1 PRODUCT OVERVIEW

SAM87RI PRODUCT FAMILY

Samsung's SAM87RI family of 8-bit single-chip CMOS microcontrollers offer fast and efficient CPU, a wide range of integrated peripherals, and supports OTP device.

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

KS86C6104/P6104 MICROCONTROLLER

The KS86C6104/P6104 microcontroller with USB function can be used in a wide range of general purpose applications. It is especially suitable for mouse or joystick controller and is available in 20-pin DIP and 24-pin SOP package.

The KS86C6104/P6104 single-chip 8-bit microcontroller is fabricated using an advanced CMOS process. It is built around the powerful SAM87RI CPU core.

Stop and Idle power-down modes were implemented to reduce power consumption. To increase on-chip register space, the size of the internal register file was logically expanded. The KS86C6104/P6104 has 4 Kbytes of program memory on-chip and 208 bytes of RAM including 16 bytes of working register.

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

— Two configurable I/O ports (11 pins)

— 7 bit-programmable pins for external interrupts

— 8-bit timer/counter with two operating modes

OTP

The KS86C6104 microcontroller is also available in OTP (One Time Programmable) version, KS86P6104. KS86P6104 microcontroller has an on-chip 4-Kbyte one-time-programmable EPROM instead of masked ROM. The KS86P6104 is comparable to KS86C6104, both in function and in pin configuration.

FEATURES

CPU

- SAM87RI CPU core

Memory

- 4-Kbyte internal program memory (ROM)
- 208-byte RAM
- 16 bytes of working register

Instruction Set

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

Instruction Execution Time

- 1.0 µs at 6 MHz f_{OSC}

Interrupts

- 12 interrupt sources with one vector
- One level, one vector interrupt structure

Oscillation Circuit Options

- 6 MHz crystal/ceramic oscillator
- External clock source

General I/O

- 11 bit-programmable I/O pins

Timer/Counter

- One 8-bit basic timer for watchdog function and programmable oscillation stabilization interval generation function
- One 8-bit timer/counter with Compare/Overflow counter

USB Serial Bus

- Compatible to USB low speed (1.5 Mbps) device 1.0 specification.
- Serial bus interface engine (SIE)
  - — Packet decoding/generation
  - — CRC generation and checking
  - — NRZI encoding/decoding and bit-stuffing
- Two 8-byte receive/transmit USB buffer

Operating Temperature Range

- – 40°C to + 85°C

Operating Voltage Range

- 4.0 V to 5.25 V

Package Types

- 20-pin DIP
- 24-pin SOP

Comparator

- 4-channel mode, 4-bit resolution
- 3-channel mode, external reference low EMI design

BLOCK DIAGRAM

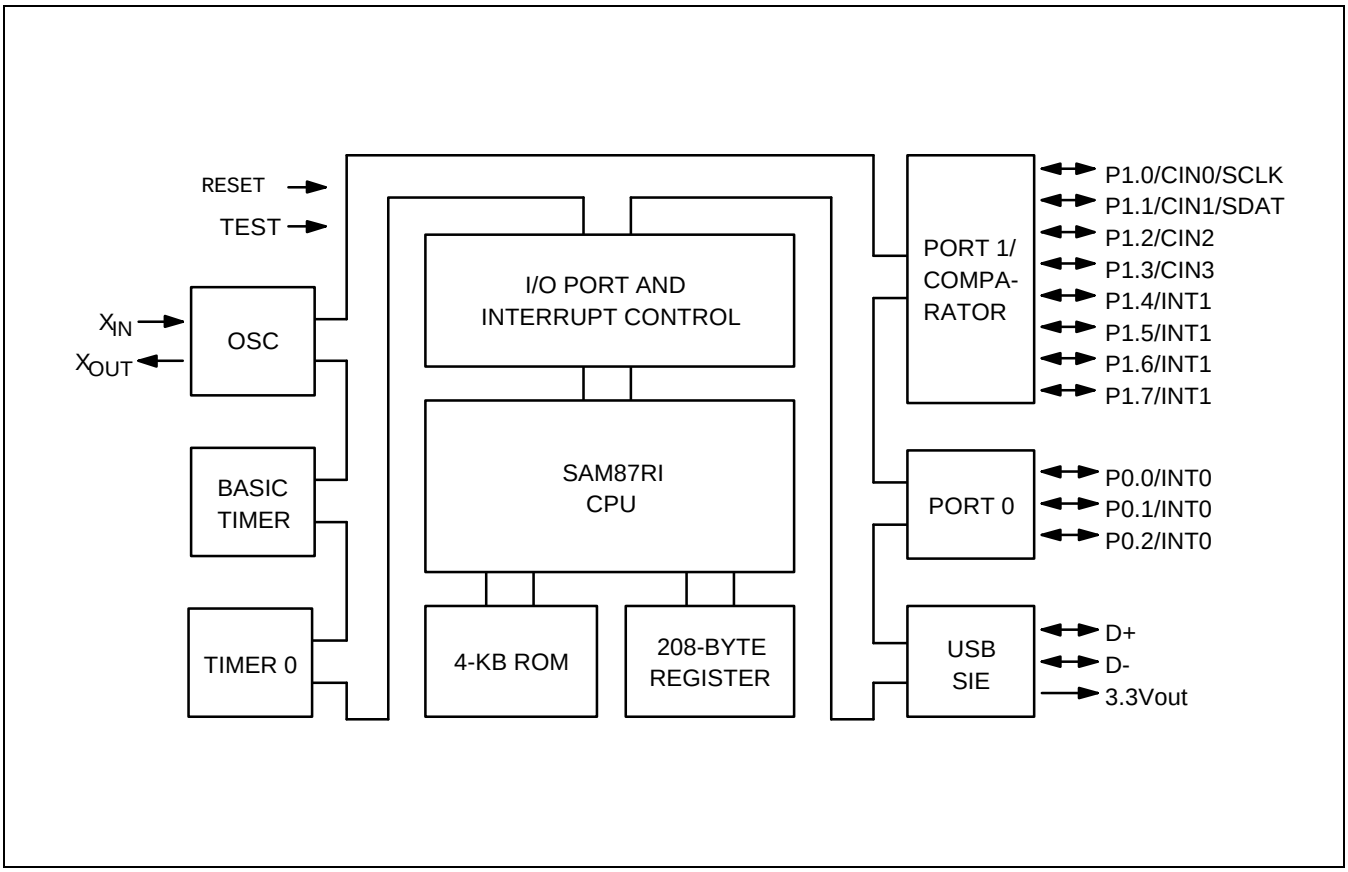


Figure 1-1. Block Diagram

PIN ASSIGNMENTS

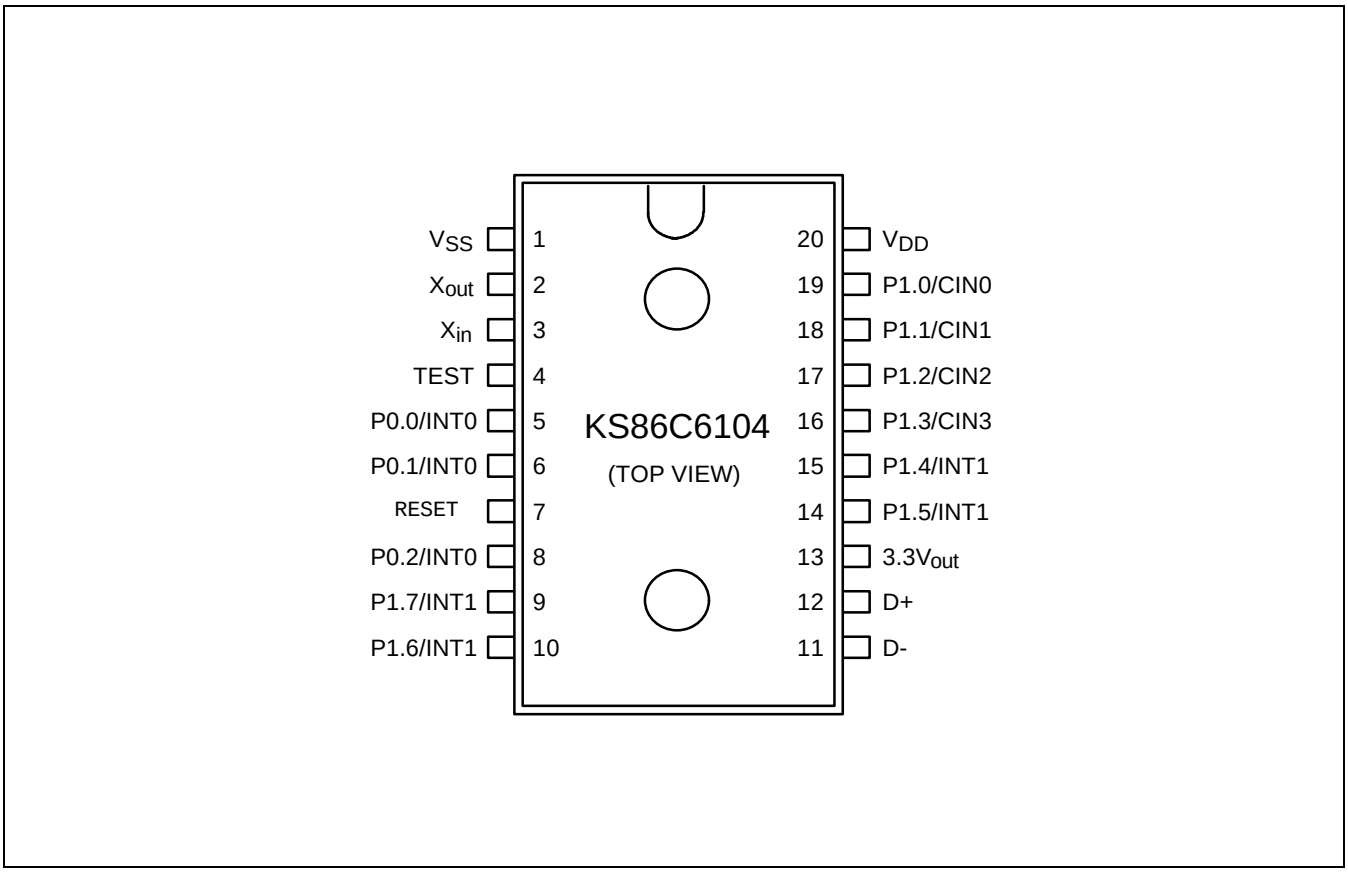


Figure 1-2. Pin Assignment Diagram (20-Pin DIP Package)

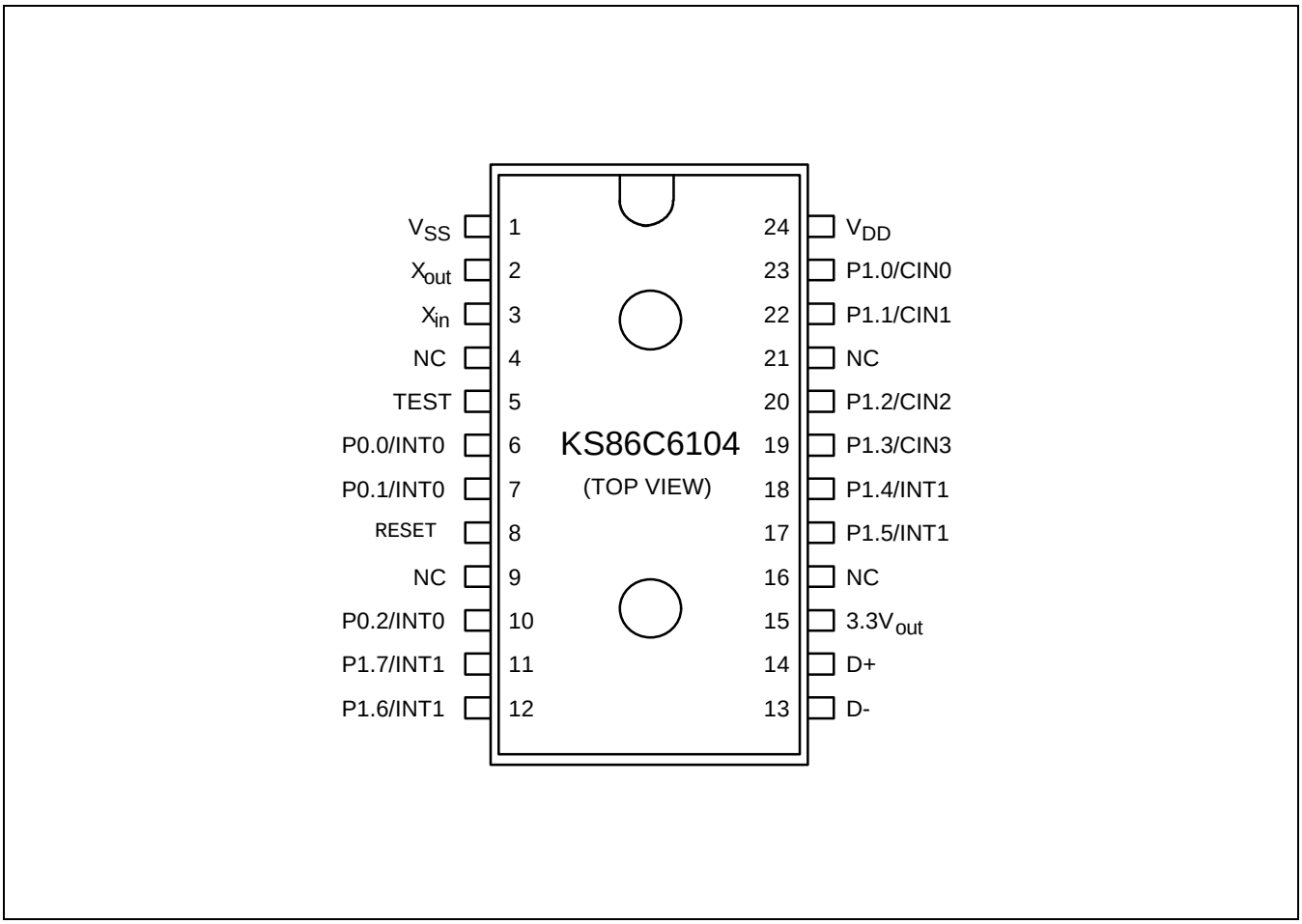


Figure 1-3. Pin Assignment Diagram (24-Pin SOP Package)

PIN DESCRIPTIONS

Table 1-1. KS86C6104/P6104 Pin Descriptions

Pin Names	Pin Type	Pin Description	Circuit Number	Pin Numbers	Share Pins
P0.0–P0.2	I/O	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are individually assignable to input pins by software and are automatically disable for output pins. Port0 can be individually configured as external interrupt inputs.	D	5, 6, 8	INT0
P1.0–P1.3	I/O	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are individually assignable to input pins by software. Port1.0–1.3 can be configured as comparator input	F-8	19–16	CIN0–CIN3
P1.4-P1.7	I/O	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are individually assignable to input pins by software and are automatically disabled for output pins. Port1.4–1.7 can be individually configured as external interrupt inputs.	D	15, 14, 10, 9	INT1
D+/D-	I/O	Only used as USB tranceive/receive port.	–	12–11	–
3.3VOUT	O	Internal regulator 3.3 V output pin for referencing the voltage	–	13	–
X_{IN}, X_{OUT}	–	System clock input and output pin (crystal/ceramic oscillator, or external clock source)	–	3–2	–
INT0	I	External interrupt for bit-programmable port0.	D	5, 6, 8	Port0
INT1	I	External interrupt for bit-programmable port1	D	9, 10, 14, 15	Port1
RESET	I	RESET signal input pin.	–	7	–
TEST	I	Test signal input pin (for factory use only; must be connected to V_{SS})	–	4	–
V_{DD}	–	Power input pin	–	20	–
V_{SS}	–	V_{SS} is a ground power for CPU core.	–	1	–

PIN CIRCUITS

Table 1-2. Pin Circuit Assignments for the KS86C6104/P6104

Circuit Number	Circuit Type	KS86C6104/P6104 Assignments
C	O	
D	I/O	Port0, Port1.4–1.7, INT0, INT1
F-8	I/O	Port1.0–1.3

NOTE: Diagrams of circuit types C–D, and F-8 are presented below.

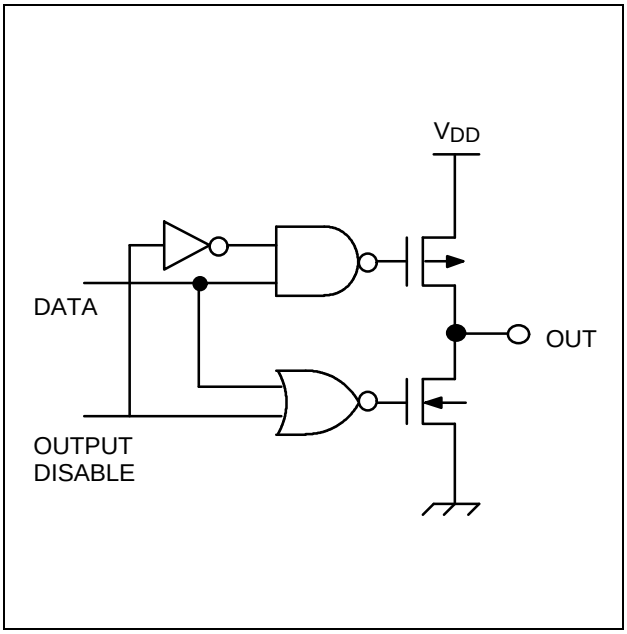


Figure 1-4. Pin Circuit Type C

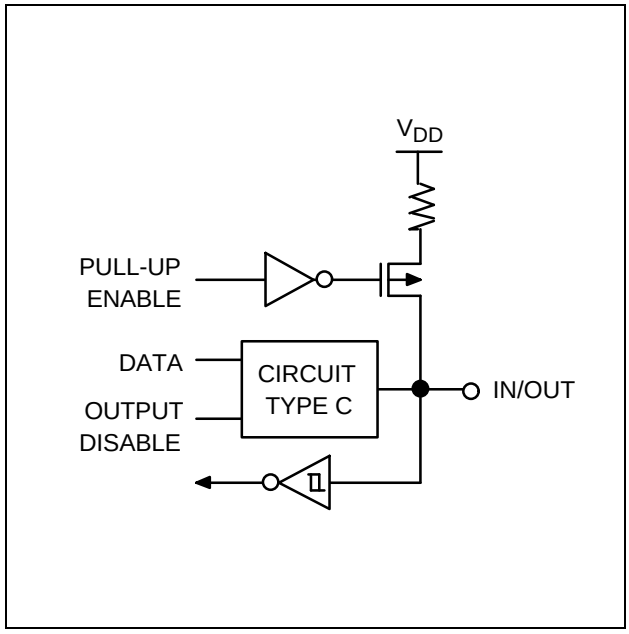


Figure 1-5. Pin Circuit Type D

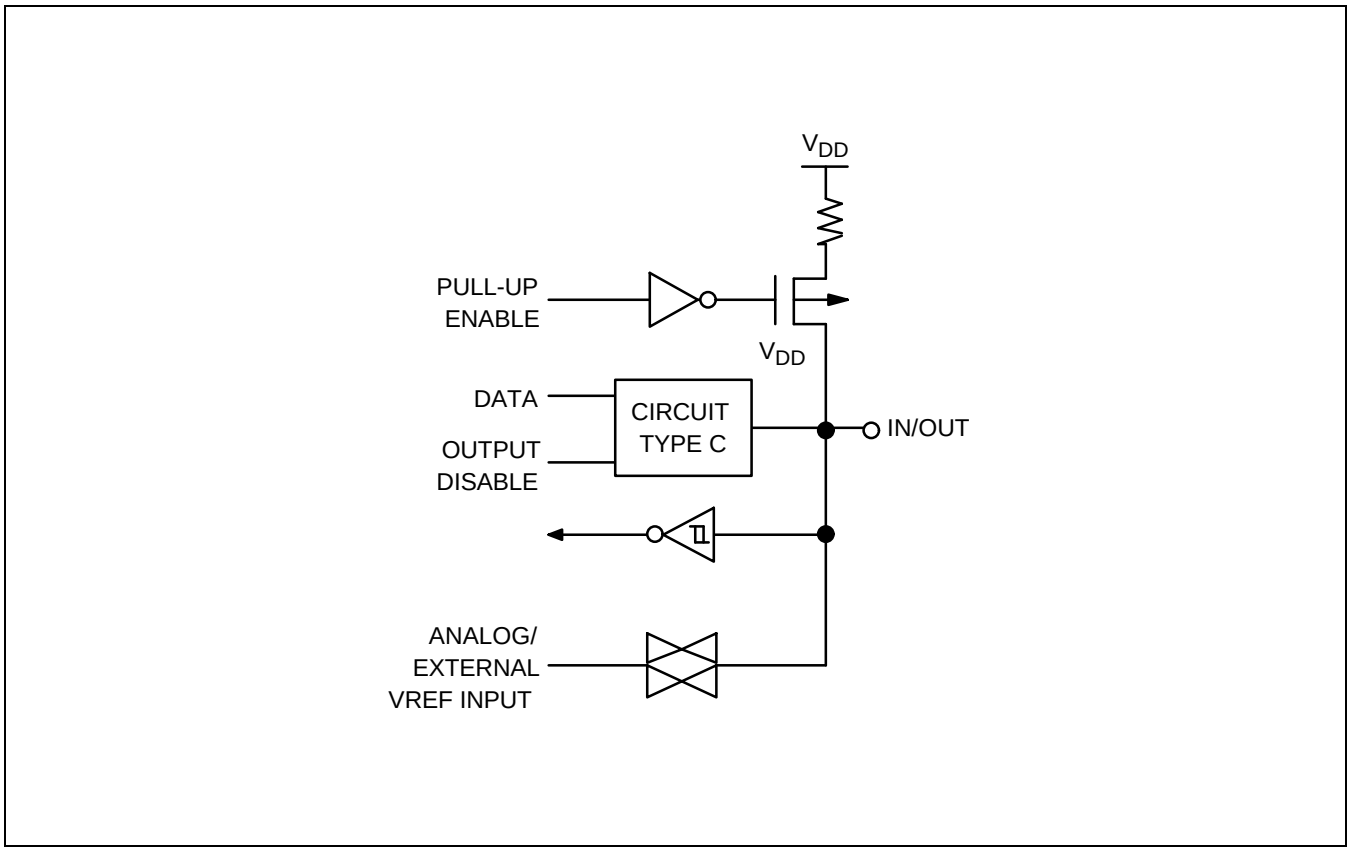


Figure 1-6. Pin Circuit Type F-8

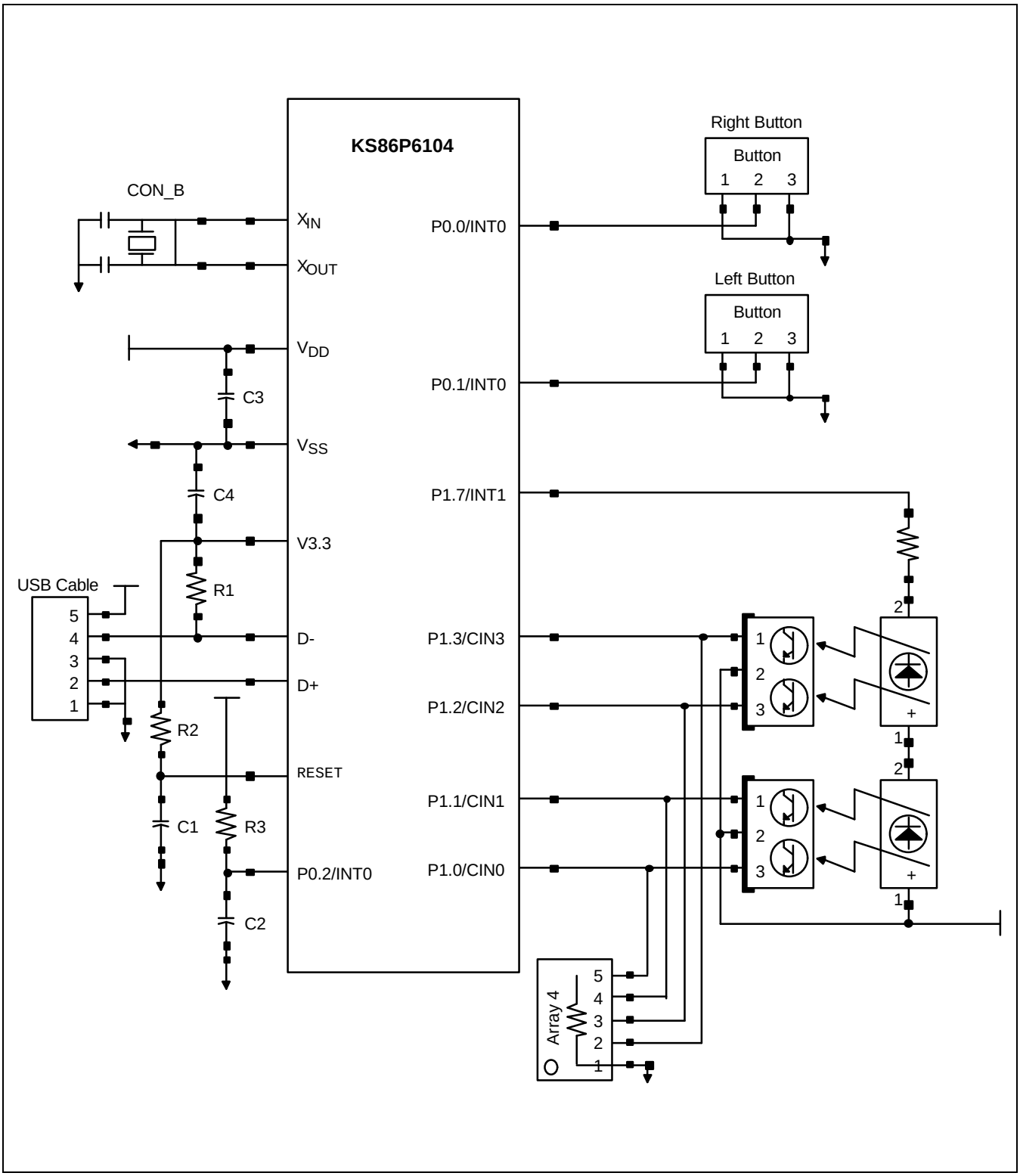


Figure 1-7. USB Mouse Circuit Diagram

NOTES

2 ADDRESS SPACES

OVERVIEW

The KS86C6104/P6104 microcontroller has two kinds of address space:

— Program memory (ROM)
— Internal register file

A 13-bit address bus supports both program memory. Special instructions and related internal logic determine when the 13-bit bus carries addresses for program memory. A separate 8-bit register bus carries addresses and data between the CPU and the internal register file.

The KS86C6104 has 4 K bytes of mask-programmable program memory on-chip. The KS86C6104/P6104 microcontroller has 192 bytes general-purpose registers in its internal register file. Forty-eight bytes in the register file are mapped for system and peripheral control functions.

PROGRAM MEMORY (ROM)

NORMAL OPERATING MODE (INTERNAL ROM)

The KS86C6104/P6104 has 4 K bytes (locations 0H–0FFFH) of internal mask-programmable program memory. The first 2 bytes of the ROM (0000H–0001H) are an interrupt vector address.
The program reset address in the ROM is 0100H.

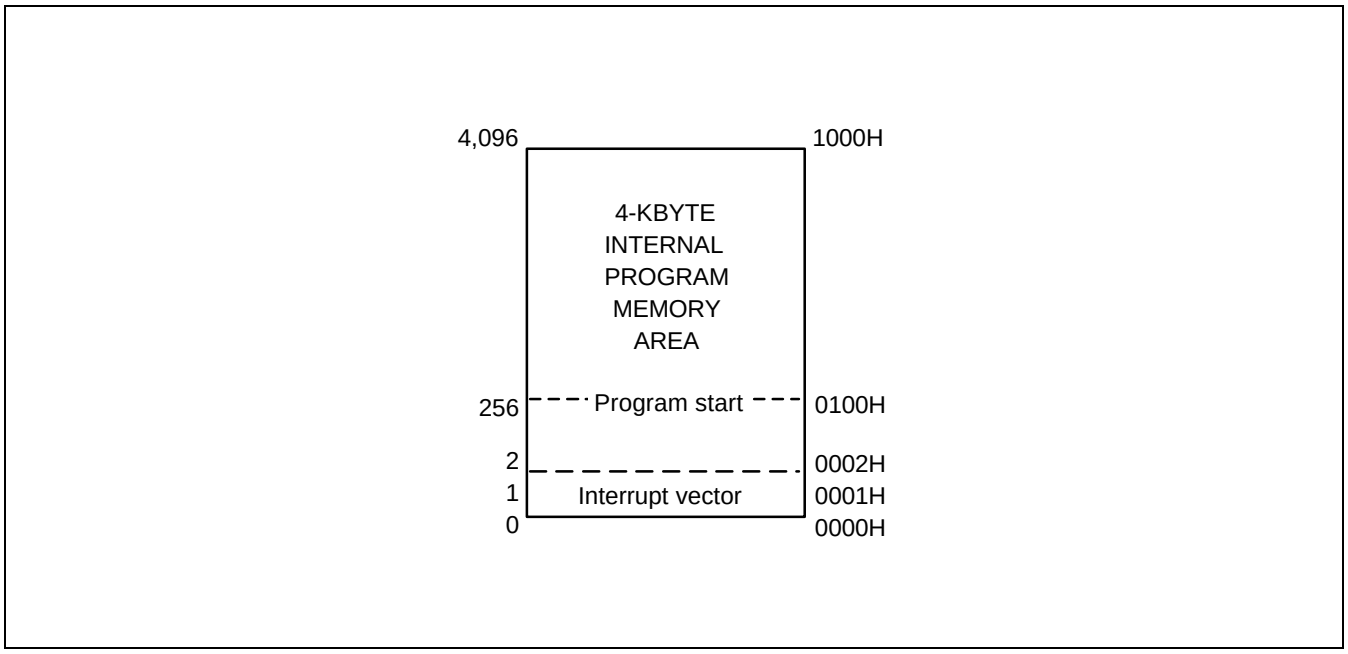


Figure 2-1. Program Memory Address Space

REGISTER ARCHITECTURE

The upper 64 bytes of the KS86C6104/P6104's internal register file are addressed as working registers, system control registers and peripheral control registers. The lower 192 bytes of internal register file (00H–BFH) is called the *general purpose register space*.

For many SAM87RI microcontrollers, the addressable area of the internal register file is further expanded by the additional of one or more register pages at general purpose register space (00H–BFH). This register file expansion is not implemented in the KS86C6104/P6104.

