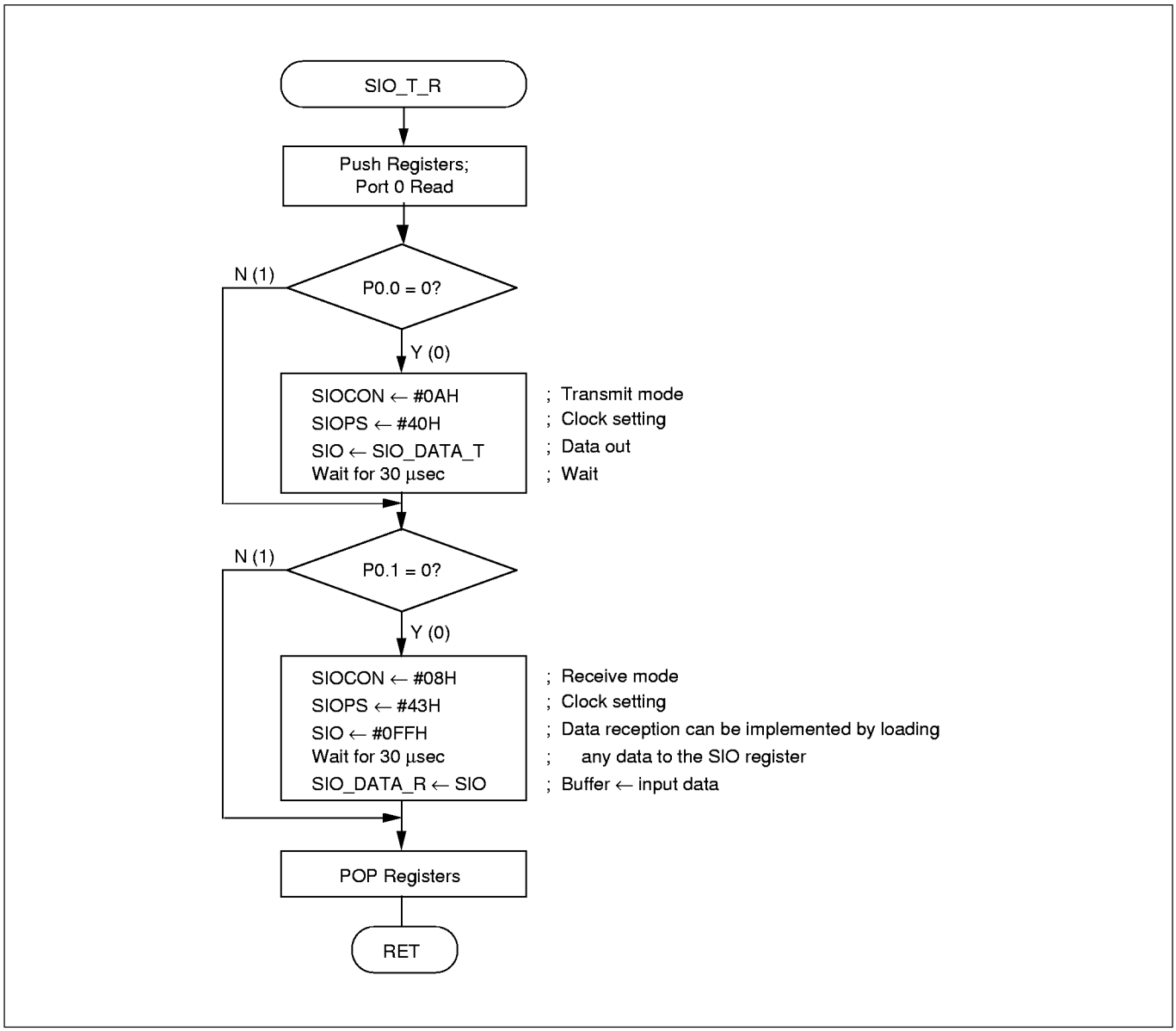


Figure 60. Decision Flowchart for SIO Programming Tip (Main Program)

☞ PROGRAMMING TIP — Programming the Serial I/O Interface (Continued)

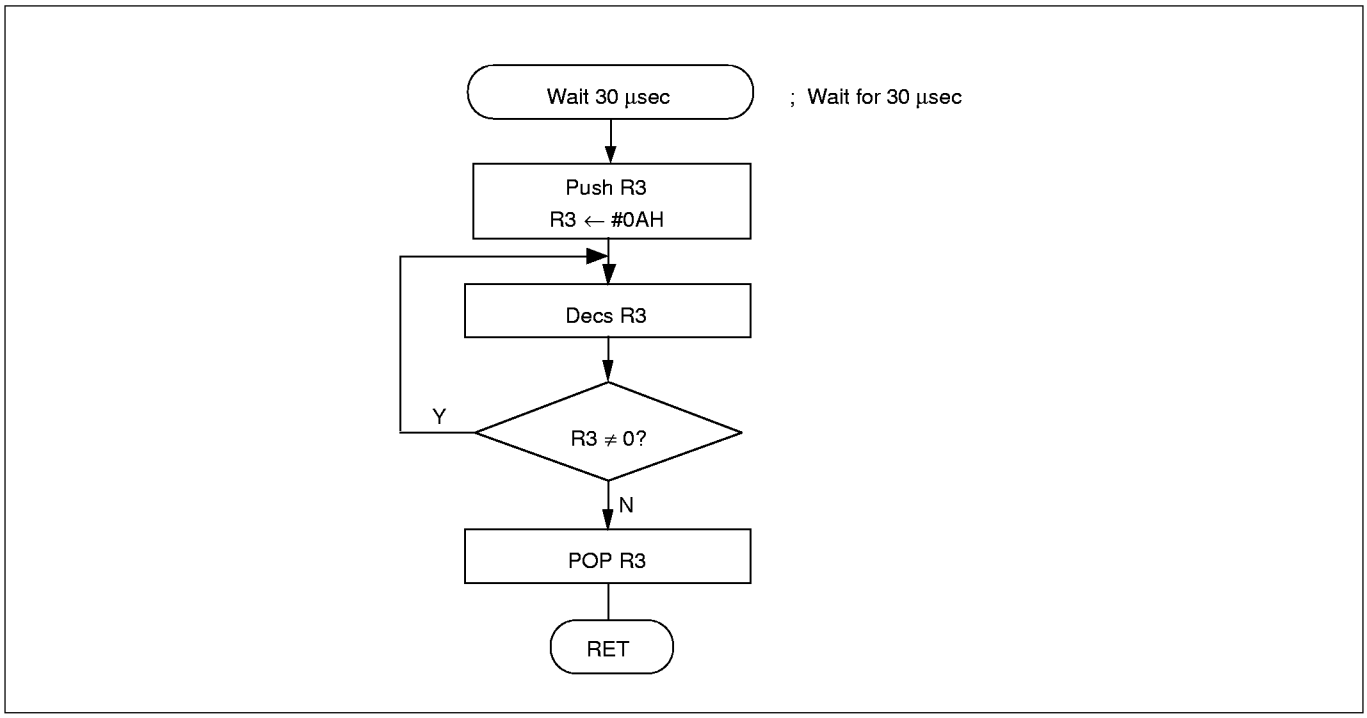


Figure 61. Decision Flowchart for SIO Programming Tip (Wait Cycle)

```
                •
                •
                •
f_SIO_T         EQU     0               ;  Transmit enable flag (P0.0), enable when low
f_SIO_R         EQU     1               ;  Receive enable flag (P0.1), enable when low
SIO_DATA_T      EQU     50H             ;  Transmit data buffer
SIO_DATA_R      EQU     51H             ;  Receive data buffer
                •
                •
                •
SIO_T_R         PUSH    PP              ;  Push page pointer to stack
                PUSH    R3              ;  Push working register R3
                SB0                     ;  Select bank 0
                PUSH    P0CON           ;  Push port 0 control register (P0CON)
                CLR     PP              ;  Select page 0
                LD      P0CON,#44H      ;  P0.0–0.7 input with pull-up selected
                LD      R3,P0           ;  Read port 0
                BTJRT   SIO_IN,R3.f_SIO_T ;  Transmit enable?
SIO_OUT         LD      SIOCON,#00001010B ;  Rising edge, push-pull output, transmit mode, and
                                        ;  interrupt disabled
                LD      SIOPS,#01000000B ;  Prescaler output as shift clock, internal CPU clock /4
                                        ;  or 500 ns (2 MHz) if CPU clock = 8 MHz. (It takes
                                        ;  4 µs for an 8-bit shift data transmit operation.)
```

