

1 PRODUCT OVERVIEW

S3C8-SERIES MICROCONTROLLERS

Samsung's S3C8 series of 8-bit single-chip CMOS microcontrollers offers a fast and efficient CPU, a wide range of integrated peripherals, and various mask-programmable ROM sizes. Among the major CPU features are:

- Efficient register-oriented architecture
- Selectable CPU clock sources
- Idle and Stop power-down mode release by interrupt
- Built-in basic timer with watchdog function

A sophisticated interrupt structure recognizes up to eight interrupt levels. Each level can have one or more interrupt sources and vectors. Fast interrupt processing (within a minimum of six CPU clocks) can be assigned to specific interrupt levels.

S3C821A/P821A MICROCONTROLLER

The S3C821A/P821A single-chip CMOS microcontroller is fabricated using the highly advanced CMOS process, based on Samsung's newest CPU architecture.

The S3C821A is a microcontroller with a 48-Kbyte mask-programmable ROM embedded.

The S3P821A is a microcontroller with a 48-Kbyte one-time-programmable ROM embedded.

Using a proven modular design approach, Samsung engineers have successfully developed the S3C821A/P821A by integrating the following peripheral modules with the powerful SAM8 core:

- Six programmable I/O ports, including five 8-bit

ports and one 7-bit port, for a total of 47 pins.

- Twelve bit-programmable pins for external interrupts.
- One 8-bit basic timer for oscillation stabilization and watchdog functions (system reset).
- One 8-bit timer/counter and one 16-bit timer/counter with selectable operating modes.
- Watch timer for real time.
- 4-input A/D converter
- Serial I/O interface

The S3C821A/P821A is versatile microcontroller for cordless phone, pager, etc. They are currently available in 80-pin TQFP and 80-pin QFP package.

OTP

The S3P821A is an OTP (One Time Programmable) version of the S3C821A microcontroller. The S3P821A microcontroller has an on-chip 48-Kbyte one-time-programmable EPROM instead of a masked ROM. The S3P821A is comparable to the S3C821A, both in function and in pin configuration.

FEATURES

CPU

- SAM8 CPU core

Memory

- Data memory: 1040-byte of internal register file (Excluding LCD RAM)
- Program memory: 48-Kbyte internal program memory (ROM)

External Interface

- 64-Kbyte external data memory area

Instruction Execution Time

- 750 ns at 8 MHz (minimum, Main oscillator)
- 183 μ s at 32,768 Hz (minimum, Sub oscillator)

Interrupts

- 7 interrupt levels and 19 interrupt sources
- 19 vectors
- Fast interrupt processing feature (for one selected interrupt level)

I/O Ports

- Five 8-bit I/O ports (P0–P4) and one 7-bit I/O port (P5) for a total of 47 bit-programmable pins

8-Bit Basic Timer

- One programmable 8-bit basic timer (BT) for oscillation stabilization control or watchdog timer (software reset) function

Watch Timer

- Time interval generation: 3.91 ms, 0.5 s at 32,768 Hz
- Four frequency outputs to BUZ pin
- Clock source generation for LCD

Timers and Timer/Counters

- One 8-bit timer/counter (Timer 0) with three operating modes: Interval, Capture, and PWM
- One 16-bit timer/counter (Timer 1) with two 8-bit timer/counter modes

LCD Controller/Driver

- Up to 32 segment pins
- 3, 4, and 8 common selectable
- Choice of duty cycle
- All dots can be switched on/off
- Internal resistor circuit for LCD bias

Serial Port

- One synchronous SIO

A/D Converter

- 8-bit conversion resolution \times 4 channel
- 34 μ s conversion time (4 MHz CPU clock, fxx/4)

Oscillation Sources

- Crystal, ceramic, or RC for main system clock
- Crystal or external oscillator for subsystem clock
- Main system clock frequency: 8 MHz
- Subsystem clock frequency: 32.768 kHz

Power-down Modes

- Main idle mode (only CPU clock stops)
- Sub idle mode
- Stop mode (main/sub system oscillation stops)

Operating Temperature Range

- -40°C to $+85^{\circ}\text{C}$

Operating Voltage Range

- 2.0 V to 5.5 V at 32 kHz (sub clock)-6 MHz (main clock)
- 2.2 V to 5.5 V at 8 MHz

Package Type

- 80-pin TQFP, 80-pin QFP

BLOCK DIAGRAM

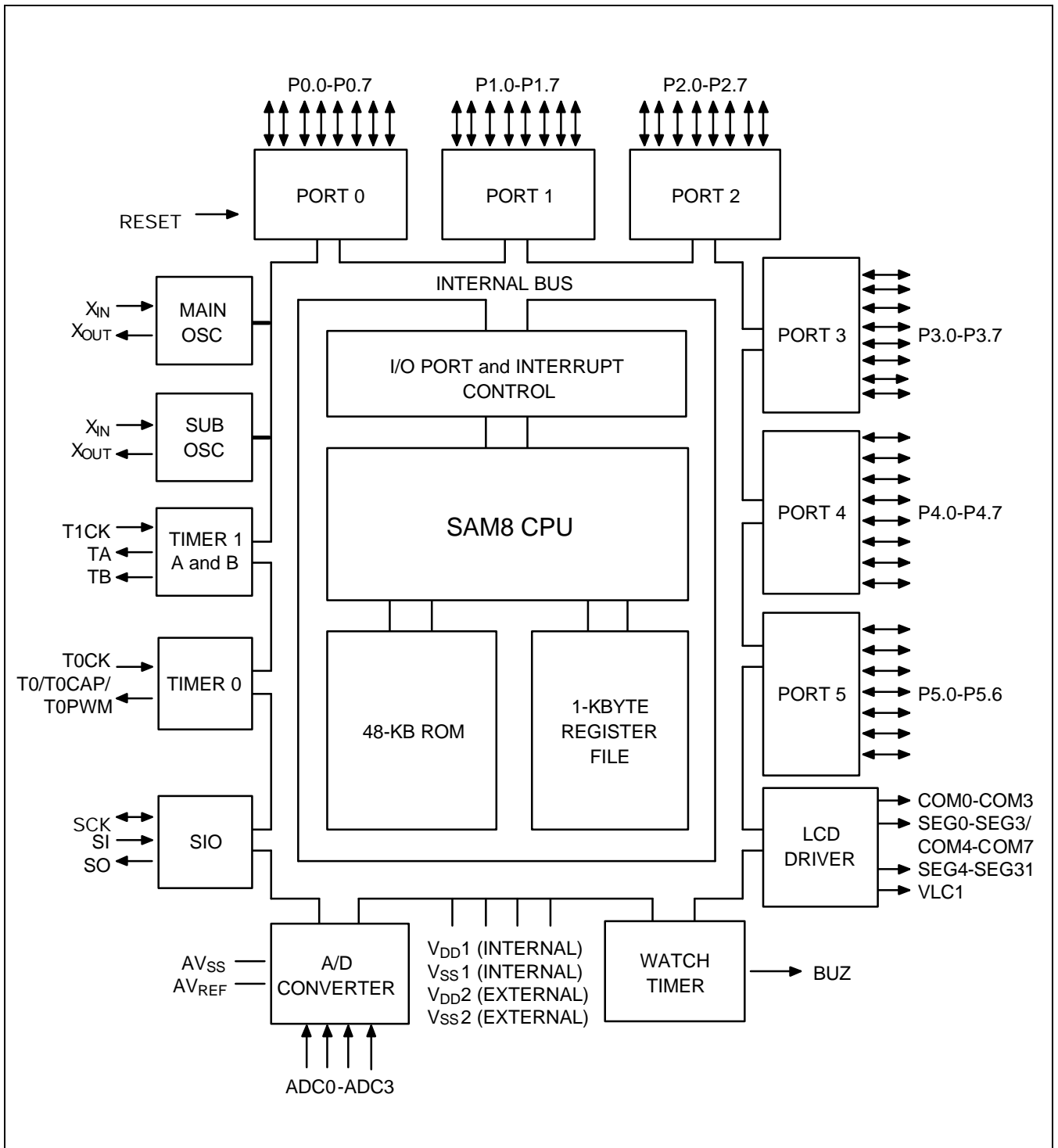


Figure 1-1. S3C821A Simplified Block Diagram

PIN ASSIGNMENTS

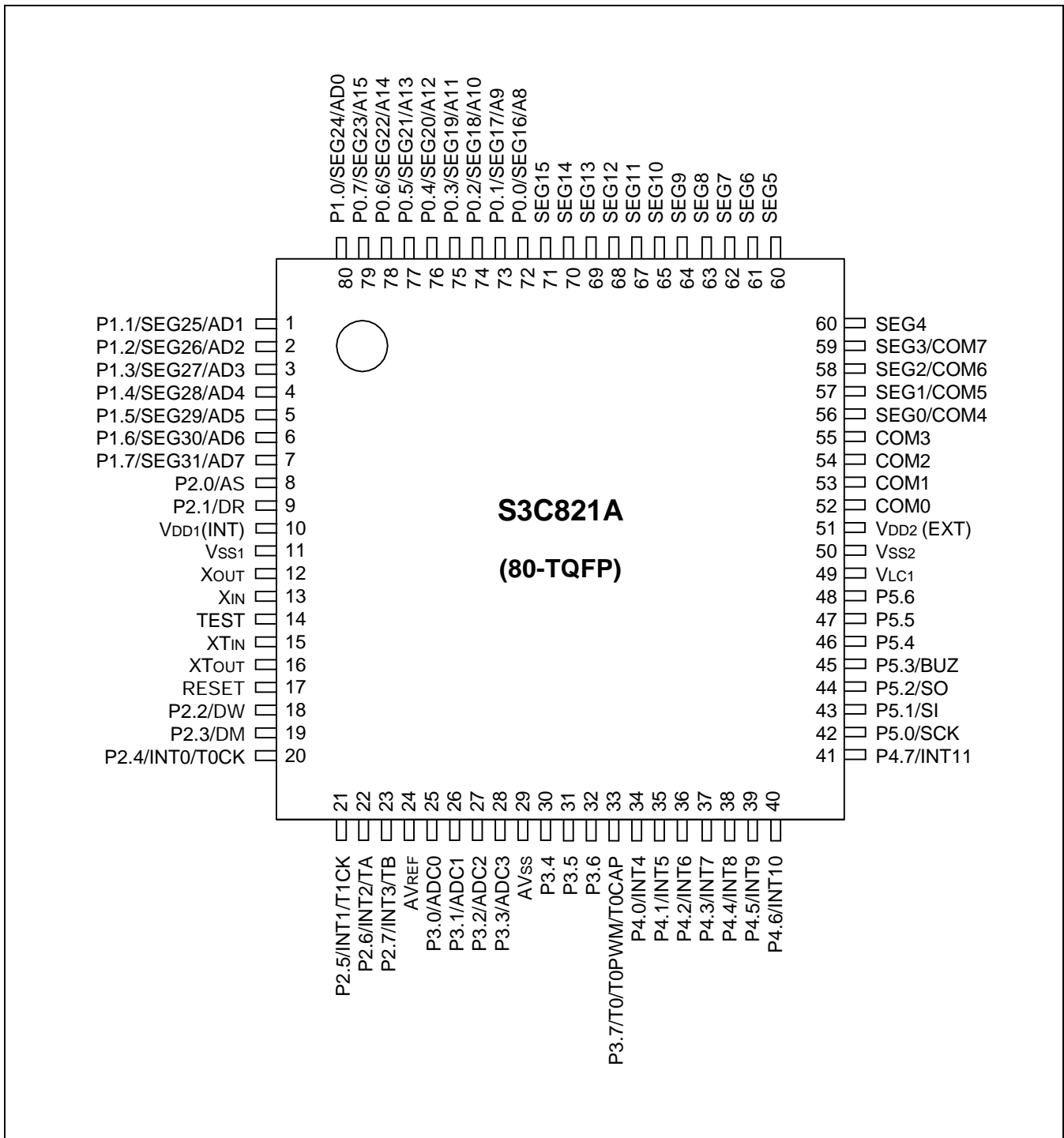


Figure 1-2. S3C821A Pin Assignments (80-TQFP-1212)

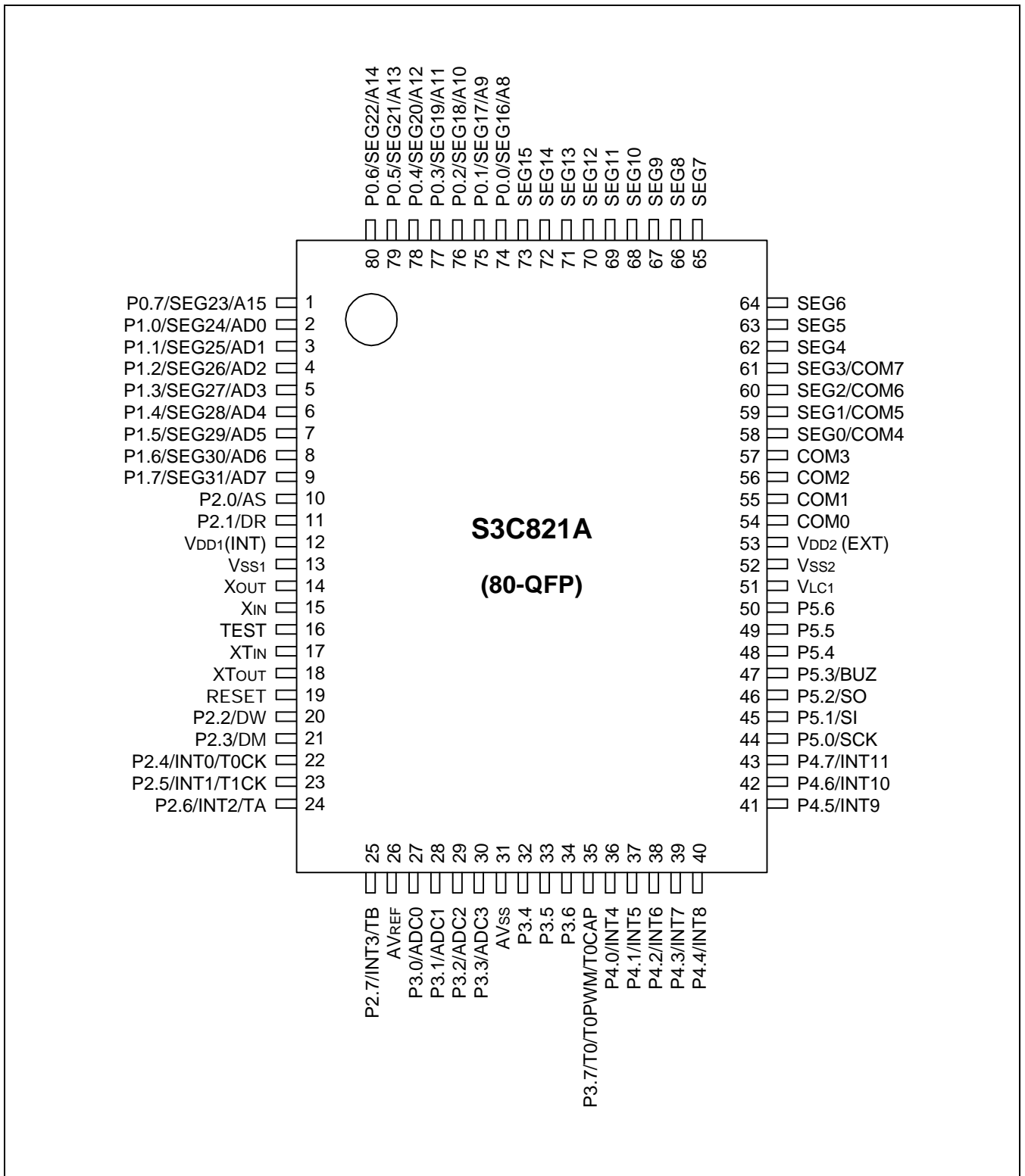


Figure 1-3. S3C821A Pin Assignments (80-QFP-1420C)

PIN DESCRIPTIONS

Table 1-1. S3C821A Pin Descriptions

Pin Names	Pin Type	Pin Description	Circuit Type	Pin Numbers (note)	Share Pins
P0.0–P0.7	I/O	4-bit-programmable I/O port. Pull-up resistors and open-drain outputs are software assignable. Pull-up resistors are automatically disabled for output pins. Configurable as LCD segments/ external interface address and data lines	H-32	72–79 (74-80, 1)	SEG16/A8 – SEG23/A15
P1.0–1.7	I/O	4-bit-programmable I/O port. Pull-up resistors and open-drain outputs are software assignable. Pull-up resistors are automatically disabled for output pins. Configurable as LCD segments/ external interface address and data lines	H-32	80, 1–7 (2-9)	SEG24/AD0 – SEG31/AD7
P2.0 P2.1 P2.2 P2.3 P2.4 P2.5 P2.6 P2.7	I/O	1-bit-programmable I/O port. Pull-up resistors are software assignable, and automatically disabled for output pins. P2.0–P2.3 can alternately be used as external interface lines. P2.4–P2.7 are configurable as alternate functions or external interrupts at falling edge with noise filters.	D-4	8 (10) 9 (11) 18 (20) 19 (21) 20 (22) 21 (23) 22 (24) 23 (25)	AS DR DW DM INT0/T0CK INT1/T1CK INT2/TA INT3/TB
P3.0–P3.3 P3.4–P3.6 P3.7	I/O	1-bit-programmable I/O port. Pull-up resistors are software assignable, and automatically disabled for output pins. P3.0–P3.3 can alternately be used as ADC. P3.7 is configurable as an alternate function.	F-16 D-4 D-4	25–28 (27–30) 30–32 (32–34) 33 (35)	ADC0–ADC3 T0/T0PWM/ T0CAP
P4.0–P4.7	I/O	1-bit-programmable I/O port. Pull-up resistors and open-drain outputs are software assignable. Pull-up resistors are automatically disabled for output pins. P4.0–P4.7 are configurable as external interrupts at a selectable edge with noise filters.	E-4	34–41 (36–43)	INT4–INT11
P5.0 P5.1 P5.2 P5.3 P5.4–P5.6	I/O	1-bit-programmable I/O port. Pull-up resistors are software assignable, and automatically disabled for output pins. P5.0–P5.3 are configurable as alternate functions. If SCK and SI are used as input, these pins have noise filters.	D-4	42 (44) 43 (45) 44 (46) 45 (47) 46–48 (48–50)	SCK SI SO BUZ

NOTE: Parentheses indicate pin number for 80-QFP package.