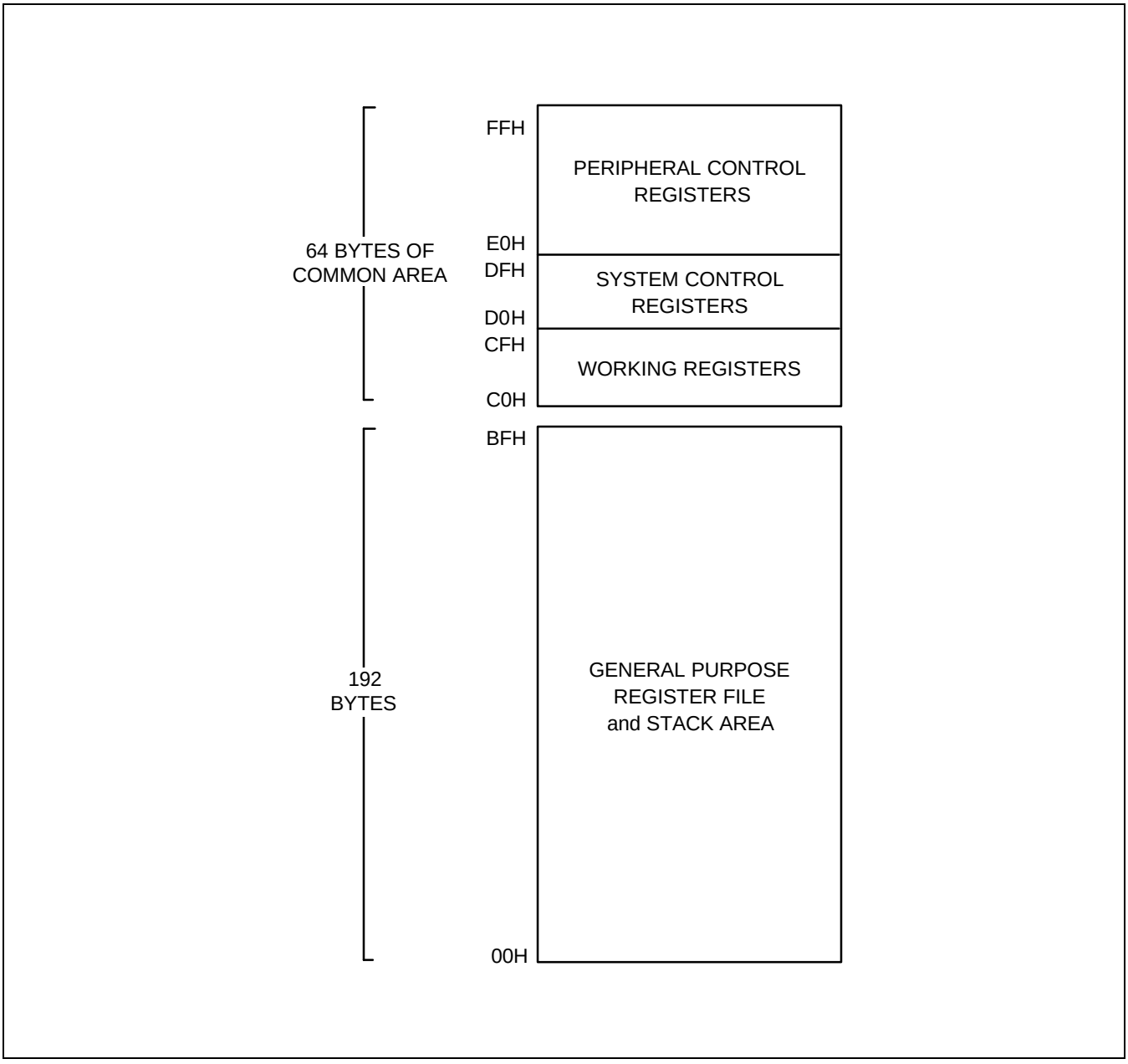


Figure 2-2. Internal Register File Organization

COMMON WORKING REGISTER AREA (C0H–CFH)

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

This16-byte address range is called *common area*. That is, locations in this area can be used as working registers by operations that address any location on any page in the register file. Typically, these working registers serve as temporary buffers for data operations between different pages. However, because the KS86C6104/P6104 uses only page 0, you can use the common area for any internal data operation.

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

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

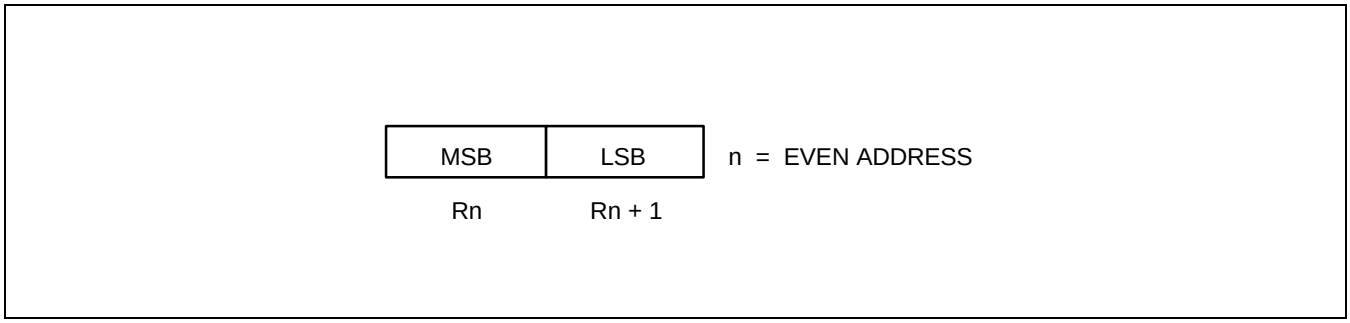


Figure 2-3. 16-Bit Register Pairs

☞ PROGRAMMING TIP — Addressing the Common Working Register Area

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

Examples:

1. LD 0C2H,40H ; Invalid addressing mode!

   Use working register addressing instead:

   LD R2,40H ; R2 (C2H) ← the value in location 40H

2. ADD 0C3H,#45H ; Invalid addressing mode!

   Use working register addressing instead:

   ADD R3,#45H ; R3 (C3H) ← R3 + 45H

SYSTEM STACK

KS86-series microcontrollers use the system stack for subroutine calls and returns and to store data. The PUSH and POP instructions are used to control system stack operations. The KS86C6104/P6104 architecture supports stack operations in the internal register file.

STACK OPERATIONS

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

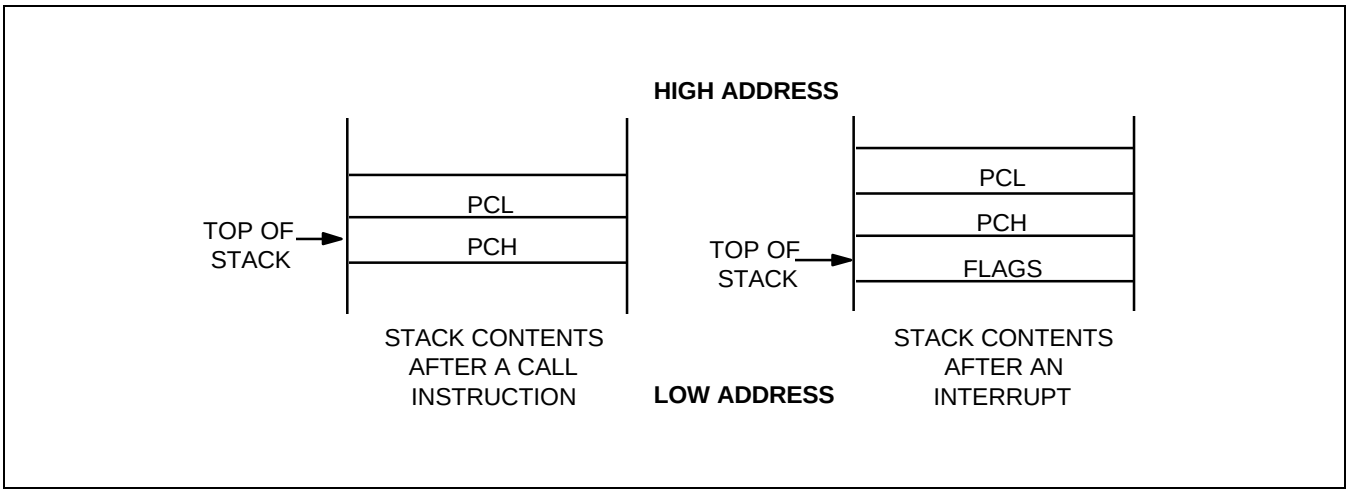


Figure 2-4. Stack Operations

STACK POINTER (SP)

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

Because only internal memory space is implemented in the KS86C6104/P6104, the SP must be initialized to an 8-bit value in the range 00H–BFH.

NOTE

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

☞ PROGRAMMING TIP — Standard Stack Operations Using PUSH and POP

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

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

3 ADDRESSING MODES

OVERVIEW

Instructions that are stored in program memory are fetched for execution using the program counter. Instructions indicate the operation to be performed and the data to be operated on. Addressing mode is the method used to determine the location of the data operand. The operands specified in SAM87RI instructions may be condition codes, immediate data, or a location in the register file, program memory, or data memory.

The SAM87RI instruction set supports six explicit addressing modes. Not all of these addressing modes are available for each instruction. The addressing modes and their symbols are as follows:

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

REGISTER ADDRESSING MODE (R)

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

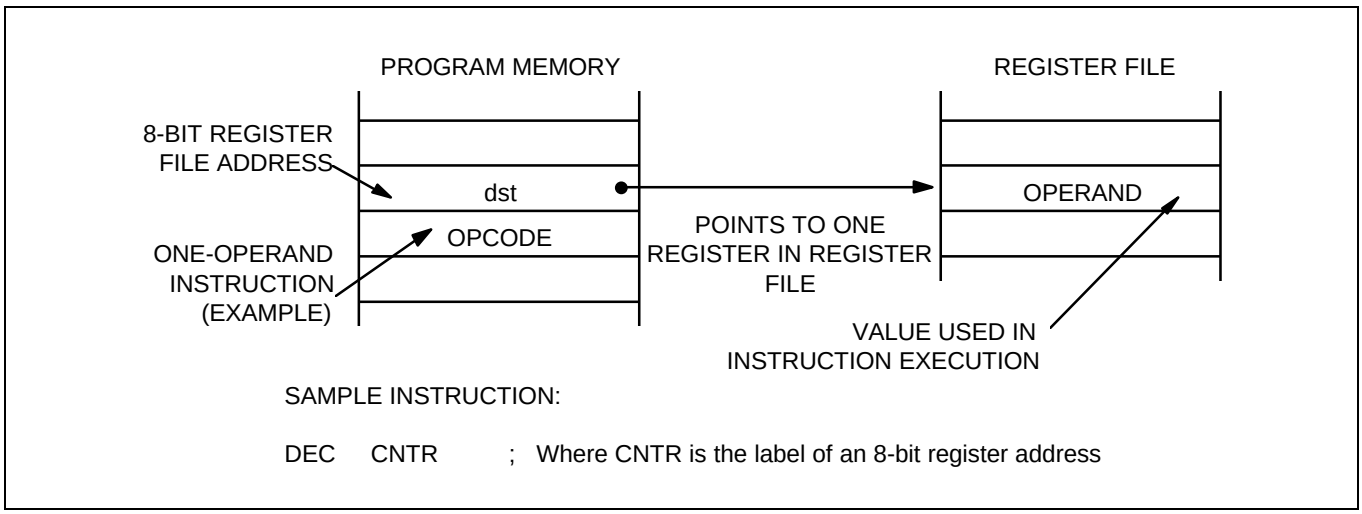


Figure 3-1. Register Addressing

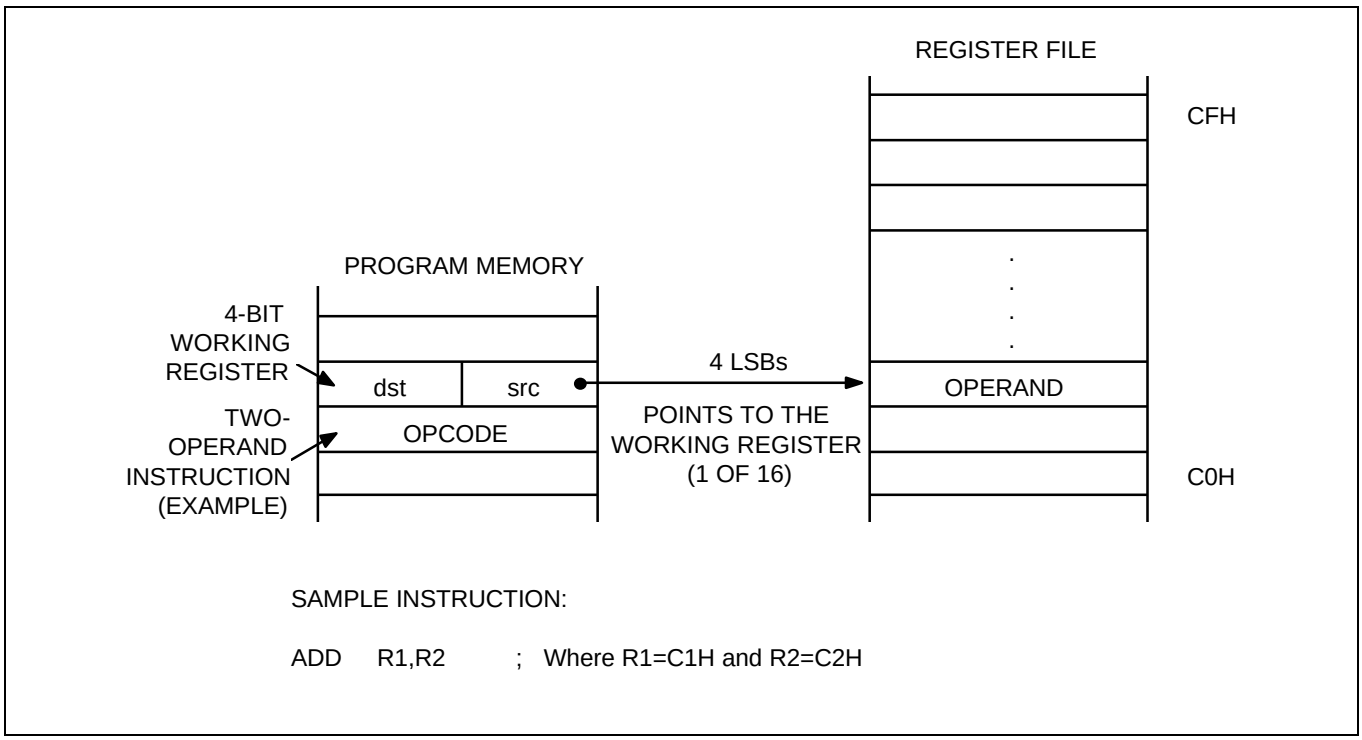


Figure 3-2. Working Register Addressing

INDIRECT REGISTER ADDRESSING MODE (IR)

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

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

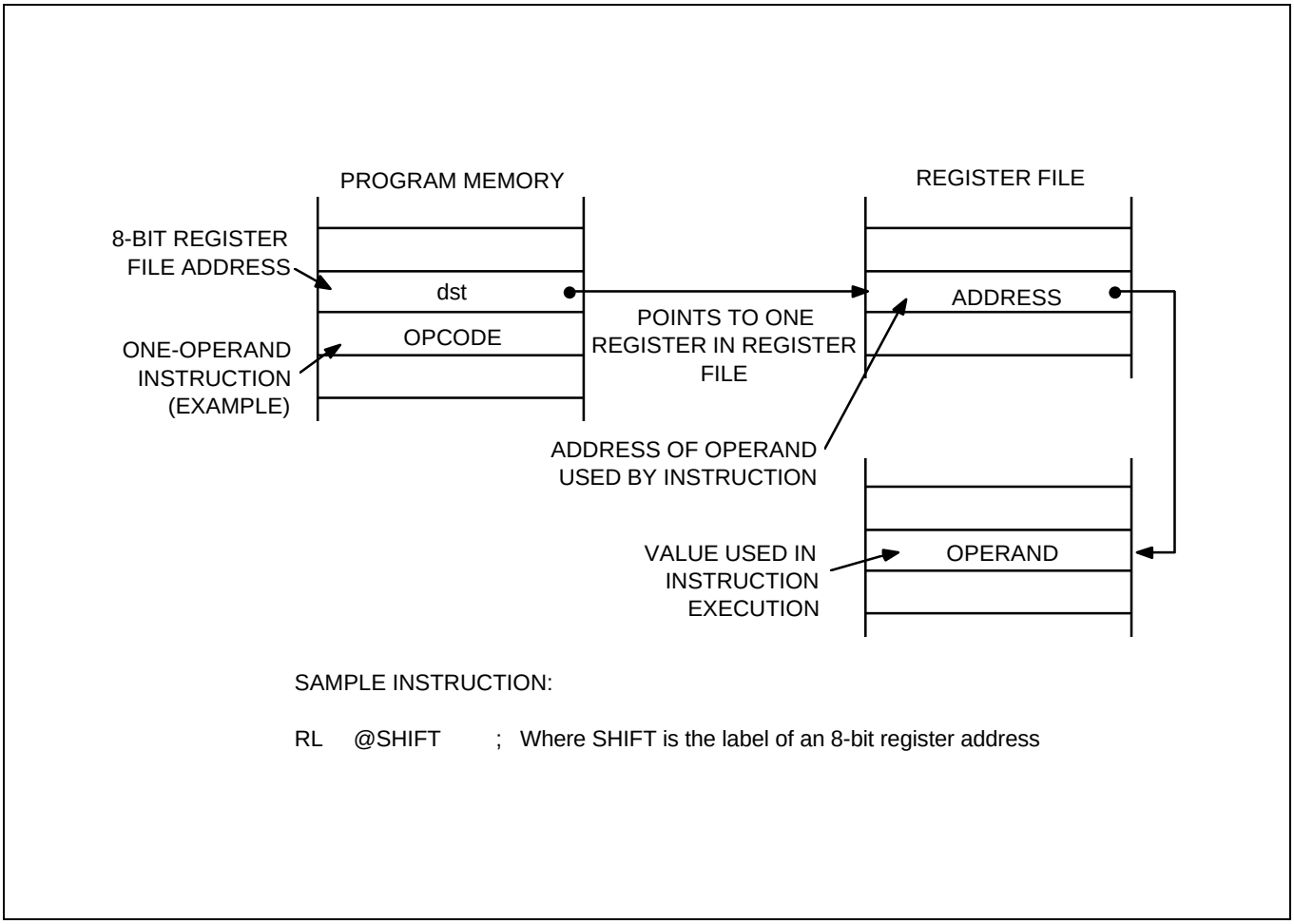


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

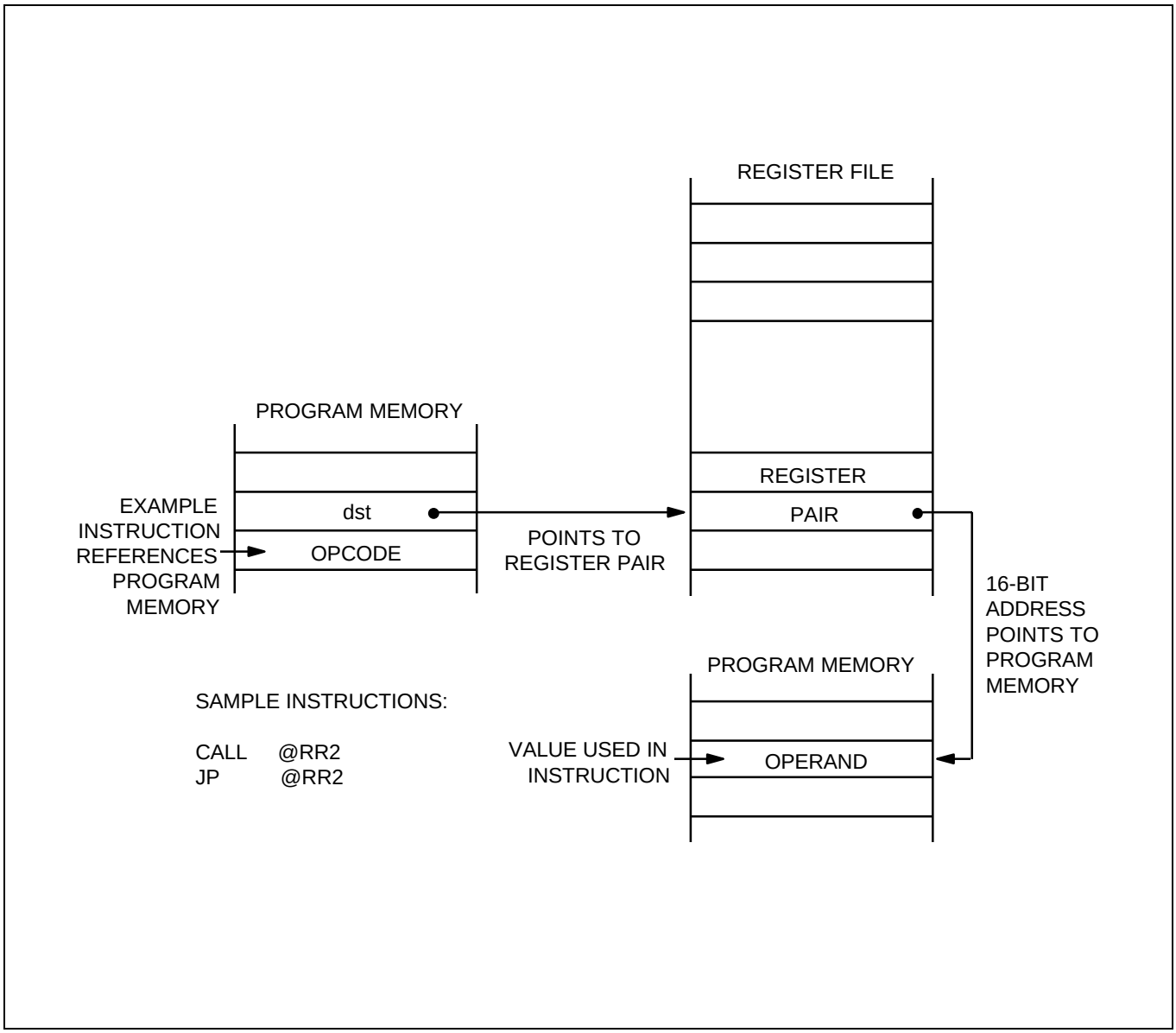


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

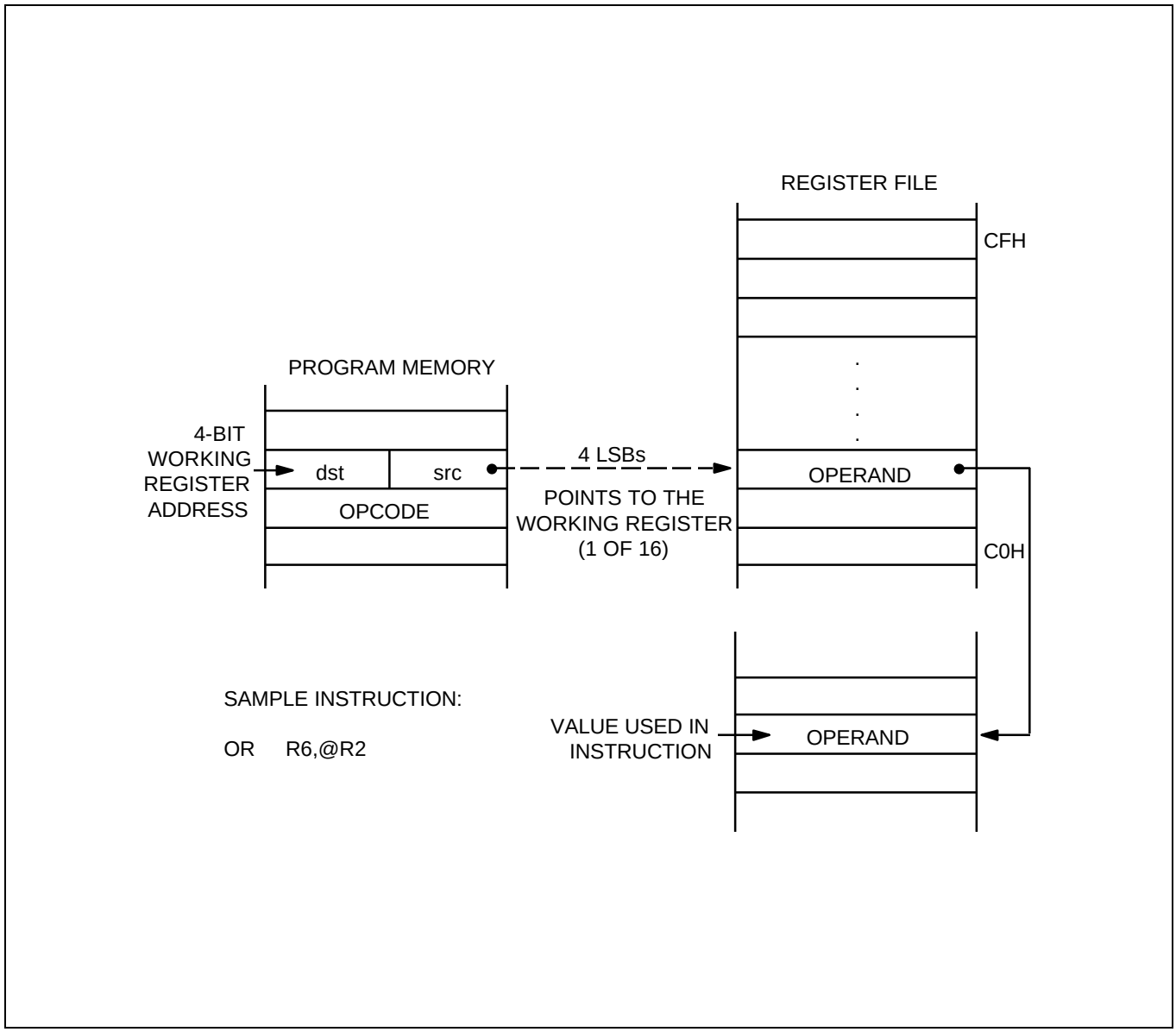


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Concluded)

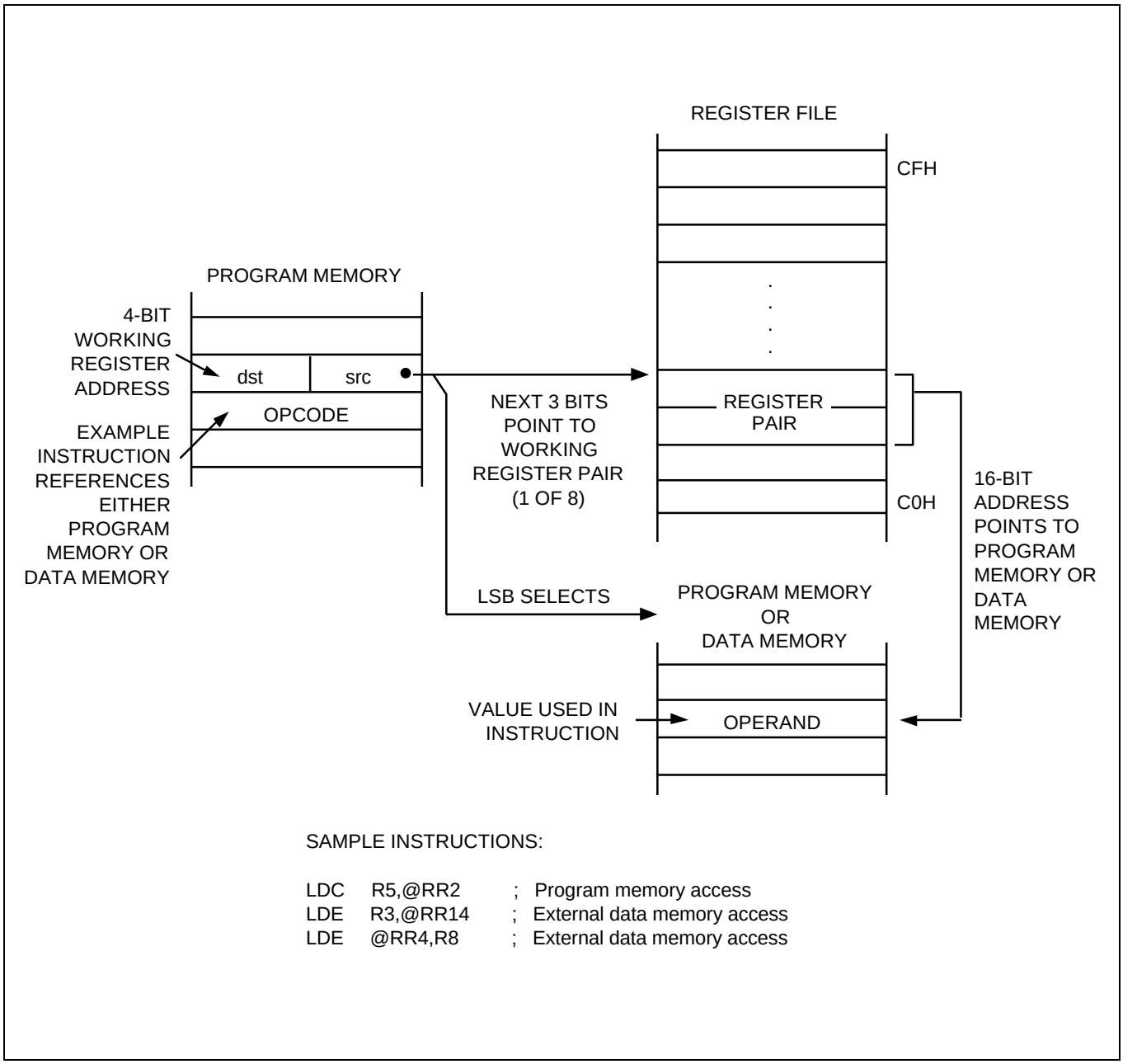


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

INDEXED ADDRESSING MODE (X)