(Continued on next page)

☞ PROGRAMMING TIP — Programming the Serial I/O Interface (Concluded)

```
                •
                •
                •
                LD      SIO,SIO_DATA_T      ; Transmit data buffer
                CALL    WAIT30μs            ; Wait for 30 μs
                LD      R3,P0
SIO_IN          BTJRT   SIO_E,R3.f_SIO_R    ; Receive enable?
                LD      SIOCON,#00001000B   ; Receive mode select
                LD      SIOPS,#01000011B    ; 2 μs (500 kHz) if CPU clock = 8 MHz. (It takes 16 μs
                                            ; to complete an 8-bit shift data receive operation.)
                LD      SIO,#0FFH           ; Serial data input from external device
                CALL    WAIT30μs            ; Wait for 30 μs
                LD      SIO_DATA_R,SIO      ; Receive buffer register ← shifted input data
SIO_E           POP     P0CON
                POP     R3
                POP     PP
                RET
WAIT30μs        PUSH    R3
                LD      R3,#0AH
LOOP            DJNZ    R3,LOOP
                POP     R3
                RET
                •
                •
                •
```

ELECTRICAL DATA

Table 16. Absolute Maximum Ratings

(T_A = 25°C)

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}		– 0.3 to + 7.0	V
Input voltage	V_{IN}	All input pins except TEST	– 0.3 to V_{DD} + 0.3	V
		TEST pin only	– 0.3 to + 10	
Output voltage	V_O	All output pins	– 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 except for port 1	+ 100	
		Total pin current for port 1	+ 100	
Operating temperature	T_A		– 20 to + 85	°C
Storage temperature	T_{STG}		– 65 to + 150	°C

Table 17. D.C. Electrical Characteristics

(T_A = – 20°C to + 85°C, V_{DD} = 4.5 V to 6.0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input high voltage	V_{IH1}	All input pins except as specified for V_{IH2}	0.8 V_{DD}		V_{DD}	V
	V_{IH2}	X_{IN}, XT_{IN}	V_{DD} – 0.5		V_{DD}	
Input low voltage	V_{IL1}	All input pins except V_{IL2}			0.2 V_{DD}	V
	V_{IL2}	X_{IN}, XT_{IN}			0.4	
Output high voltage	V_{OH1}	V_{DD} = 4.5 V to 6.0 V I_{OH} = – 1 mA Port 1, T2M3, and T2M5.2	V_{DD} – 1.0			V
		If I_{OH} = – 200 µA	V_{DD} – 0.5			
	V_{OH2}	V_{DD} = 4.5 V to 6.0 V I_{OH} = – 200 µA All output pins except V_{OH1}	V_{DD} – 1.0			
		If I_{OH} = – 60 µA	V_{DD} – 0.5			

Table 17. D.C. Electrical Characteristics (Continued)

(T_A = −20°C to +85°C, V_{DD} = 4.5 V to 6.0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Output low voltage	V_{OL1}	V_{DD} = 4.5 V to 6.0 V I_{OL} = 15 mA Port 1, T2M3, and T2M5.2			1.0	V
	V_{OL2}	I_{OL} = 2 mA All output pins except V_{OL1}			0.4	
Input high leakage current	I_{LIH1}	V_{IN} = V_{DD} All input pins except X_{IN} and XT_{IN}			3	µA
	I_{LIH2}	V_{IN} = V_{DD} X_{IN} and XT_{IN}			20	
Input low leakage current	I_{LIL1}	V_{IN} = 0 V All input pins except X_{IN}, XT_{IN}, and $\overline{RESET}$			−3	µA
	I_{LIL2}	V_{IN} = 0 V X_{IN}, and XT_{IN}			−20	
Output high leakage current	I_{LOH}	V_{OUT} = V_{DD} All output pins			3	µA
Output low leakage current	I_{LOL}	V_{OUT} = 0 V			−3	µA
Pull-up resistor	R_{L1}	V_{IN} = 0 V; V_{DD} = 5 V ± 10% Port 0, 1, 2, 3, and 4	25	47	70	KΩ
	R_{L2}	V_{DD} = 5 V ± 10% $\overline{RESET}$ only	120	220	320	
Supply current (see Note)	I_{DD1}	V_{DD} = 5 V ± 10% 8-MHz crystal oscillator		19	45	mA
	I_{DD2}	Idle mode; V_{DD} = 5 V ± 10% 8-MHz crystal oscillator		2.7	10	
	I_{DD3}	Stop mode; V_{DD} = 5 V ± 10% 32-kHz crystal suboscillator		90	200	µA
		If V_{DD} = 3 V ± 10%		10	50	
	I_{DD4}	Stop mode; XT_{IN} = 0 V Crystal suboscillator stopped V_{DD} = 5 V ± 10%		1.5	20	µA
		If V_{DD} = 3 V ± 10%		0.1	5	

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

Table 18. A.C. Electrical Characteristics

(T_A = −20°C to +85°C, V_{DD} = 4.5 V to 6.0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Interrupt input high, low width	t_{INTH}, t_{INTL}	P2.0–P2.7, P4.0–P4.7, T1M0, T2M0–T2M4	3			t_{CPU} (1)
$\overline{RESET}$ input low width	t_{RSL}	Input	22			

NOTES:
1. The unit t_{CPU} means one CPU clock period.
2. The oscillator frequency (f_{OSC}) is identical to the CPU clock frequency.

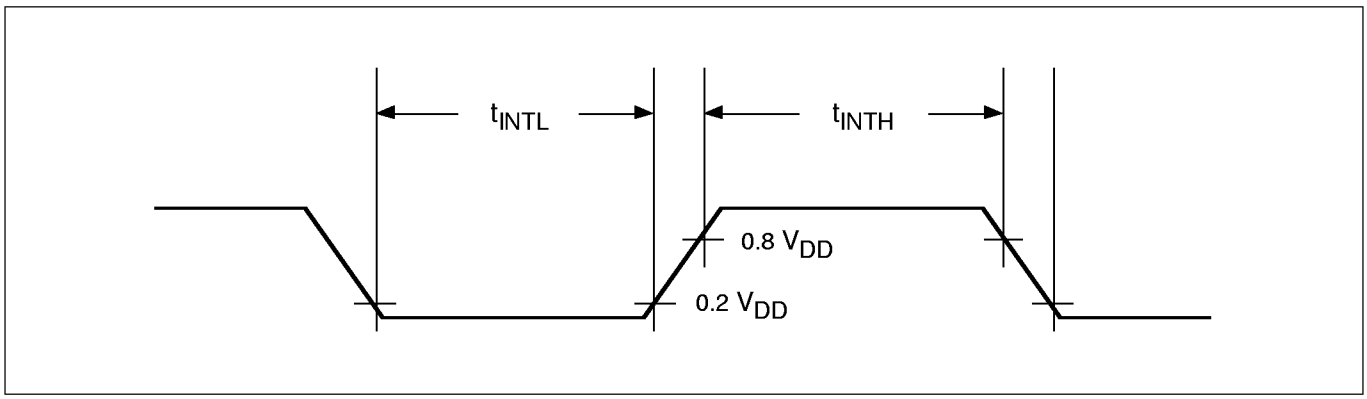


Figure 62. Input Timing for External Interrupts (Port 4)