Table 1-1. S3C821A Pin Descriptions (Continued)

Pin Names	Pin Type	Pin Description	Circuit Type	Pin Numbers ^(note)	Share Pins
V_{SS1} , V_{DD1}	–	Power input pins for internal power block	–	10, 11 (12, 13)	–
X_{OUT} , X_{IN}	–	Main oscillator pins	–	12, 13 (14, 15)	–
TEST	–	Chip test input pin Hold GND when the device is operating	–	14 (16)	–
XT_{IN} , XT_{OUT}	–	Sub oscillator pins for sub-system clock	–	15, 16 (17,18)	–
RESET	I	RESET signal input pin. Schmitt trigger input with internal pull-up resistor.	B	17 (19)	–
INT0–INT3	I/O	External interrupts input with noise filter.	D-4	20–23 (22–25)	P2.4–P2.7
T0CK	I/O	8Bit Timer 0 external clock input.	D-4	20 (22)	P2.4
T1CK	I/O	Timer 1/A external clock input.	D-4	21 (23)	P2.5
TA	I/O	Timer 1/A clock output	D-4	22 (24)	P2.6
TB	I/O	Timer B clock output	D-4	23 (25)	P2.7
T0	I/O	Timer 0 clock output	D-4	33 (35)	P3.7
T0PWM	I/O	Timer 0 PWM output	D-4	33 (35)	P3.7
T0CAP	I/O	Timer 0 capture input	D-4	33 (35)	P3.7
ADC0–ADC3	I/O	Analog input pins for A/D converts module	F-16	25–28 (27–30)	P3.0–P3.3
AV_{REF} , AV_{SS}	–	A/D converter reference voltage and ground	–	24, 29 (26, 31)	–
INT4–INT11	I/O	External interrupts input with noise filter.	E-4	34–41 (36–43)	P4.0–P4.7
BUZ	I/O	Buzzer signal output	D-4	45 (47)	P5.3
SCK, SI, SO	I/O	Serial clock, serial data input, serial data output	D-4	42–44 (44–46)	P5.0–P5.2
V_{LC1}	–	LCD bias voltage input pins	–	49 (51)	–
V_{SS2} , V_{DD2}	–	Power input pins for external power block	–	50, 51 (52, 53)	–
COM0–COM3	O	LCD Common signal output	H-30	52–55 (54–57)	–
SEG0–SEG3 (COM4–COM7)	O	LCD Common or Segment signal output	H-31	56–59 (58–61)	–
SEG4–SEG15	O	LCD segment signal output	H-29	60–71 (62–73)	–

NOTE: Parentheses indicate pin number for 80-QFP package.

Table 1-1. S3C821A Pin Descriptions (Continued)

Pin Names	Pin Type	Pin Description	Circuit Type	Pin Numbers	Share Pins
SEG16–SEG23	I/O	LCD segment signal output	H-32	72–79 (74–80, 1)	P0.0–P0.7
SEG24–SEG31	I/O	LCD segment signal output	H-32	80, 1–7 (2–9)	P1.0–P1.7
A8–A15	I/O	External interface address lines	H-32	72–79 (74–80, 1)	P0.0–P0.7
AD0–AD7	I/O	External interface address/data lines	H-32	80, 1–7 (2–9)	P1.0–P1.7
AS	I/O	Address strobe	D-4	8 (10)	P2.0
DR	I/O	Data read	D-4	9 (11)	P2.1
DW	I/O	Data write	D-4	18 (20)	P2.2
DM	I/O	Data memory select	D-4	19 (21)	P2.3

NOTE: Parentheses indicate pin number for 80-QFP package.

PIN CIRCUITS

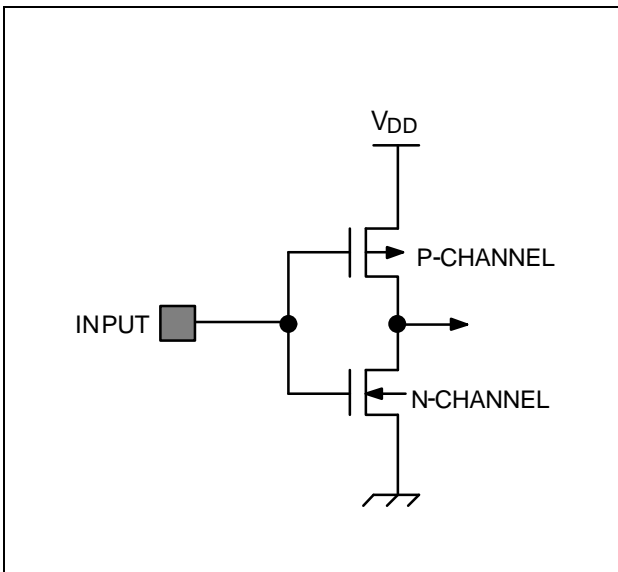


Figure 1-4. Pin Circuit Type A

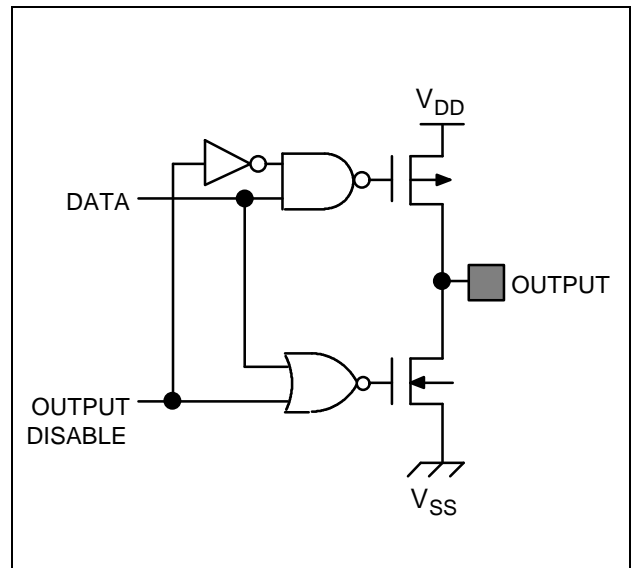


Figure 1-6. Pin Circuit Type C

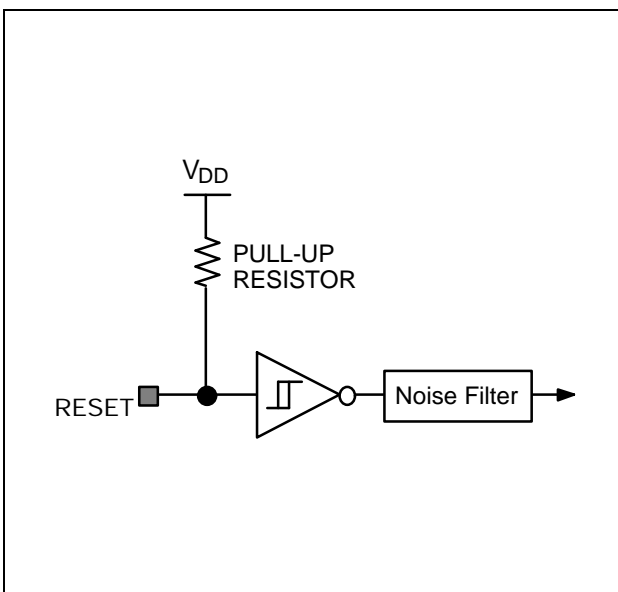


Figure 1-5. Pin Circuit Type B

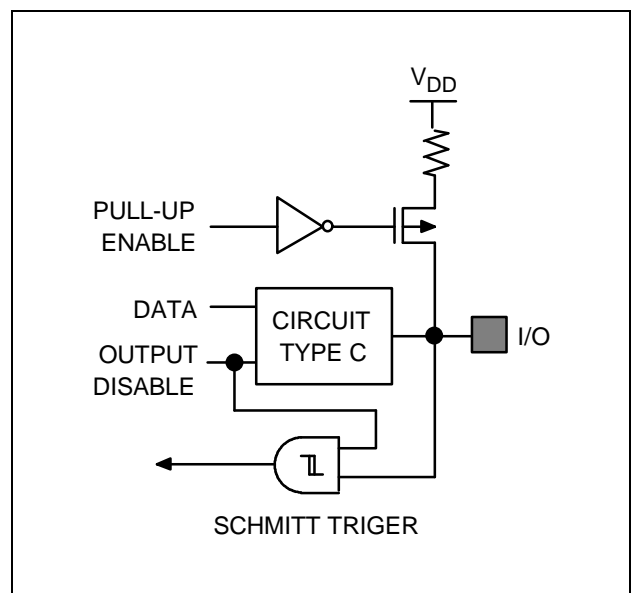


Figure 1-7. Pin Circuit Type D-4

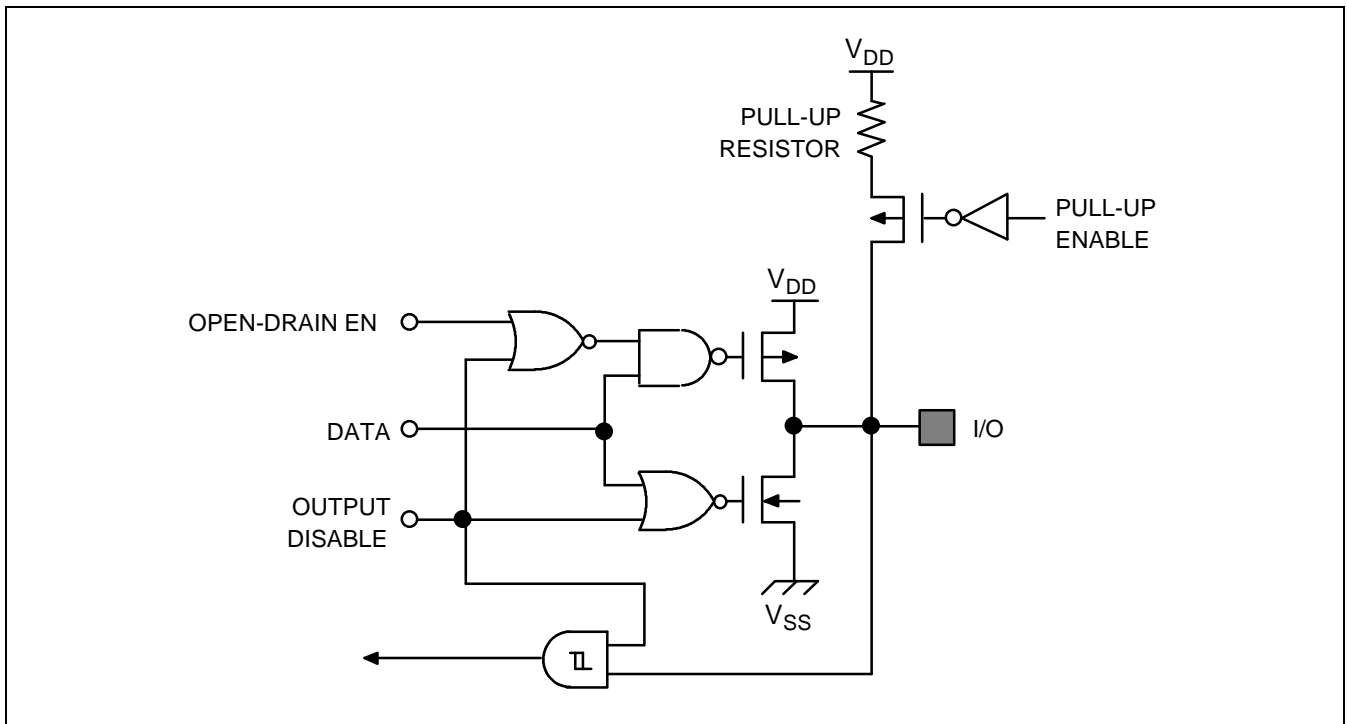


Figure 1-8. Pin Circuit Type E-4

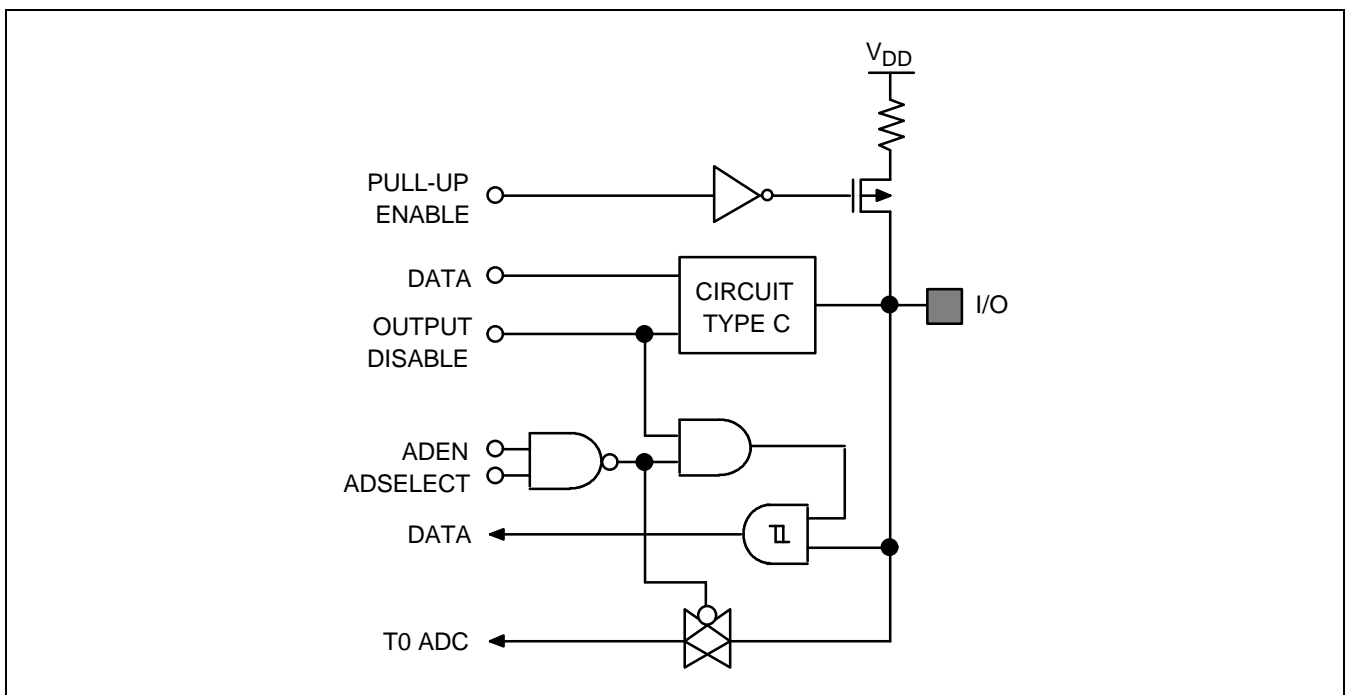


Figure 1-9. Pin Circuit Type F-16

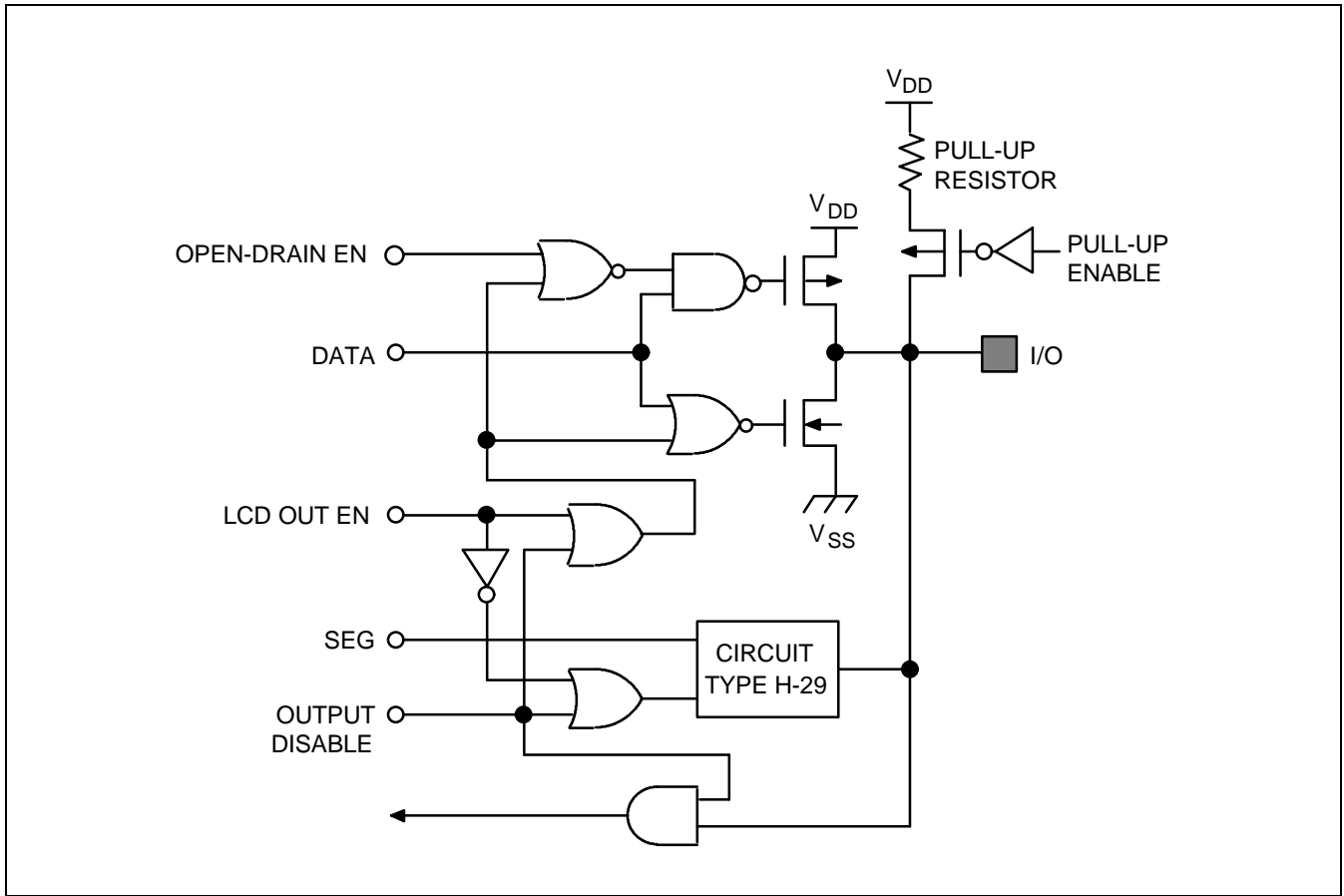


Figure 1-13. Pin Circuit Type H-32

2 ADDRESS SPACES

OVERVIEW

The S3C821A microcontroller has three types of address space:

- Internal program memory (ROM)
- Internal register file
- External data memory

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

The S3C821A has an internal 48-Kbyte mask-programmable ROM. An external data memory interface is implemented.