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

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

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

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

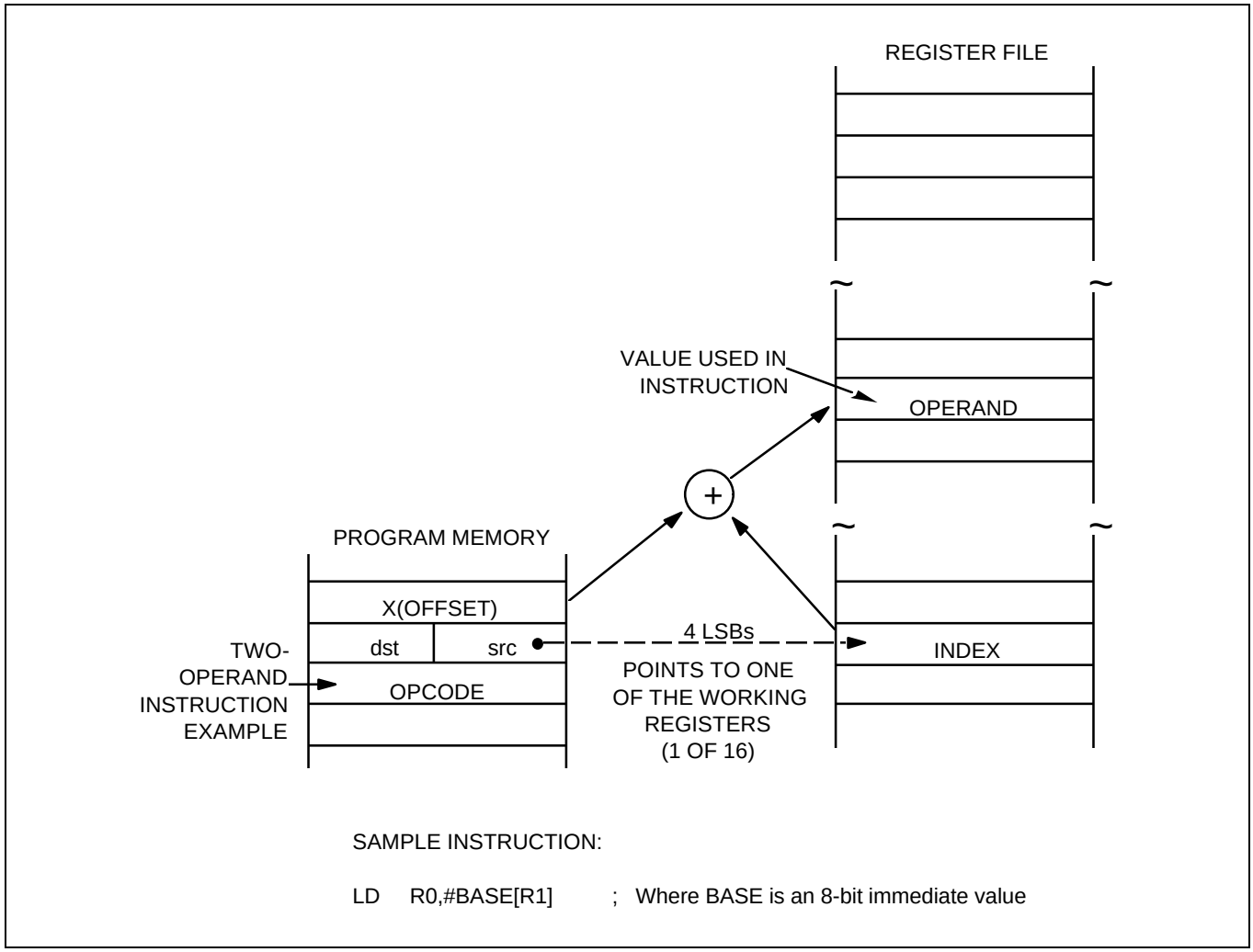
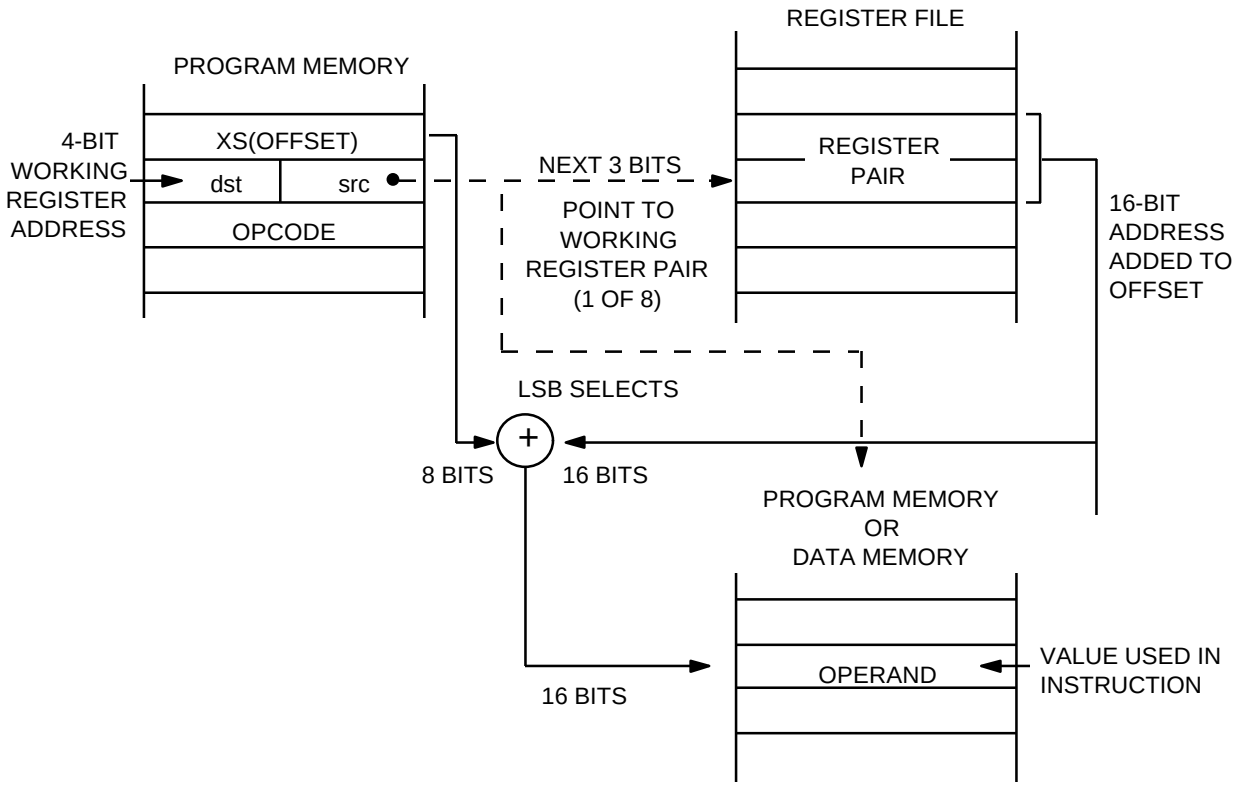


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

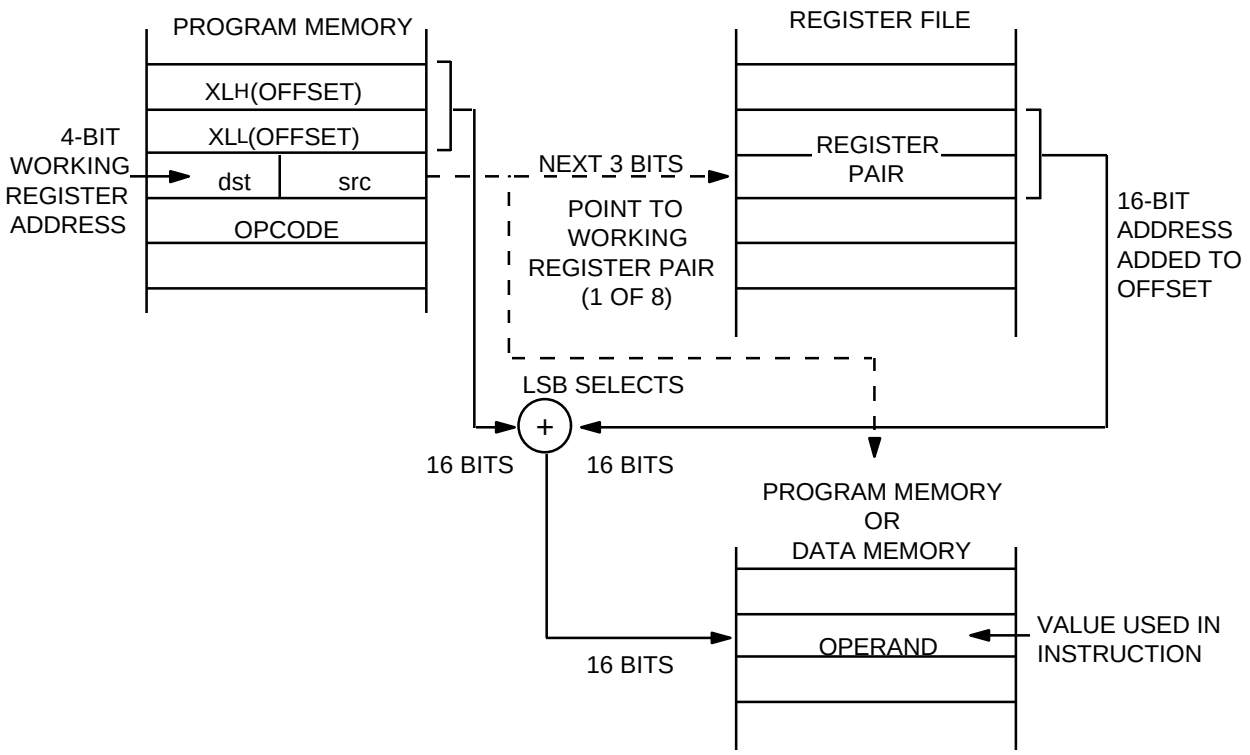


SAMPLE INSTRUCTIONS:

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

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

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

INDEXED ADDRESSING MODE (Concluded)



SAMPLE INSTRUCTIONS:

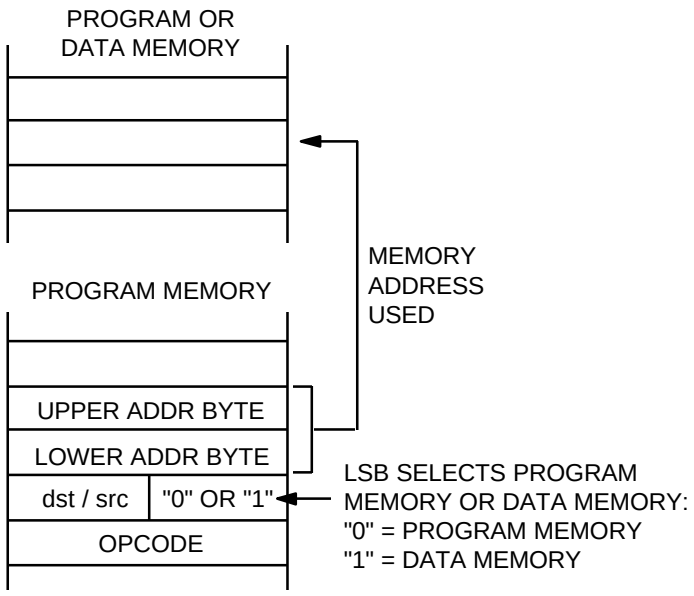
```
LDC     R4,#1000H[RR2]      ;  The values in the program address (RR2 + #1000H)
                               are loaded into register R4.

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

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

DIRECT ADDRESS MODE (DA)

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

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

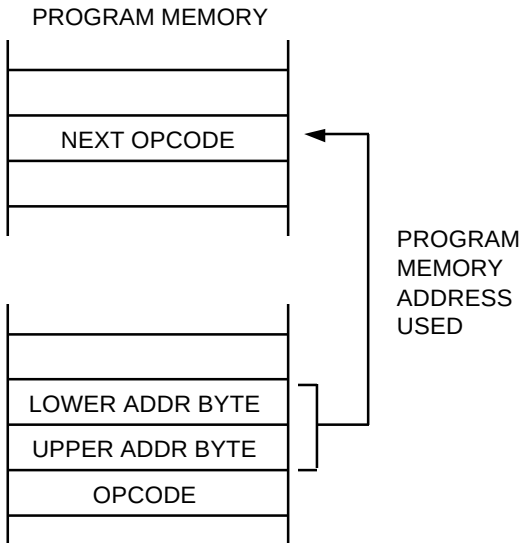


SAMPLE INSTRUCTIONS:

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

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

Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)



SAMPLE INSTRUCTIONS:

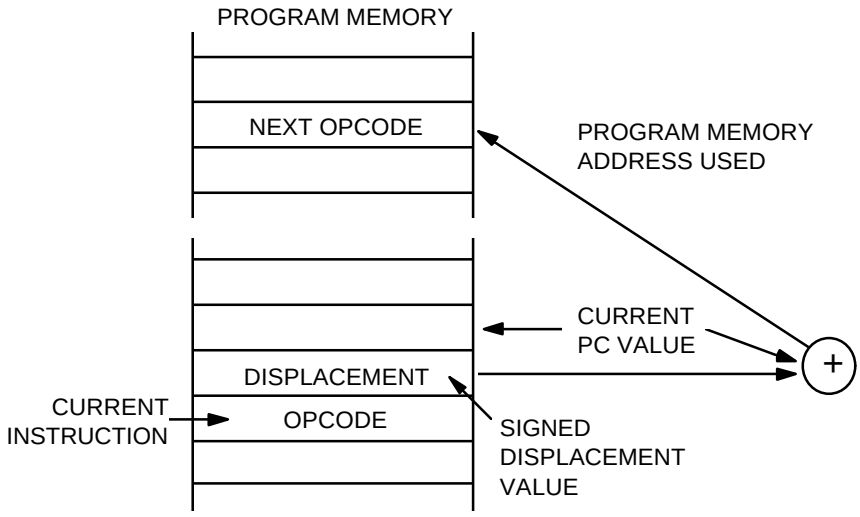
```
JP      C,JOB1      ;  Where JOB1 is a 16-bit immediate address

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

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

RELATIVE ADDRESS MODE (RA)

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

The instructions that support RA addressing is JR.



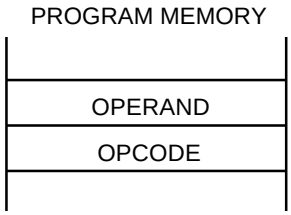
SAMPLE INSTRUCTION:

JR ULT,$+OFFSET ; Where OFFSET is a value in the range +127 to –128

Figure 3-12. Relative Addressing

IMMEDIATE MODE (IM)

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



(THE OPERAND VALUE IS IN THE INSTRUCTION)

SAMPLE INSTRUCTION: LD R0,#0AAH

Figure 3-13. Immediate Addressing

4 CONTROL REGISTERS

OVERVIEW

In this section, detailed descriptions of the KS86C6104/P6104 control registers are presented in an easy-to-read format. These descriptions will help familiarize you with the mapped locations in the register file. You can also use them as a quick-reference source when writing application programs.

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

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

Table 4-1. System and Peripheral Control Registers

Register Name	Mnemonic	Hex	R/W
Timer 0 counter register	T0CNT	D0H	R
Timer 0 data register	T0DATA	D1H	R/W
Timer 0 control register	T0CON	D2H	R/W
Location D3H is not mapped.			
Clock control register	CLKCON	D4H	R/W
System flags register	FLAGS	D5H	R/W
Locations D6H–D8H are not mapped.			
Stack pointer	SP	D9H	R/W
Locations DAH–DBH are not mapped.			
Basic timer control register	BTCON	DCH	R/W
Basic timer counter	BTCNT	DDH	R
Location DEH is not mapped.			
System mode register	SYM	DFH	R/W
Port 0 data register	P0	E0H	R/W
Port 1 data register	P1	E1H	R/W
Locations E2H–E4H are not mapped.			
Comparison result register	CDATA	E5H	R
Port 0 control register	P0CON	E6H	R/W
Comparator control mode register	CCON	E7H	R/W
Port 1 control register (high nibble)	P1CONH	E8H	R/W
Port 1 control register (low nibble)	P1CONL	E9H	R/W
Port 0 interrupt control register	P0INT	EAH	R/W
Port 0 interrupt pending register	P0PND	EBH	R/W
Port 1 interrupt control register	P1INT	ECH	R/W
Port 1 interrupt pending register	P1PND	EDH	R/W
Locations EEH–EFH are not mapped.			
USB function address register	FADDR	F0H	R/W
Control endpoint status register	EP0CSR	F1H	R/W
Interrupt endpoint status register	EP1CSR	F2H	R/W
Control endpoint byte count register	EP0BCNT	F3H	R/W
Control endpoint FIFO register	EP0FIFO	F4H	R/W
Interrupt endpoint FIFO register	EP1FIFO	F5H	R/W
USB interrupt pending register	USBPND	F6H	R/W
USB interrupt enable register	USBINT	F7H	R/W
USB power management register	PWRMGR	F8H	R/W
Locations F9H–FEH are not mapped.			
USB reset register	USBRST	FFH	R/W

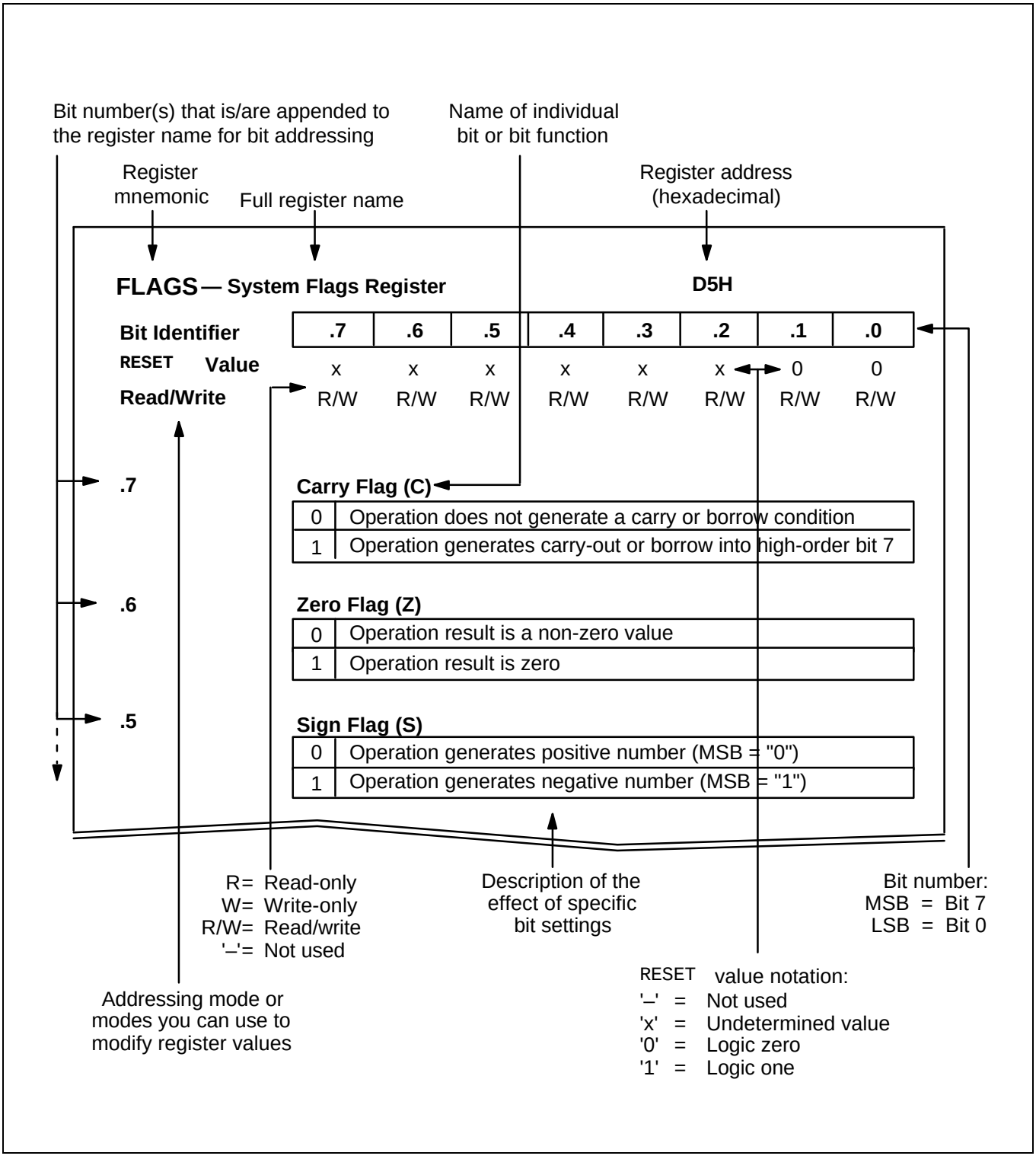


Figure 4-1. Register Description Format

BTCON — Basic Timer Control Register DCH

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

.7 – .4 Watchdog Timer Enable Bits

1	0	1	0	Disable watchdog function
Any other value				Enable watchdog function

.3 – .2 Basic Timer Input Clock Selection Bits

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

.1 Basic Timer Counter Clear Bit (note)

0	No effect
1	Clear BTCNT

.0 Basic Timer Divider Clear Bit (note)

0	No effect
1	Clear both dividers

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

CCON — Comparator Mode Register E7H

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

.7 Comparator Enable Bit

0	Disable comparator
1	Enable comparator

.6 Conversion Time

0	Conversion time (4×2^7/fx)
1	Conversion time (4×2^4/fx)

.5 External Reference Voltage

0	Internal reference voltage
1	External reference voltage

.4

Always logic zero

.3 – .0 Reference voltage (Vref) selection

$V_{DD} \times (n + 0.7)/16$, n = 0 to 15

CLKCON — System Clock Control Register D4H

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

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

0	Enable IRQ for main system oscillator wake-up in power down mode
1	Disable IRQ for main system oscillator wake-up in power down mode

.6 and .5

Not used for KS86C6104/P6104

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

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

.2 – .0

Not used for KS86C6104/P6104

EP0CSR — Control Endpoint Status Register F1H

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

.7 SETUP_END Clear Bit

0	No effect (when write)
1	Clear SETUP_END (bit4) bit

.6 OUT_PKT_RDY Clear Bit

0	No effect (when write)
1	Clear OUT_PKT_RDY (bit0) bit

.5 STALL Signal Sending Bit

0	No effect (when write)
1	Send STALL signal to host

.4 Setup Transfer End Bit

0	No effect (when write)
1	SIE sets this bit when a control transfer ends before DATA_END (bit3) is set

.3 Setup Data End Bit

0	No effect (when write)
1	MCU set this bit after loading or unloading the last packet data into the FIFO

.2 STALL Signal Receive Bit

0	MCU clear this bit to end the STALL condition
1	SIE sets this bit if a control transaction is ended due to a protocol violation

.1 In Packet Ready Bit

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into Endpoint0 FIFO

.0 Out Packet Ready Bit

0	No effect (when write)
1	SIE sets this bit once a valid token is written to the FIFO

EP1CSR — Interrupt Endpoint Status Register F2H

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

.7 DATA_TOGGLE Clear Bit

0	No effect (when write)
1	Clears the data toggle sequence bit

.6 – .3 Maximum Packet Size Bits

0	No effect (when write)
1	Indicates the maximum packet size for interrupt endpoint

.2 FIFO Flush Bit

0	No effect (when write)
1	FIFO is flushed, and IN_PKT_RDY cleared

.1 Force STALL Bit

0	MCU clears this bit to end the STALL condition
1	Issues a STALL handshake to USB

.0 In Packet Ready Bit

0	SIE clear this bit once the packet has been successfully sent to the host
1	MCU sets this bit after writing a packet of data into Endpoint1 FIFO

FLAGS — System Flags Register D5H

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

.7 Carry Flag (C)

0	Operation does not generate a carry or borrow condition

.6 Zero Flag (Z)

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

.5 Sign Flag (S)

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

.4 Overflow Flag (V)

0	Operation result is $\leq$ +127 or $\geq$ –128
1	Operation result is $\geq$ +127 or $\leq$ –128

.3 – .0

Not used for KS86C6104/P6104

P0CON — Port 0 Control Register E6H

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

.7 and .6

Not used for KS86C6104/P6104

.5 and .4 **Port 0, P0.2 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up resistor
1	0	Output mode, push-pull
1	1	Not used

.3 and .2 **Port 0, P0.1 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up resistor
1	0	Outmode, push-pull
1	1	Not used

.1 and .0 **Port 0, P0.0 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up resistor
1	0	Output mode, push-pull
1	1	Not used

P0INT — Port 0 Interrupt Control Register EAH

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

.7 – .3

Not used for KS86C6104/P6104

.2 P0.2 Interrupt Enable Bits

0	External interrupt disable
1	External interrupt enable

.1 P0.1 Interrupt Enable Bits

0	External interrupt disable
1	External interrupt enable

.0 P0.0 Interrupt Enable Bits

0	External interrupt disable
1	External interrupt enable

P0PND — Port 0 Interrupt Pending Register EBH

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

.7 – .3

Not used for KS86C6104/P6104

.2 P0.2 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.1 P0.1 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.0 P0.0 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

P1CONH — Port 1 Control Register (High Byte) E8H

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

.7 and .6 **Port 1, P1.7 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up register
1	0	Output mode, push-pull
1	1	Not used

.5 and .4 **Port 1, P1.6 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up register
1	0	Output mode, push-pull
1	1	Not used

.3 and .2 **Port 1, P1.5 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up register
1	0	Output mode, push-pull
1	1	Not used

.1 and .0 **Port 1, P1.4 Configuration Bits**

0	0	Input, rising edge external interrupt
0	1	Input, falling edge external interrupt with pull-up register
1	0	Output mode, push-pull
1	1	Not used

P1CONL — Port 1 Control Register (Low Byte) E9H

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

.7 and .6 **Port 1, P1.3 Configuration Bits**

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up resistor
1	0	Push-pull output mode
1	1	Comparator input, analog input

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

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up resistor
1	0	Push-pull output mode
1	1	Comparator input, analog input

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

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-up resistor
1	0	Push-pull output mode
1	1	Comparator input, analog input

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

0	0	Schmitt trigger input
0	1	Schmitt trigger input with pull-upresistor
1	0	Push-pull output mode
1	1	Comparator input, analog input

P1INT — Port 1 Interrupt Control Register ECH

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

.7 – .4

Not used for KS86C6104/P6104

.3 P1.7 Interrupt Enable Bit

0	External interrupt disable
1	External interrupt enable

.2 P1.6 Interrupt Enable Bit

0	External interrupt disable
1	External interrupt enable

.1 P1.5 Interrupt Enable Bit

0	External interrupt disable
1	External interrupt enable

.0 P1.4 Interrupt Enable Bit

0	External interrupt disable
1	External interrupt enable

P1PND — Port 1 Interrupt Pending Register EDH

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

.7 – .4

Not used for KS86C6104/P6104

.3 P1.7 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.2 P1.6 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.1 P1.5 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

.0 P1.4 Interrupt Pending Bit

0	No pending (when read)/clear pending bit (when write)
1	Pending (when read)/no effect (when write)

PWRMGR — USB Power Management Register F8H

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

.7 – .2

Always logic zero

.1 **RESUME Signal Sending Bit**

0	RESUME signal is ended
1	While in suspend state, if the MCU wants to initiate a resume, it writes a 1 to this register for 10ms (maximum of 15ms), and clears this register. In suspend mode, if this bit is set to “1”, USB generates resume signaling.

.0 **SUSPEND Status Bit**

0	Cleared automatically when MCU writes a zero to RESUME signal sending bit or when function receives resume signal from the host while in suspend mode
1	This bit is set when SUSPEND interrupt occur

SYM — System Mode Register DFH

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

.7 – .3

Not used for KS86C6104/P6104

.2 Global Interrupt Enable Bit

0	Disable global interrupt processing
1	Enable global interrupt processing

.1 and .0 Page Selection Bits

0	0	Page 0
0	1	Page 1 (not used in KS86C6104)
1	0	Page 2 (not used in KS86C6104)
1	1	Page 3 (not used in KS86C6104)

T0CON — Timer 0 Control Register D2H

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

.7 and .6 T0 Counter Input Clock Selection Bits

0	0	$f_{OSC}/4096$
0	1	$f_{OSC}/256$
1	0	$f_{OSC}/8$
1	1	Not used for KS86C6104/P6104

.5 and .4 T0 Operating Mode Selection Bits

0	0	Interval timer mode (The counter is automatically cleared whenever T0DATA value equals to T0CNT value)
0	1	Invalid selection
1	0	
1	1	Overflow mode (OVF interrupt can occur)

.3 T0 Counter Clear Bit (T0CLR)

0	No effect
1	Clear T0 counter (when write)

.2 T0 Overflow Interrupt Enable Bit (T0OVF)

0	Disable T0 overflow interrupt
1	Enable T0 overflow interrupt

.1 T0 Match Interrupt Enable Bit (T0INT)

0	Disable T0 match interrupt
1	Enable T0 match interrupt

.0 T0 Interrupt Pending Bit (T0PND)

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

NOTE: When you write a "1" to T0CON.3, the timer 0 counter is cleared. The bit is then cleared automatically to "0".

USBINT — USB Interrupt Enable Register F7H

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

.7 – .3

Not used for KS86C6104/P6104

.2 SUSPEND/RESUME Interrupt Enable Bit

0	Disable SUSPEND and RESEME interrupt (default)
1	Enable SUSPEND and RESEME interrupt

.1 ENDPOINT1 Interrupt Pending Bit

0	Disable ENDPOINT 1 interrupt
1	Enable ENDPOINT 1 interrupt (default)

.0 ENDPOINT0 Interrupt Pending Bit

0	Disable ENDPOINT 0 interrupt
1	Enable ENDPOINT 0 interrupt (default)

USBPND — USB Interrupt Pending Register F6H

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

.7 – .4

Not used for KS86C6104/P6104

.3 RESUME Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, if RESUME signaling is received while in SUSPEND mode

.2 SUSPEND Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, when suspend signaling is received

.1 ENDPOINT1 Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, when endpoint1 needs to be serviced

.0 ENDPOINT0 Interrupt Pending Bit

0	No effect (once read, this bit is cleared automatically)
1	This bit is set, while endpoint 0 needs to serviced. It is set under the following conditions: — OUT_PKT_RDY is set — IN_PKT_RDY get cleared — SENT_STALL gets set — DATA_END gets cleared — SETUP_END gets set

USBRST — USB RESET Register FFH

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

.7 – .1	Not used for KS86C6104/P6104

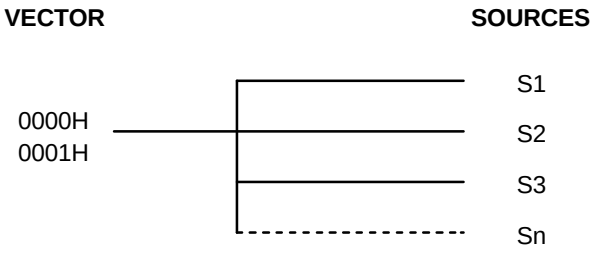
.0 USB Reset Signal Receive Bit

0	Clear reset signal bit
1	This bit is set when host send USB reset signal

5 INTERRUPT STRUCTURE

OVERVIEW

The SAM87RI interrupt structure has two basic components: a vector, and sources. The number of interrupt sources can be serviced through a interrupt vector which is assigned in ROM address 0000H–0001H.



NOTES:
1. The SAM87RI interrupt has only one vector address (0000H-0001H)
2. The number of Sn value is expandable.

Figure 5-1. KS86-Series Interrupt Type

INTERRUPT PROCESSING CONTROL POINTS

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

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

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

ENABLE/DISABLE INTERRUPT INSTRUCTIONS (EI, DI)

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

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

INTERRUPT PENDING FUNCTION TYPES

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

INTERRUPT PRIORITY

Because there is not a interrupt priority register in SAM87RI, the order of service is determined by a sequence of source which is executed in interrupt service routine.