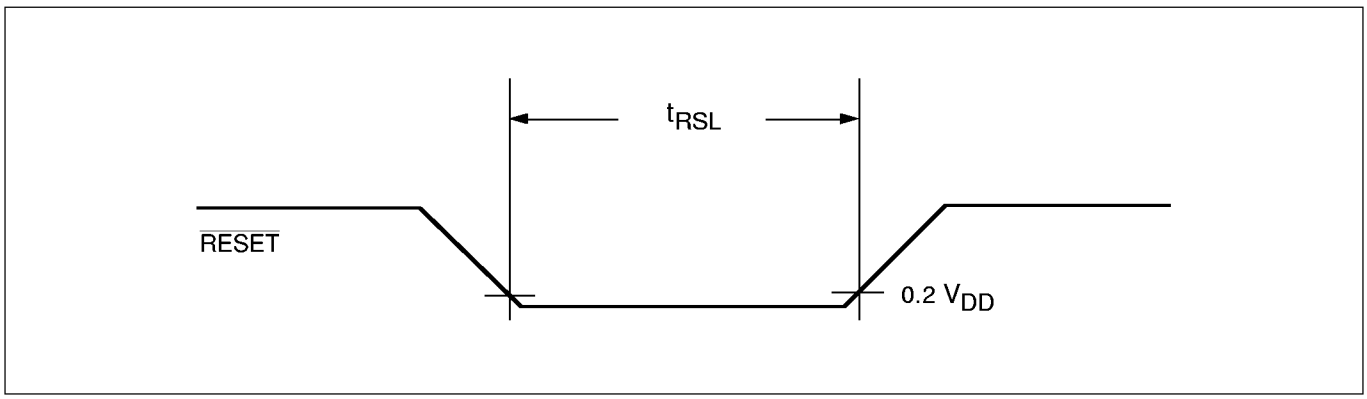


Figure 63. Input Timing for $\overline{RESET}$

Table 19. Input/Output Capacitance

(T_A = −20°C to +85°C, V_{DD} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input capacitance	C_{IN}	f = 1 MHz; unmeasured pins are returned to V_{SS}			10	pF
Output capacitance	C_{OUT}					
I/O capacitance	C_{IO}					

Table 20. Data Retention Supply Voltage in Stop Mode

(T_A = −20°C to +85°C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}		2		6	V
Data retention supply current	I_{DDDR}	V_{DDDR} = 2 V			5	µA

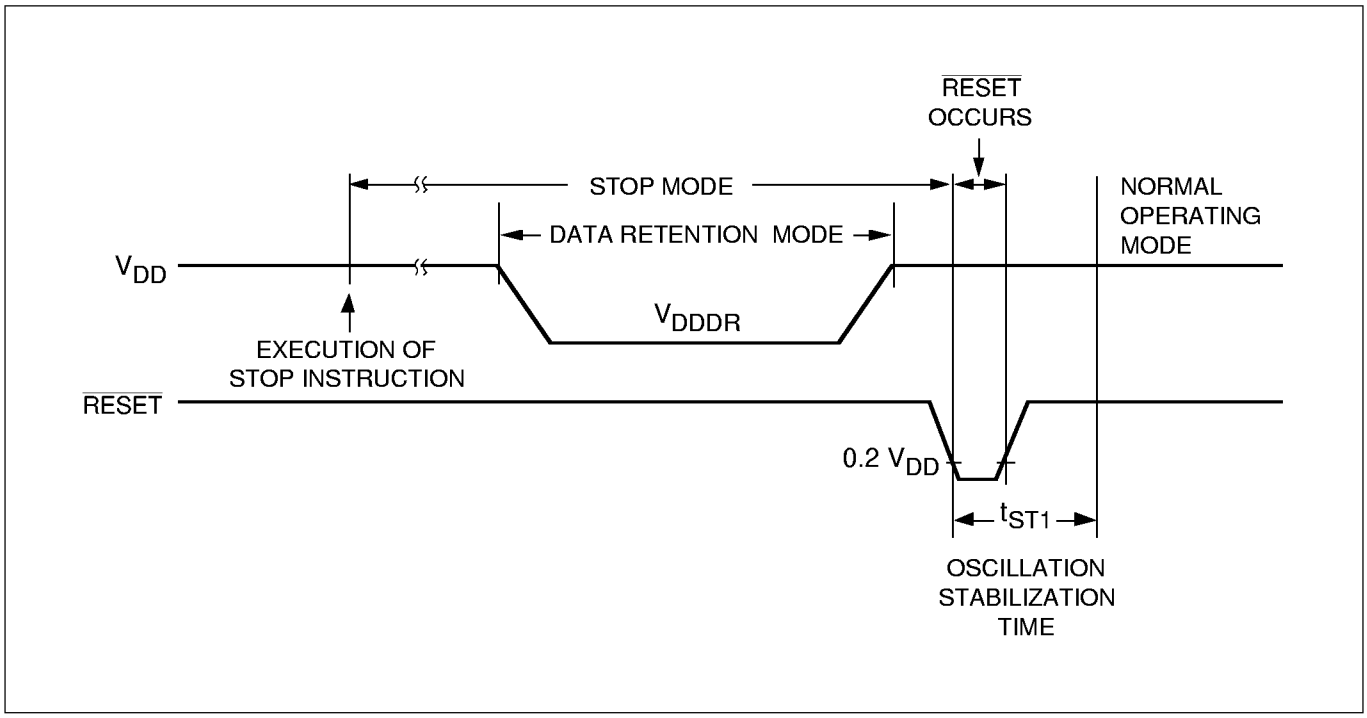


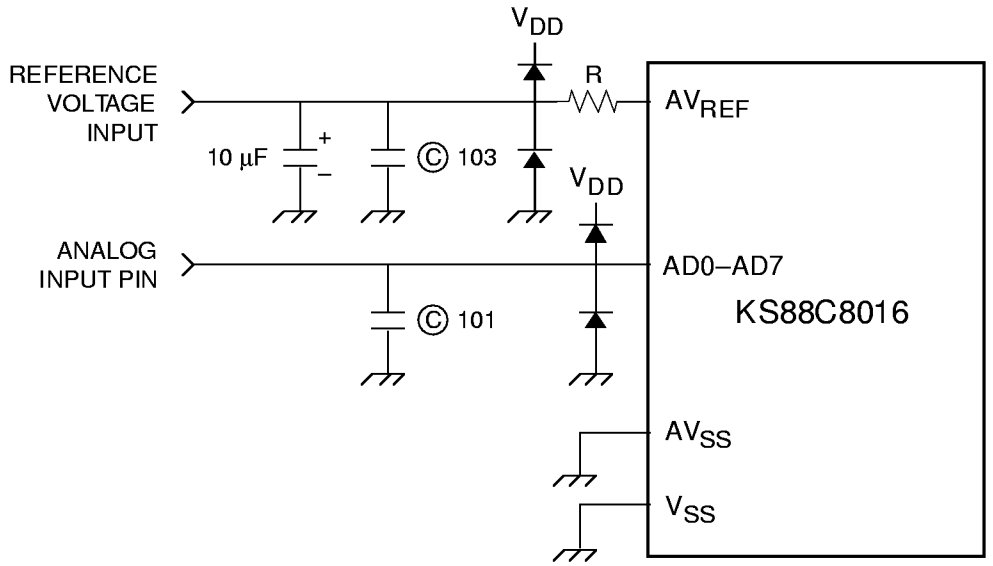
Figure 64. Stop Mode Release Timing When Initiated by RESET