The 256-byte physical register space is expanded into an addressable area of 320 bytes using addressing modes.

A 32-byte LCD display register file is implemented.

There are 1,125 mapped registers in the internal register file. Of these, 1,040 are for general-purpose. (This number includes a 16-byte working register common area used as a "scratch area" for data operations, four 192-byte prime register areas, and four 64-byte areas (Set 2)). Nineteen 8-bit registers are used for the CPU and the system control, and 34 registers are mapped for peripheral controls and data registers. Eleven register locations are not mapped.

PROGRAM MEMORY (ROM)

Program memory (ROM) stores program codes or table data. The S3C821A has 48 K bytes of internal mask-programmable program memory. The program memory address range is therefore 0H–BFFFH (see Figure 2-1).

The first 256 bytes of the ROM (0H–0FFH) are reserved for interrupt vector addresses. Unused locations in this address range can be used as normal program memory. If you use the vector address area to store a program code, be careful not to overwrite the vector addresses stored in these locations.

The ROM address at which a program execution starts after a reset is 0100H.

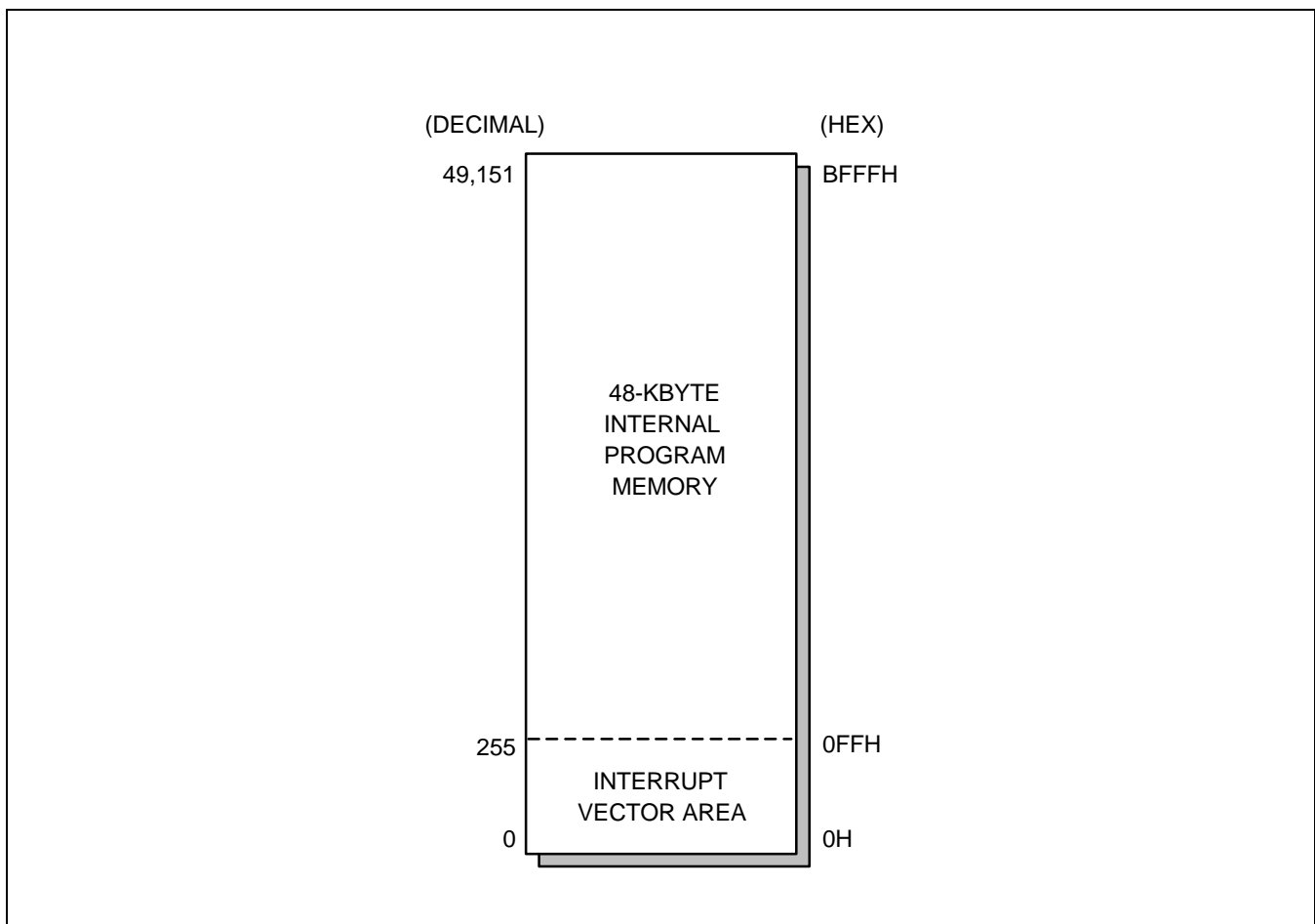


Figure 2-1. Program Memory Address Space

REGISTER ARCHITECTURE

In the S3C821A implementation, the upper 64-byte area of register files is expanded two 64-byte areas, called *set 1* and *set 2*. The upper 32-byte area of set 1 is further expanded two 32-byte register banks (bank 0 and bank 1), and the lower 32-byte area is a single 32-byte common area. In addition, set 2 is logically expanded four separately addressable register pages, *page 0–page 3*.

The total number of addressable 8-bit registers is 1125. Of these 1125 registers, 19 bytes are for CPU and system control registers, 32 bytes are for LCD data registers, 34 bytes are for peripheral control and data registers, 16 bytes are used as a shared working registers, and 1024 registers are for general-purpose use.

You can always address set 1 register locations, regardless of which of the four register pages is currently selected. Set 1 locations, however, can only be addressed using direct addressing modes.

The extension of register space into separately addressable areas (sets, banks, and pages) is supported by various addressing mode restrictions, the select bank instructions (SB0 and SB1) and the register page pointer (PP).

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

Table 2-1. S3C821A Register Type Summary

Register Type	Number of Bytes
General-purpose registers (including the 16-byte common working register area, four 192-byte prime register area, and four 64-byte set 2 area)	1,040
LCD data registers	32
CPU and system control registers	19
Mapped clock, peripheral, I/O control, and data registers	34
Total Addressable Bytes	1,125

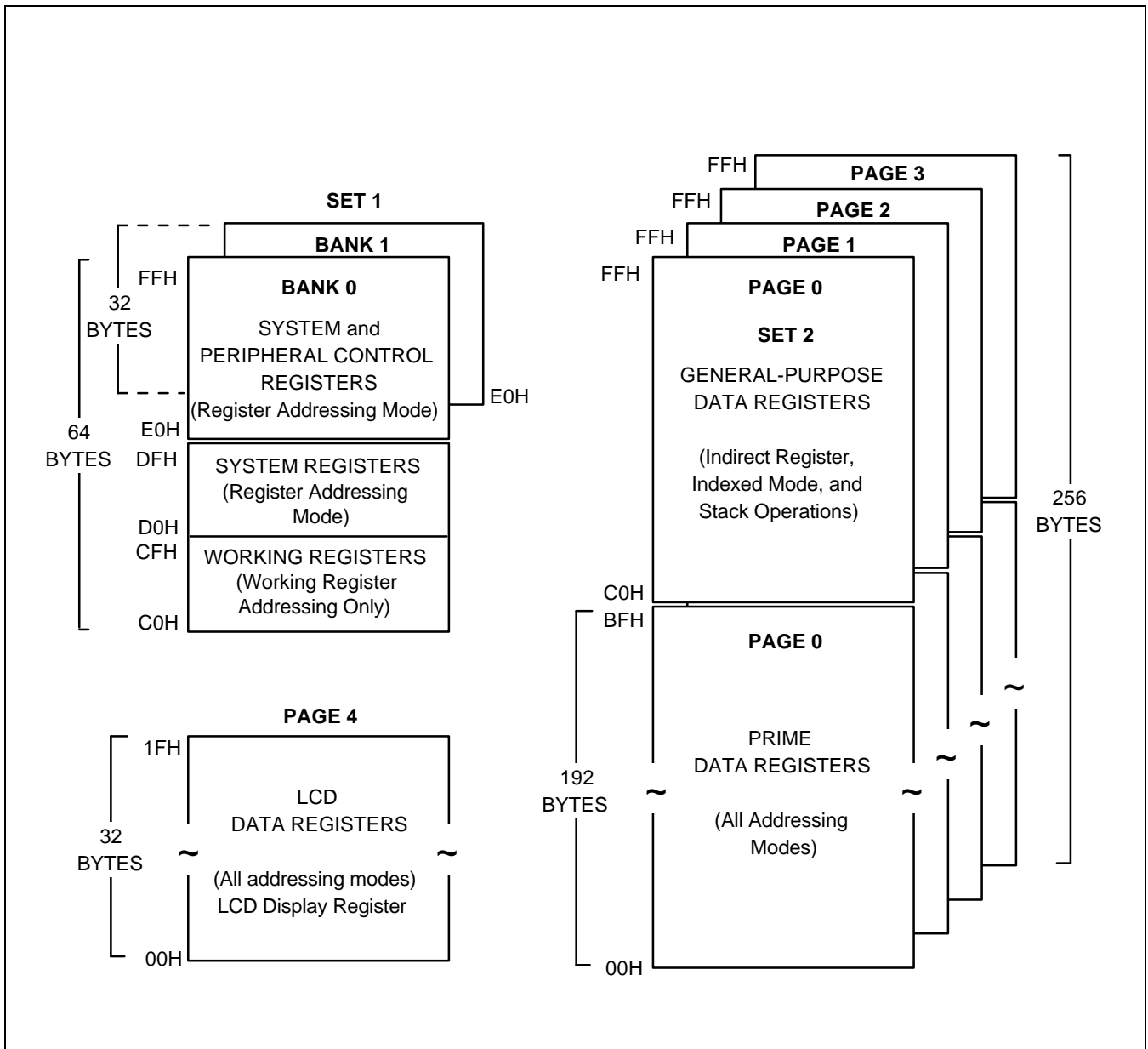


Figure 2-2. Internal Register File Organization

REGISTER PAGE POINTER (PP)

The S3C8-series architecture supports the logical expansion of the physical 256-byte internal register file (using an 8-bit data bus) into as many as 16 separately addressable register pages. Page addressing is controlled by the register page pointer (PP, DFH). In the S3C821A microcontroller, a paged register file expansion is implemented for LCD data registers, and the register page pointer must be changed to address other pages.

After a reset, the page pointer's source value (lower nibble) and the destination value (upper nibble) are always "0000B", automatically selecting page 0 as the source and destination page for register addressing.

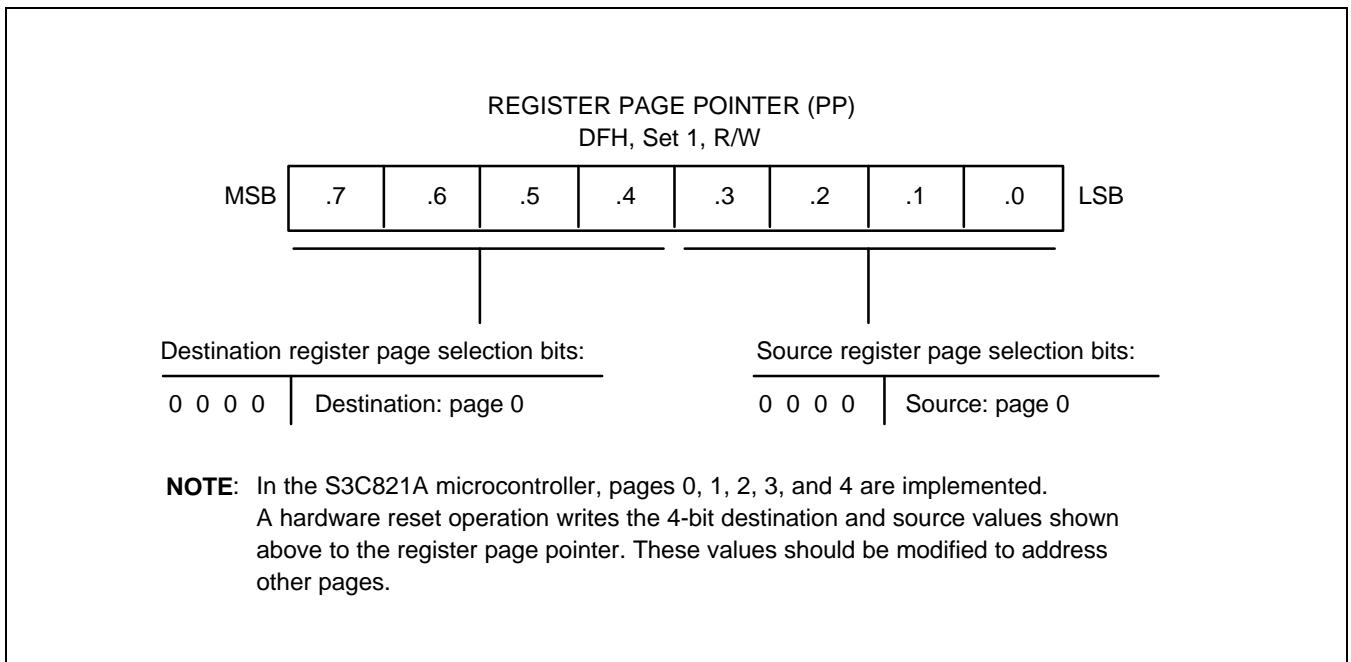


Figure 2-3. Register Page Pointer (PP)

PROGRAMMING TIP — Using the Page Pointer for RAM clear (Page 0, Page 1)

```

RAMCLO
LD      PP,#00H           ; Destination ← 0, Source ← 0
SRP    #0C0H
LD      R0,#0FFH        ; Page0 RAM clear starts
CLR    @R0
DJNZ   R0,RAMCLO
CLR    @R0               ; R0 = 00H

RAMCL1
LD      PP,#10H          ; Destination ← 1, Source ← 0
LD      R0,#0FFH        ; Page1 RAM clear starts
CLR    @R0
DJNZ   R0,RAMCL1
CLR    @R0               ; R0 = 00H

```

NOTE: You should refer to page 6-39 and use DJNZ instruction properly when DJNZ instruction is used in your program.

REGISTER SET 1

The term *set 1* refers to the upper 64 bytes of the register file, locations C0H–FFH.

The upper 32-byte area of this 64-byte space (E0H–FFH) is expanded two 32-byte register banks, *bank 0* and *bank 1*. The set register bank instructions, SB0 or SB1, are used to address one bank or the other. A hardware reset operation always selects bank 0 addressing.

The upper 32-byte area (bank 0 and bank 1, E0H–FFH) of set 1 contains 37 mapped system and peripheral control registers. The lower 32-byte area contains 16 system registers (D0H–DFH) and a 16-byte common working register area (C0H–CFH). You can use the common working register area as a “scratch” area for data operations being performed in other areas of the register file.

Registers in set 1 locations are directly accessible at all times using Register addressing mode. The 16-byte working register area can only be accessed using working register addressing (For more information about working register addressing, please refer to Chapter 3, “Addressing Modes.”)

REGISTER SET 2

The same 64-byte physical space that is used for set 1 locations C0H–FFH is logically duplicated to add another 64 bytes of register space. This expanded area of the register file is called *set 2*. For the S3C821A, the set 2 address range (C0H–FFH) is accessible on pages 0-3.

The logical division of set 1 and set 2 is maintained by means of addressing mode restrictions. You can use only Register addressing mode to access set 1 locations. In order to access registers in set 2, you must use Register Indirect addressing mode or Indexed addressing mode.

The set 2 register area on page 0 is commonly used for stack operations.

PRIME REGISTER SPACE

The lower 192 bytes (00H–BFH) of the S3C821A's four 256-byte register pages is called *prime register area*. Prime registers can be accessed using any of the seven addressing modes (see Chapter 3, "Addressing Modes.")

The prime register area on page 0 is immediately addressable following a reset. In order to address prime registers on pages 0, 1, 2, or 3 you must set the register page pointer (PP) to the appropriate source and destination values.