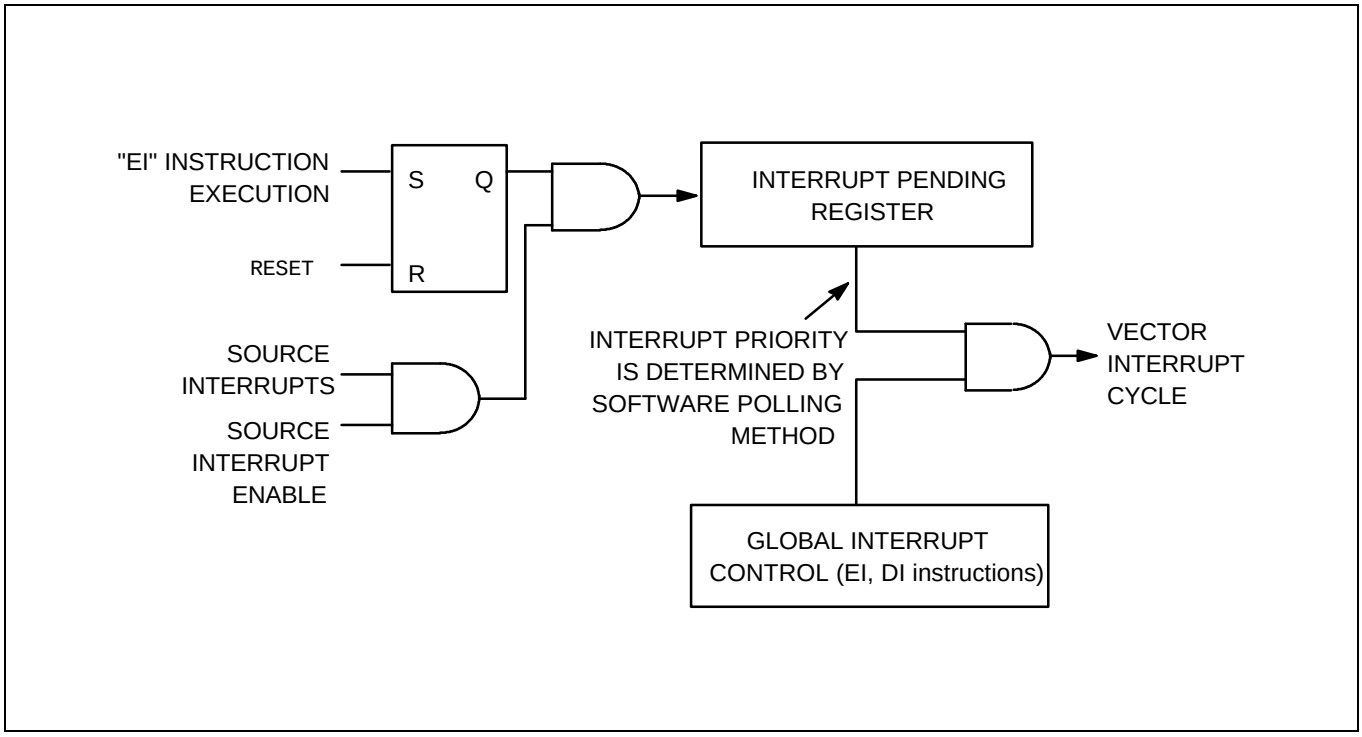


Figure 5-2. Interrupt Function Diagram

INTERRUPT SOURCE SERVICE SEQUENCE

The interrupt request polling and servicing sequence is as follows:

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

INTERRUPT SERVICE ROUTINES

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

— Interrupt processing must be enabled (EI, SYM.2 = "1")
— Interrupt must be enabled at the interrupt's source (peripheral control register)

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

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

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

GENERATING INTERRUPT VECTOR ADDRESSES

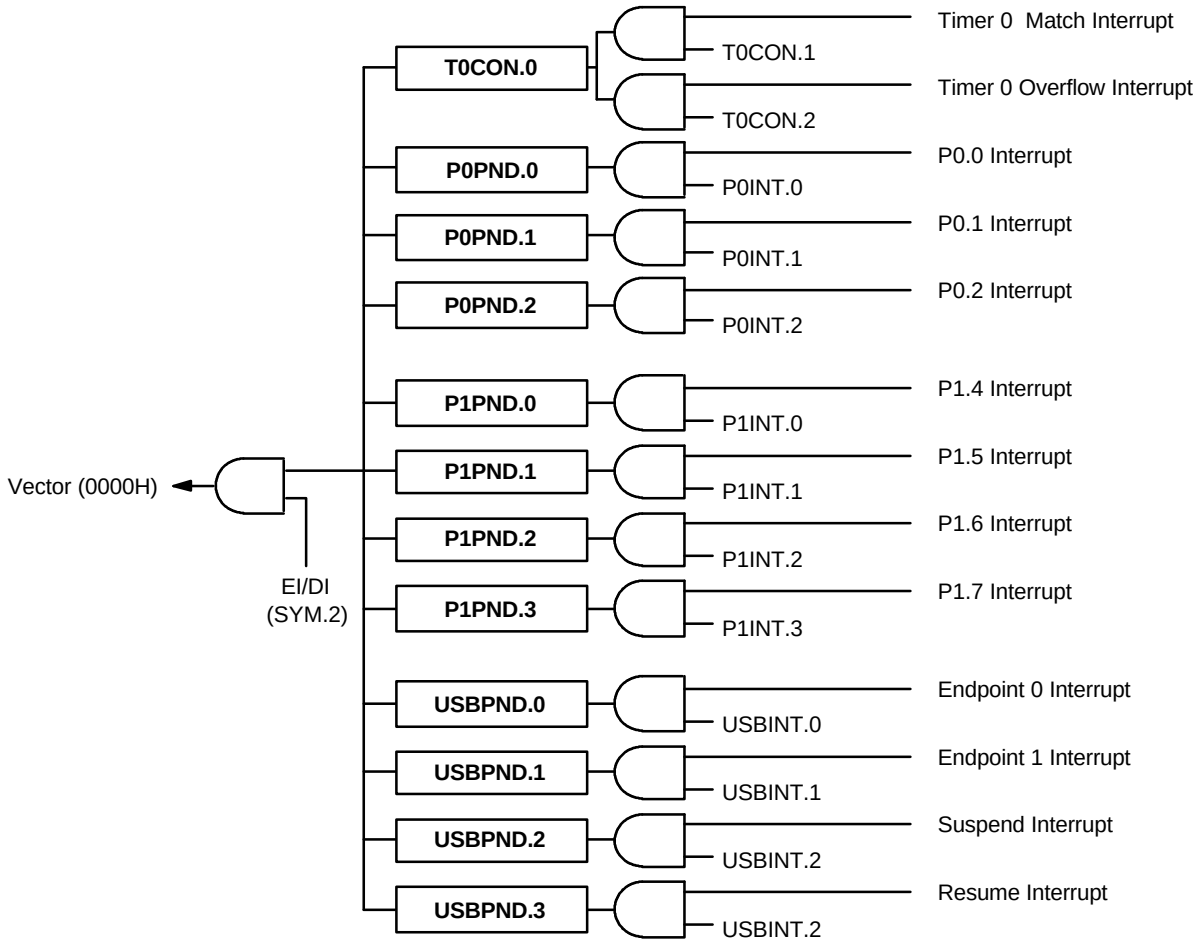
The interrupt vector area in the ROM contains the address of the interrupt service routine. Vectored interrupt processing follows this sequence:

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

KS86C6104/P6104 INTERRUPT STRUCTURE

The KS86C6104/P6104 microcontroller has thirteen peripheral interrupt sources:

— Timer 0 match interrupt
— Timer 0 overflow interrupt
— Suspend interrupt
— Resume interrupt
— Two Endpoint interrupts for Endpoint 0 and Endpoint 1
— Three external interrupts for port 0, P0.0–P0.2
— Four external interrupts for port 1, P1.4–P1.7



NOTE: Interrupt service priority should be managed by S/W in the interrupt service routine.

Figure 5-3. KS86C6104/P6104 Interrupt Structure

7 CLOCK CIRCUIT

OVERVIEW

The KS86C6104/P6104 has two oscillation circuit options, a crystal/ceramic oscillation and an external clock source. The crystal or ceramic oscillation source provides a maximum 6 MHz clock. The X_{IN} and X_{OUT} pins connect the oscillation source to the on-chip clock circuit. External clock and crystal/ceramic oscillator circuits are shown in Figures 7-1 and 7-2.

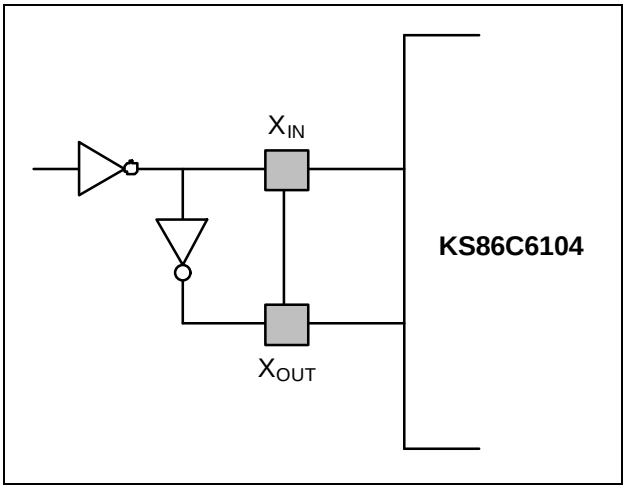


Figure 7-1. External Oscillator

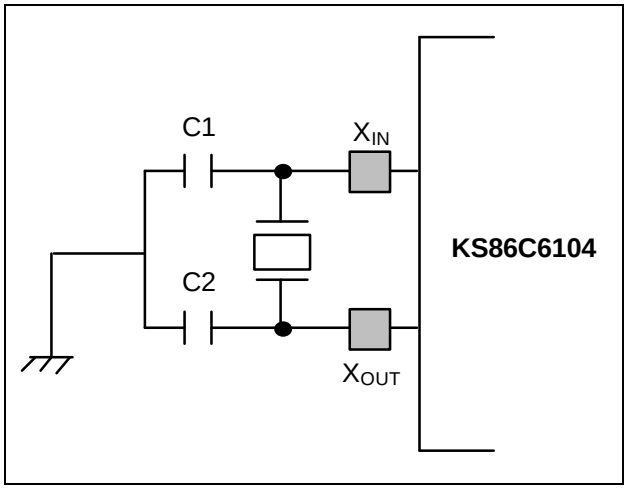


Figure 7-2. Main Oscillator Circuit (Crystal/Ceramic Oscillator)

MAIN OSCILLATOR LOGIC

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

CLOCK STATUS DURING POWER-DOWN MODES

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

— In Stop mode, the main oscillator "freezes," halting the CPU and peripherals. The contents of the register file and current system register values are retained. RESET operation releases the Stop mode, and starts the oscillator.

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

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

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

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

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

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

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

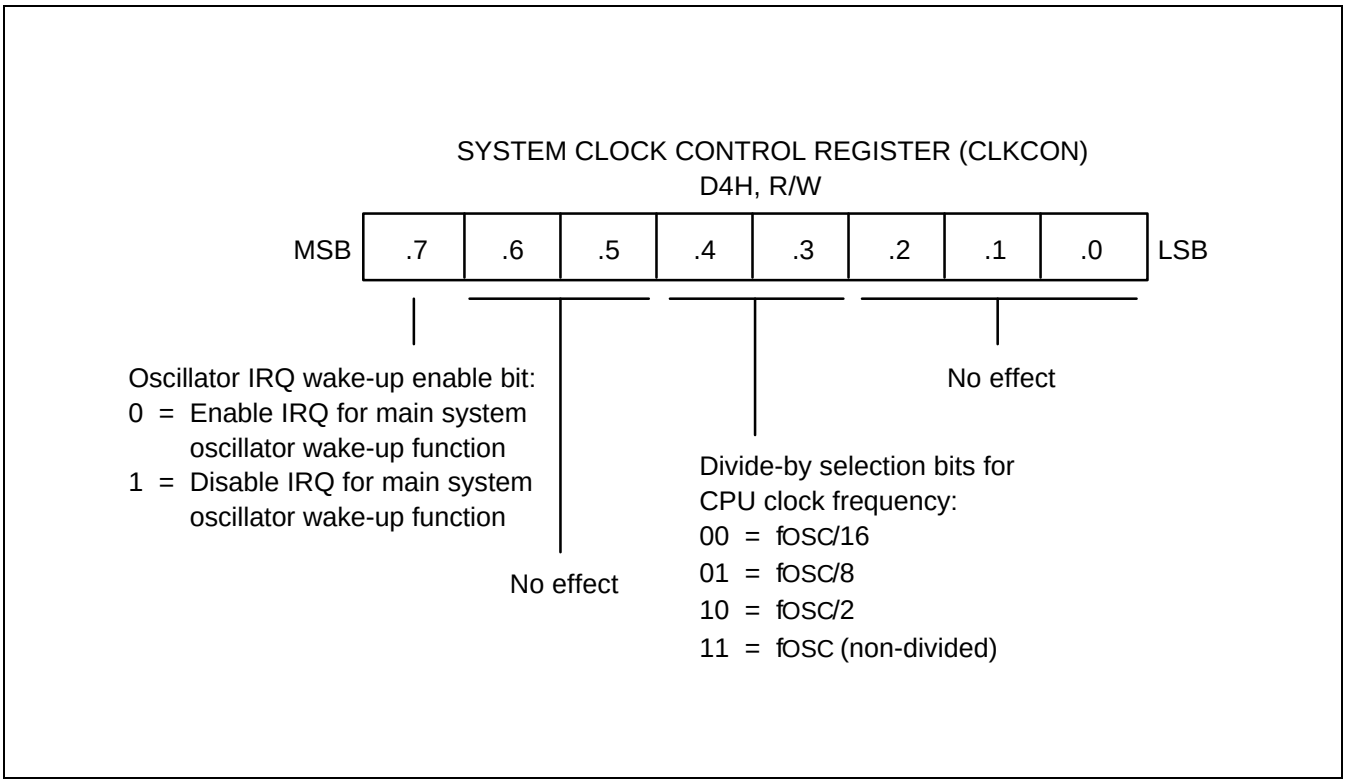


Figure 7-3. System Clock Control Register (CLKCON)

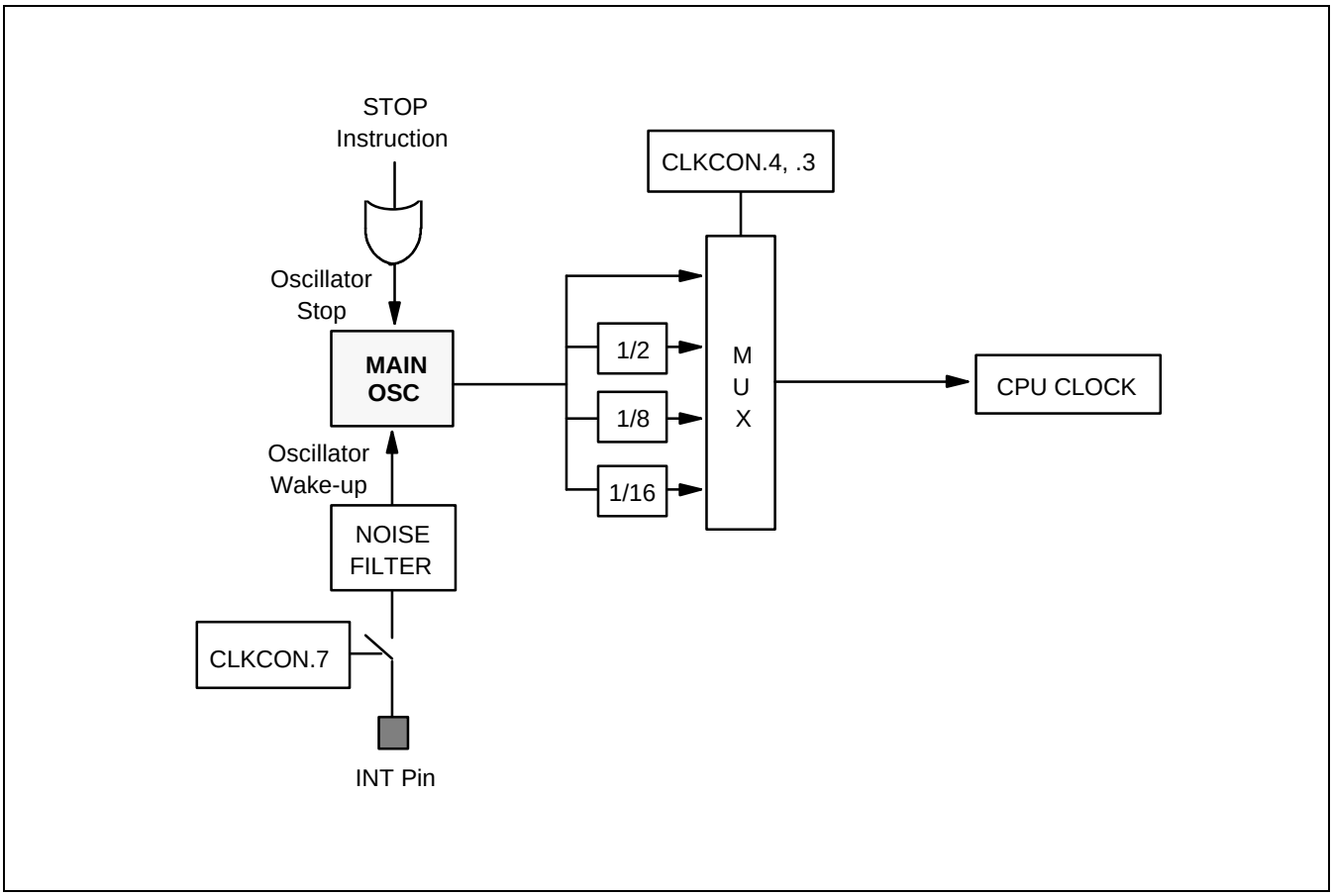


Figure 7-4. System Clock Circuit Diagram

NOTES

8 RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

During a power-on reset, the voltage at V_{DD} is High level and the RESET pin is forced to Low level. The RESET signal is input through a filter circuit where it is then synchronized with the CPU clock. This brings the KS86C6104/P6104 into a known operating status.

The RESET pin must be held to Low level for a minimum time interval after the power supply comes within tolerance in order to allow time for internal CPU clock oscillation to stabilize. The minimum required oscillation stabilization time for a reset is approximately 10ms ($\cong 2^{16}/f_{OSC}$, f_{OSC} = 6 MHz).

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

The following sequence of events occur during a reset operation:

— All interrupts are disabled.
— The watchdog function (basic timer) is enabled.
— Ports 0 and 1 are set to Schmitt trigger input mode and all pull-up resistors are disabled.
— Peripheral control and data registers are disabled and reset to their initial values.
— The program counter is loaded with the ROM reset address, 0100H.
— When the programmed oscillation stabilization time interval has elapsed, the address stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

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

POWER-DOWN MODES

STOP MODE

Stop mode is invoked by the instruction STOP (opcode 7FH). In Stop mode, the operation of the CPU and all peripherals is halted. That is, the on-chip main oscillator stops and the supply current is reduced to less than 120 µA. All system functions are halted when the clock "freezes," but data stored in the internal register file is retained. Stop mode can be released in one of two ways: by a RESET signal or by an external interrupt.

Using RESET to Release Stop Mode

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

Using an External Interrupt to Release Stop Mode

Only external interrupts with an RC-delay noise filter circuit can be used to release Stop mode (Clock-related external interrupts cannot be used). External interrupts in the KS86C6104/P6104 interrupt structure does not meet this criteria.

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

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

NOTE

Do not use the STOP mode when external clock source is being used as the oscillation circuit option.

IDLE MODE

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

There are two ways to release Idle mode:

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

NOTE

Only external interrupts that are not clock-related can be used to release Stop mode. To release Idle mode, however, any type of interrupt (that is, internal or external) can be used.

HARDWARE RESET VALUES

Tables 8-1 through 8-3 list the values for CPU and system registers, peripheral control registers, and peripheral data registers following a reset operation in normal operating mode. The following notation is used in these tables to represent specific reset values:

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

— An 'x' means that the bit value is undefined following RESET.

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

Table 8-1. Register Values after RESET

Register Name	Mnemonic	Address	Bit Values After RESET							
			7	6	5	4	3	2	1	0
General purpose registers(page 0)	–	00H–7FH	x	x	x	x	x	x	x	x
Working registers	–	C0H–CFH	x	x	x	x	x	x	x	x
Timer 0 counter	T0CNT	D0H	0	0	0	0	0	0	0	0
Timer 0 data register	T0DATA	D1H	1	1	1	1	1	1	1	1
Timer 0 control register	T0CON	D2H	0	0	0	0	0	0	0	0
Location D3H is not mapped.										
Clock control register	CLKCON	D4H	0	–	–	0	0	–	–	–
System flags register	FLAGS	D5H	0	0	0	0	–	–	–	–
Locations D6H – D8H are not mapped.										
Stack pointer	SP	D9H	x	x	x	x	x	x	x	x
Locations DAH – DBH are not mapped.										
Basic timer control register	BTCON	DCH	0	0	0	0	0	0	0	0
Basic timer counter	BTCNT	DDH	0	0	0	0	0	0	0	0
Location DEH is not mapped.										
System mode register	SYM	DFH	–	–	–	–	–	0	0	0
Port 0 data register	P0	E0H	x	x	x	x	0	0	0	0
Port 1 data register	P1	E1H	0	0	0	0	0	0	0	0
Locations E2H – E4H are not mapped.										

NOTE: The timer 0 counter, T0CNT, the basic timer counter, BTCNT, and comparison result, CDATA, are read-only. All other registers are read/write addressable.

Table 8-1. Register Values after RESET (continued)

Bank 0 Register Name	Mnemonic	Address	Bit Values After RESET							
			7	6	5	4	3	2	1	0
Comparison result register	CDATA	E5H	x	x	x	x	0	0	0	0
Port 0 control register	P0CON	E6H	–	–	0	0	0	0	0	0
Comparator control mode register	CCON	E7H	0	0	0	0	0	0	0	0
Port 1 control register (high nibble)	P1CONH	E8H	0	0	0	0	0	0	0	0
Port 1 control register (low nibble)	P1CONL	E9H	0	0	0	0	0	0	0	0
Port 0 interrupt control register	P0INT	EAH	–	–	–	–	–	0	0	0
Port 0 interrupt pending register	P0PND	EBH	–	–	–	–	–	0	0	0
Port 1interrupt control register	P1INT	ECH	–	–	–	–	0	0	0	0
Port 1 interrupt pending register	P1PND	EDH	–	–	–	–	0	0	0	0
Locations EEH –EFH are not mapped.										
USB function address register	FADDR	F0H	0	0	0	0	0	0	0	0
Control Endpoint status register	EP0CSR	F1H	0	0	0	0	0	0	0	0
Interrupt Endpoint status register	EP1CSR	F2H	0	0	0	0	0	0	0	0
Control Endpoint byte count register	EP0BCNT	F3H	0	0	0	0	0	0	0	0
Control Endpoint FIFO register	EP0FIFO	F4H	x	x	x	x	x	x	x	x
Interrupt Endpoint FIFO register	EP1FIFO	F5H	x	x	x	x	x	x	x	x
USB interrupt pending register	USBPND	F6H	–	–	–	–	0	0	0	0
USB interrupt enable register	USBINT	F7H	–	–	–	–	–	0	1	1
USB power management register	PWRMGR	F8H	0	0	0	0	0	0	0	0
Locations F9H – FEH are not mapped.										
USB reset register	USBRST	FFH	–	–	–	–	–	–	–	0

9 I/O PORTS

OVERVIEW

The KS86C6104/P6104 has two I/O ports (Port 0 and Port 1), 11 pins total. You access these ports directly by writing or reading port data register addresses.

For mouse applications, ports 1.0–1.3 are usually configured as mouse sensing input. Port 0 is used for button data input.

Table 9-1. KS86C6104/P6104 Port Configuration Overview

Port	Function Description	Programmability
0	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are individually assignable to input pins by software. Port0 can also be configured as external interrupt inputs.	Bit
1	Bit-programmable I/O port for Schmitt trigger input or push-pull output. Pull-up resistors are individually assignable to input pins by software and, they are automatically disabled for output pins. Port1.0–Port1.3 can be configured as comparator input. Port1.4–Port1.7 can be individually configured as external interrupt inputs.	Bit

PORT DATA REGISTERS

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

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Hex	R/W
Port 0 data register	P0	E0H	R/W
Port 1 data register	P1	E1H	R/W

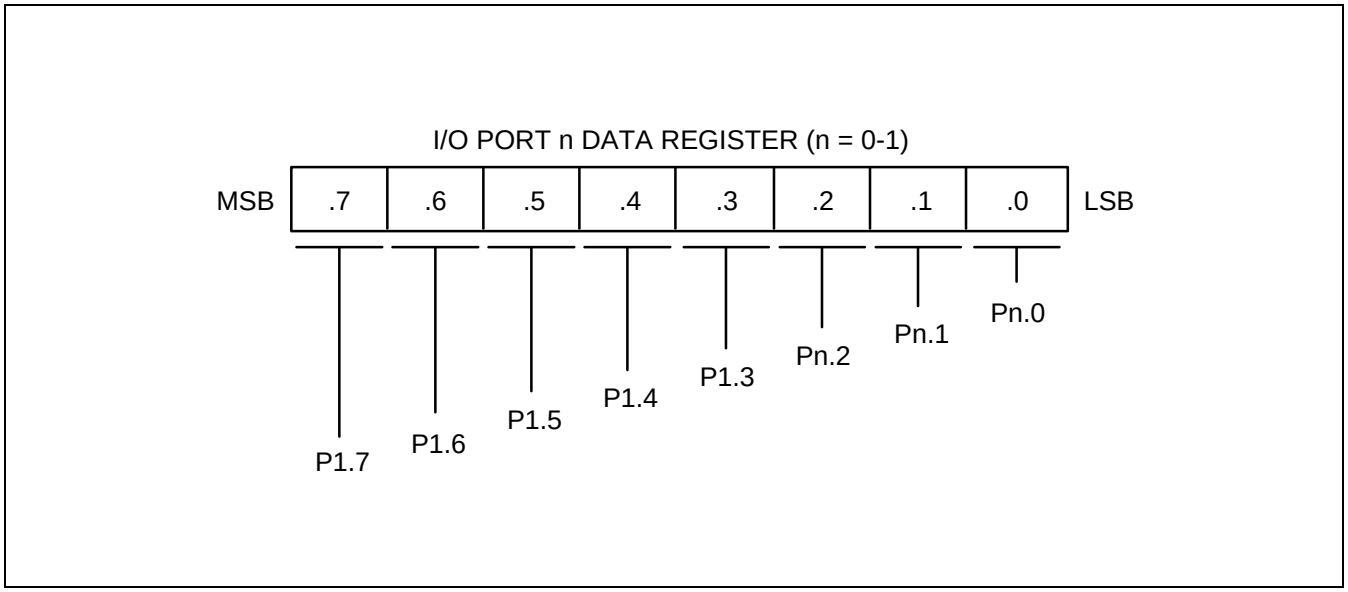
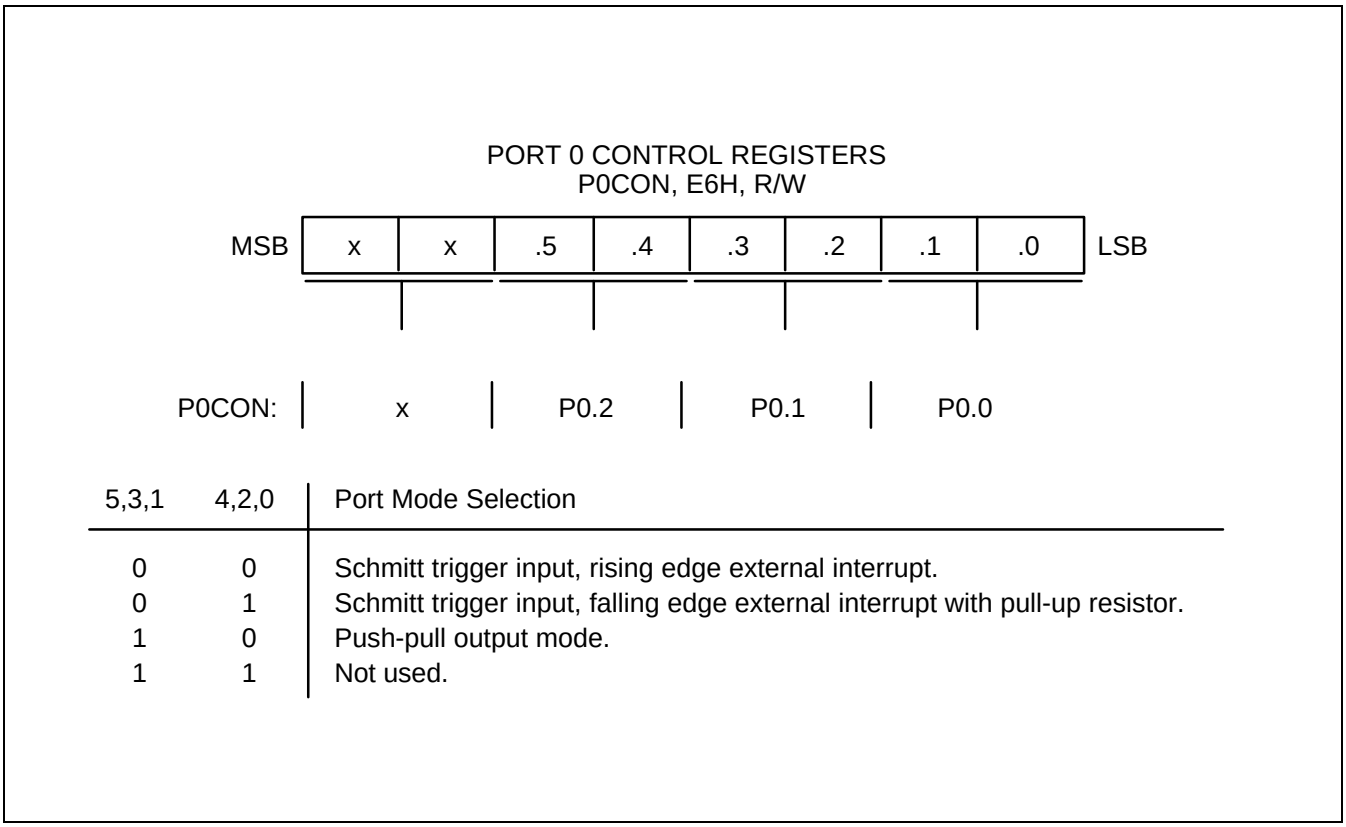


Figure 9-1. Port Data Register Format

PORT 0 AND PORT 1

Ports 0 and 1are bit-programmable, general-purpose, I/O ports. You can select Schmitt trigger input mode or push-pull output mode. Port1.0 to Port1.3 can be configured as comparator input.

You access ports 0 and 1 directly by writing or reading the corresponding port data registers — P0 (E0H) and P1 (E1H). RESET clears the port control registers P0CON, P1CONH, and P1CONL to '00H', configuring all port 0 and port 1 pins as Schmitt trigger inputs.



5,3,1	4,2,0	Port Mode Selection
0	0	Schmitt trigger input, rising edge external interrupt.
0	1	Schmitt trigger input, falling edge external interrupt with pull-up resistor.
1	0	Push-pull output mode.
1	1	Not used.

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

PORT 1 CONTROL REGISTERS
P1CONH, E8H, R/W, P1CONL, E9H, R/W

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

P1CONH:	P1.7	P1.6	P1.5	P1.4
P1CONL:	P1.3	P1.2	P1.1	P1.0

7,5,3,1	6,4,2,0	Port Mode Selection
0	0	Schmitt trigger input, rising edge external interrupt.
0	1	Schmitt trigger input, falling edge external interrupt with pull-up resistor.
1	0	Push-pull output mode.
1	1	Comparator input (not used in P1CONH).