Table 21. A/D Converter Electrical Characteristics

(T_A = −20°C to +85°C, V_{DD} = 4.5 V to 6.0 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Resolution			8	8	8	bit
Absolute accuracy (1)		CPU clock = 8 MHz			3	LSB
Conversion time (2)	t_{CON}		t_{CPU} × 192 (3)			µs
Analog input voltage	V_{IAN}		V_{SS}		V_{DD}	V
Analog input impedance	R_{AN}		2			MΩ

NOTES:
1. Excluding quantization error, absolute accuracy values are within ± 1/2 LSB. Absolute accuracy of 3 LSB can only be guaranteed when the offset error is adjusted to its optimal value (see Figure 65).
2. 'Conversion time' is the time required from the moment a conversion operation starts until it ends.
3. The unit t_{CPU} means one CPU clock period.



NOTE: The symbol 'R' signifies an offset resistor with a value of 50 Ω to 100 Ω. If this resistor is omitted, the absolute accuracy will be maximum of 6 LSBs.

Figure 65. Recommended A/D Converter Circuit for Highest Absolute Accuracy

Table 22. Serial I/O Timing Characteristics

(T_A = –20°C to +85°C, V_{DD} = 4.5 V to 6.0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Instruction cycle time	t_{CY}	CPU clock = 5–10 MHz	0.6		1.2	μs
SCLK cycle time	t_{KCY}	Input	6			t_{CPU}
		Output	4			
SCLK high, low width	t_{KH}, t_{KL}	Input	3 × t_{CPU} – 20 ns			ns
		Output	2 × t_{CPU} – 20 ns			
Data setup time	t_S	Input	100			ns
		Output	150			
Data hold time	t_H	Input	3			t_{CPU}
		Output	2			

NOTES:
1. The unit t_{CPU} means one CPU clock period.
2. The oscillator frequency (f_{OSC}) is identical to the CPU clock frequency.

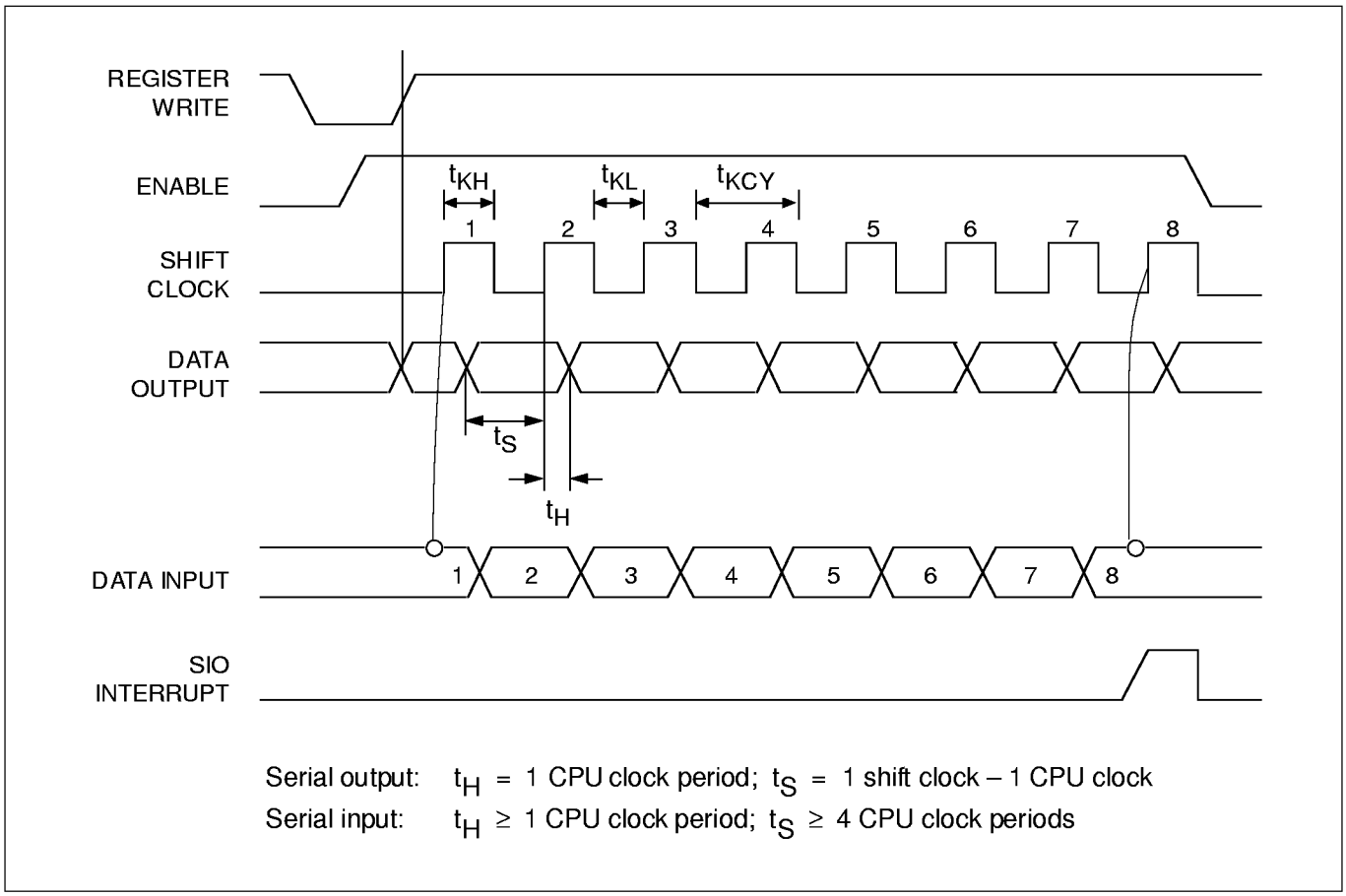


Figure 66. Serial I/O Timing Diagram (Transmit/Receive Mode; Rising Edge Selection)

Table 23. Main Oscillator Frequency (f_{OSC1})

(T_A = −20°C + 85°C, V_{DD} = 4.5 V to 6.0 V)

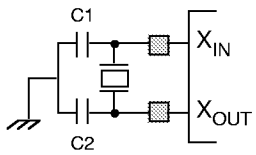
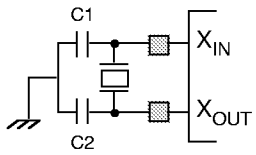
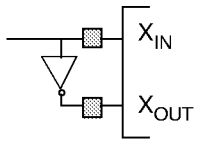
Oscillator	Clock Circuit	Test Condition	Min	Typ	Max	Unit
Crystal		CPU clock frequency	1	8	12	MHz
Ceramic		CPU clock frequency	1	8	12	MHz
External clock		X_{IN} input frequency	1	8	12	MHz

Table 24. Main Oscillator Clock Stabilization Time (t_{ST1})

(T_A = −20°C + 85°C, V_{DD} = 4.5 V to 6.0 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	V_{DD} = 4.5 V to 6.0 V			10	ms
Ceramic	Stabilization occurs when V_{DD} is equal to the minimum oscillator voltage range.			4	ms
External clock	X_{IN} input high and low level width (t_{XH}, t_{XL})	41.7		500	ns

NOTE: Oscillation stabilization time (t_{ST1}) is the time required for the CPU clock to return to its normal oscillation frequency after a power-on occurs, or when Stop mode is released by a reset.