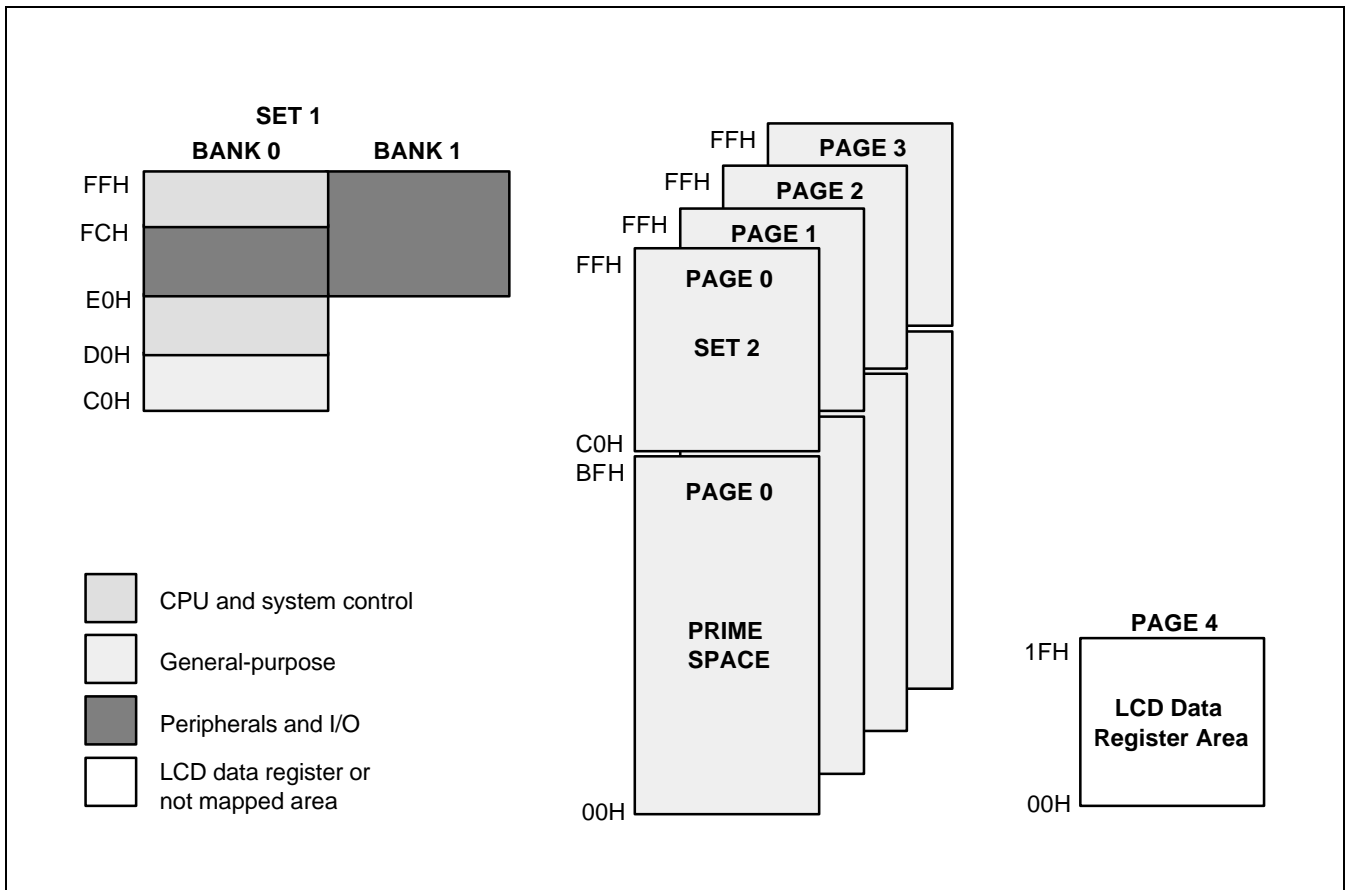


Figure 2-4. Set 1, Set 2, Prime Area Register, and LCD Data Register Map

WORKING REGISTERS

Instructions can access specific 8-bit registers or 16-bit register pairs using either 4-bit or 8-bit address fields. When 4-bit working register addressing is used, the 256-byte register file can be seen by the programmer as one that consists of thirty-two 8-byte register groups or "slices." Each slice comprises of eight 8-bit registers.

Using the two 8-bit register pointers, RP1 and RP0, two working register slices can be selected at any one time to form a 16-byte working register block. Using the register pointers, you can move this 16-byte register block anywhere in the addressable register file, except for the set 2 area.

The terms *slice* and *block* are used in this manual to help you visualize the size and relative locations of selected working register spaces:

- One working register *slice* is 8 bytes (eight 8-bit working registers, R0–R7 or R8–R15)
- One working register *block* is 16 bytes (sixteen 8-bit working registers, R0–R15)

All the registers in an 8-byte working register slice have the same binary value for their five most significant address bits. This makes it possible for each register pointer to point to one of the 32 slices in the register file. The base addresses for the two selected 8-byte register slices are contained in register pointers RP0 and RP1.

After a reset, RP0 and RP1 always point to the 16-byte common area in set 1 (C0H–CFH).

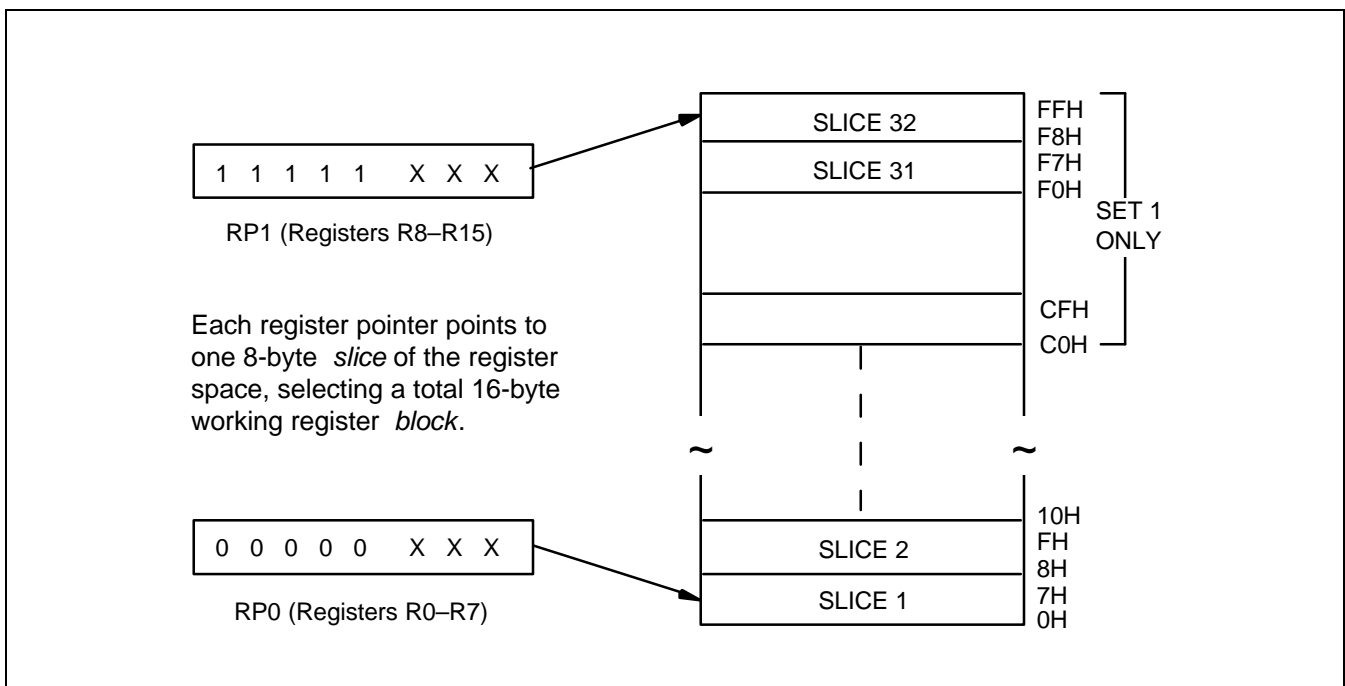


Figure 2-5. 8-Byte Working Register Areas (Slices)

USING THE REGISTER POINTERS

Register pointers RP0 and RP1, mapped to addresses D6H and D7H in set 1, are used to select two movable 8-byte working register slices in the register file. After a reset, they point to the working register common area: RP0 points to addresses C0H–C7H, and RP1 points to addresses C8H–CFH.

To change a register pointer value, you load a new value to RP0 and/or RP1 using an SRP or LD instruction (see Figures 2-6 and 2-7).

With working register addressing, you can only access those two 8-bit slices of the register file that are currently pointed to by RP0 and RP1. You cannot, however, use the register pointers to select a working register space in set 2, C0H–FFH, because these locations can be accessed only using the Indirect Register or Indexed addressing modes.

The selected 16-byte working register block usually consists of two contiguous 8-byte slices. As a general programming guideline, it is recommended that RP0 point to the "lower" slice and RP1 point to the "upper" slice (see Figure 2-6). In some cases, it may be necessary to define working register areas in different (non-contiguous) areas of the register file. In Figure 2-7, RP0 points to the "upper" slice and RP1 to the "lower" slice.

Because a register pointer can point to either of the two 8-byte slices in the working register block, you can flexibly define the working register area to support program requirements.

PROGRAMMING TIP — Setting the Register Pointers

SRP	#70H	; RP0 ← 70H, RP1 ← 78H
SRP1	#48H	; RP0 ← no change, RP1 ← 48H,
SRP0	#0A0H	; RP0 ← A0H, RP1 ← no change
CLR	RP0	; RP0 ← 00H, RP1 ← no change
LD	RP1,#0F8H	; RP0 ← no change, RP1 ← F8H

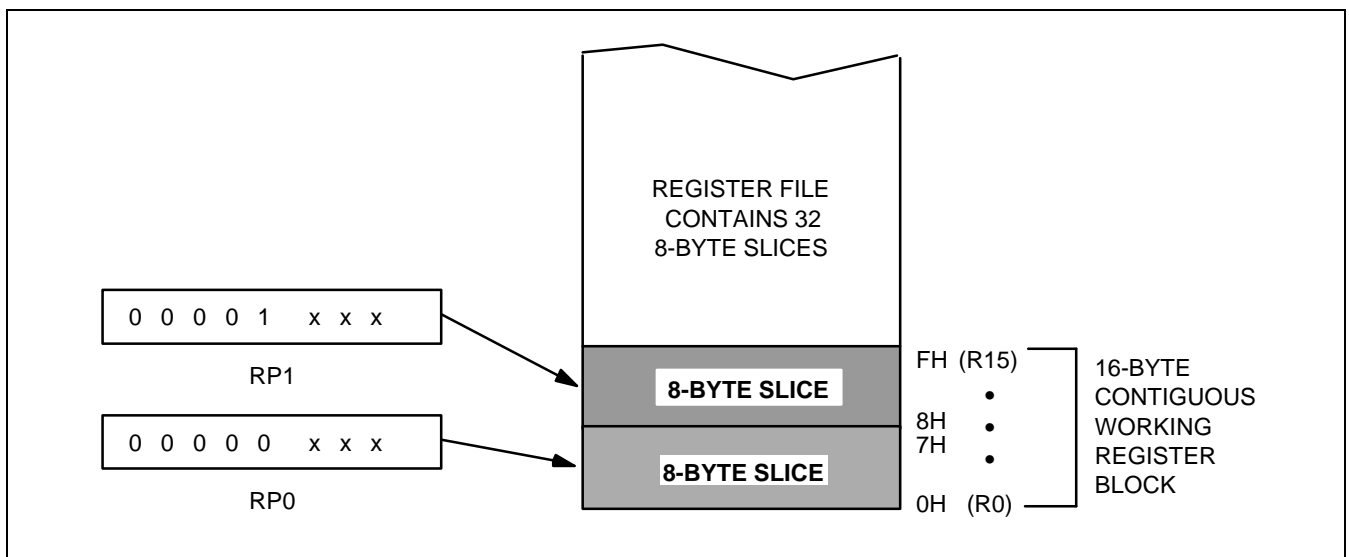


Figure 2-6. Contiguous 16-Byte Working Register Block

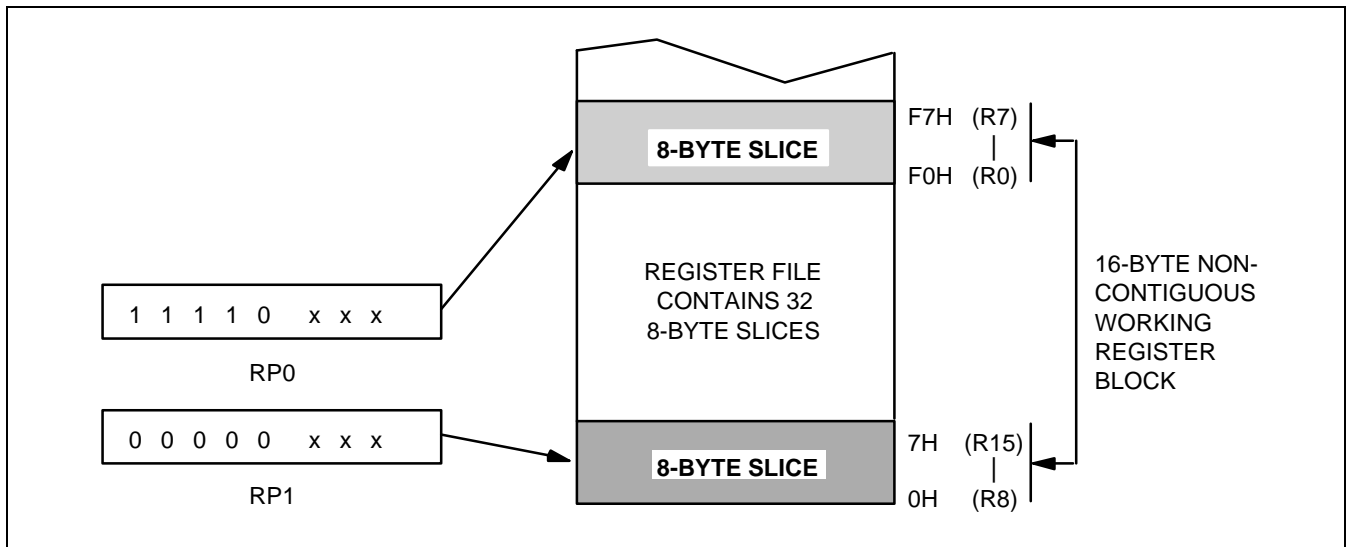


Figure 2-7. Non-Contiguous 16-Byte Working Register Block

 **Programming Tip — Using the RPs to Calculate the Sum of a Series of Registers**

Calculate the sum of registers 80H-85H using the register pointer. The register addresses from 80H through 85H contain the values 10H, 11H, 12H, 13H, 14H, and 15 H, respectively:

```
SRP0    #80H           ; RP0 ← 80H
ADD     R0,R1          ; R0 ← R0 + R1
ADC     R0,R2          ; R0 ← R0 + R2 + C
ADC     R0,R3          ; R0 ← R0 + R3 + C
ADC     R0,R4          ; R0 ← R0 + R4 + C
ADC     R0,R5          ; R0 ← R0 + R5 + C
```

The sum of these six registers, 6FH, is located in the register R0 (80H). The instruction string used in this example takes 12 bytes of instruction code and its execution time is 36 cycles. If the register pointer is not used to calculate the sum of these registers, the following instruction sequence would have to be used:

```
ADD     80H,81H        ; 80H ← (80H) + (81H)
ADC     80H,82H        ; 80H ← (80H) + (82H) + C
ADC     80H,83H        ; 80H ← (80H) + (83H) + C
ADC     80H,84H        ; 80H ← (80H) + (84H) + C
ADC     80H,85H        ; 80H ← (80H) + (85H) + C
```

Now, the sum of the six registers is also located in register 80H. However, this instruction string takes 15 bytes of instruction code rather than 12 bytes, and its execution time is 50 cycles rather than 36 cycles.

REGISTER ADDRESSING

The S3C8-series register architecture provides an efficient method of working register addressing that takes full advantage of shorter instruction formats to reduce execution time.

With Register (R) addressing mode, in which the operand value is the content of a specific register or register pair, you can access any location in the register file except for set 2. With working register addressing, you use a register pointer to specify an 8-byte working register space in the register file and an 8-bit register within that space.

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

Working register addressing differs from Register addressing as it uses a register pointer to identify a specific 8-byte working register space in the internal register file and a specific 8-bit register within that space.