Figure 9-3. Port 1 Control Registers (P1CONH, P1CONL)

☞ PROGRAMMING TIP — Configuring KS86C6104/P6104 Port Pins to Specification

This example shows how to configure ports 0–1 to specification. The programming parameters are as follows:

Examples:

1. Set port 0 push-pull output mode
```
LD      P0CON,#0AAH         ;  P0.0–P0.3 ← Push-pull output
```

2. Set port 1.4–port 1.7 schmitt trigger input mode
```
LD      P1CONH,#00H         ;  P1.4–P1.7 ← Schmitt trigger input
```

3. Set port 1.0–port 1.3 comparator input mode
```
LD      P1CONL,#0FFH        ;  P1.0–P1.3 ← Comparator input
```

10 BASIC TIMER and TIMER 0

MODULE OVERVIEW

The KS86C6104/P6104 has two default timers: an 8-bit *basic timer* and one 8-bit general-purpose timer/counter. The 8-bit timer/counter is called *timer 0.*

Basic Timer (BT)

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

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

The functional components of the basic timer block are:

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

Timer 0

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

— Interval timer mode
— Overflow mode

Timer 0 has the following functional components:

— Clock frequency divider (f_{OSC} divided by 4096, 256, or 8) with multiplexer
— 8-bit counter (T0CNT), 8-bit comparator, and 8-bit reference data register (T0DATA)
— Timer 0 overflow interrupt (T0OVF) and match interrupt (T0INT) generation
— Timer 0 control register, T0CON

BASIC TIMER CONTROL REGISTER (BTCON)

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

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

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

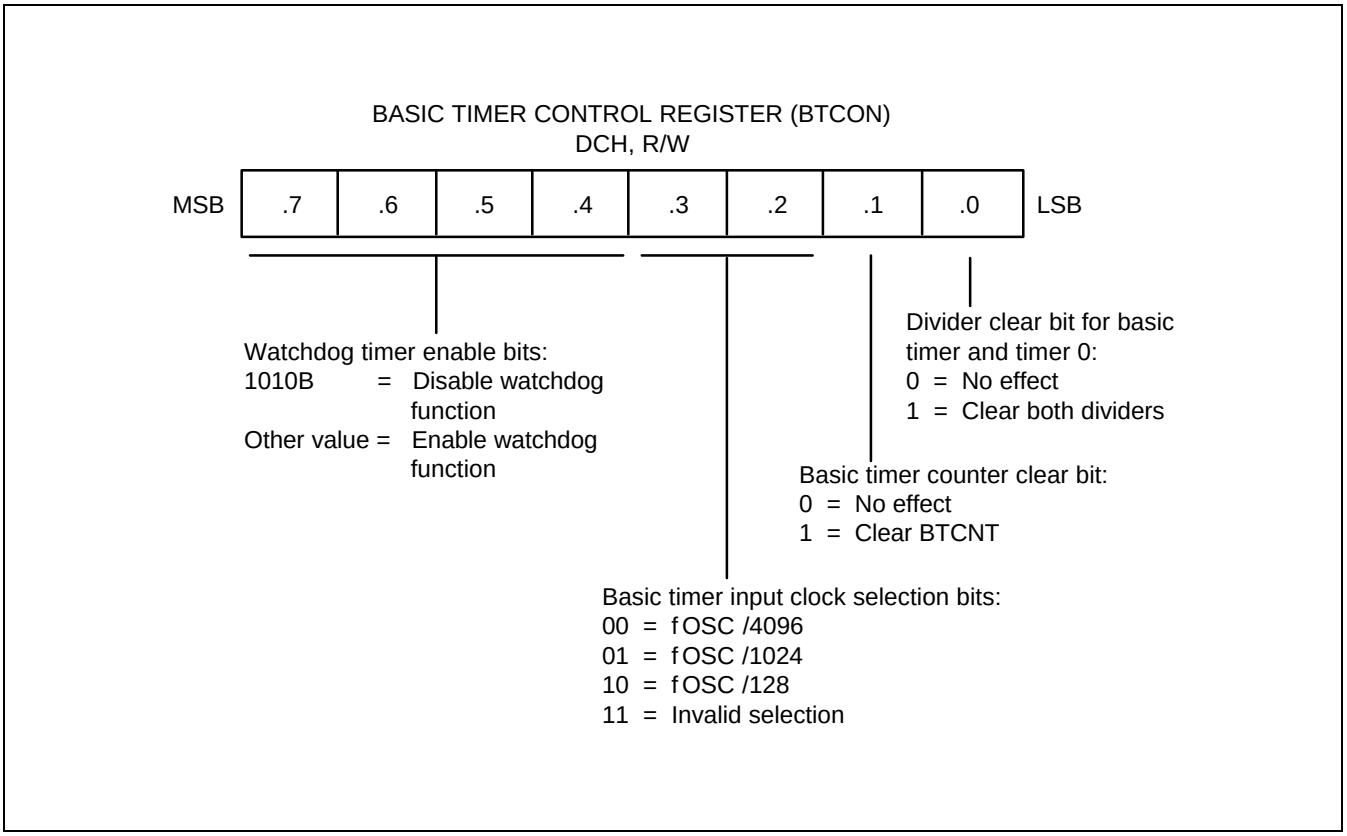


Figure 10-1. Basic Timer Control Register (BTCON)

BASIC TIMER FUNCTION DESCRIPTION

Watchdog Timer Function

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

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

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

Oscillation Stabilization Interval Timer Function

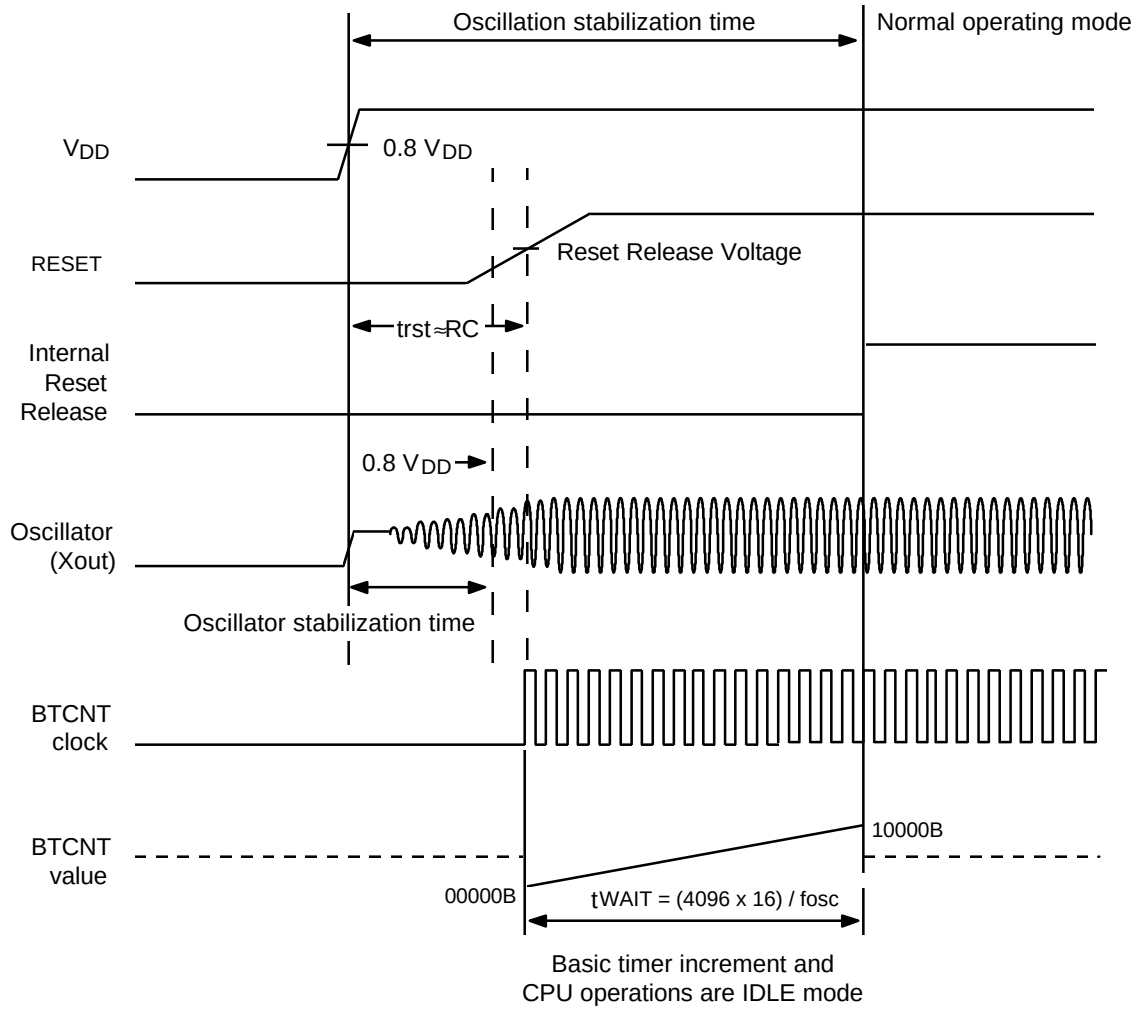
You can also use the basic timer to program a specific oscillation stabilization interval following a reset or when Stop mode has been released by an external interrupt.

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

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

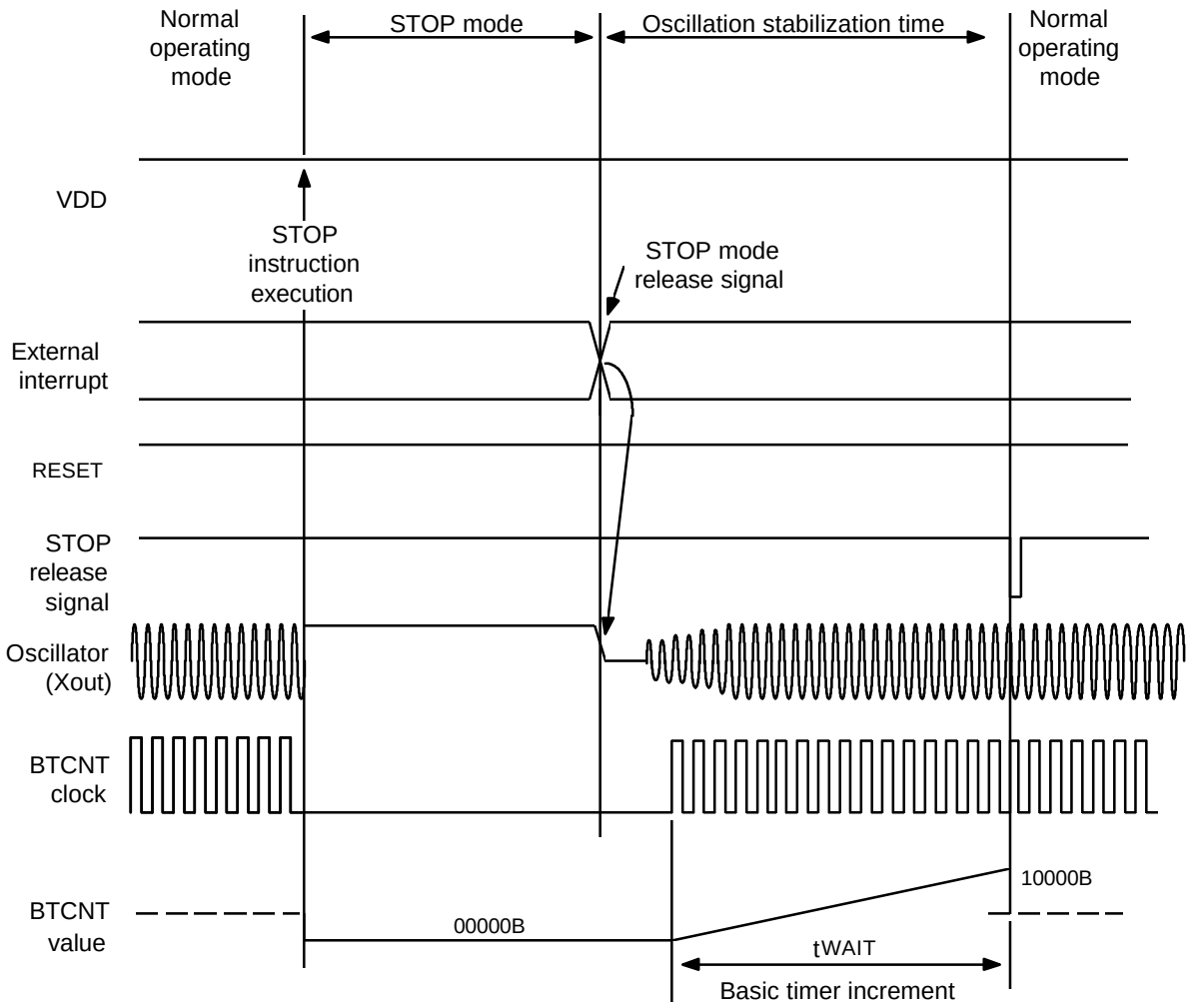
1. During Stop mode, a power-on reset or an external interrupt occurs to trigger the Stop mode release and oscillation starts.
2. If a power-on reset occurred, the basic timer counter will increase at the rate of f_{OSC} /4096. If an external interrupt is used to release Stop mode, the BTCNT value increases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

Figure 10-2 and 10-3 show the oscillation stabilization time on RESET and STOP mode release, respectively.



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

Figure 10-2. Oscillation Stabilization Time on RESET



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

BTCON.3	BTCON.2	tWAIT	tWAIT (When fosc is 6 MHz)
0	0	(4096 x 16) / fosc	10.92 ms
0	1	(1024 x 16) / fosc	2.7 ms
1	0	(128 x 16) / fosc	0.341 ms
1	1	(16 x 16) / fosc	42.6 us

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

TIMER 0 CONTROL REGISTER (T0CON)

T0CON is located at address D2H, and is read/write addressable.

A reset clears T0CON to '00H'. This sets timer 0 to normal interval match mode, selects an input clock frequency of $f_{OSC}/4096$, and disables the timer 0 overflow interrupt and match interrupt. You can clear the timer 0 counter at any time during normal operation by writing a "1" to T0CON.3.

The timer 0 overflow interrupt can be enabled by writing a "1" to T0CON.1. When a timer 0 overflow interrupt occurs and is serviced by the CPU, the pending condition must be cleared by software by writing a "0" to the timer 0 interrupt pending bit, T0CON.0.

To enable the timer 0 match interrupt, you must write T0CON.1 to "1". To detect an interrupt pending condition, the application program polls T0CON.0. When a "1" is detected, a timer 0 match/ capture interrupt is pending. When the interrupt request has been serviced, the pending condition must be cleared by software by writing a "0" to the timer 0 interrupt pending bit, T0CON.0.

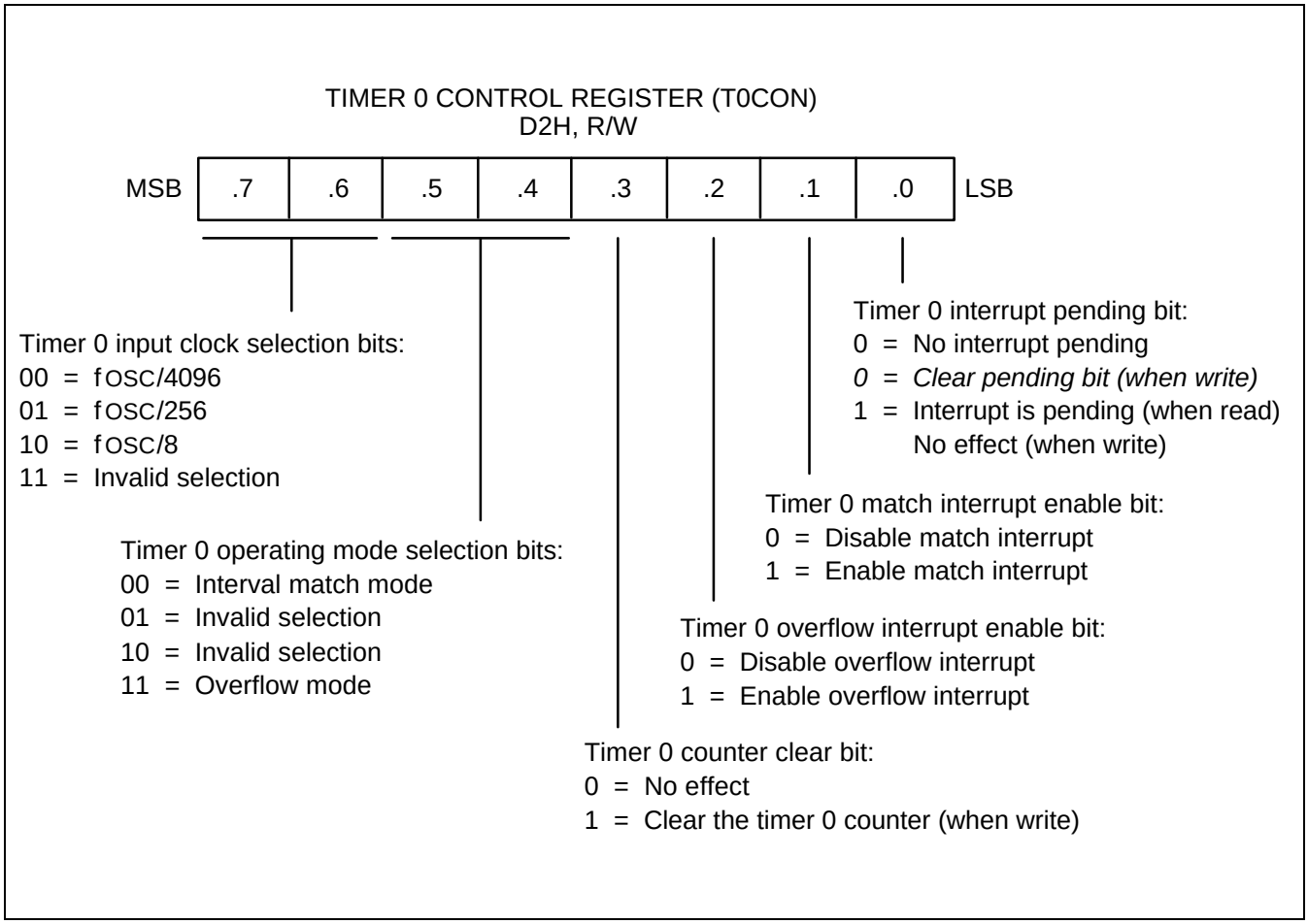


Figure 10-4. Timer 0 Control Register (T0CON)

TIMER 0 FUNCTION DESCRIPTION

Interval Match Mode

In interval match mode, a match signal is generated when the counter value is identical to the value written to the T0 reference data register, T0DATA. The match signal generates a timer 0 match interrupt and then clears the counter. If for example, you write the value '10H' to T0DATA, the counter will increment until it reaches '10H'. At this point, the T0 match interrupt is generated, the counter value is reset and counting resumes.

Overflow Mode

In overflow mode, a overflow signal is generated regardless of the value written to the T0 reference data register when the counter value is overflowed. The overflow signal generates a timer 0 overflow interrupt and then T0 counter is cleared.

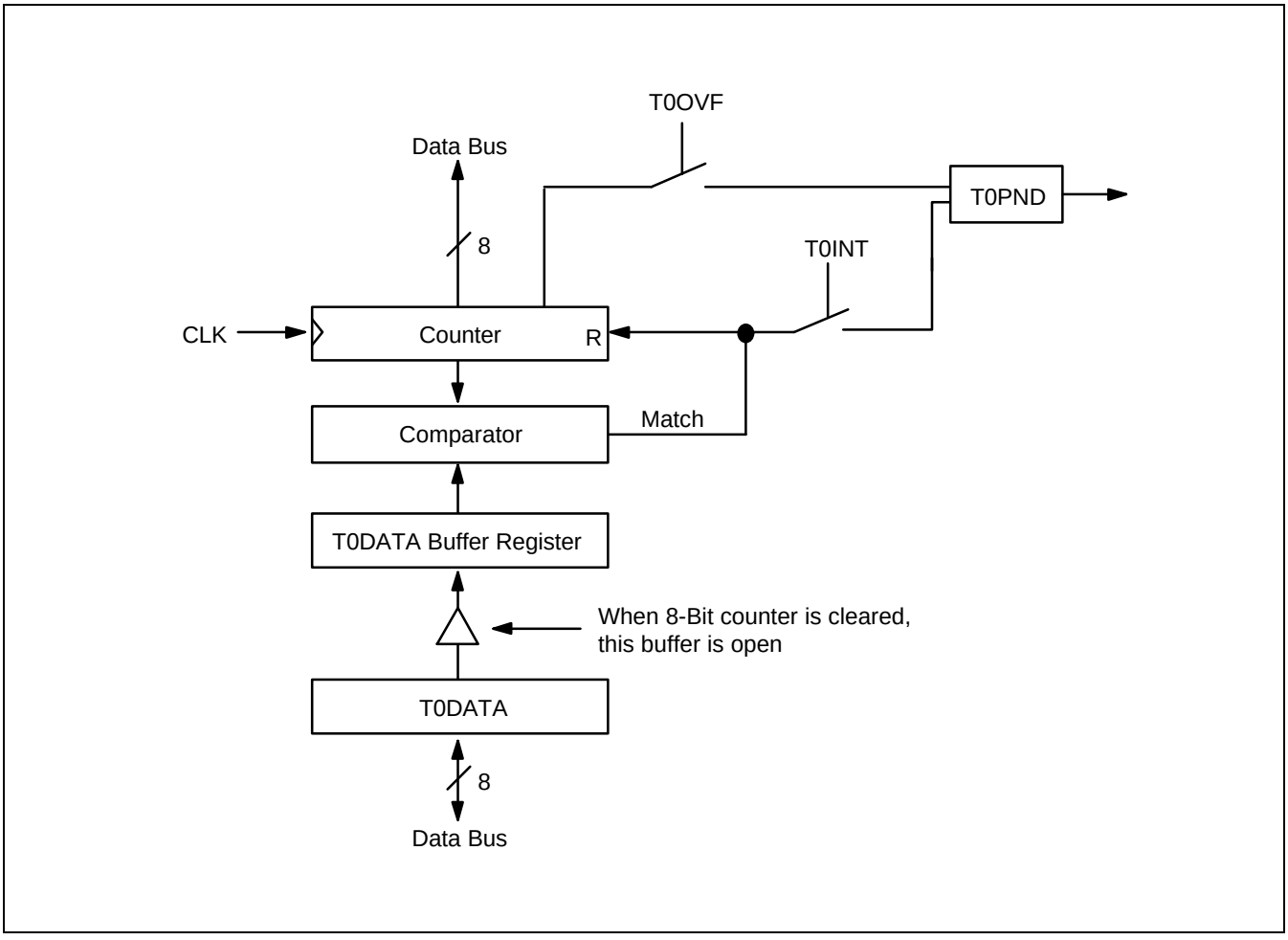


Figure 10-5. Simplified Timer 0 Function Diagram: Interval Timer Mode

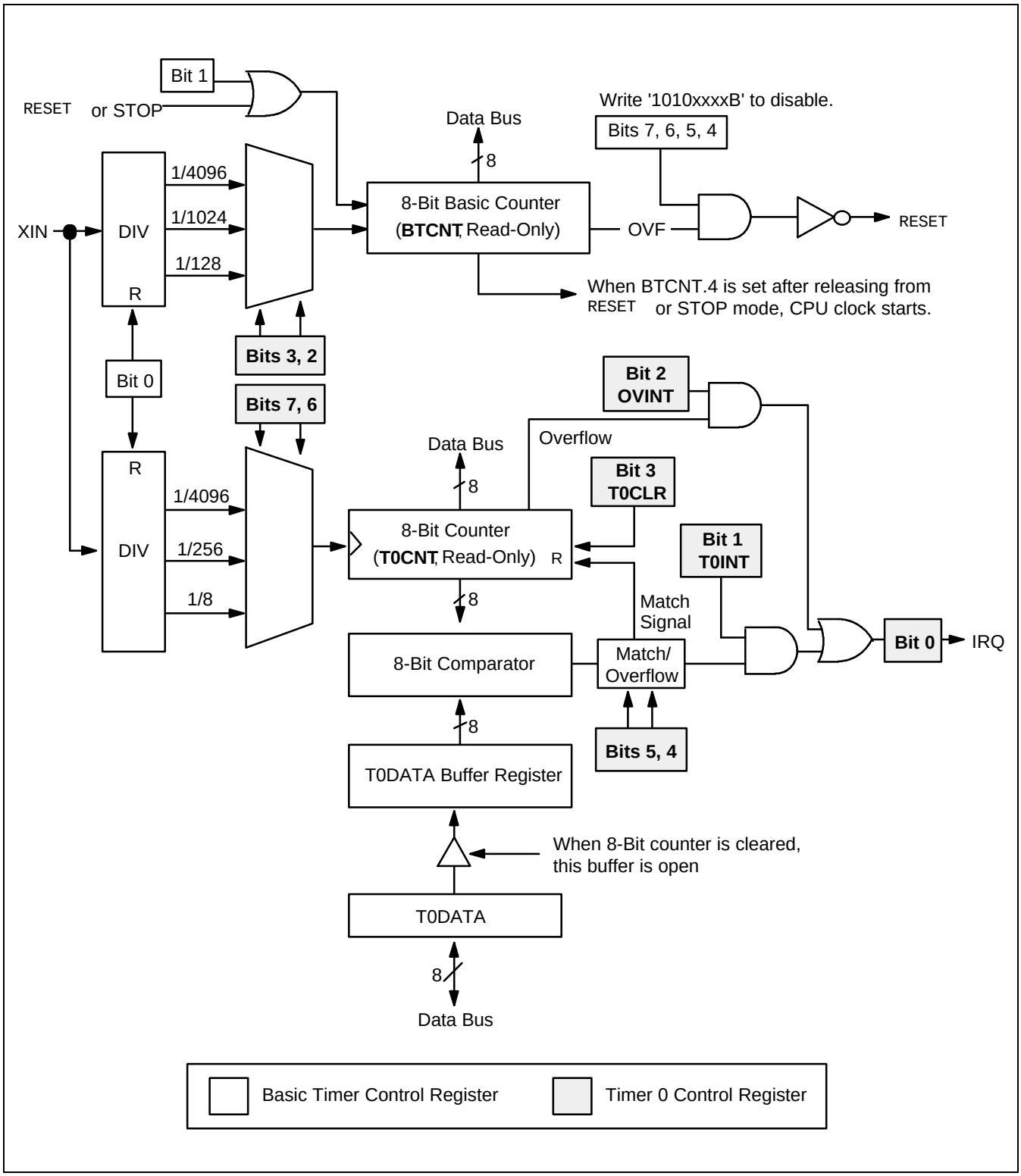


Figure 10-6. Basic Timer and Timer 0 Block Diagram

11 UNIVERSAL SERIAL BUS

OVERVIEW

Universal Serial Bus (USB) is a communication architecture that supports data transfer between a host computer and a wide range of PC peripherals. USB is actually a cable bus in which the peripherals share its bandwidth through a host scheduled token based protocol.

The USB module in KS86C6104/P6104 is designed to serve at a low speed transfer rate (1.5 Mbs) USB device as described in the Universal Serial Bus Specification Revision 1.0. KS86C6104/P6104 can be briefly describe as a microcontroller with SAM 87Ri core with an on-chip USB peripheral as can be seen in figure 11-1.

The KS86C6104/P6104 comes equipped with Serial Interface Engine (SIE), which handles the communication protocol of the USB. The KS86C6104/P6104 supports the following control logic: packet decoding/generation, CRC generation/checking, NRZI encoding/decoding, Sync detection, EOP (end of packet) detection and bit stuffing.

KS86C6104/P6104 supports two types of data transfers; control and interrupt. Two endpoints are used in this device; Endpoint 0 and Endpoint 1. Please refer to the USB specification revision 1.0 for detail description of USB.

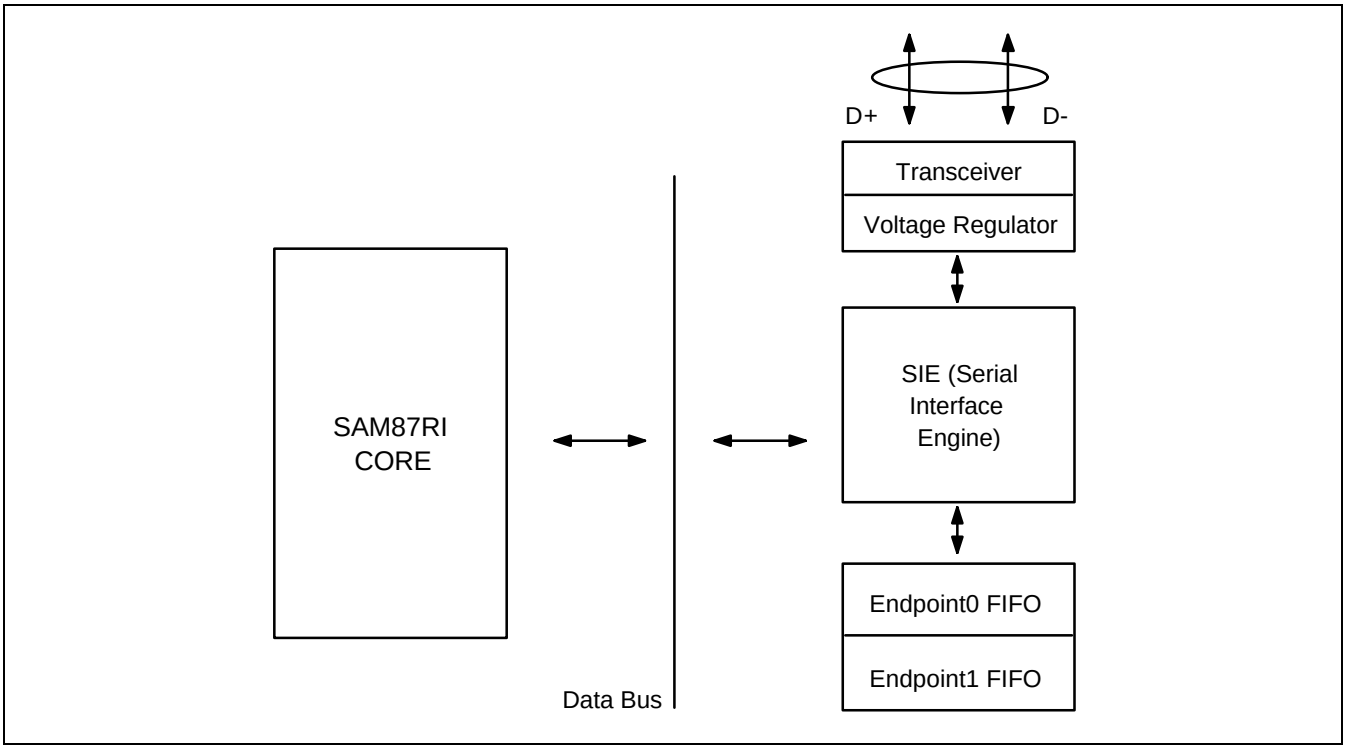


Figure 11-1. USB Peripheral Interface

Serial Bus Interface Engine (SIE)

The Serial Interface Engine interfaces to the USB serial data and handles, deserialization/serialization of data, NRZI encoding/decoding, clock extraction, CRC generation and checking, bit stuffing and other specifications pertaining to the USB protocol such as handling inter packet time out and PID decoding.