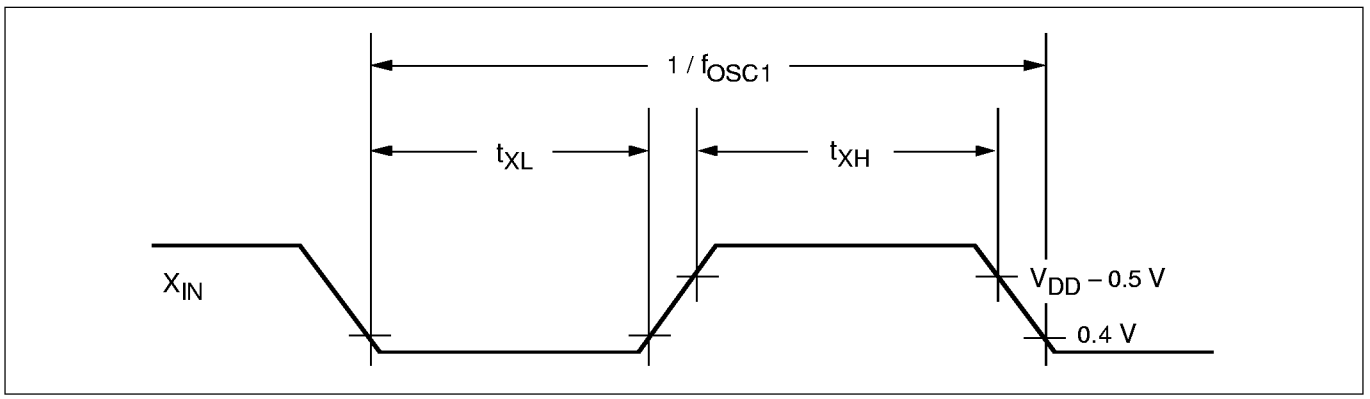


Figure 67. Clock Timing Measurement at X_{IN}

Table 25. Suboscillator Clock Frequency (f_{OSC2})

(T_A = −20°C +85°C, V_{DD} = 4.5 V to 6.0 V)

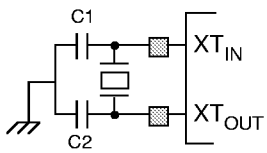
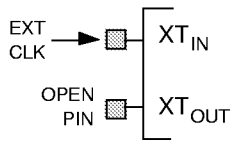
Oscillator	Clock Circuit	Test Condition	Min	Typ	Max	Unit
Crystal		Suboscillator clock frequency when used as backup timer clock	30	32.7	35	kHz
External clock		XT_{IN} input frequency	30	32.7	500	kHz

Table 26. Suboscillator Clock Stabilization Time (t_{ST2})

(T_A = −20°C +85°C, V_{DD} = 4.5 V to 6.0 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	V_{DD} = 4.5 V to 6.0 V		1.0	2	s
External clock	XT_{IN} input high, low level width (t_{XTH}, t_{XTL})	1		16.7	µs

NOTE: Suboscillator clock stabilization time (t_{ST2}) is the time required for the backup timer clock to return to its normal oscillation frequency after a power-on occurs.

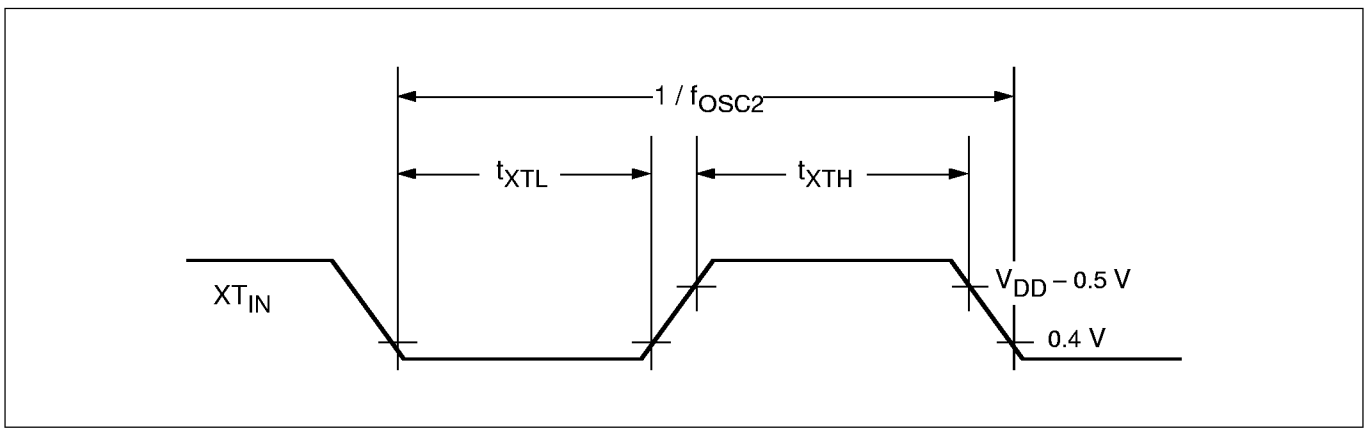


Figure 68. Clock Timing Measurement at XT_{IN}

CHARACTERISTIC CURVES

NOTE

The characteristic values shown in the following graphs are based on actual test measurements using an external clock source. They do not, however, represent guaranteed operating values.