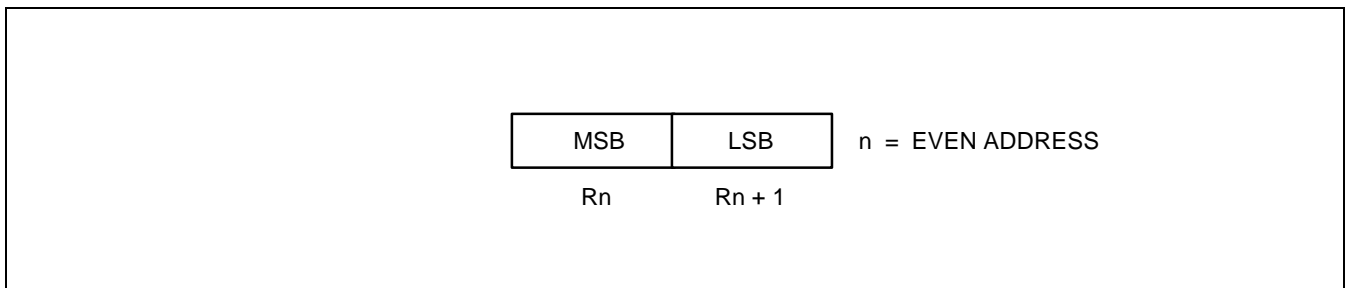


Figure 2-8. 16-Bit Register Pair

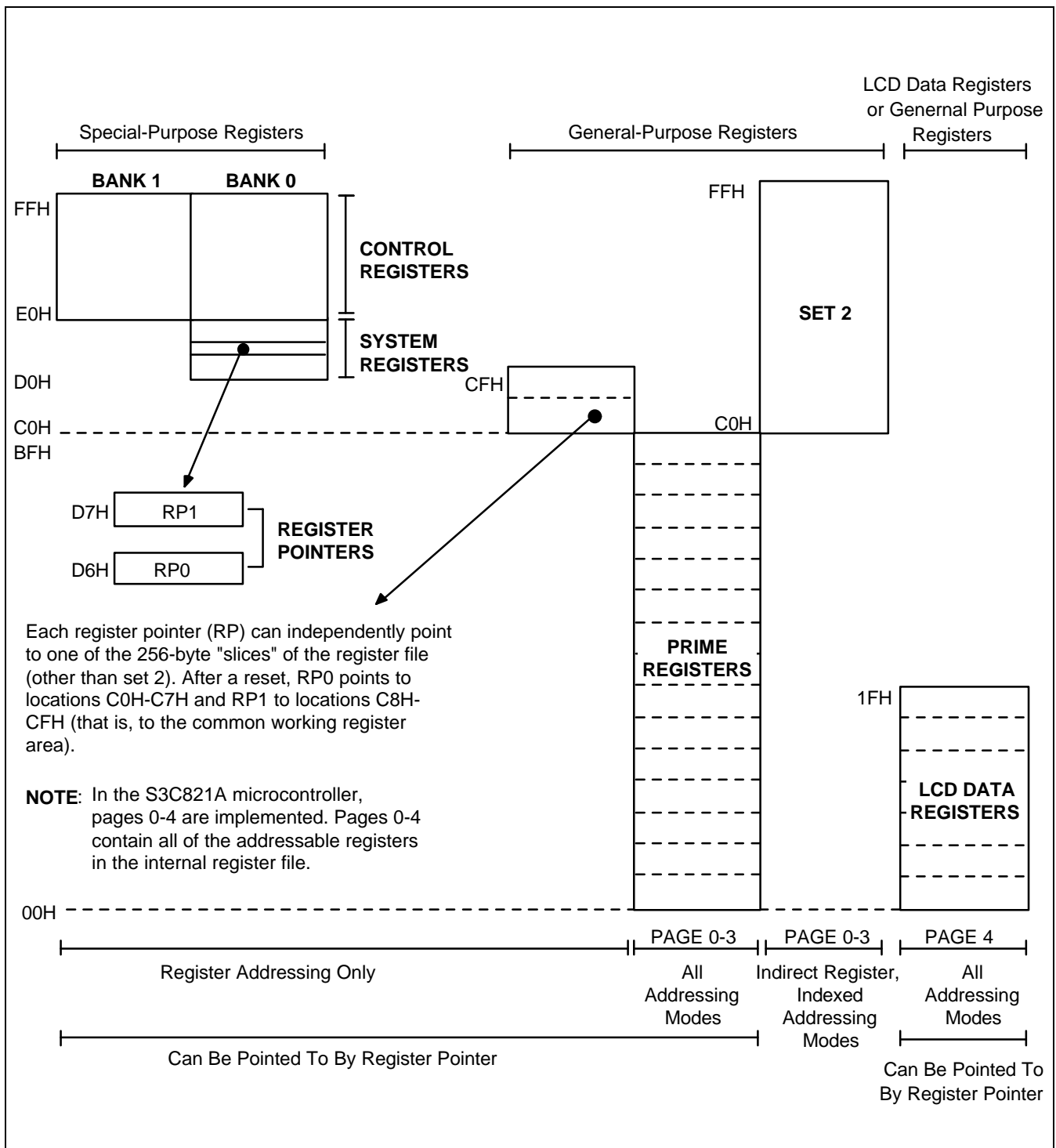


Figure 2-9. Register File Addressing

COMMON WORKING REGISTER AREA (C0H–CFH)

After a reset, register pointers RP0 and RP1 automatically select two 8-byte register slices in set 1, locations C0H-CFH, as the active 16-byte working register block:

RP0 → C0H-C7H

RP1 → C8H-CFH

This 16-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.

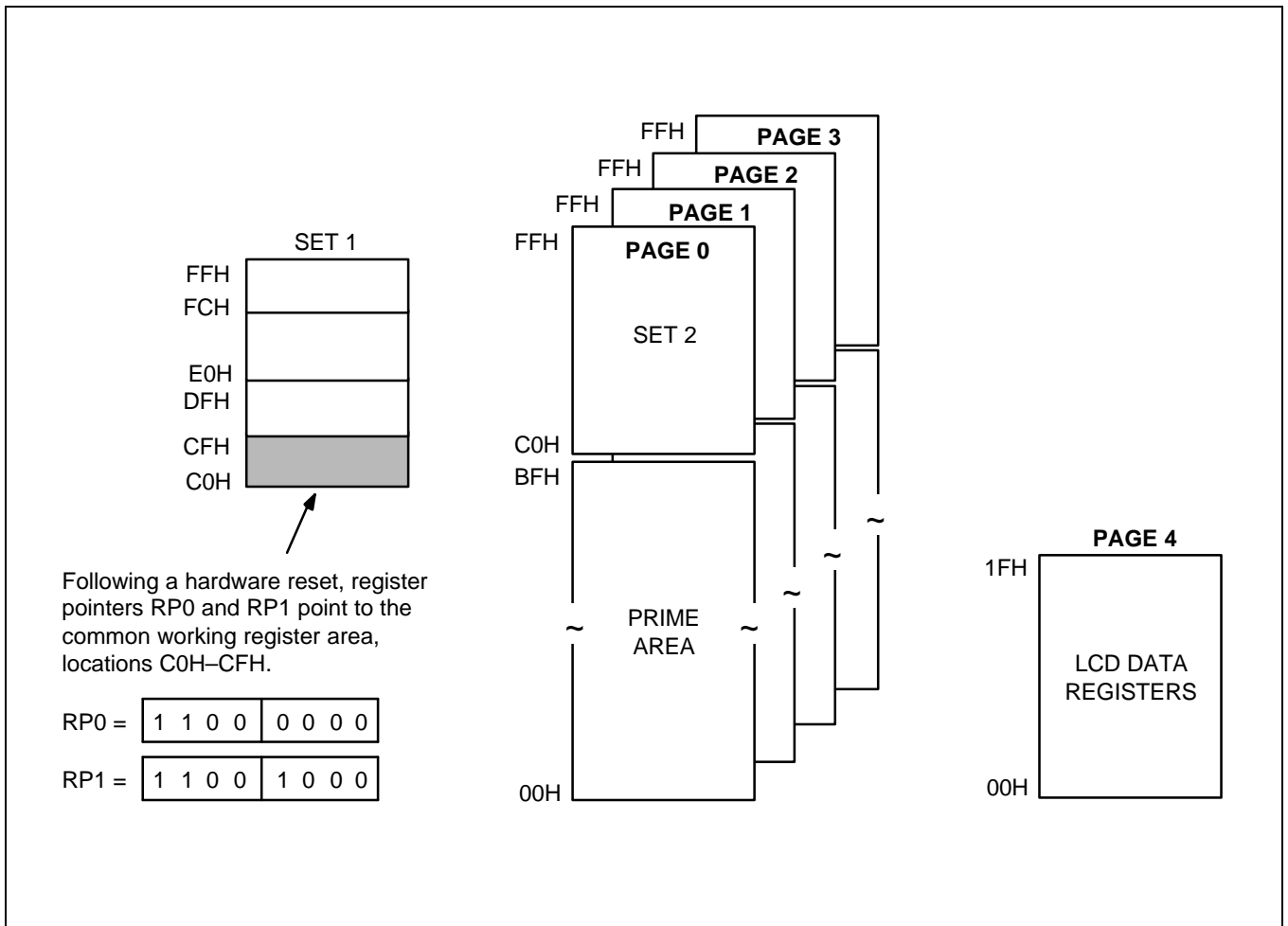


Figure 2-10. Common Working Register Area

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:

```
SRP    #0C0H
LD     R2,40H ; R2 (C2H) ← the value in location 40H
```

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

Use working register addressing instead:

```
SRP    #0C0H
ADD    R3,#45H ; R3 (C3H) ← R3 + 45H
```

4-BIT WORKING REGISTER ADDRESSING

Each register pointer defines a movable 8-byte slice of working register space. The address information stored in a register pointer serves as an addressing "window" that makes it possible for instructions to access working registers very efficiently using short 4-bit addresses. When an instruction addresses a location in the selected working register area, the address bits are concatenated in the following way to form a complete 8-bit address:

- The high-order bit of the 4-bit address selects one of the register pointers ("0" selects RP0, "1" selects RP1).
- The five high-order bits in the register pointer select an 8-byte slice of the register space.
- The three low-order bits of the 4-bit address select one of the eight registers in the slice.

As shown in Figure 2-11, the result of this operation is that the five high-order bits from the register pointer are concatenated with the three low-order bits from the instruction address to form the complete address. As long as the address stored in the register pointer remains unchanged, the three bits from the address will always point to an address in the same 8-byte register slice.

Figure 2-12 shows a typical example of 4-bit working register addressing. The high-order bit of the instruction "INC R6" is "0", which selects RP0. The five high-order bits stored in RP0 (01110B) are concatenated with the three low-order bits of the instruction's 4-bit address (110B) to produce the register address 76H (01110110B).

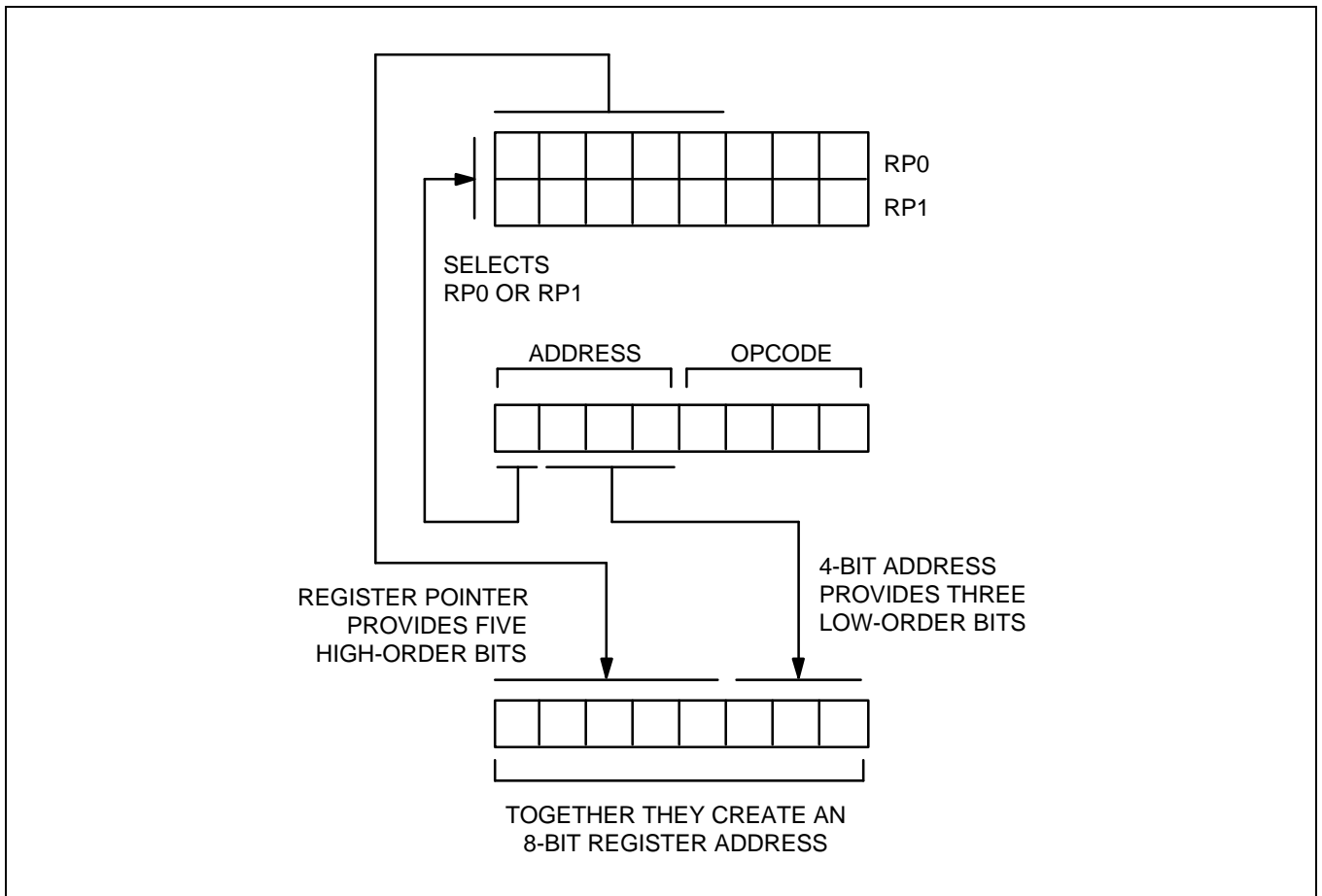


Figure 2-11. 4-Bit Working Register Addressing

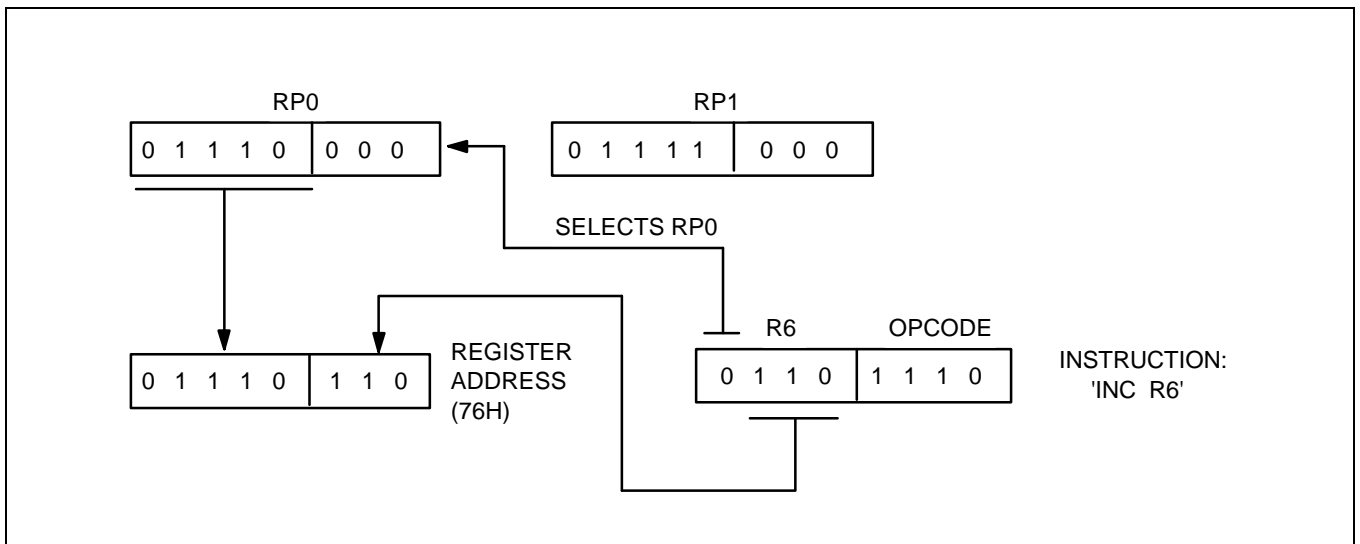


Figure 2-12. 4-Bit Working Register Addressing Example

8-BIT WORKING REGISTER ADDRESSING

You can also use 8-bit working register addressing to access registers in a selected working register area. To initiate 8-bit working register addressing, the upper four bits of the instruction address must contain the value "1100B." This 4-bit value (1100B) indicates that the remaining four bits have the same effect as 4-bit working register addressing.

As shown in Figure 2-13, the lower nibble of the 8-bit address is concatenated in much the same way as for 4-bit addressing: Bit 3 selects either RP0 or RP1, which then supplies the five high-order bits of the final address; the three low-order bits of the complete address are provided by the original instruction.

Figure 2-14 shows an example of 8-bit working register addressing. The four high-order bits of the instruction address (1100B) specify 8-bit working register addressing. Bit 4 ("1") selects RP1 and the five high-order bits in RP1 (10101B) become the five high-order bits of the register address. The three low-order bits of the register address (011) are provided by the three low-order bits of the 8-bit instruction address. The five address bits from RP1 and the three address bits from the instruction are concatenated to form the complete register address, 0ABH (10101011B).