Control Logic

The USB control logic manages data movements between the CPU and the transceiver by manipulating the transceiver and the endpoint register. This includes both transmit and receive operations on the USB. The logic contains byte count buffers for transmit operations that load the active transmit endpoint's byte count and use this to determine the number of bytes to transfer. The same buffer is used for receive transactions to count the number of bytes received and transfer that number to the receive endpoint's byte count register at the end of the transaction.

The control logic in KS86C6104/P6104, when transmitting, manages parallel to serial conversion, packet generation, CRC generation, NRZI encoding and bit stuffing.

When receiving, the control logic in KS86C6104/P6104 handles Sync detection, packet decoding, EOP (end of packet) detection, bit (un)stuffing, NRZI decoding, CRC checking and serial to parallel conversion

Bus Protocol

All bus transactions involve the transmission of packets. KS86C6104/P6104 supports three packet types; Token, Data and Handshake. Each transaction starts when the host controller sends a Token Packet to the USB device. The Token packets are generated by the USB host and decoded by the USB device. A Token Packet includes the type description, direction of the transaction, USB device address and the endpoint number.

Data and Handshake packets are both decoded and generated by the USB device. In any transaction, the data is transferred from the host to a device or from a device to the host. The transaction source then sends a Data Packet or indicates that it has no data to transfer. The destination then responds with a Handshake Packet indicating whether the transfer was successful.

Data Transfer Types

USB data transfer occurs between the host software and a specific endpoint on the USB device. An endpoint supports a specific type of data transfer. The KS86C6104/P6104 supports two data transfer endpoints: control and interrupt.

Control transfer configures and assigns an address to the device when detected. Control transfer also supports status transaction, returning status information from device to host.

Interrupt transfer refers to a small, spontaneous data transfer from USB device to host.

Endpoints

Communication flows between the host software and the endpoints on the USB device. Each endpoint on a device has an identifier number. In addition to the endpoint number, each endpoint supports a specific transfer type. KS86C6104/P6104 supports two endpoints: Endpoint 0 supports control transfer, and Endpoint 1 supports interrupt transfer.

USB FUNCTION ADDRESS REGISTER (USBADDR)

This register holds the USB address assigned by the host computer. USBADDR is located at address F0H and is read/write addressable.

Bit7 Not used

Bit6–0 **FADDR:** MCU updates this register once it decodes a SET_ADDRESS command. MCU must write this register before it clears OUT_PKT_RDY (bit0) and sets DATA_END (bit3) in the EP0STU register. The function controller use this register's value to decode USB Token packet address. At reset, if the device is not yet configured the value is reset to 0.

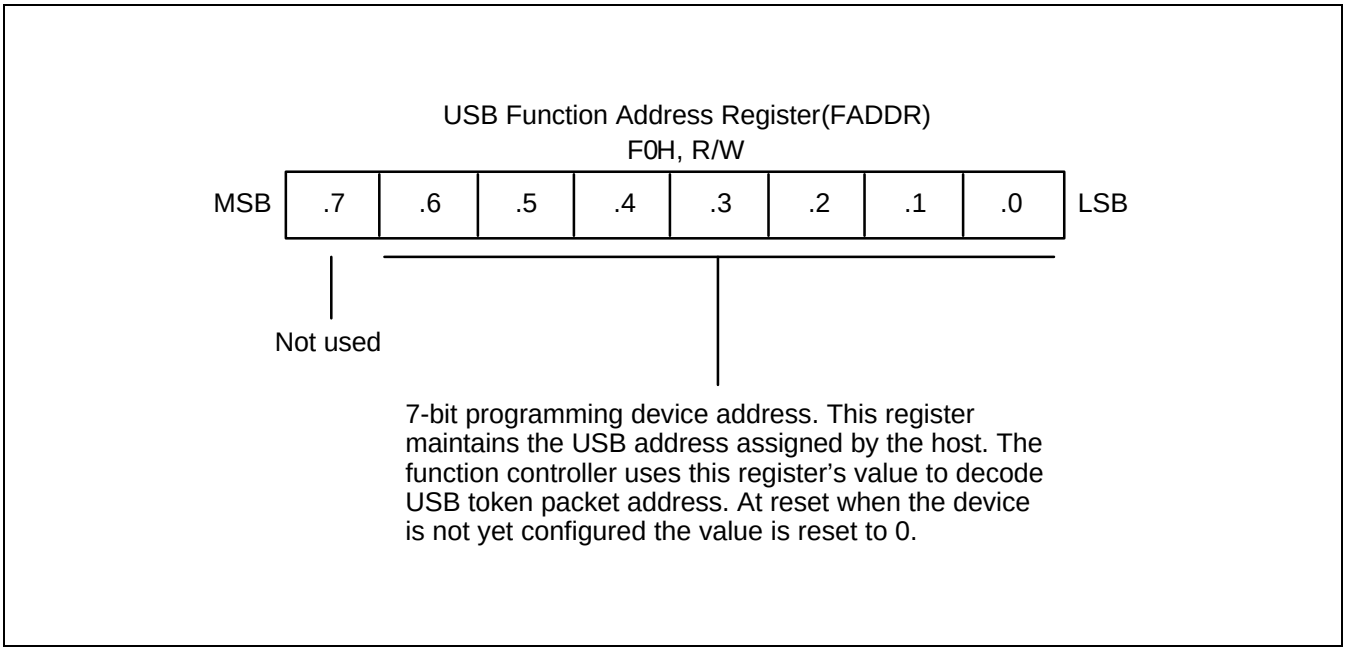


Figure 11-2. USB Function Address Register (FADDR)

CONTROL ENDPOINT STATUS REGISTER (EP0CSR)

EP0CSR register controls Endpoint 0 (Control Endpoint), and also holds status bits for Endpoint 0. EP0CSR is located at F1H and is read/write addressable.

Bit7 **CLEAR_SETUP_END:** MCU writes “1” to this bit to clear SETUP_END bit (bit4). This bit is automatically cleared after writing "1" by USB block.

Bit6 **CLEAR_OUT_PKT_RDY:** MCU writes “1” to this bit to clear OUT_PKT_RDY bit (bit0). This bit is automatically cleared after writing "1" by USB block.

Bit5 **SEND_STALL:** MCU writes “1” to this bit to send STALL signal to the Host, at the same time it clears OUT_PKT_RDY (bit0), if it decodes an invalid token. USB issues a STALL Handshake to the current control transfer. This bit gets cleared once a STALL Handshake is issued to the current control transfer.

Bit4 **SETUP_END:** USB sets this bit, when a control transfer ends before DATA_END bit (bit3) is set. MCU clears this bit, by writing a “1” to SERVICED_SETUP_END bit (bit7). When USB sets this bit, an interrupt is generated to MCU. When such condition occurs, USB flushes the FIFO, and invalidates MCU’s access to FIFO.

Bit3 **DATA_END:** MCU sets this bit:

— After loading the last packet of data into the FIFO, and at the same time IN_PKT_RDY bit is set.
— While it clears OUT_PKT_RDY bit after unloading the last packet of data.
— For a zero length data phase, when it clears OUT_PKT_RDY bit, and sets IN_PKT_RDY bit.

Bit2 **SENT_STALL:** USB sets this bit, if a control transaction has ended due to a protocol violation. An interrupt is generated when this bit gets set. MCU clears this bit to end the STALL condition.

Bit1 **IN_PKT_RDY:** MCU sets this bit, after writing a packet of data into Endpoint 0 FIFO. USB clears this bit, once the packet has been successfully sent to the host. An interrupt is generated when USB clears this bit so that MCU can load the next packet. For a zero length data phase, MCU sets IN_PKT_RDY bit and DATA_END bit at the same time.

Bit0 **OUT_PKT_RDY:** USB sets this bit, once a valid token is written to FIFO. An interrupt is generated, when USB sets this bit. MCU clears this bit by writing "1” to SERVICED_OUT_PKT_RDY bit.

NOTES:
1. In control transfer case, where there is no data phase, MCU after unloading the setup token, sets IN_PKT_RDY, and DATA_END at the same time it clears OUT_PKT_RDY for the setup token.
2. When SETUP_END bit is set, OUT_PKT_RDY bit may also be set. This happens when the current transfer has ended, and a new control transfer is received before MCU can service the interrupt. In such case, MCU should first clear SETUP_END bit, and then start servicing the new control transfer.

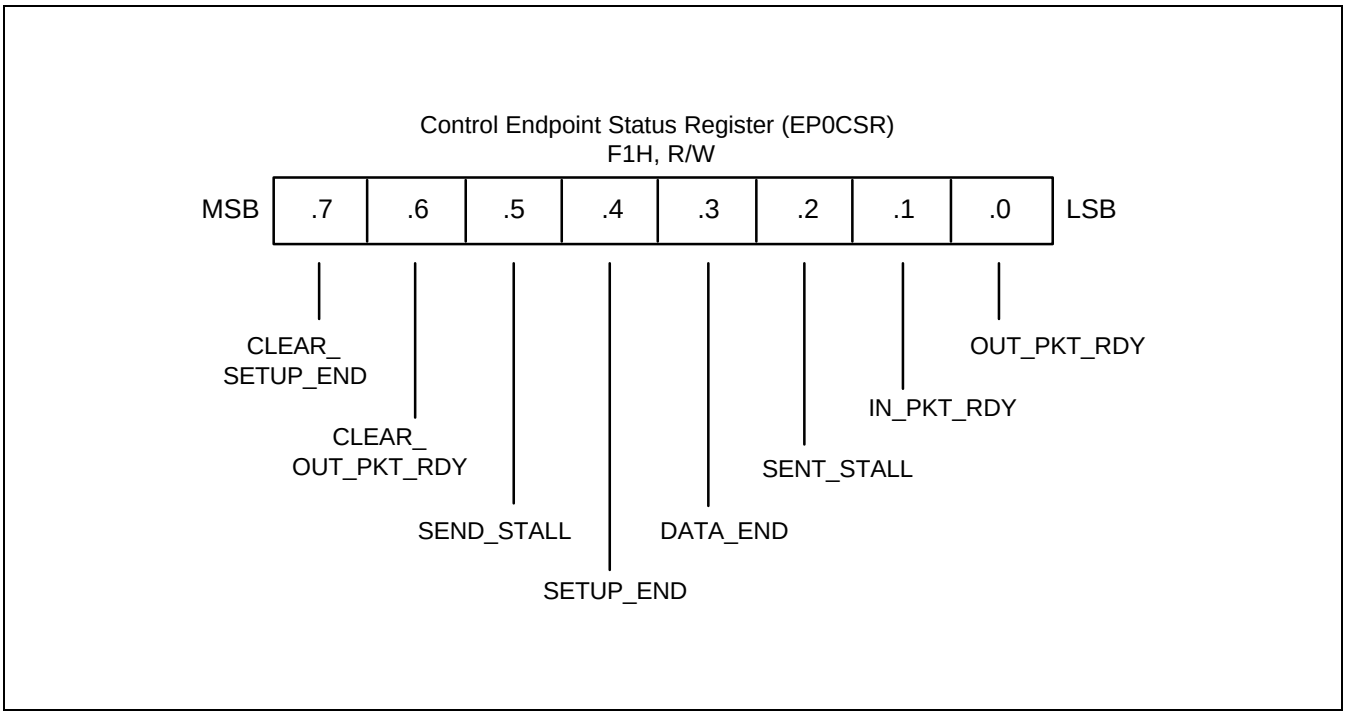


Figure 11-3. Control Endpoint Status Register (EP0CSR)

INTERRUPT ENDPOINT STATUS REGISTER (EP1CSR)

EP1CSR is the control register for Endpoint 1, Interrupt Endpoint. This register is located at address F2H and is read/write addressable.

Bit7 **CLR_DATA_TOGGLE:** MCU writes “1” to this bit to clear the data toggle sequence bit. When the MCU writes a 1 to this register, the data toggle bit is initialized to DATA0.

Bit6–3 **MAXP:** These bits indicate the maximum packet size for IN endpoint, and needs to be updated by MCU before it sets IN_PKT_RDY. Once set, the contents are valid till MCU re-writes them.

Bit2 **FLUSH_FIFO:** When MCU writes “1” to this register, the FIFO is flushed, and IN_PKT_RDY cleared. The MCU should wait for IN_PKT_RDY to be cleared for the flush to take place.

Bit1 **FORCE_STALL:** MCU writes “1” to this register to issue a STALL Handshake to USB. MCU clears this bit, to end the STALL condition.

Bit0 **IN_PKT_RDY:** MCU sets this bit, after writing a packet of data into Endpoint 1 FIFO. USB clears this bit, once the packet has been successfully sent to the Host. An interrupt is generated when USB clears this bit, so MCU can load the next packet.

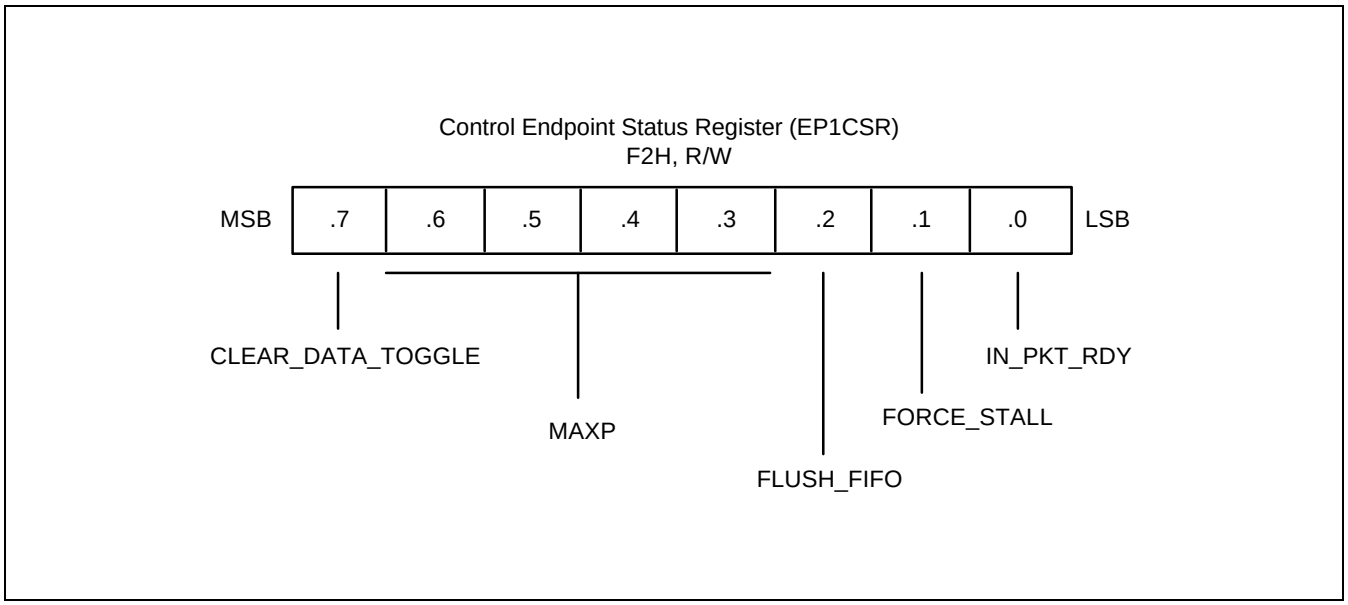


Figure 11-4. Interrupt Endpoint Status Register (EP1CSR)

CONTROL ENDPOINT BYTE COUNT REGISTER (EP0BCNT)

EP0BCNT register has the number of valid bytes in Endpoint 0 FIFO. It is located at address F3H read only addressable. Once the MCU receives a OUT_PKT_RDY (Bit0 of EP0CSR) for Endpoint 0, then it can read this register to find out the number of bytes to be read from Endpoint 0 FIFO.

CONTROL ENDPOINT FIFO REGISTER (EP0FIFO)

This register is bi-directional, 8 byte depth FIFO used to transfer Control Endpoint data. EP0FIFO is located at address F4H and is read/write addressable.

Initially, the direction of the FIFO, is from the Host to the MCU. After a setup token is received for a control transfer, that is, after MCU unload the setup token bytes, and clears OUT_PKT_RDY, the direction of FIFO is changed automatically from MCU to the Host.

INTERRUPT ENDPOINT FIFO REGISTER (EP1FIFO)

EP1FIFO is an uni-direction 8-byte depth FIFO used to transfer data from the MCU to the Host. MCU writes data to this register, and when finished set IN_PKT_RDY. This register is located at address F5H.

USB INTERRUPT PENDING REGISTER (USBPND)

USBPND register has the interrupt bits for endpoints and power management. *This register is cleared once read by MCU.* While any one of the bits is set, an interrupt is generated. USBPND is located at address F6H.

Bit7–4 Not used

Bit3 **RESUME_PND:** While in suspend mode, if resume signaling is received this bit gets set.

Bit2 **SUSPEND_PND:** This bit is set, when suspend signaling is received.

Bit1 **ENDPT1_PND:** This bit is set, when Endpoint 1 needs to be serviced.

Bit0 **ENDPT0_PND:** This bit is set, when Endpoint 0 needs to be serviced. It is set under any one of the following conditions:

- OUT_PKT_RDY is set.
- IN_PKT_RDY gets cleared.
- SENT_STALL gets set.
- DATA_END gets cleared.
- SETUP_END gets set.

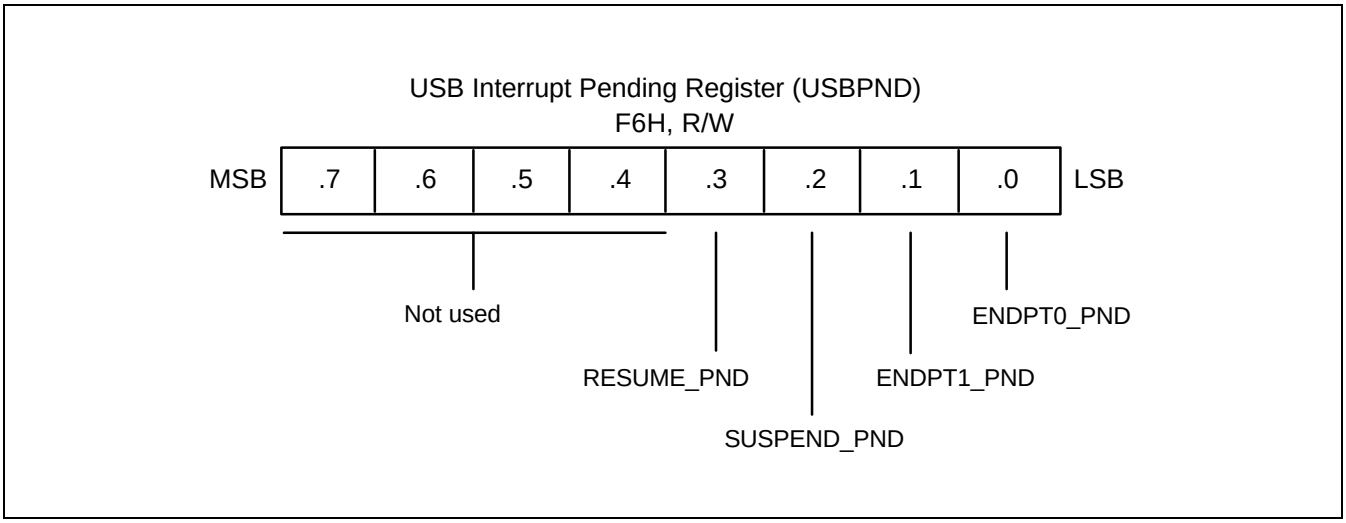


Figure 11-5. USB Interrupt Pending Register (USBPND)

USB INTERRUPT ENABLE REGISTER (USBINT)

USBINT is located at address F7H and is read/write addressable. This register serves as an interrupt mask register. If the corresponding bit = 1 then the respective interrupt is enabled.

By default, all interrupts except suspend interrupt is enabled. Interrupt enables bits for suspend and resume is combined into a single bit (bit 2).

Bit7–3 Not used

Bit2 **ENABLE_SUSPEND_RESUME_INT:**
1 Enable SUSPEND and RESUME INTERRUPT
0 Disable SUSPEND and RESUME INTERRUPT (default)

Bit1 **ENABLE_ENDPT1_INT:**
1 Enable ENDPOINT 1 INTERRUPT (default)
0 Disable ENDPOINT 1 INTERRUPT

Bit0 **ENABLE_ENDPT0_INT:**
1 Enable ENDPOINT 0 INTERRUPT (default)
0 Disable ENDPOINT 0 INTERRUPT

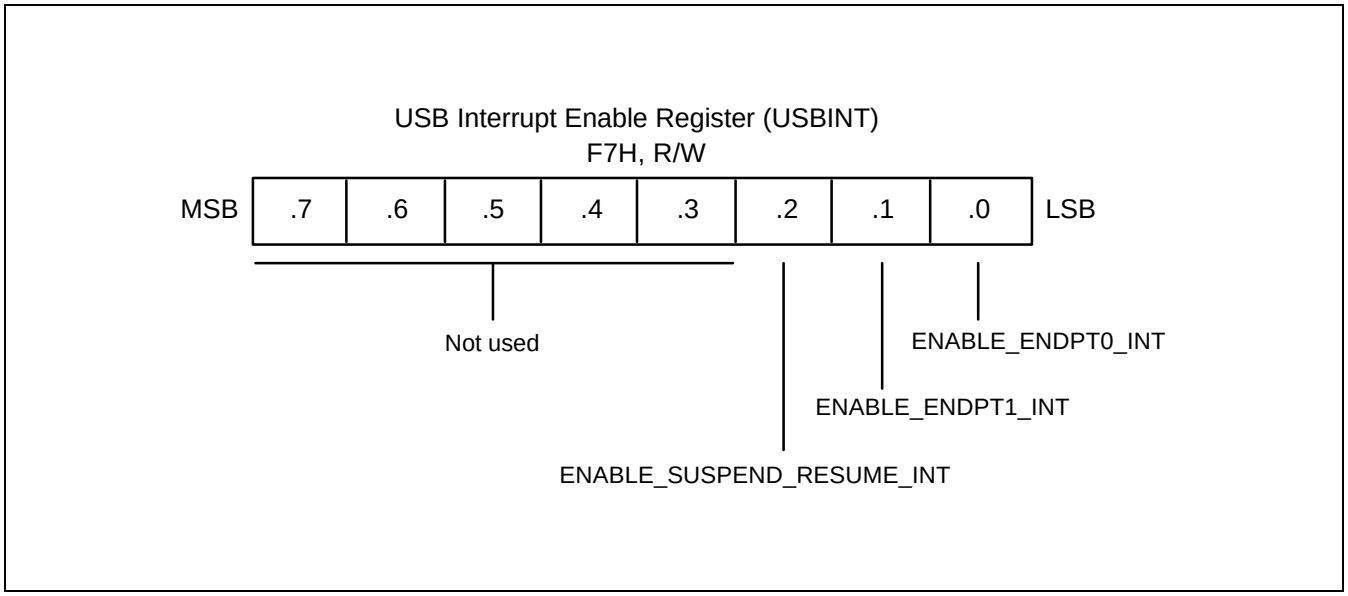


Figure 11-6. USB Interrupt Enable Register (USBINT)

USB POWER MANAGEMENT REGISTER (PWRMGR)

PWRMGR register interacts with the Host's power management system to execute system power events such as SUSPEND or RESUME. This register is located at address F8H and is read/write addressable.

Bit7–2 **RESERVED:** The value read from this bit is zero.

Bit1 **SEND_RESUME:** While in SUSPEND state, if the MCU wants to initiate RESUME, it writes "1" to this register for 10ms (maximum of 15ms), and clears this register. In SUSPEND mode if this bit reads "1", USB generates RESUME signaling.

Bit0 **SUSPEND_STATE:** Suspend state is set when the MCU sets suspend interrupt. This bit is cleared automatically when:

— MCU writes "0" to SEND_RESUME bit to end the RESUME signaling (after SEND_RESUME is set for 10ms).
— MCU receives RESUMES signaling from the Host while in SUSPEND mode.

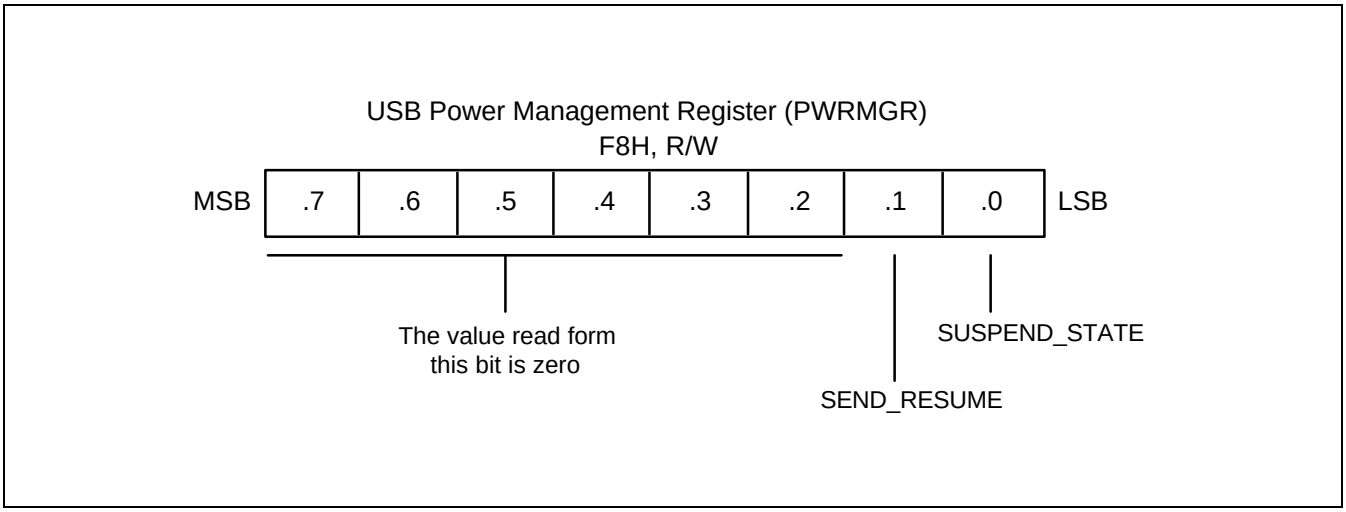


Figure 11-7. USB Power Management Register (PWRMGR)

USB RESET REGISTER (USBRST)

USBRST register receives a reset signal from the Host. This register is located at address FFH and is read/write addressable.

Bit7–1 Not used

Bit0 **USBRST:** This bit is set when the Host issues an USB reset signal.

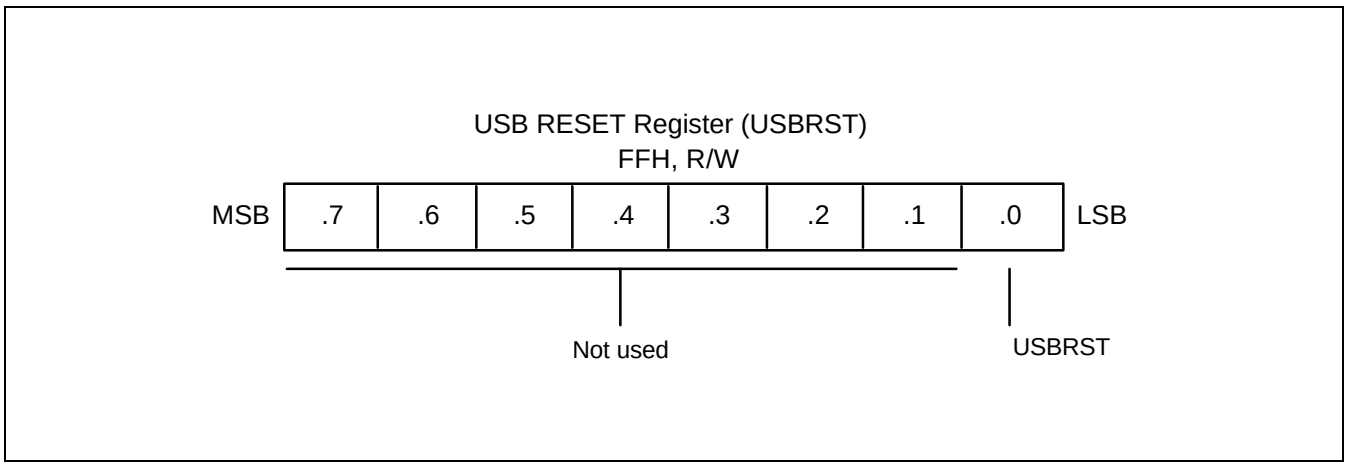


Figure 11-8. USB RESET Register (USBRST)

12 COMPARATOR

OVERVIEW

P1.0–P1.3 can be used as a analog input port for a comparator. The reference voltage for the 4-channel comparator can be supplied either internally or externally at P1.3. When an internal reference voltage is used, four channels (P1.0–P1.3) are used for analog inputs and the internal reference voltage is varied in 16 levels. If an external reference voltage is input at P1.3, the other three pins (P1.0–P1.2) in port x are used analog input. Unused port x pins should be connected to V_{DD} or V_{SS} for current saving.

When a conversion is completed, the result is saved in the comparison result register CDATA. The initial values of the CDATA are undefined and the comparator operation is disabled by a RESET

— Analog comparator
— Internal reference voltage generator (4-bit resolution)
— External reference voltage source at P1.3
— Comparator mode register (CCON)
— Four multiplexed analog data input pins (CIN0–CIN3)
— 4–channel conversion data result register (CDATA)
— 4–bit digital input port (alternatively, I/O port)

FUNCTION DESCRIPTION

The comparator compares analog voltage input at CIN0–CIN3 with an external or internal reference voltage (VREF) that is selected by CCON register. The result is written to the comparison result register CDATA at address E5H. The comparison result is calculated as follows.

If “1” Analog input voltage ≥ V_{REF} + 100 mV

If “0” Analog input voltage ≤ V_{REF} – 100 mV

To obtain a comparison result, the data must be read out from the CDATA register after V_{REF} is updated by changing the CCON value after a conversion time has elapsed.

COMPARATOR CONTROL REGISTER (CCON)

The comparator control register CCON is an 8–bit register that is used to set the operation mode of the comparator. To initiate a comparison procedure, you write the reference voltage selection data in the comparator control register CCON and set the comparison start of enable bit, CCON.7.