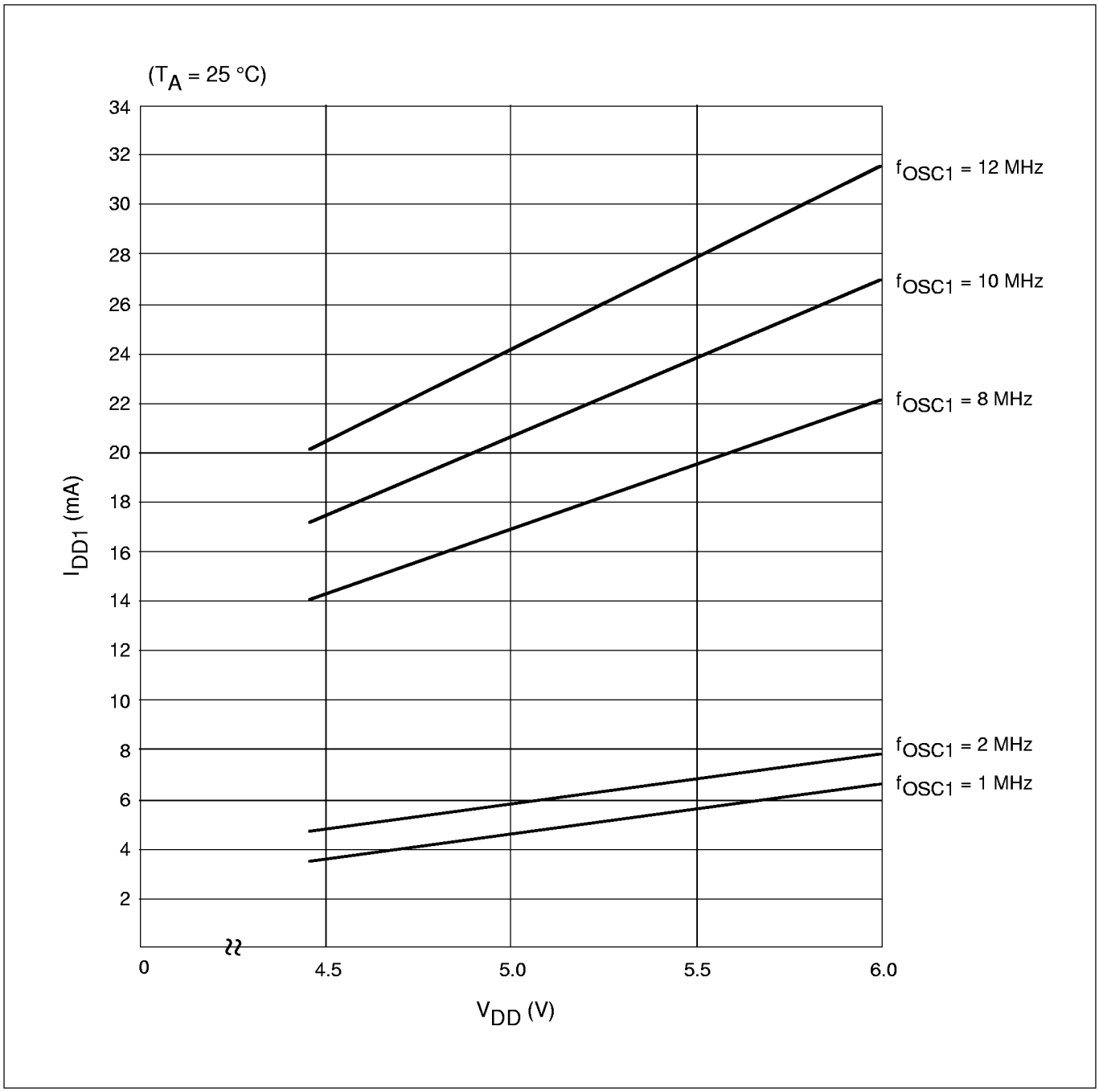


Figure 69. I_{DD1} vs. V_{DD}

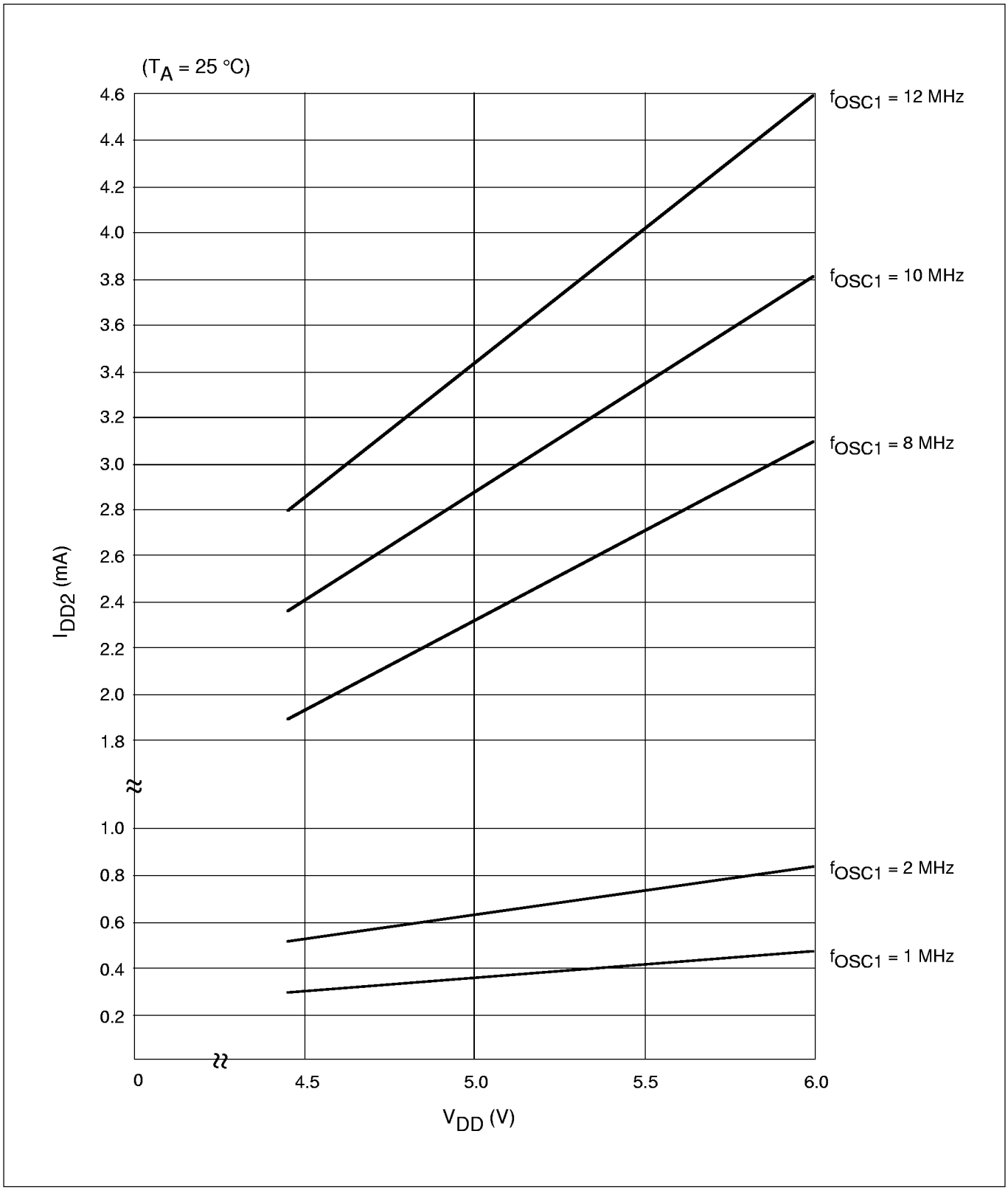


Figure 70. I_{DD2} vs. V_{DD}

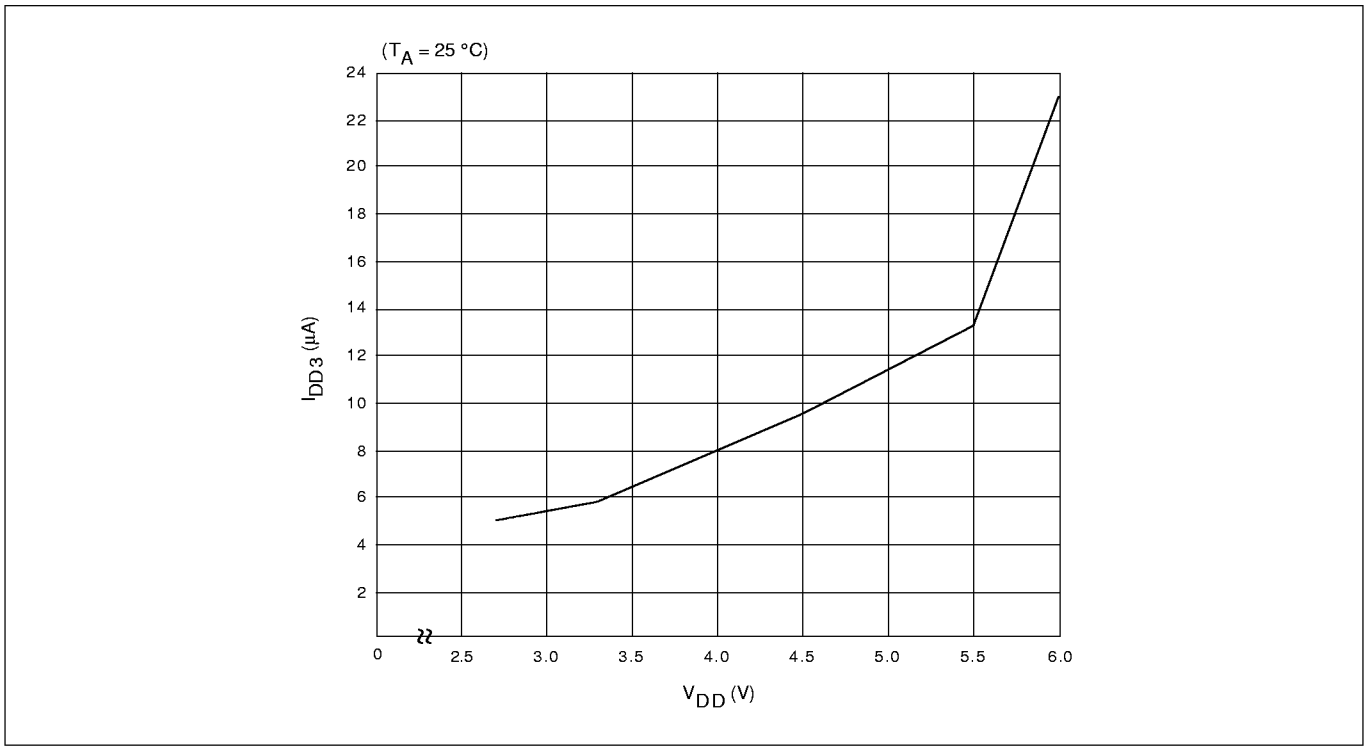