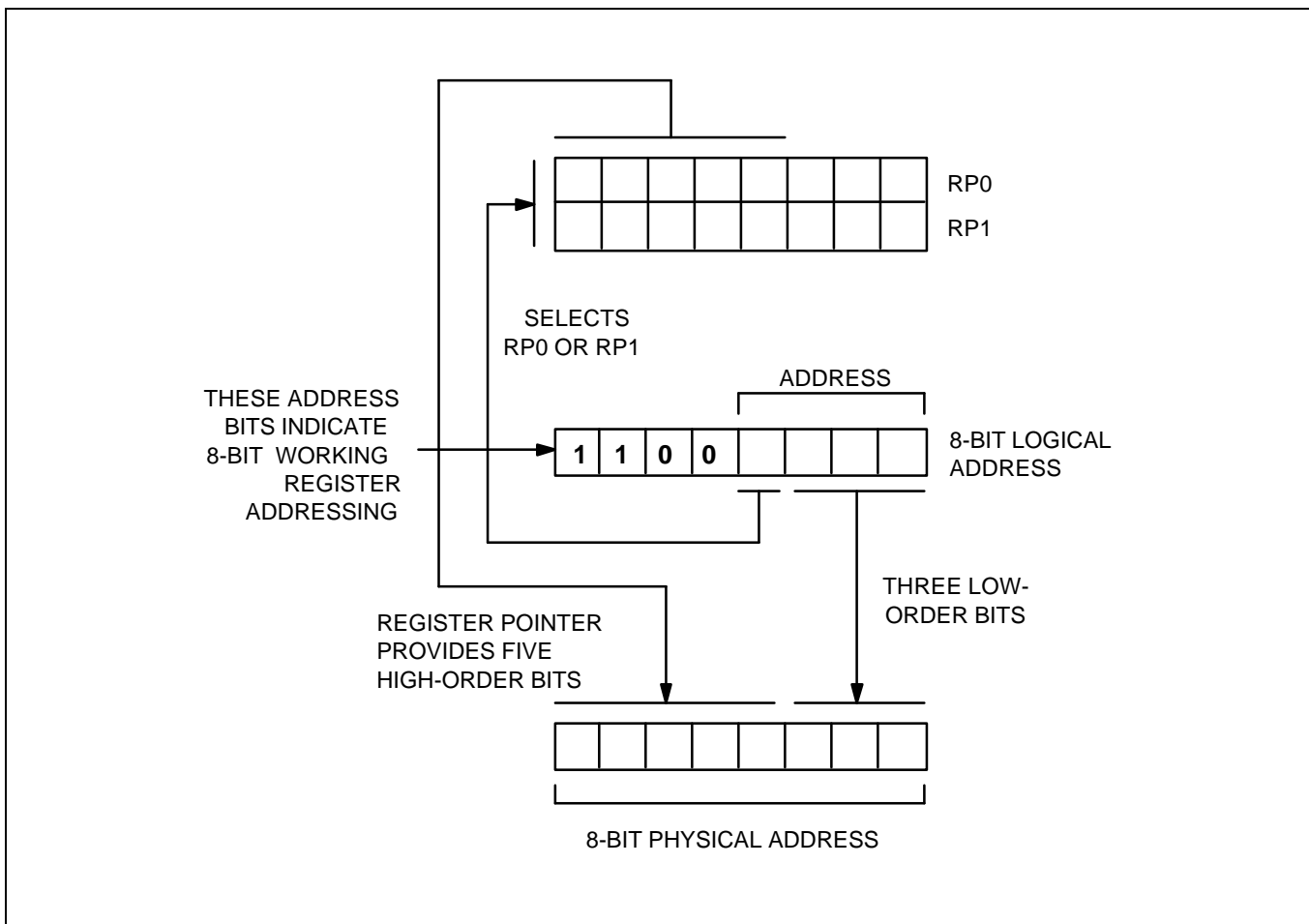


Figure 2-13. 8-Bit Working Register Addressing

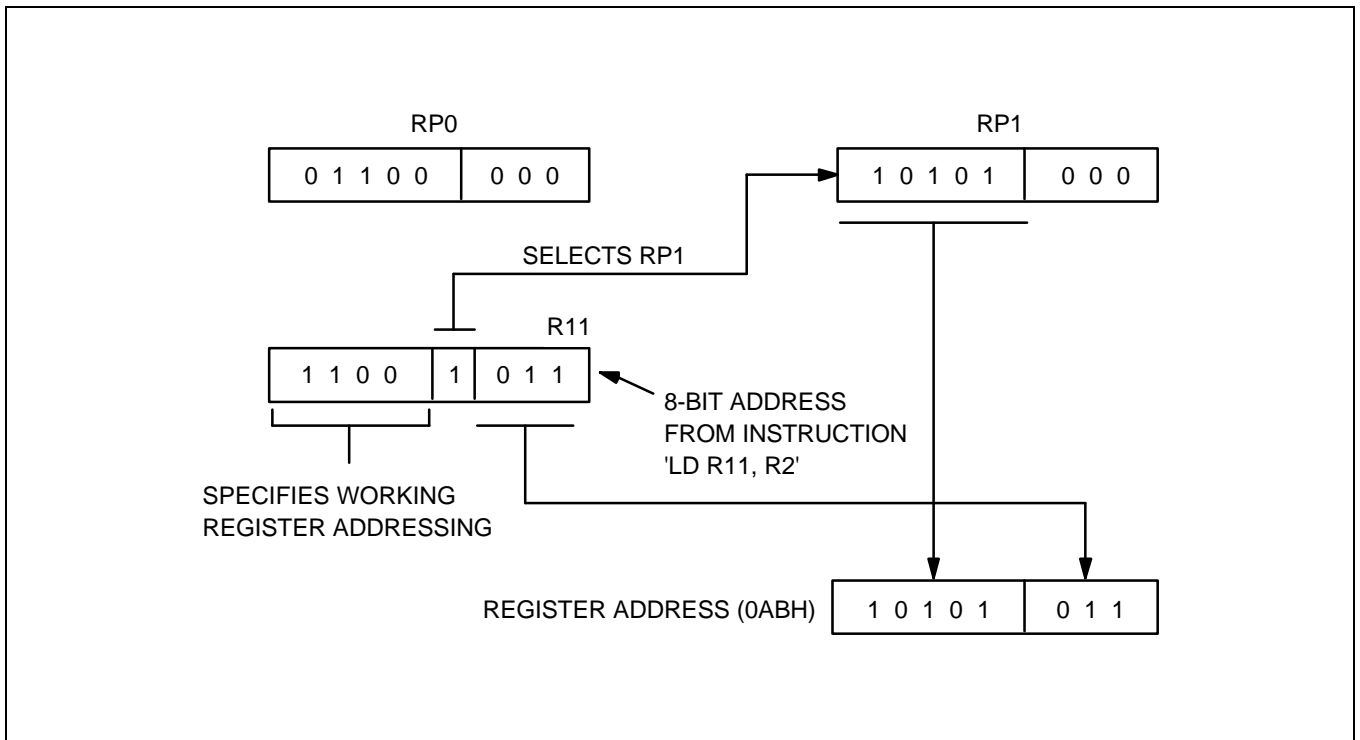


Figure 2-14. 8-Bit Working Register Addressing Example

SYSTEM AND USER STACKS

The S3C8-series microcontrollers use the system stack for data storage, subroutine calls and returns. The PUSH and POP instructions are used to control system stack operations. The S3C821A architecture supports stack operations in the internal register file.

Stack Operations

Return addresses for procedure calls, 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 value is always decreased by one before a push operation and increased by one *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 2-15.

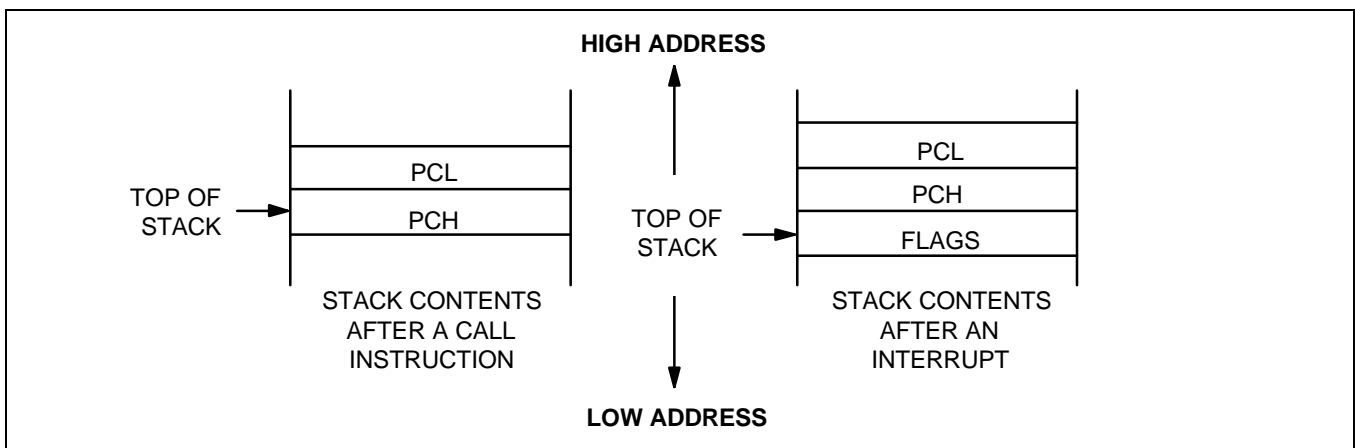


Figure 2-15. Stack Operations

User-Defined Stacks

You can freely define stacks in the internal register file as data storage locations. The instructions PUSHUI, PUSHUD, POPUI, and POPUD support user-defined stack operations.

Stack Pointers (SPL, SPH)

Register locations D8H and D9H contain the 16-bit stack pointer (SP) that is used for system stack operations. The most significant byte of the SP address, SP15-SP8, is stored in the SPH register (D8H), and the least significant byte, SP7-SP0, is stored in the SPL register (D9H). After a reset, the SP value is undetermined.

If only internal memory space is implemented in the S3C821A, the SPL must be initialized to an 8-bit value in the range 00H-FFH. The SPH register is not needed and can be used as a general-purpose register, if necessary. If external memory is implemented, both SPL and SPH must be initialized with a full 16-bit address.

When the SPL register contains the only stack pointer value (that is, when it points to a system stack in the register file), you can use the SPH register as a general-purpose data register. However, if an overflow or underflow condition occurs as a result of increasing or decreasing the stack address value in the SPL register during normal stack operations, the value in the SPL register will overflow (or underflow) to the SPH register, overwriting any other data that is currently stored there. To avoid overwriting data in the SPH register, you can initialize the SPL value to "FFH" instead of "00H".