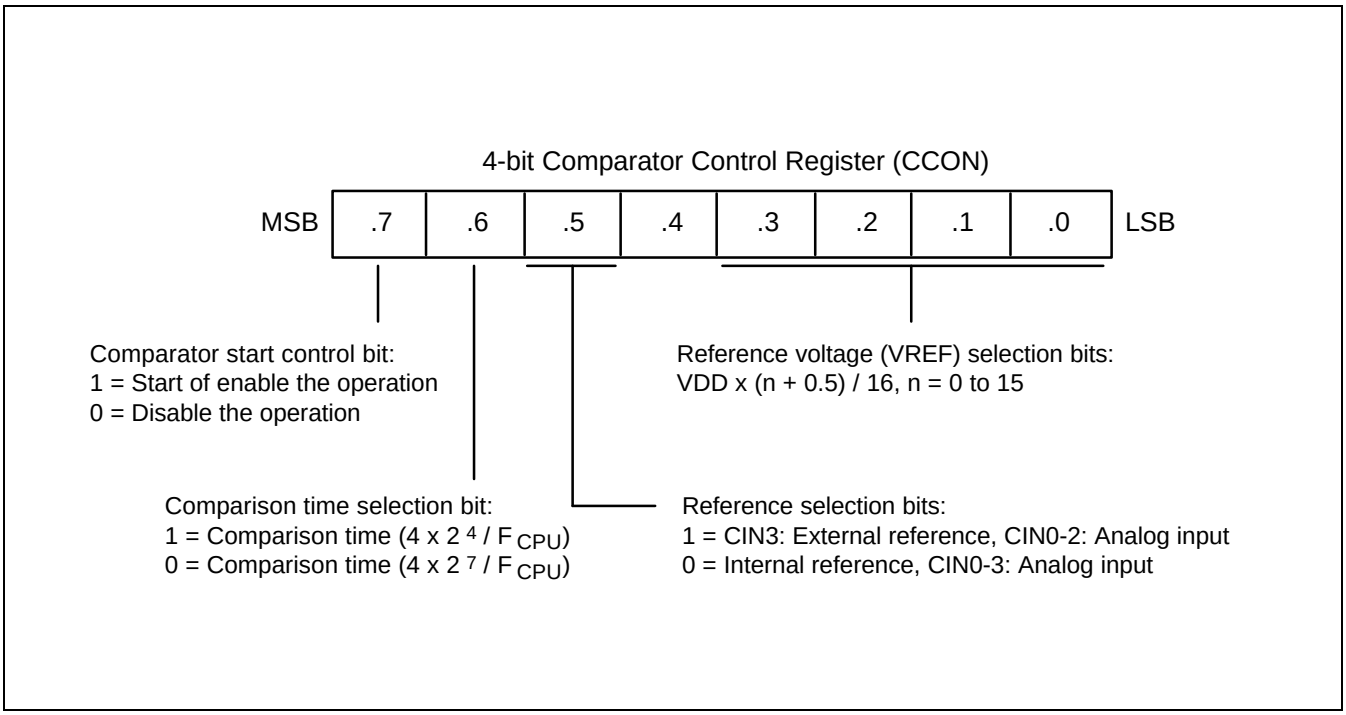


Figure 12-1. Comparison Control Register (CCON)

BLOCK DIAGRAM

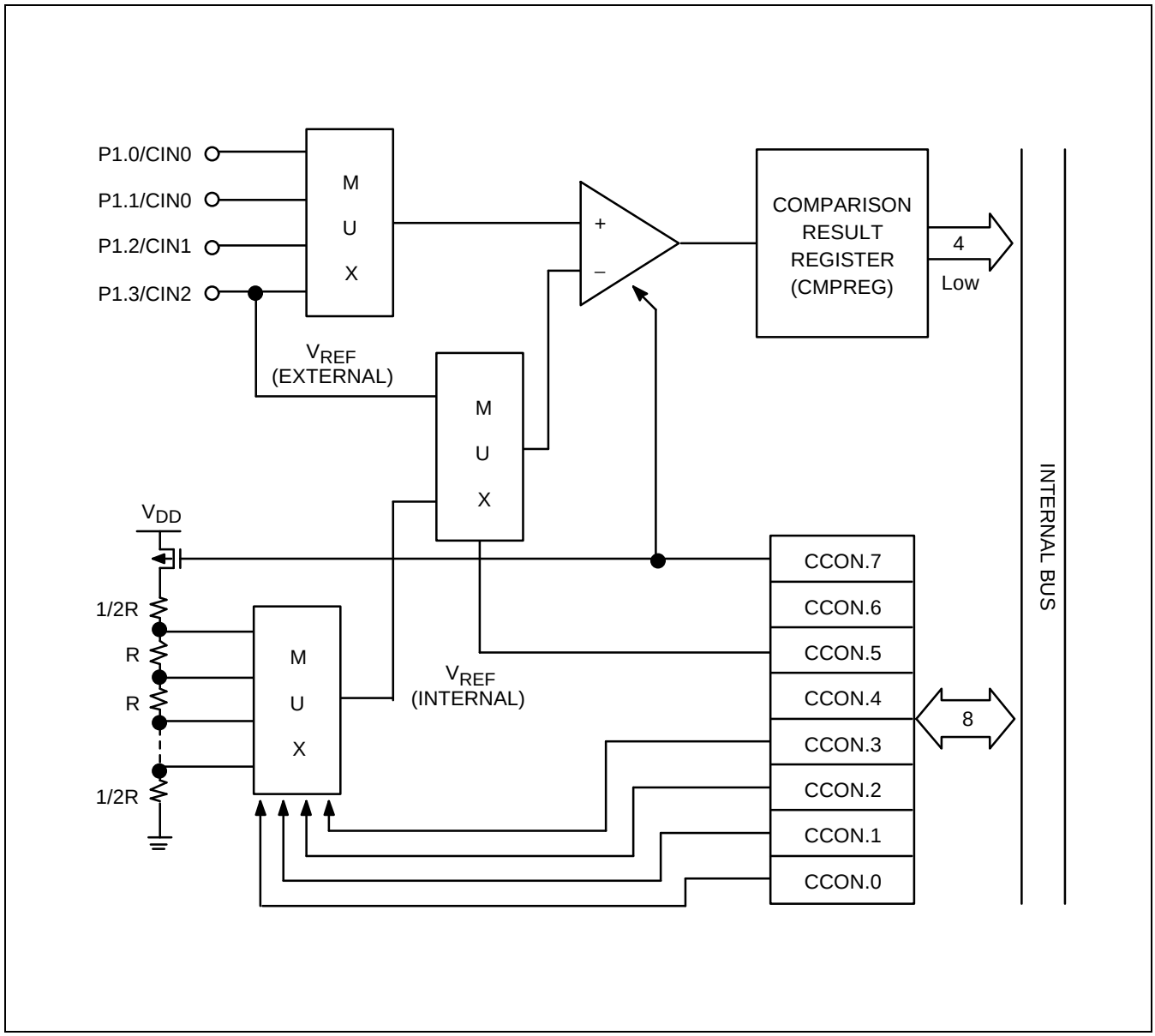


Figure 12-2. Comparator Functional Block Diagram

NOTES

13 ELECTRICAL DATA

OVERVIEW

In this section, the following KS86C6104/P6104 electrical characteristics are presented in tables and graphs:

— Absolute maximum ratings
— D.C. electrical characteristics
— I/O capacitance
— A.C. electrical characteristics
— Input timing for RESET
— Oscillator characteristics
— Operating voltage range
— Oscillation stabilization time
— Clock timing measurement points at X_{IN}
— Data retention supply voltage in Stop mode
— Stop mode release timing when initiated by a RESET
— Stop mode release timing when initiated by an external interrupt
— Characteristic curves
— Comparator Electrical Characteristics

Table 13-1. Absolute Maximum Ratings

(T_A = 25°C)

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

Table 13-2. D.C. Electrical Characteristics

(T_A = – 40°C to + 85°C, V_{DD} = 4.0 V to 5.25 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input highvoltage	V_{IH1}	All input pins except V_{IH2}, D+, D–	0.8 V_{DD}	–	V_{DD}	V
	V_{IH2}	X_{IN}	V_{DD} – 0.5		V_{DD}	
Input low voltage	V_{IL1}	All input pins except V_{IL2}, D+, D–	–	–	0.2 V_{DD}	V
	V_{IL2}	X_{IN}	–	–	0.4	
Output high voltage	V_{OH}	V_{DD} = 4.5 V – 5.5 V I_{OH} = – 200 µA All output ports except D+, D–	V_{DD} – 1.0	–	–	V
Output low voltage	V_{OL}	V_{DD} = 4.5 V – 5.5 V I_{OL} = 2 mA All output ports except D+, D–	–	–	0.4	V
Input high leakage current	I_{LIH1}	V_{IN} = V_{DD} All inputs except I_{LIH2} except D+, D–	–	–	3	µA
	I_{LIH2}	V_{IN} = V_{DD} X_{IN}, X_{OUT}	–	–	20	
Input low leakage current	I_{LIL1}	V_{IN} = 0 V All inputs except I_{LIL2} except D+, D–	–	–	– 3	µA
	I_{LIL2}	V_{IN} = 0 V X_{OUT}, X_{IN}	–	–	– 20	
Output high leakage current	I_{LOH}	V_{OUT} = V_{DD} All output pins except D+, D–	–	–	3	µA
Output low leakage current	I_{LOL}	V_{OUT} = 0 V All output pins except D+, D–	–	–	– 3	µA
Pull-up resistors	R_{L1}	V_{IN} = 0 V; V_{DD} = 5.0 V,	25	50	100	KΩ
	R_{L2}	V_{IN} = 0 V; V_{DD} = 5.0 V, RESET only	100	220	400	
Supply current (note)	I_{DD1}	Normal operation mode 6-MHz CPU clock	–	6.5	15	mA
	I_{DD2}	Idle mode; 6-MHz CPU clock	–	4	8	mA
	I_{DD3}	Stop mode; oscillator stop	–	150	300	µA

NOTES:
1. Supply current does not include current drawn through internal pull-up resistors or external output current load.
2. This parameter is guaranteed, but not tested (include D+, D–).
3. Only in 4.2 V to 5.25 V, D+ and D– satisfy the USB spec 1.0.

Table 13-3. Input/Output Capacitance

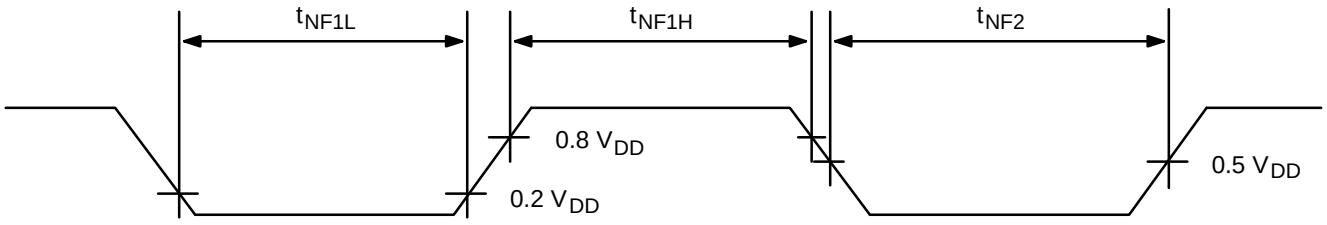
(T_A = – 40°C to + 85°C, V_{DD} = 0 V)

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

Table 13-4. A.C. Electrical Characteristics

(T_A = – 40°C to + 85°C, V_{DD} = 4.0 V to 5.25 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Noise filter	t_{NF1H}, t_{NF1L}	P1 (RC delay)	100	–	200	ns
	t_{NF2}	RESET only (RC delay)	–	800	–	
RESET input low width	t_{RSL}	Input	10	–	–	µs



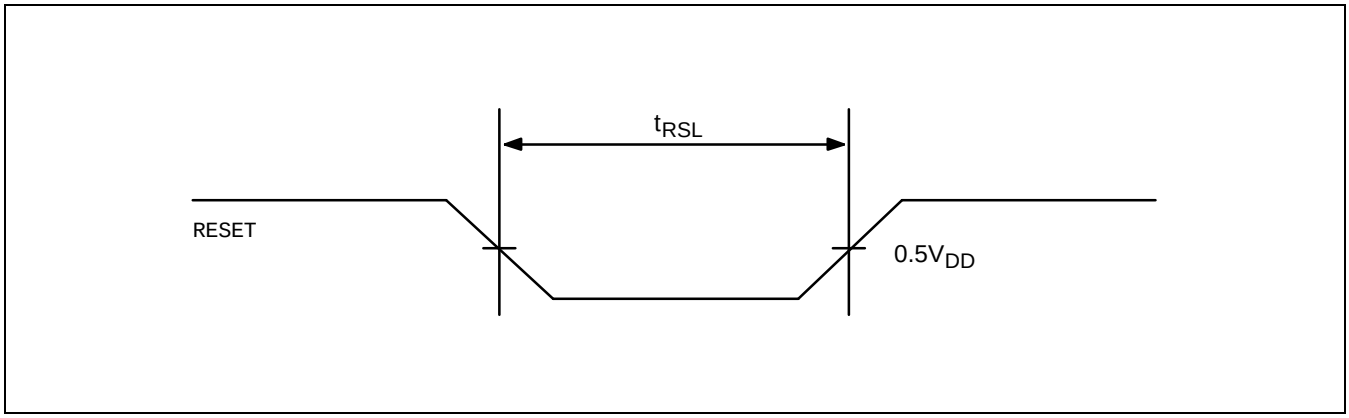


Figure 13-1. Input Timing for RESET

Table 13-5. Oscillator Characteristics

(T_A = – 40°C + 85°C)

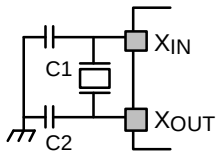
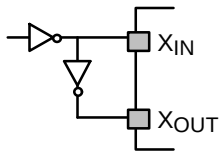
Oscillator	Clock Circuit	Test Condition	Min	Typ	Max	Unit
Main crystal Main ceramic (f_{OSC})		Oscillation frequency V_{DD} = 4.0 V – 5.25 V	–	6.0	–	MHz
External clock		Oscillation frequency V_{DD} = 4.0 V – 5.25 V	–	6.0	–	

Table 13-6. Oscillation Stabilization Time

(T_A = – 40°C + 85°C, V_{DD} = 4.0 V to 5.25 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Main crystal Main ceramic	V_{DD} = 4.5 V to 5.5 V, f_{OSC} > 6.0 MHz (Oscillation stabilization occurs when V_{DD} is equal to the minimum oscillator voltage range.)	–	–	10	ms
Oscillator stabilization wait time	t_{WAIT} stop mode release time by a reset	–	$2^{16}/f_{OSC}$	–	
	t_{WAIT} stop mode release time by an interrupt	–	–	–	

NOTE: The oscillator stabilization wait time, t_{WAIT}, when it is released by an interrupt, is determined by the setting in the basic timer control register, BTCON.

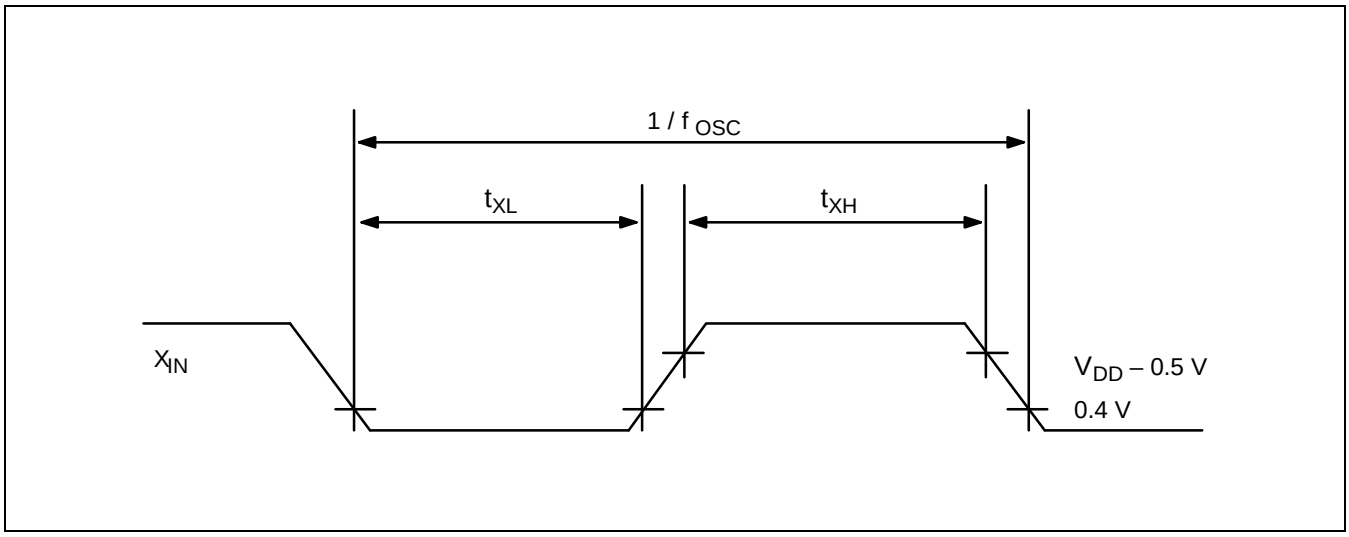


Figure 13-2. Clock Timing Measurement Points at X_{IN}

Table 13-7. Data Retention Supply Voltage in Stop Mode

(T_A = 0°C to + 70°C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	Stop mode	2.0	–	6	V
Data retention supply current	I_{DDDR}	Stop mode; V_{DDDR} = 2.0 V	–	–	5	µA

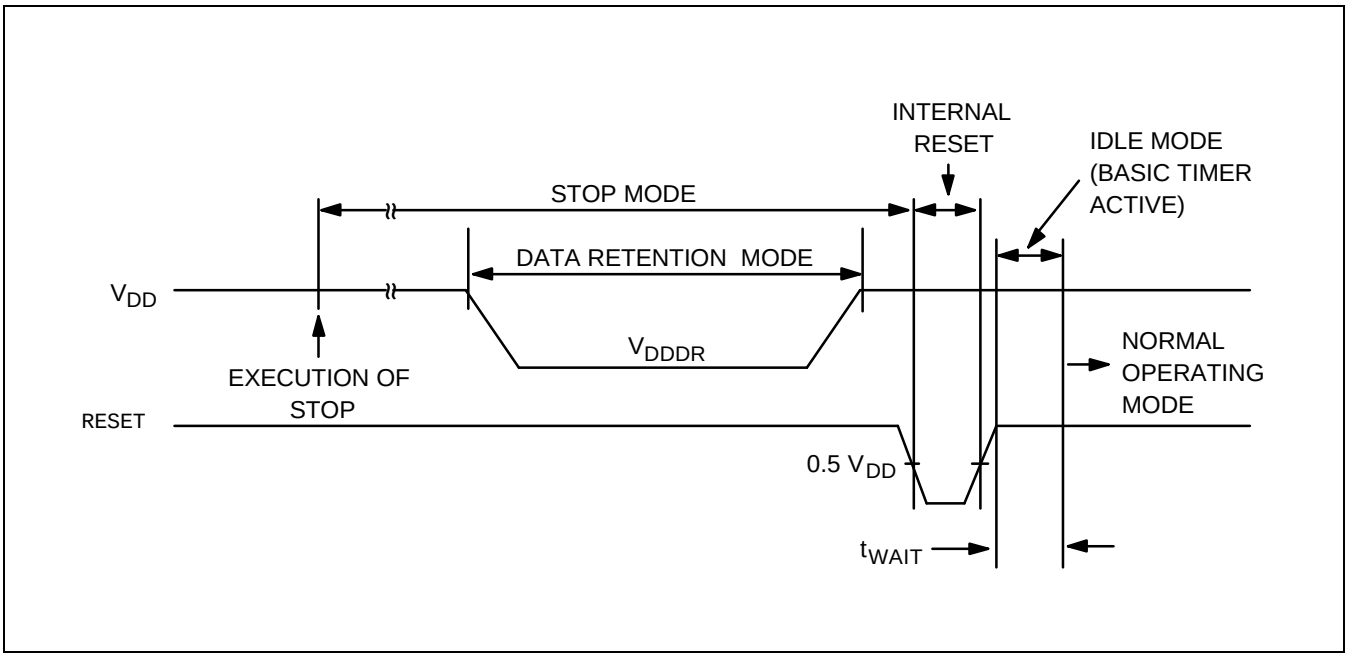


Figure 13-3. Stop Mode Release Timing When Initiated by a RESET

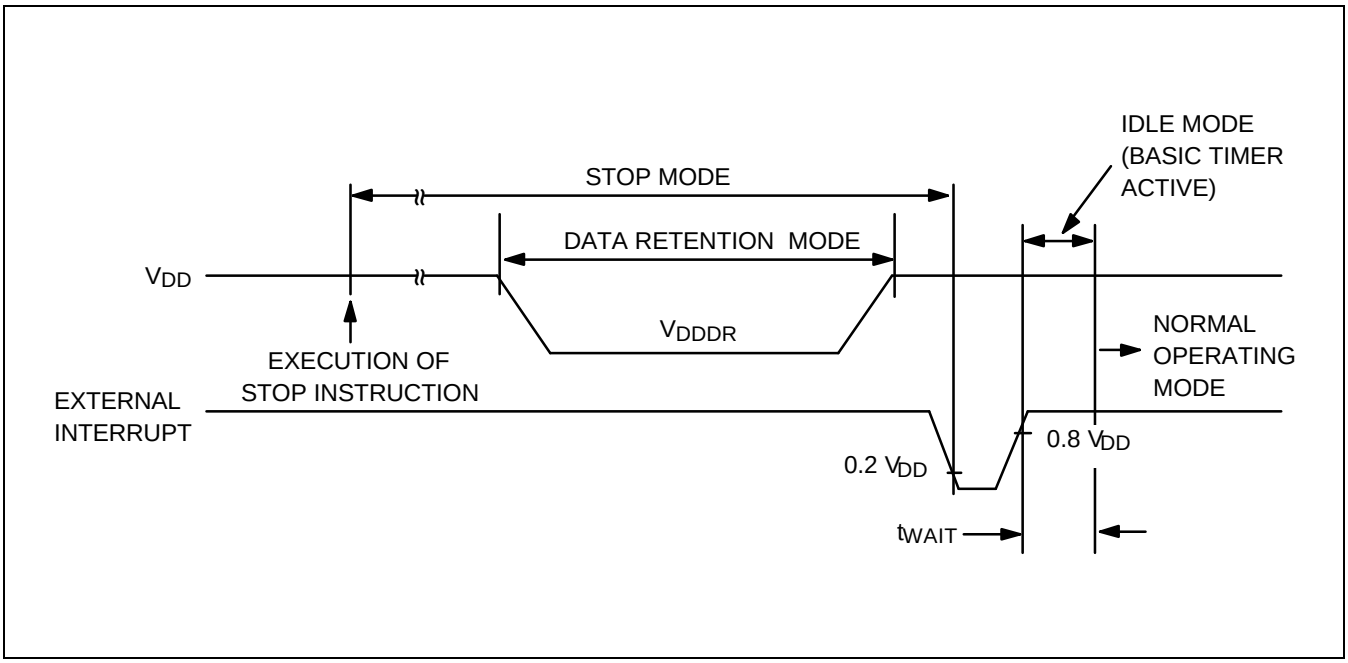


Figure 13-4. Stop Mode Release Timing When Initiated by an External Interrupt

Table 13-8. Comparator Electrical Characteristics

(T_A = – 40°C to + 85°C, V_{DD} = 4.0 V to 5.25 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Conversion time (1)	tCON	–	–	4×2^4 or 4×2^7	–	F_{CPU}
Comparator input voltage	V_{ICN}	–	V_{SS}	–	V_{DD}	V
Comparator input impedance	R_{CN}	–	2	1000	–	MΩ
Comparator reference voltage	V_{REF}	–	1.8	–	V_{DD}	V
Comparator input current	I_{CIN}	V_{DD} = 5 V	– 3	–	3	µA
Reference input current	I_{REF}	V_{DD} = 5 V	– 3	–	3	µA
Comparator block current (2)	I_{COM}	V_{DD} = 5.5 V	–	1	2	mA
		V_{DD} = 4.5 V		0.5	1	mA
		V_{DD} = 5 V (when power down mode)		100	500	nA

NOTES:
1. Conversion time is the time required from the moment a conversion operation starts until it ends.
2. I_{COM} is an operating current during conversion.

Table 13-9. Low Speed Source Electrical Characteristics (USB)

(T_A = – 40°C to + 85°C, Voltage Regulator Output V_{33out} = 2.8 V to 3.5 V, typ 3,3 V)

Parameter	Symbol	Conditions	Min	Max	Unit
Transition Time:					
Rise Time	Tr	CL = 50 pF	75	–	ns
		CL = 350 pF	–	300	
Fall Time	Tf	CL = 50 pF	75	–	
		CL = 350 pF	–	300	
Rise/Fall Time Matching	Trfm	(Tr/Tf) CL = 50 pF	70	130	%
Output Signal Crossover Voltage	Vcrs	CL = 50 pF	1.3	2.0	V
Voltage Regulator Output Voltage	V33OUT	with V33OUT to GND 0.1 µF capacitor	2.8	3.5	V

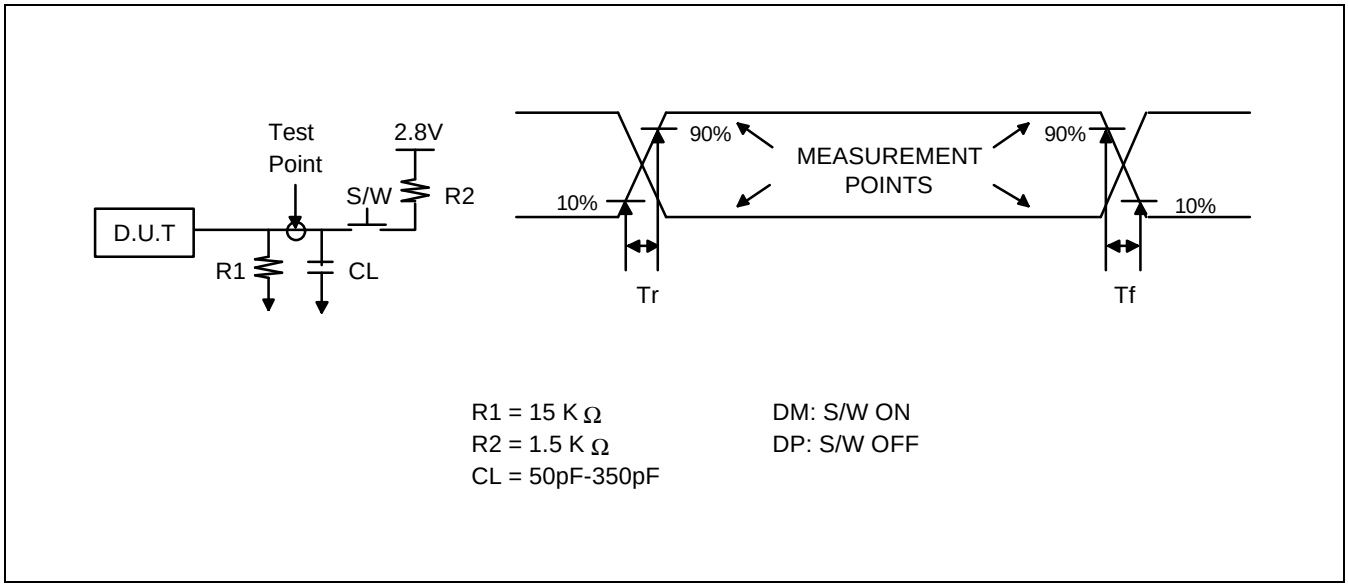


Figure 13-5. USB Data Signal Rise and Fall Time

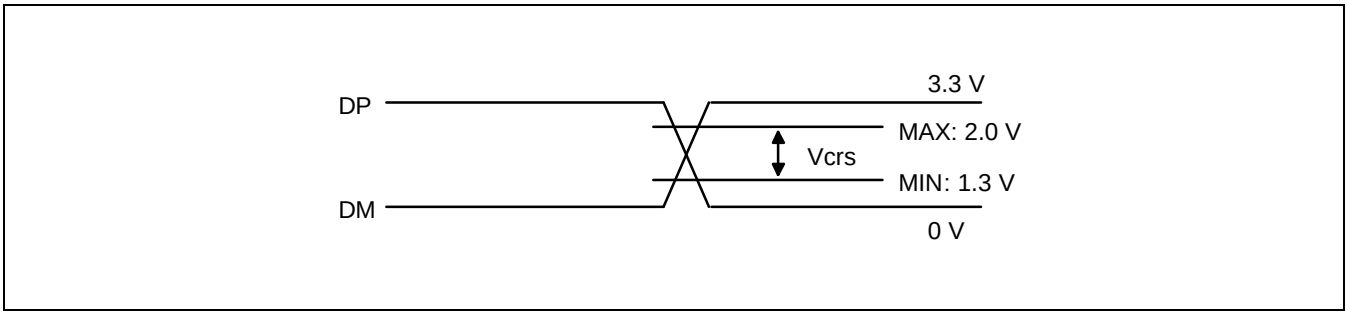


Figure 13-6. USB Output Signal Crossover Point Voltage

NOTES

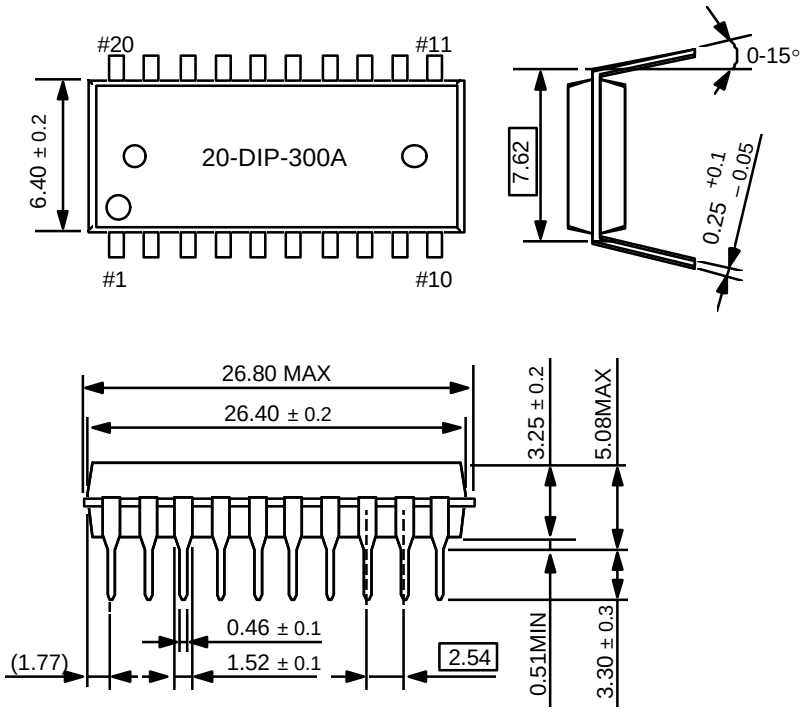
14 MECHANICAL DATA

OVERVIEW

This section contains the following information about the device package:

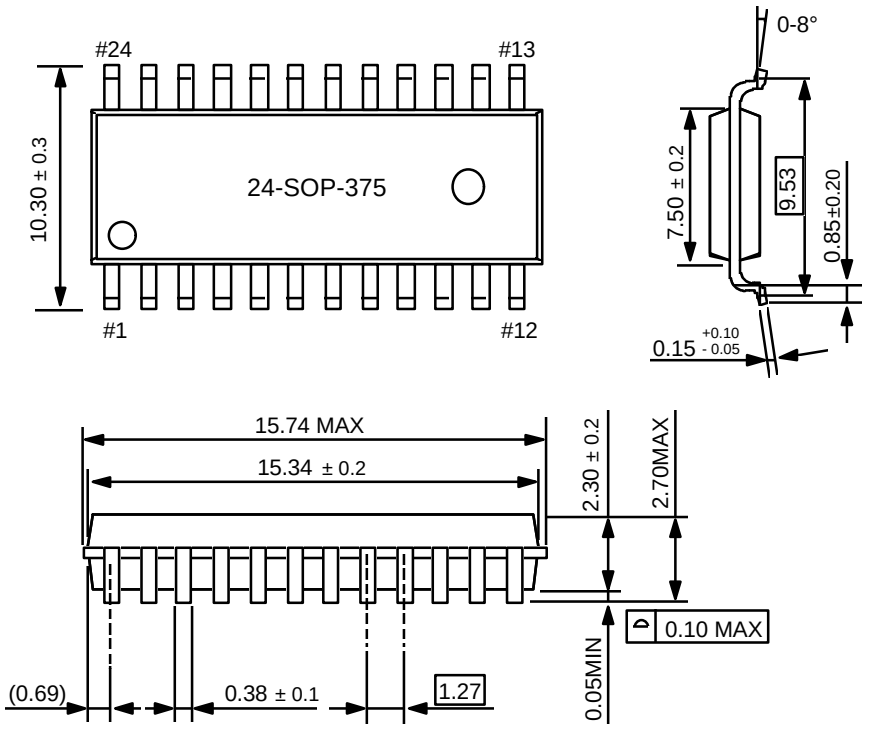
— Package dimensions in millimeters

— Pad diagram



NOTE: Dimensions are in millimeters.