Figure 71. I_{DD3} vs. V_{DD}

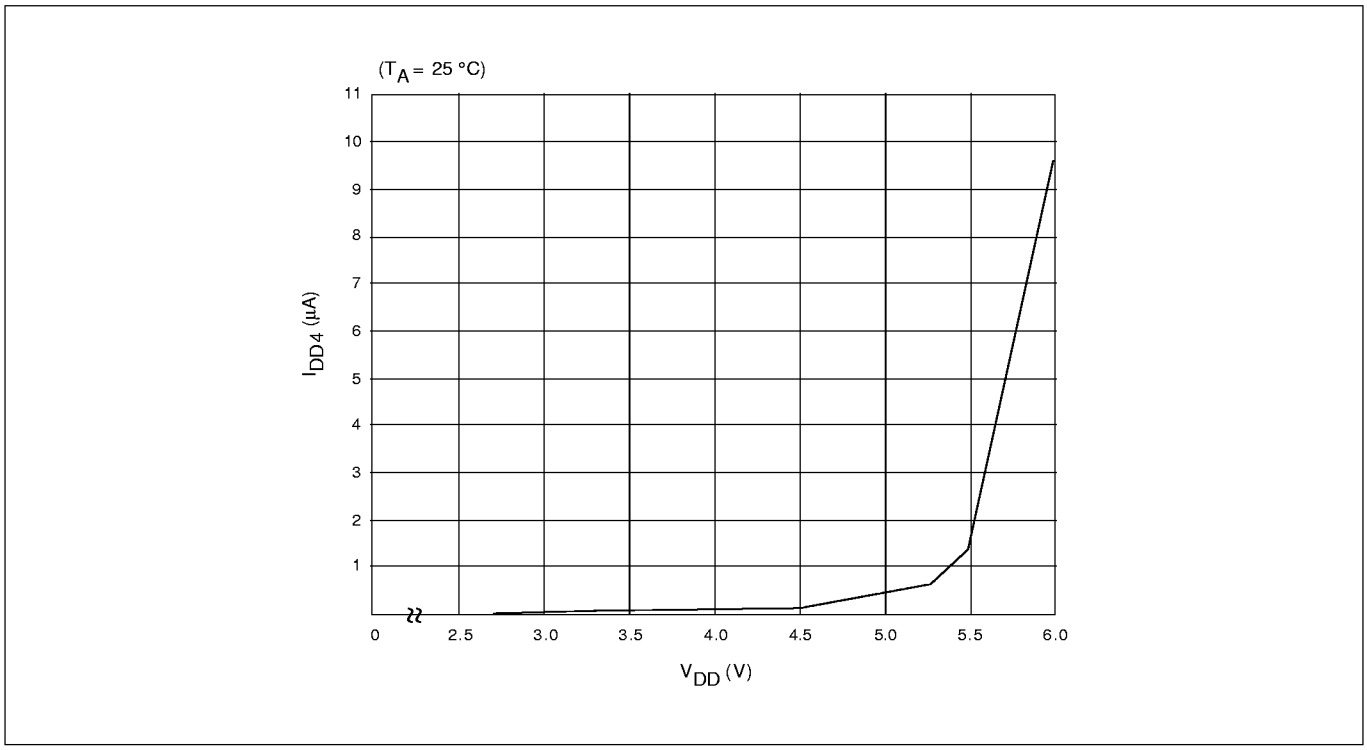


Figure 72. I_{DD4} vs. V_{DD}

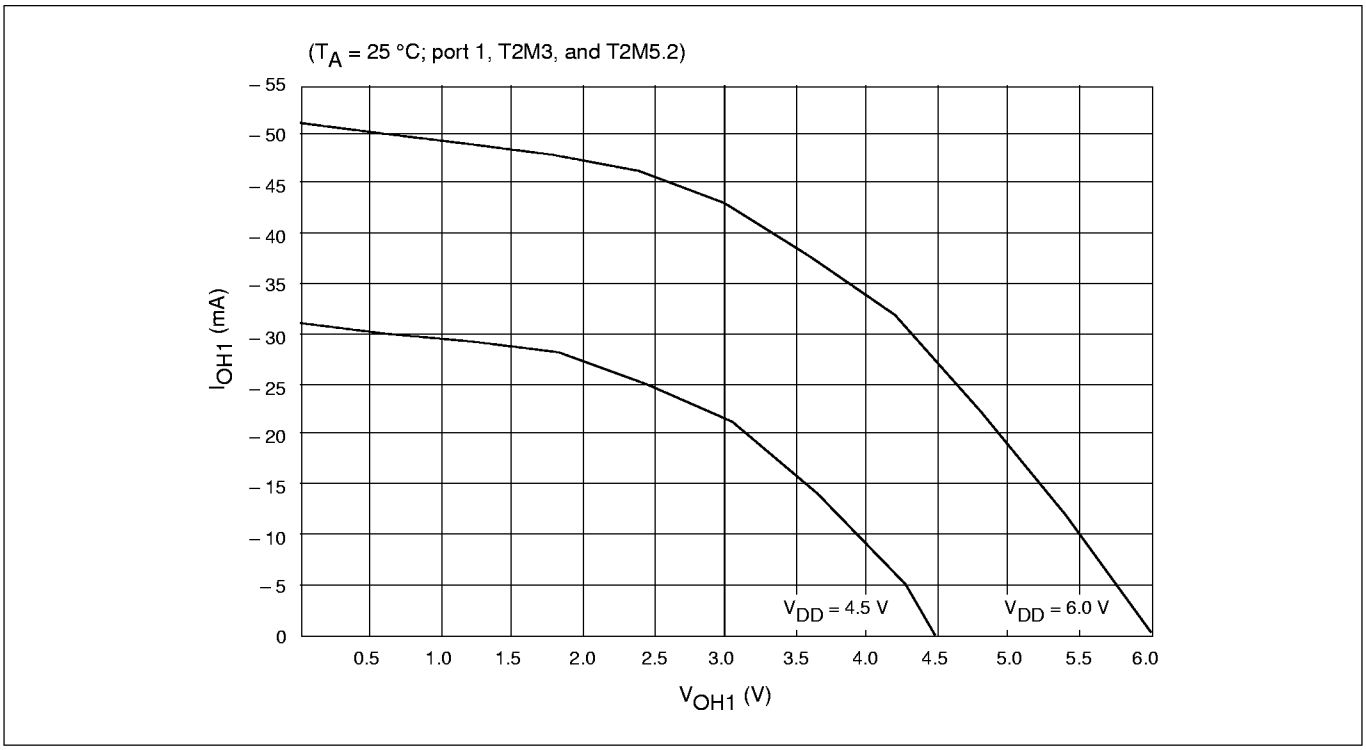


Figure 73. I_{OH1} vs. V_{OH1}

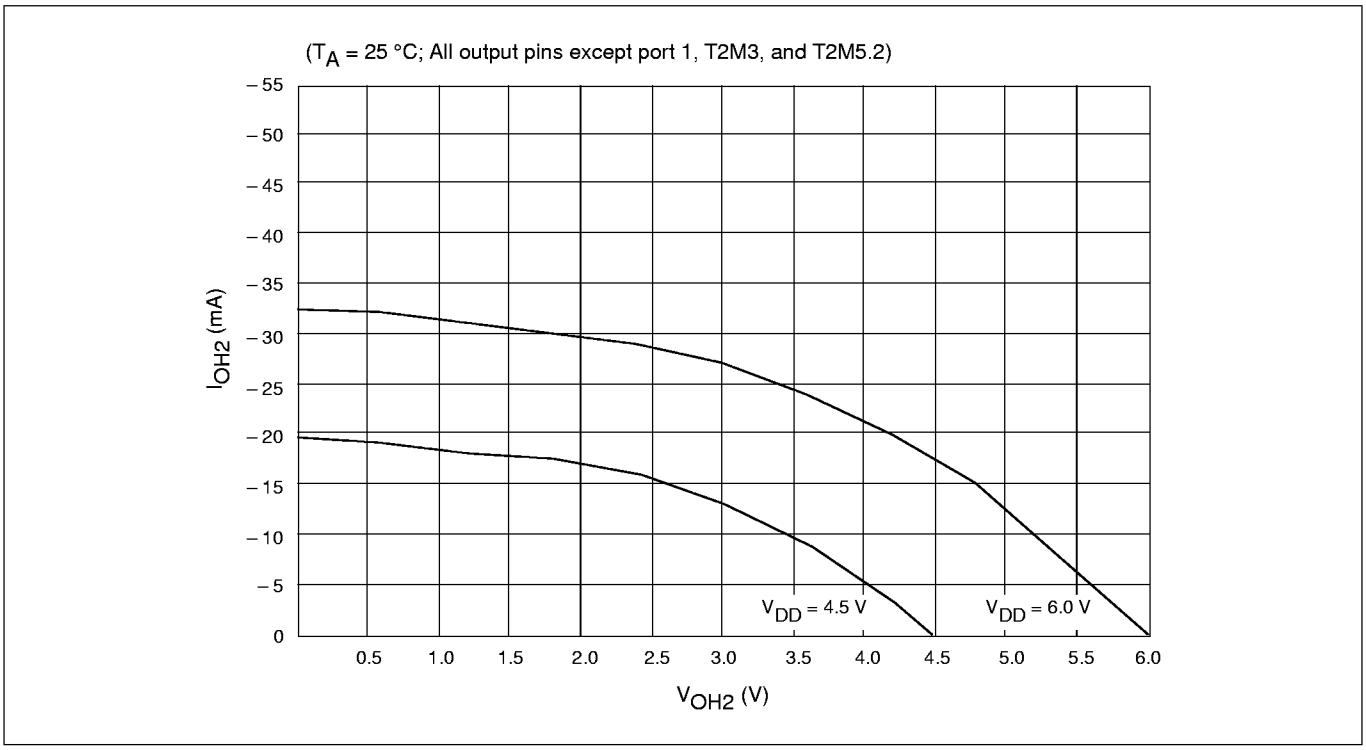