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      SPL,#0FFH      ; SPL ← FFH
                        ; (Normally, the SPL is set to 0FFH by the initialization
                        ; routine)
.
.
.
PUSH   PP              ; Stack address 0FEH ← PP
PUSH   RP0             ; Stack address 0FDH ← RP0
PUSH   RP1             ; Stack address 0FCH ← RP1
PUSH   R3              ; Stack address 0FBH ← R3
.
.
.
POP    R3              ; R3 ← Stack address 0FBH
POP    RP1             ; RP1 ← Stack address 0FCH
POP    RP0             ; RP0 ← Stack address 0FDH
POP    PP              ; PP ← Stack address 0FEH

```

3 ADDRESSING MODES

OVERVIEW

The program counter is used to fetch instructions that are stored in program memory for execution. 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 instructions may be condition codes, immediate data, or a location in the register file, program memory, or data memory.

The S3C8-series instruction set supports seven explicit addressing modes. Not all of these addressing modes are available for each instruction:

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

REGISTER ADDRESSING MODE (R)

In Register addressing mode, the operand is the content of a specified register or register pair (see Figure 3-1). Working register addressing differs from Register addressing as it uses a register pointer to specify an 8-byte working register space in the register file and an 8-bit register within that space (see Figure 3-2).

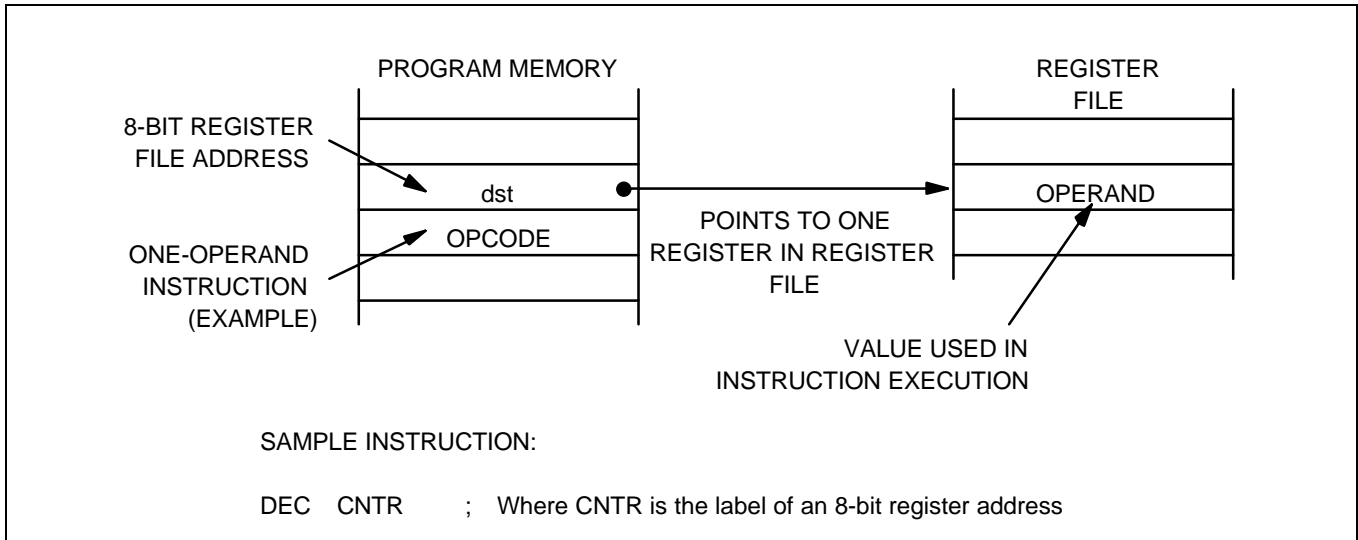


Figure 3-1. Register Addressing

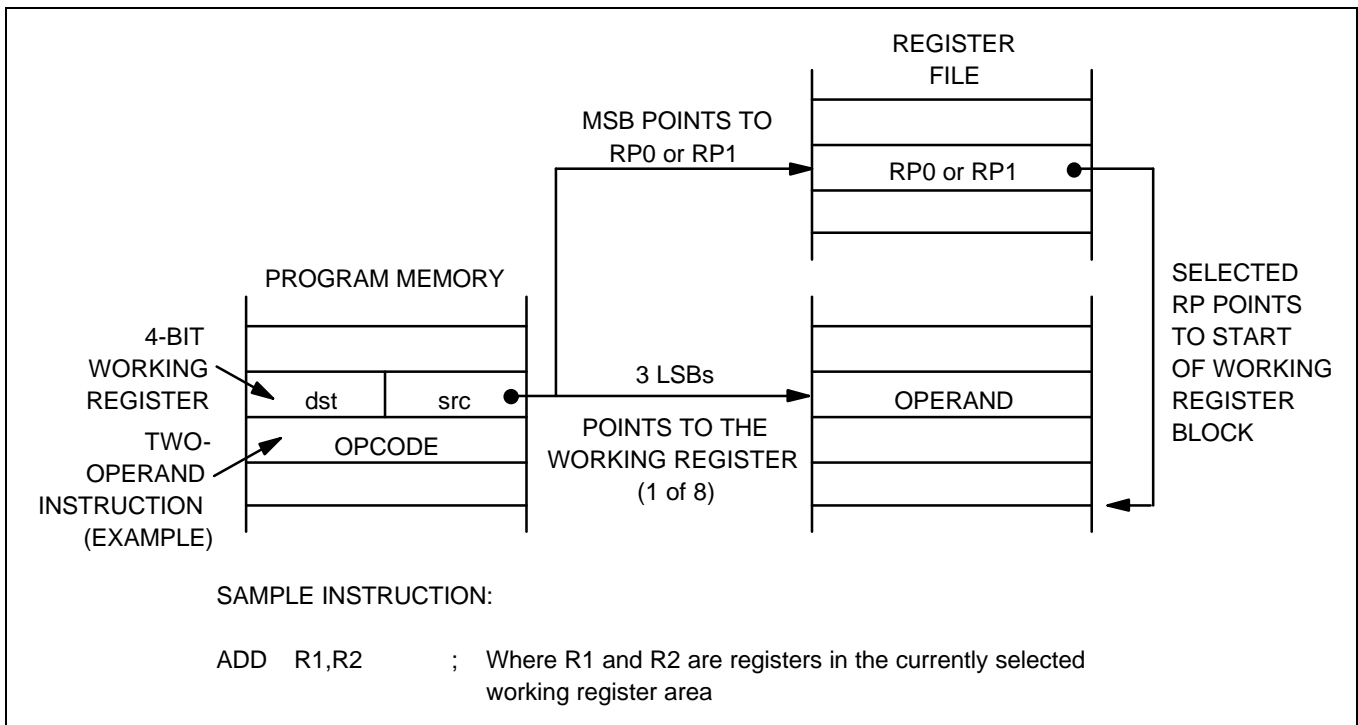


Figure 3-2. Working Register Addressing

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, if implemented (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. Remember, however, that locations C0H–FFH in set 1 cannot be accessed using Indirect Register addressing mode.

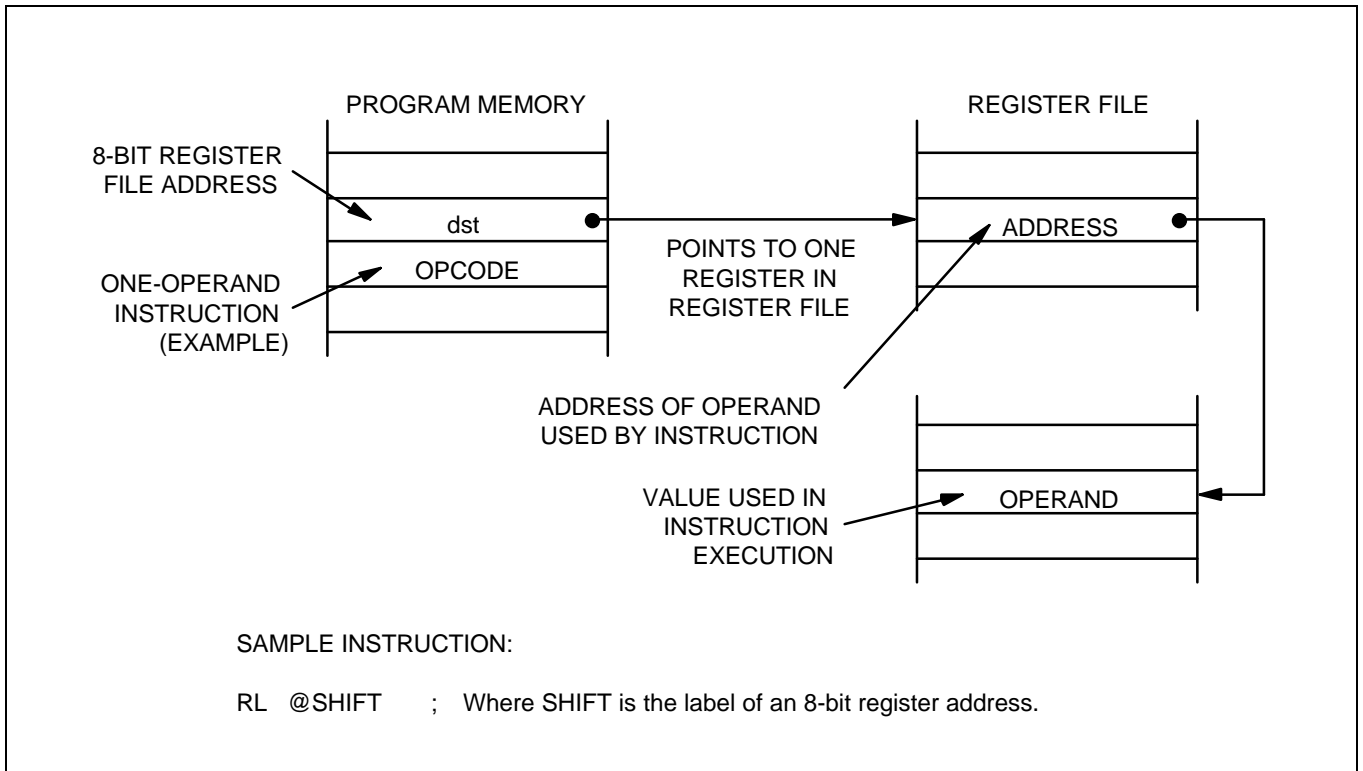


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

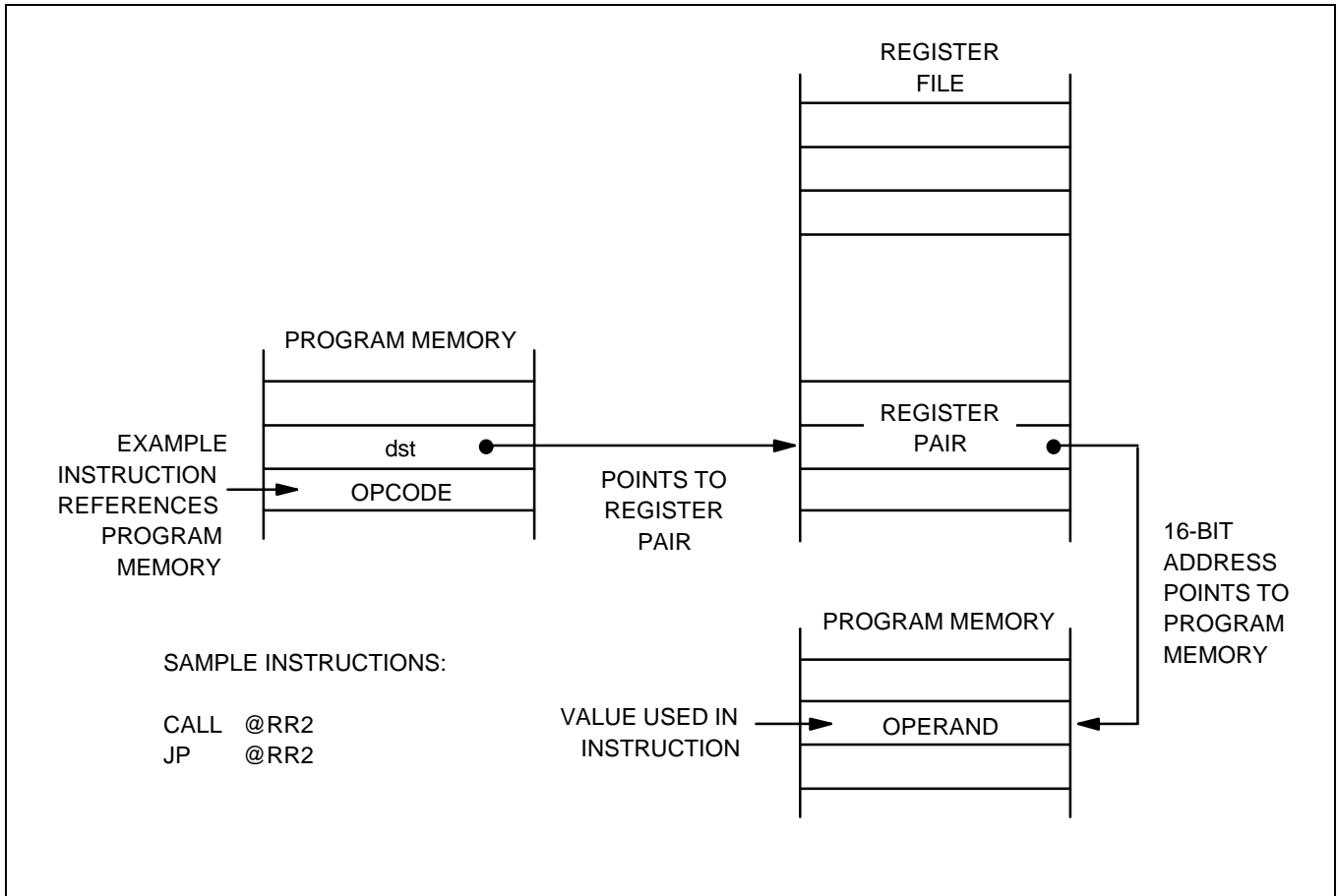


Figure 3-4. Indirect Register Addressing to Program Memory

INDIRECT REGISTER ADDRESSING MODE (Continued)