Figure 14-1. 20-DIP0300A Package Dimensions



NOTE: Dimensions are in millimeters.

Figure 14-2. 24-SOP-375 Package Dimensions

15 KS86P6104 OTP

OVERVIEW

The KS86P6104 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the KS86C6104 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The KS86P6104 is fully compatible with the KS86C6104, both in function and in pin configuration. Because of its simple programming requirements, the KS86P6104 is ideal for use as an evaluation chip for the KS86C6104.

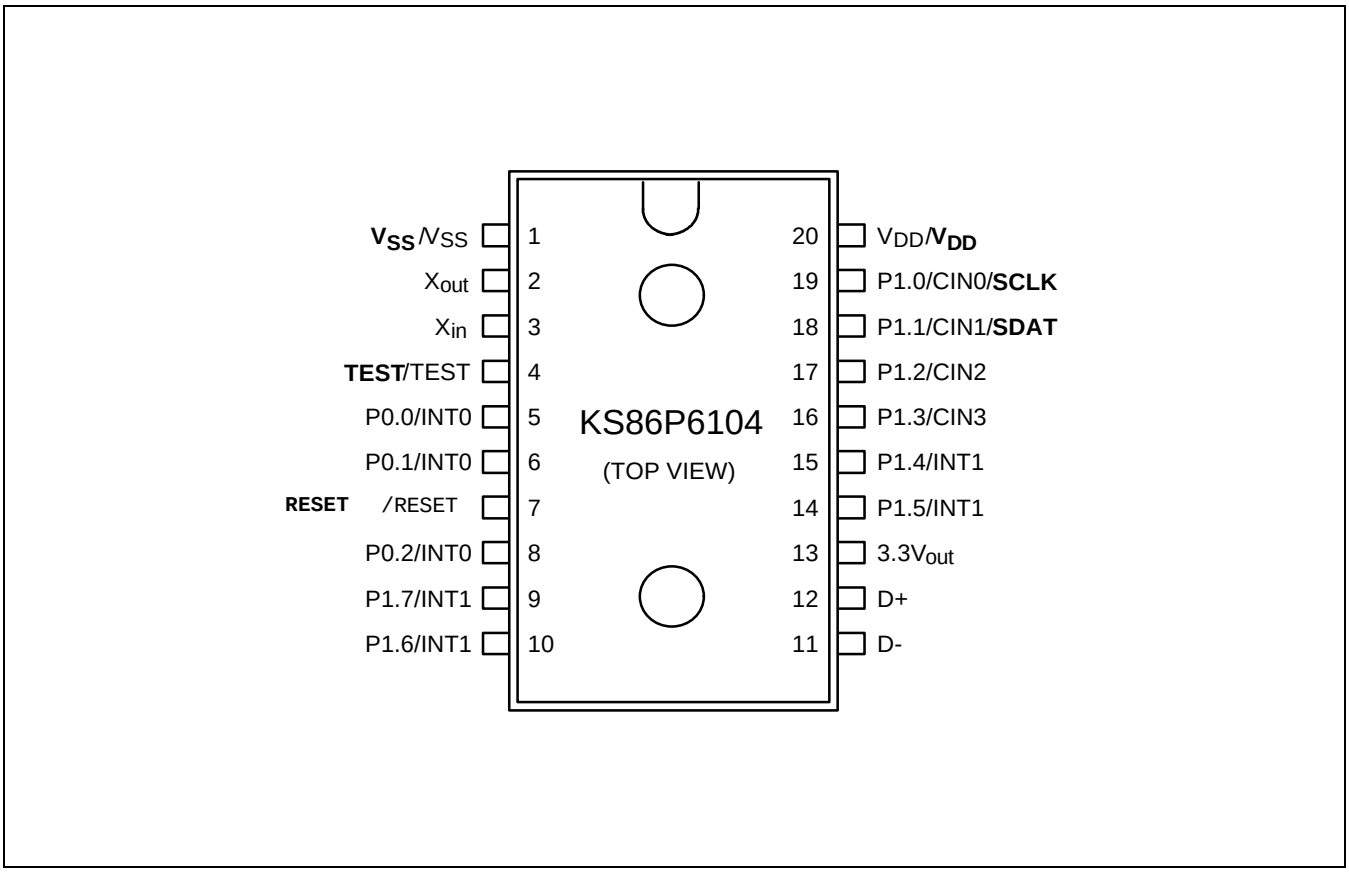


Figure 15-1. KS86P6104 Pin Assignments (20-DIP Package)

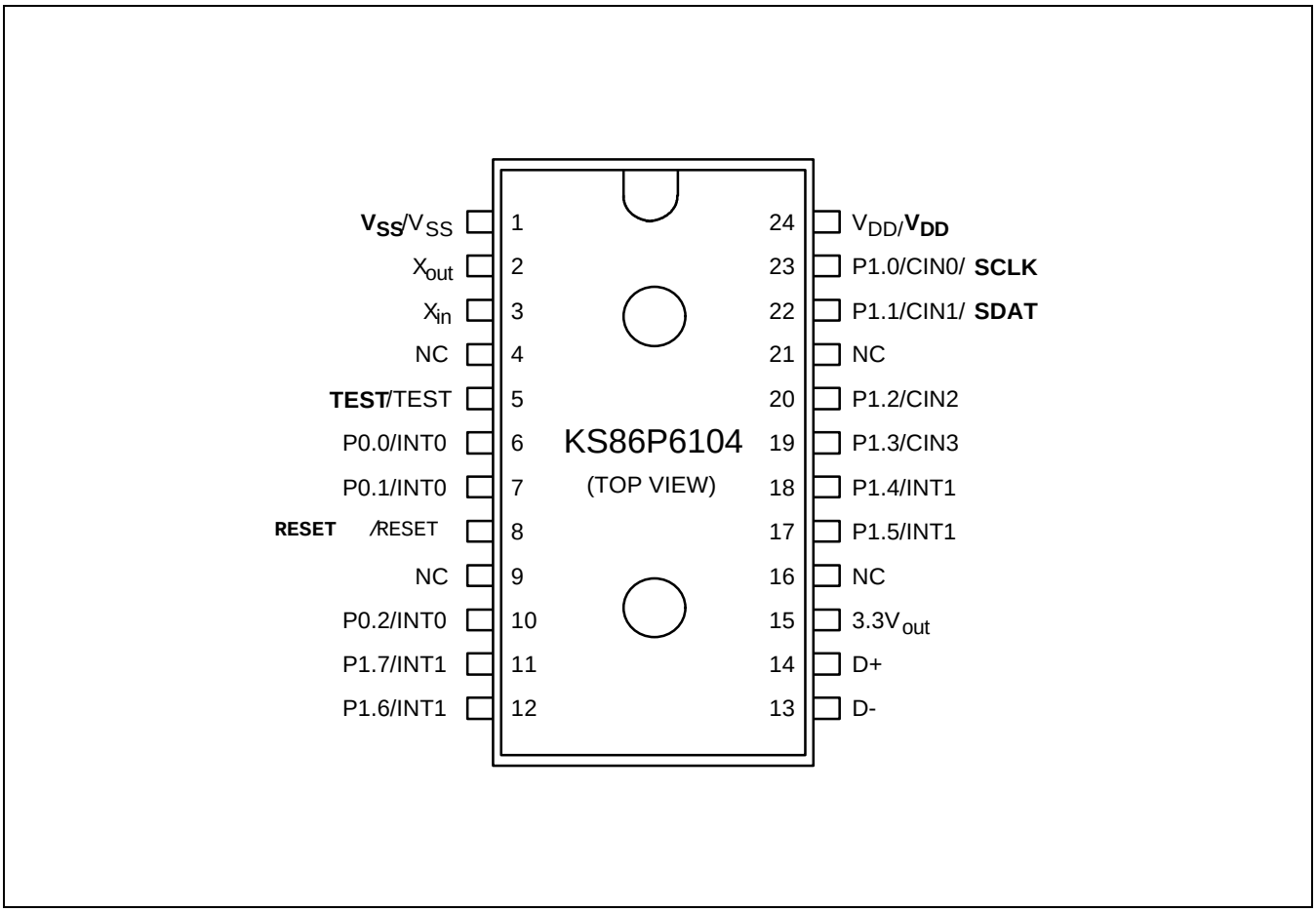


Figure 15-2. KS86P6104 Pin Assignments (24-SOP Package)

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

Main Chip Pin Name	During Programming			
	Pin Name	Pin No. (20 DIP)	I/O	Function
P1.0 (Pin 18)	SDAT	18	I/O	Serial Data Pin (Output when reading, Input when writing) Input and Push-pull Output Port can be assigned
P1.1 (Pin 19)	SCLK	19	I/O	Serial Clock Pin (Input Only Pin)
TEST	V_{PP} (TEST)	4	I	0V : OTP write and test mode 5V : Operating mode
RESET	RESET	7	I	Chip Initialization and EPROM Cell Writing Power Supply Pin (Indicates OTP Mode Entering) When writing 12.5V is applied and when reading.
V_{DD}/V_{SS}	V_{DD}/V_{SS}	20/1	I	Logic Power Supply Pin.

Table 15-2. Comparison of KS86P6104 and KS86C6104 Features

Characteristic	KS86P6104	KS86C6104
Program Memory	4 K byte EPROM	4 K byte mask ROM
Operating Voltage (V_{DD})	4.0 V to 5.25 V	4.0 V to 5.25 V
OTP Programming Mode	V_{DD} = 5 V, V_{PP} (RESET)=12.5V	
Pin Configuration	20 DIP/24 SOP	20 DIP/24 SOP
EPROM Programmability	User Program 1 time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V_{PP} (RESET) pin of the KS86P6104, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 14-3 below.

Table 15-3. Operating Mode Selection Criteria

V_{DD}	V_{PP} (RESET)	REG/ MEM	Address (A15-A0)	R/W	Mode
5 V	5 V	0	0000H	1	EPROM read
	12.5 V	0	0000H	0	EPROM program
	12.5 V	0	0000H	1	EPROM verify
	12.5 V	1	0E3FH	0	EPROM read protection

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

Table 15-4. D.C. Electrical Characteristics

(T_A = – 40°C to + 85°C, V_{DD} = 4.0 V to 5.25 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Supply Current (note)	I_{DD1}	Normal mode; 6 MHz CPU clock	–	6.5	15	mA
	I_{DD2}	Idle mode; 6 MHz CPU clock		4	8	
	I_{DD3}	Stop mode; oscillator stop		150	300	µA

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

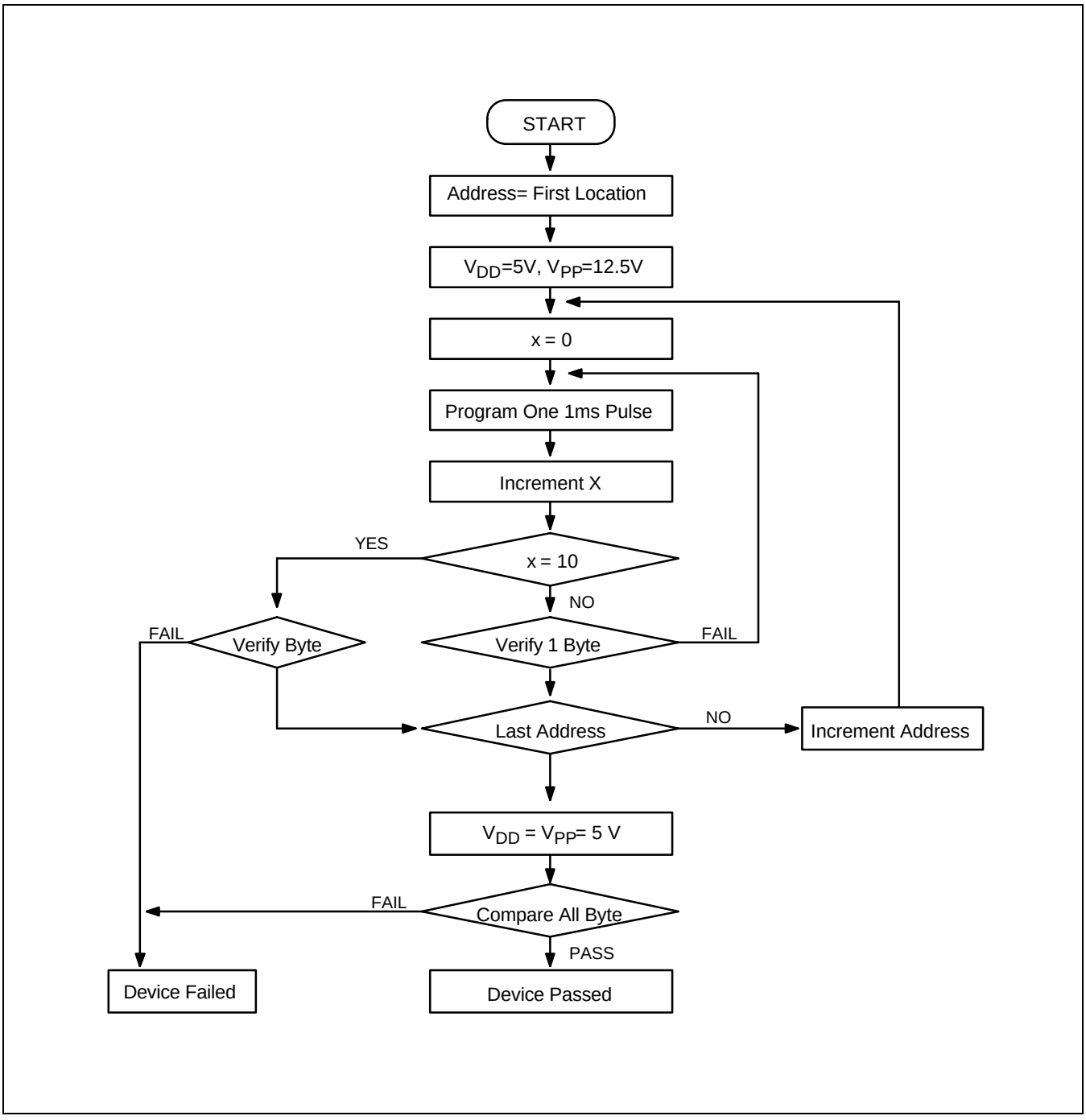


Figure 15-3. OTP Programming Algorithm

NOTES

16 DEVELOPMENT TOOLS

OVERVIEW

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

SHINE

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

SAMA ASSEMBLER

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

SASM86

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

HEX2ROM

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

TARGET BOARDS

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

OTPs

One times programmable microcontrollers (OTPs) are under development for KS86C6104 microcontroller.

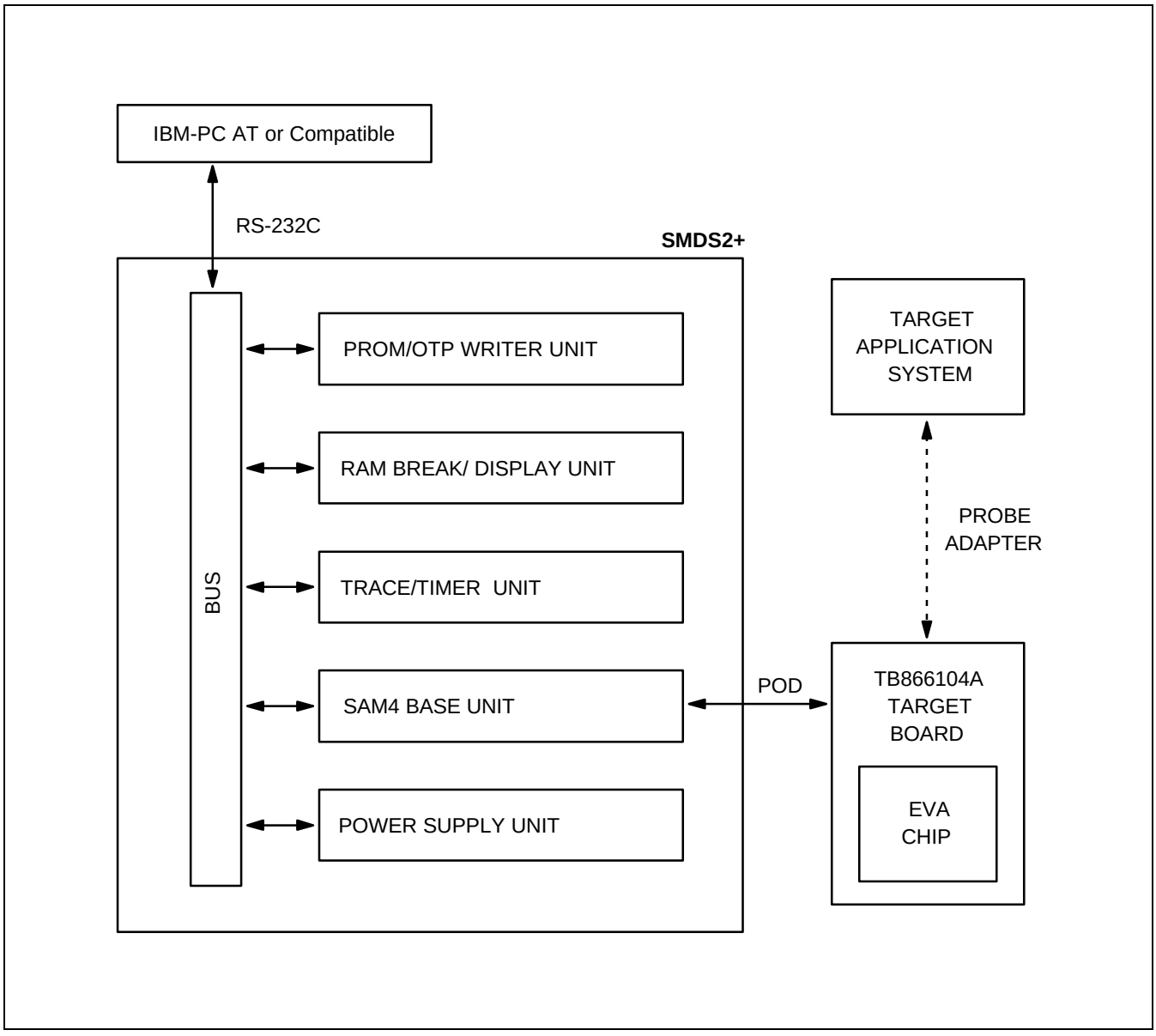


Figure 16-1. SMDS Product Configuration (SMDS2+)

TB866104A TARGET BOARD

The TB866104A target board is used for the KS86C6104 microcontrollers. It is supported by the SMDS2+ development systems. The TB866104A target board can also be used for KS86C6104.

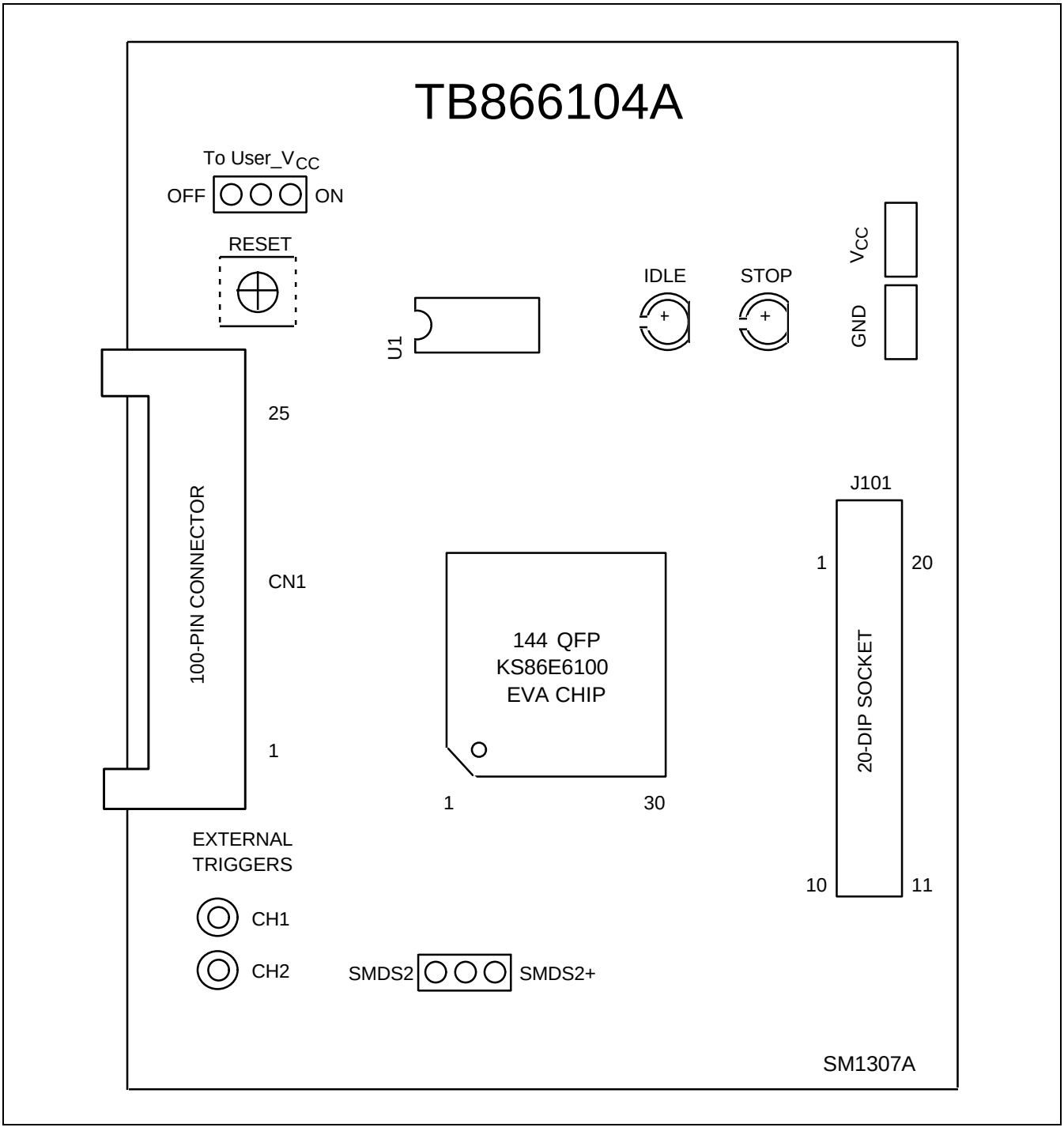
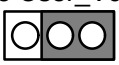
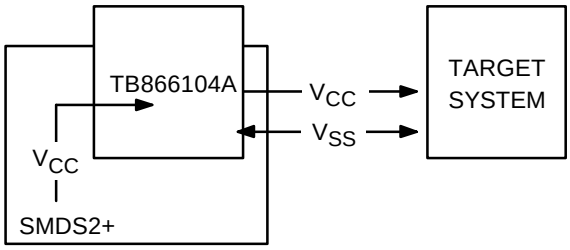
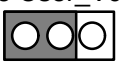
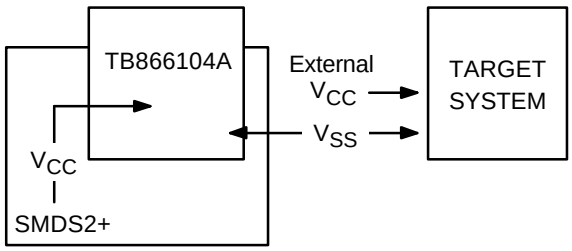
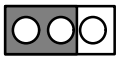


Figure 16-2. TB866104A Target Board Configuration

Table 16-1. Power Selection Settings for TB866104A

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

NOTE: The following symbol in the "To User_V_{CC}" Setting column indicates the electrical short (off) configuration:



SMDS2+ Selection (SAM8)

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

Table 16-2. The SMDS2+ Tool Selection Setting

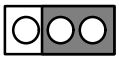
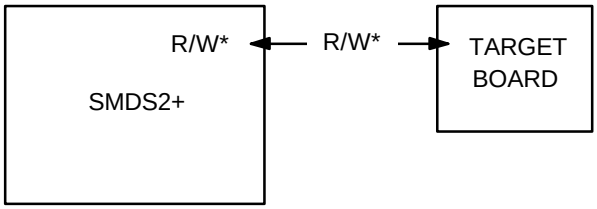
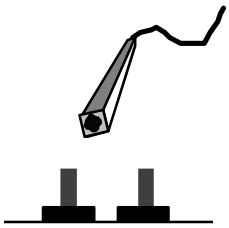
"SW1" Setting	Operating Mode
SMDS2  SMDS2+	

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

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

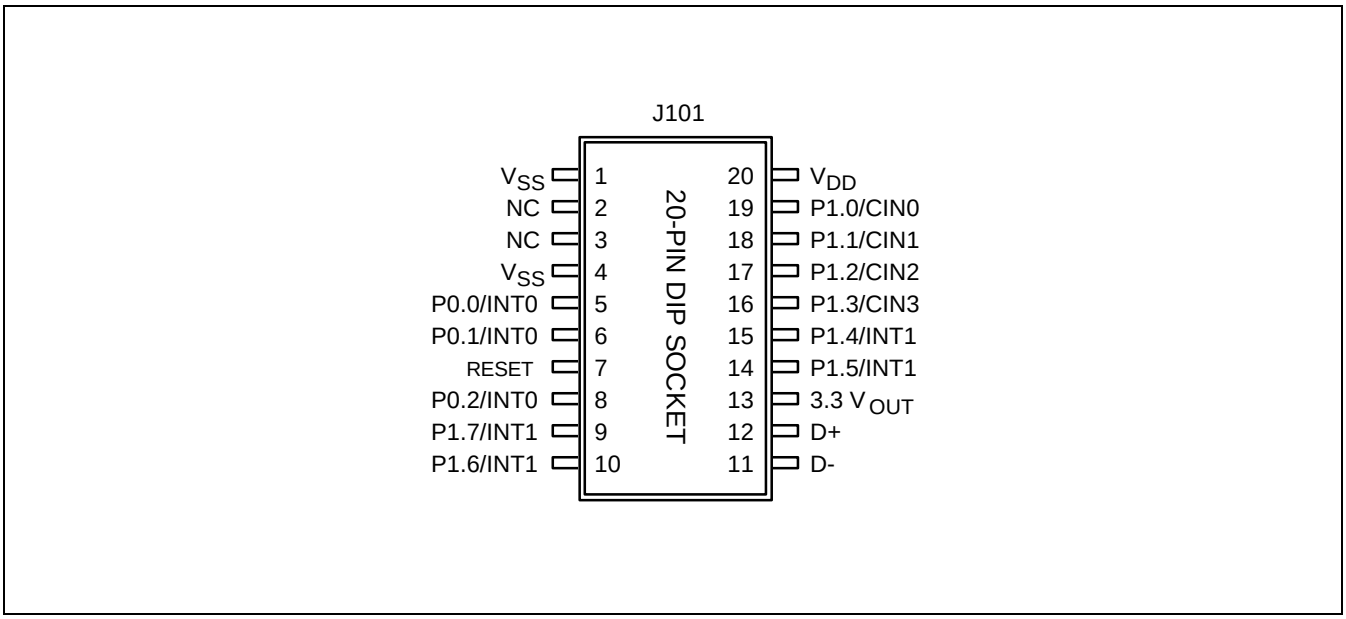


Figure 16-3. 20-Pin DIP Socket for TB866104A

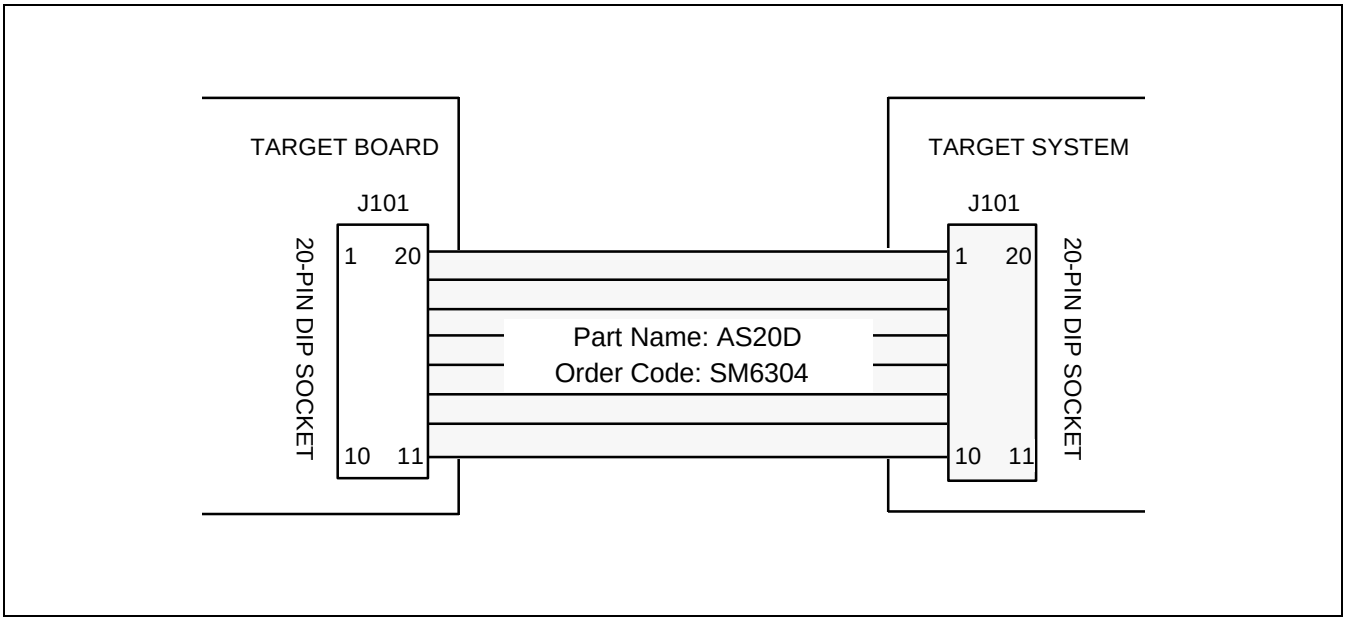


Figure 16-4. KS86C6104 Probe Adapter Cable for 20-DIP Package