Figure 74. I_{OH2} vs. V_{OH2}

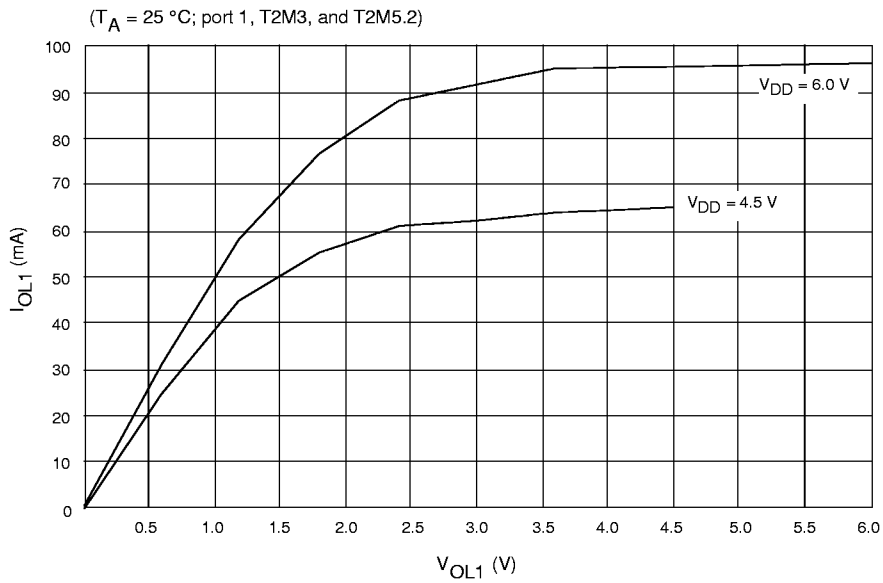


Figure 75. I_{OL1} vs. V_{OL1}

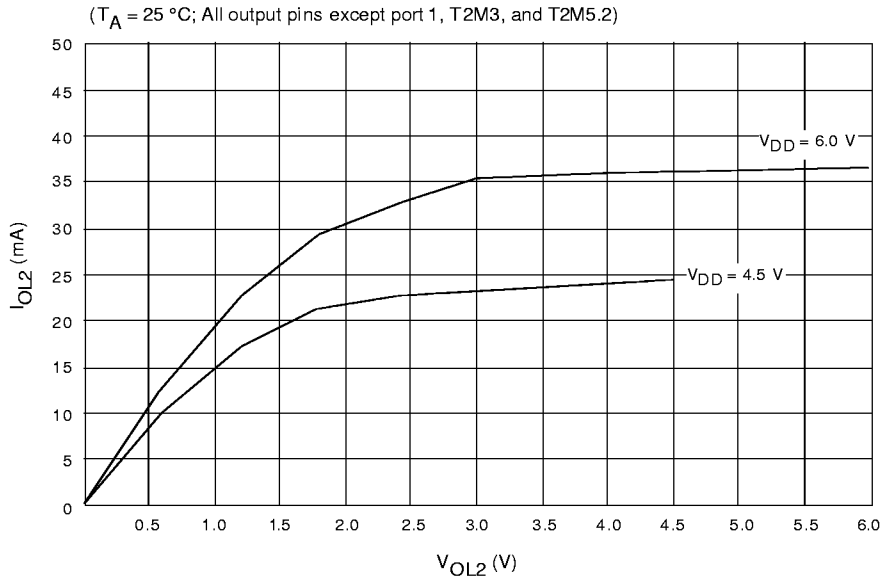


Figure 76. I_{OL2} vs. V_{OL2}