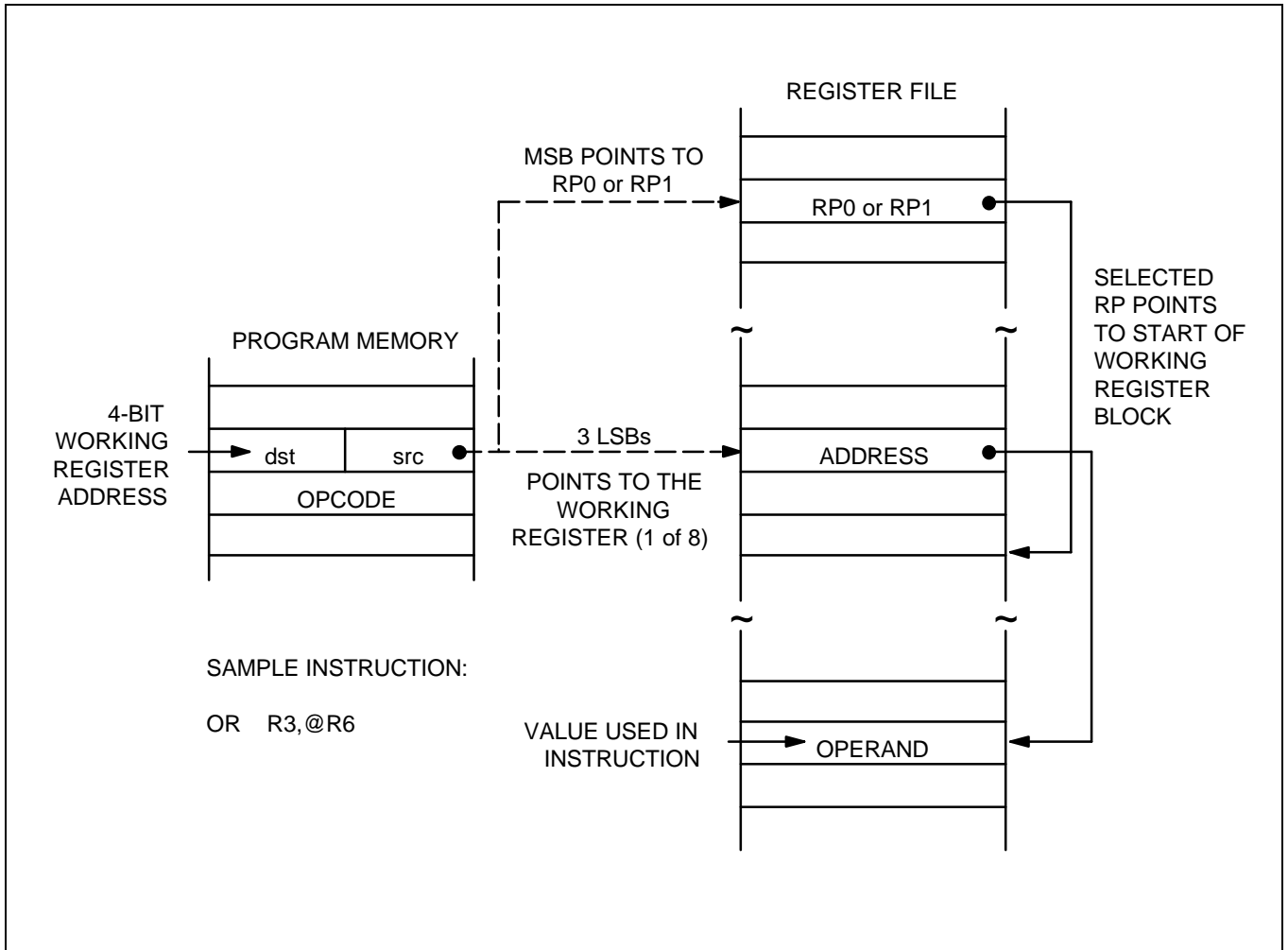


Figure 3-5. Indirect Working Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

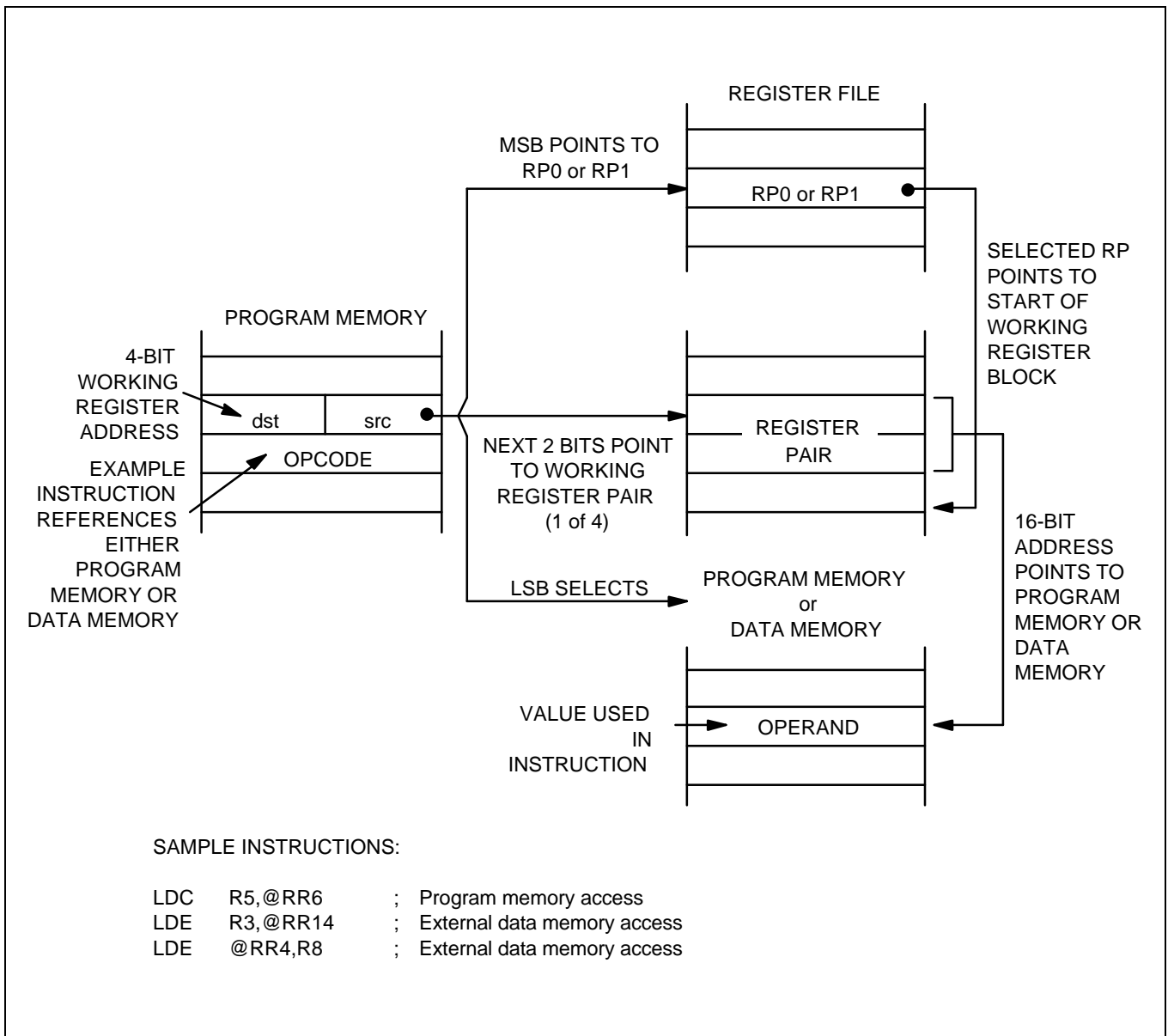


Figure 3-6. Indirect Working Register Addressing to Program or Data Memory

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 (if implemented). You cannot, however, access locations C0H-FFH in set 1 using Indexed addressing.

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

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

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

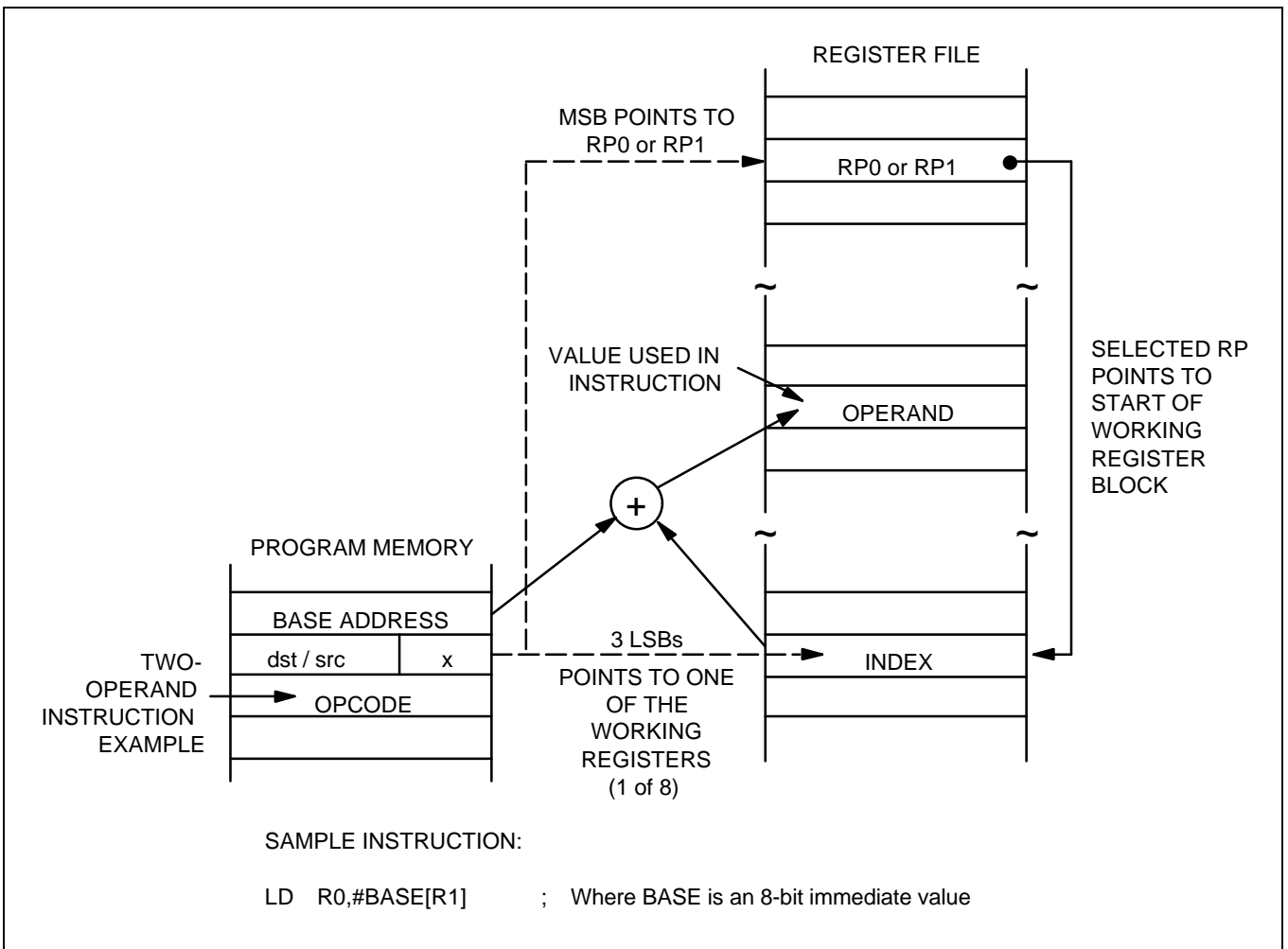


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

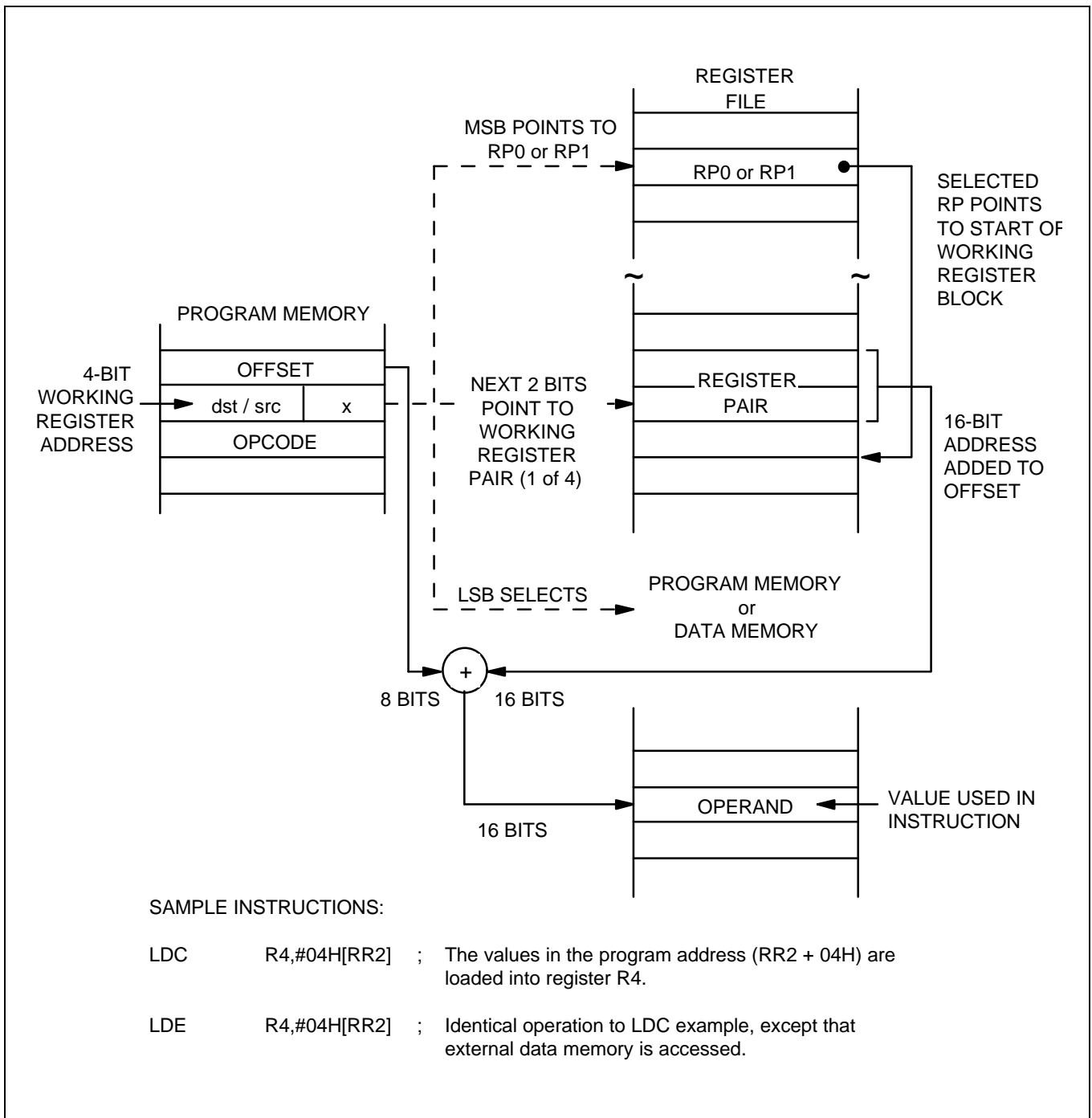


Figure 3-8. Indexed Addressing to Program or Data Memory with Short Offset

INDEXED ADDRESSING MODE (Continued)

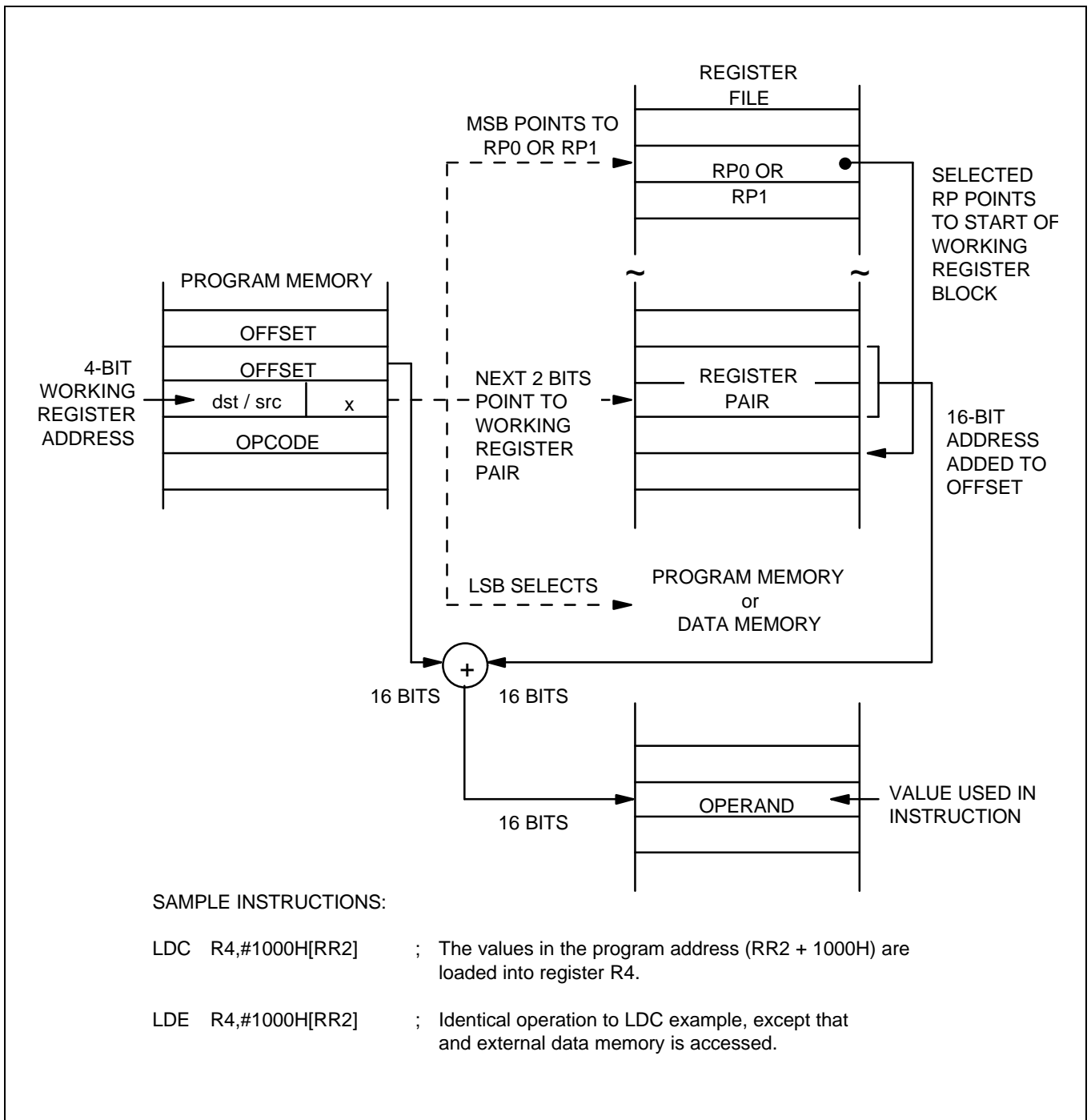


Figure 3-9. Indexed Addressing to Program or Data Memory

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.

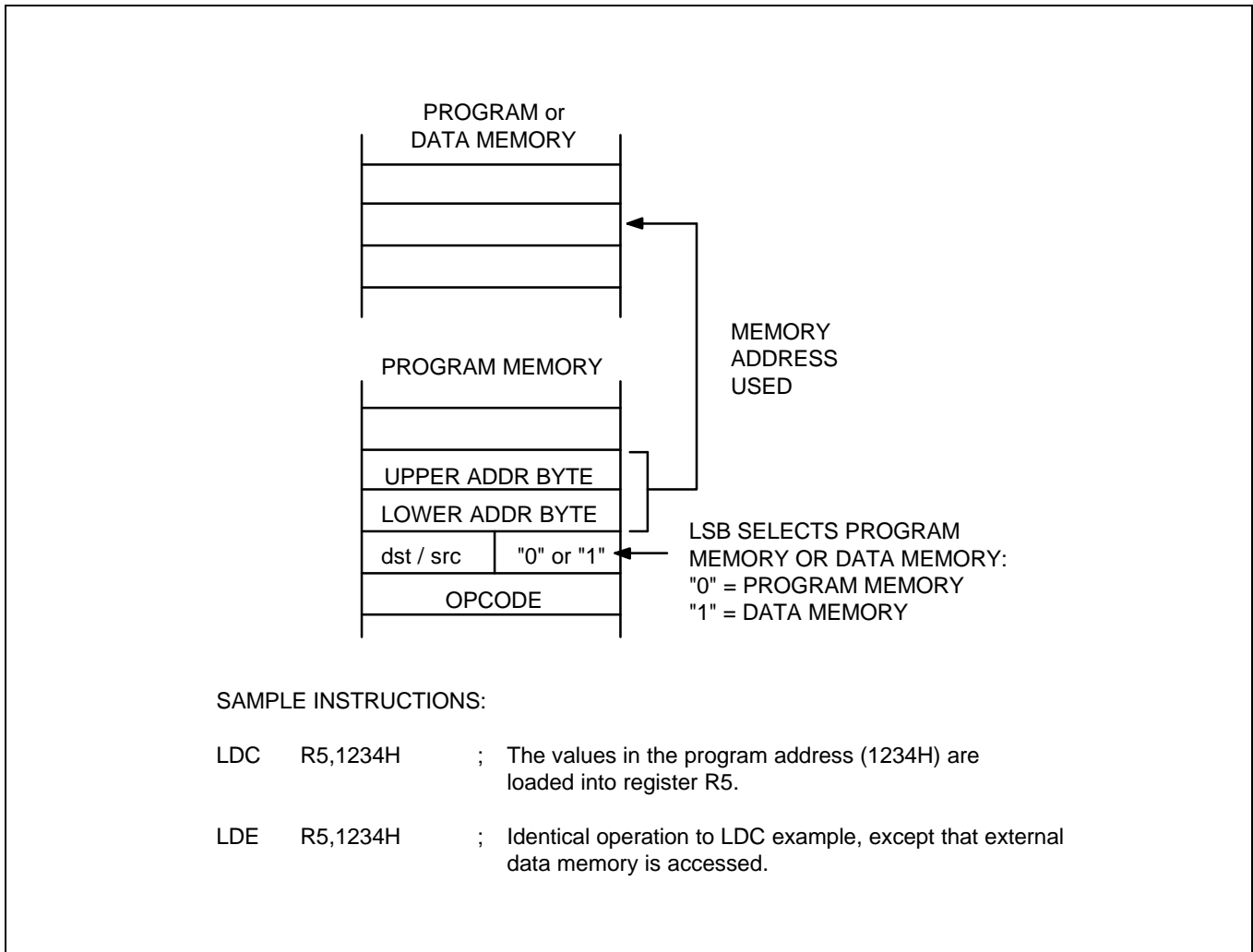


Figure 3-10. Direct Addressing for Load Instructions

DIRECT ADDRESS MODE (Continued)

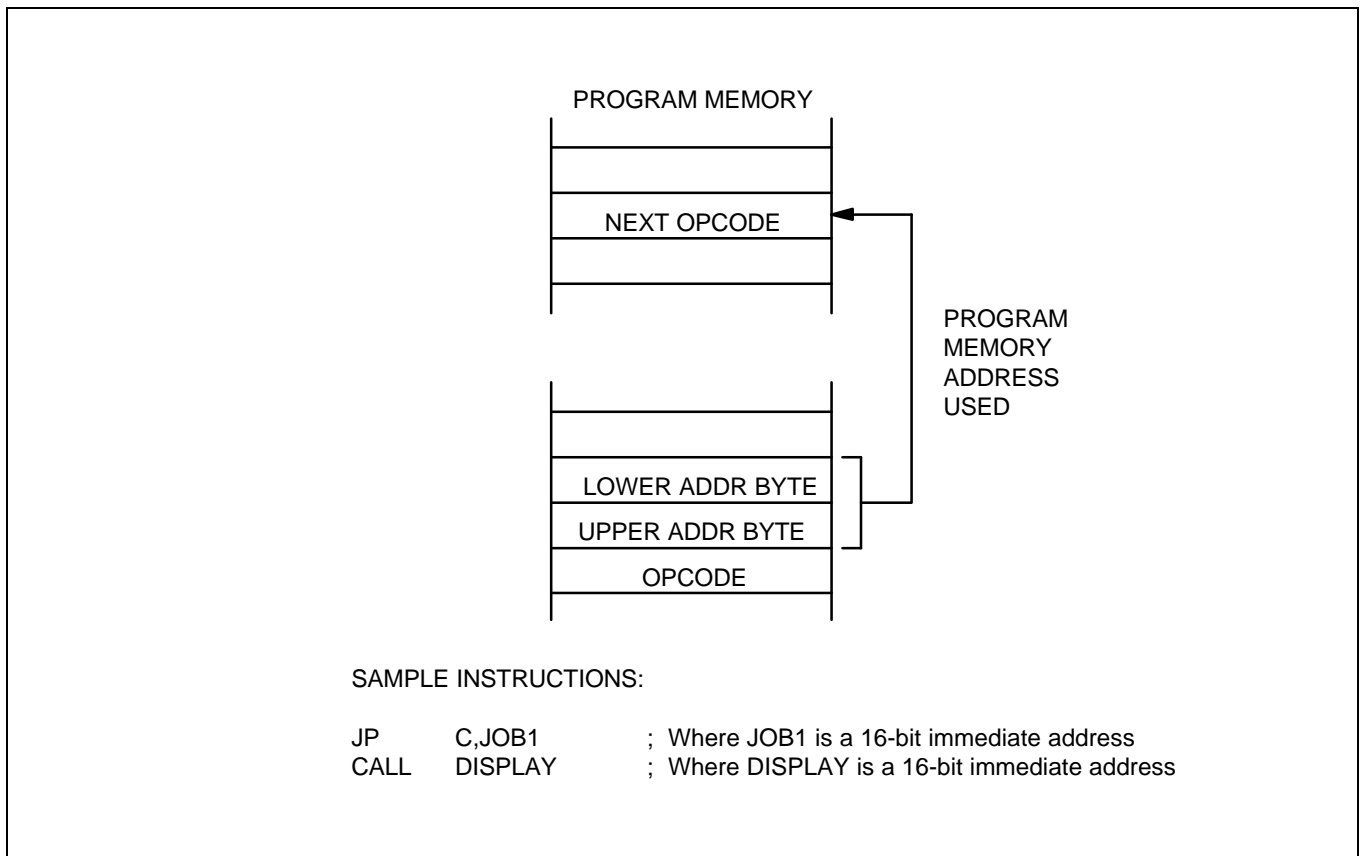


Figure 3-11. Direct Addressing for Call and Jump Instructions

INDIRECT ADDRESS MODE (IA)

In Indirect Address (IA) mode, the instruction specifies an address located in the lowest 256 bytes of the program memory. The selected pair of memory locations contains the actual address of the next instruction to be executed. Only the CALL instruction can use the Indirect Address mode.

Because the Indirect Address mode assumes that the operand is located in the lowest 256 bytes of program memory, only an 8-bit address is supplied in the instruction; the upper bytes of the destination address are assumed to be all zeros.

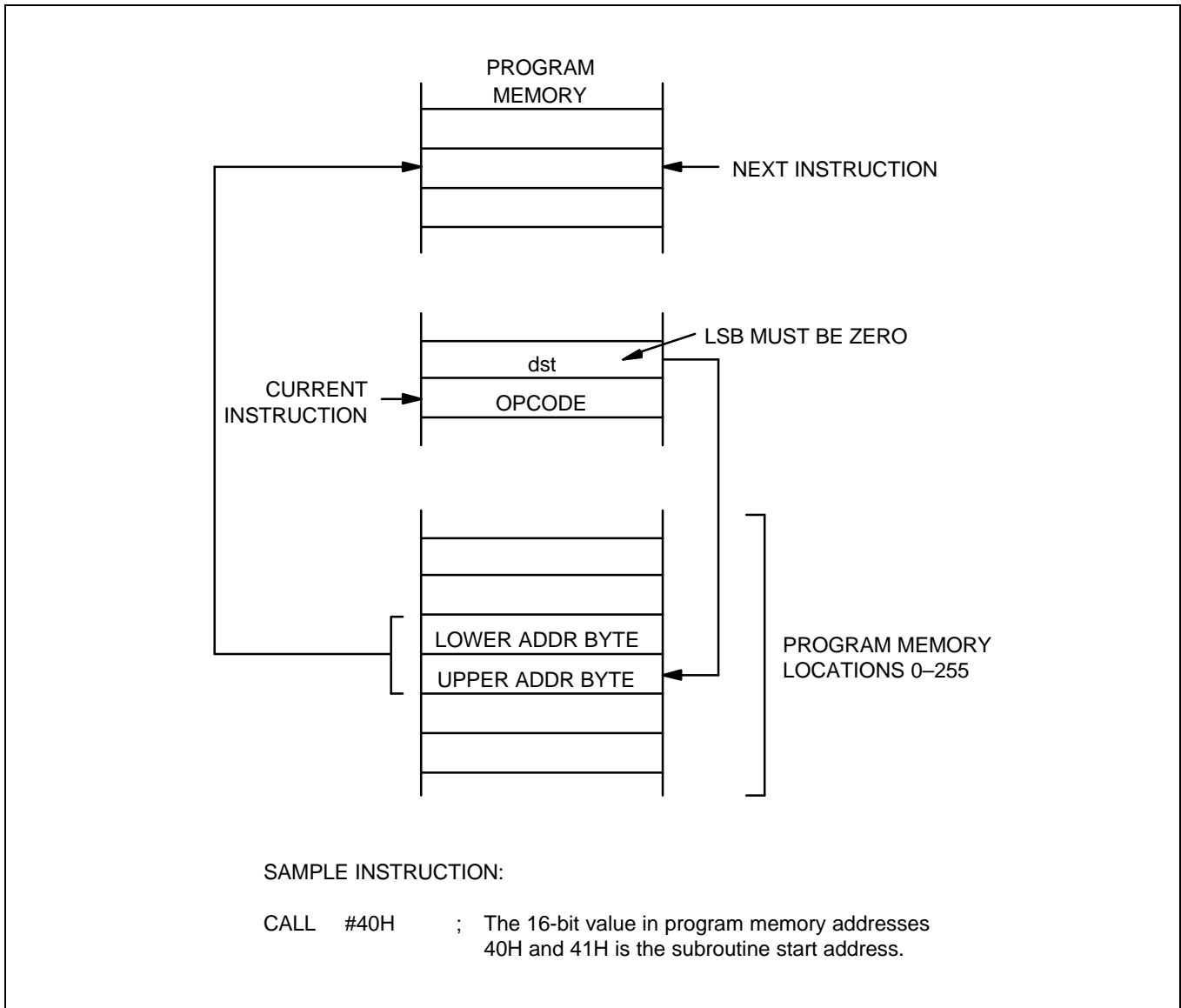


Figure 3-12. Indirect Addressing

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.

Several program control instructions use the Relative Address mode to perform conditional jumps. The instructions that support RA addressing are BTJRF, BTJRT, DJNZ, CPIJE, CPIJNE, and JR.

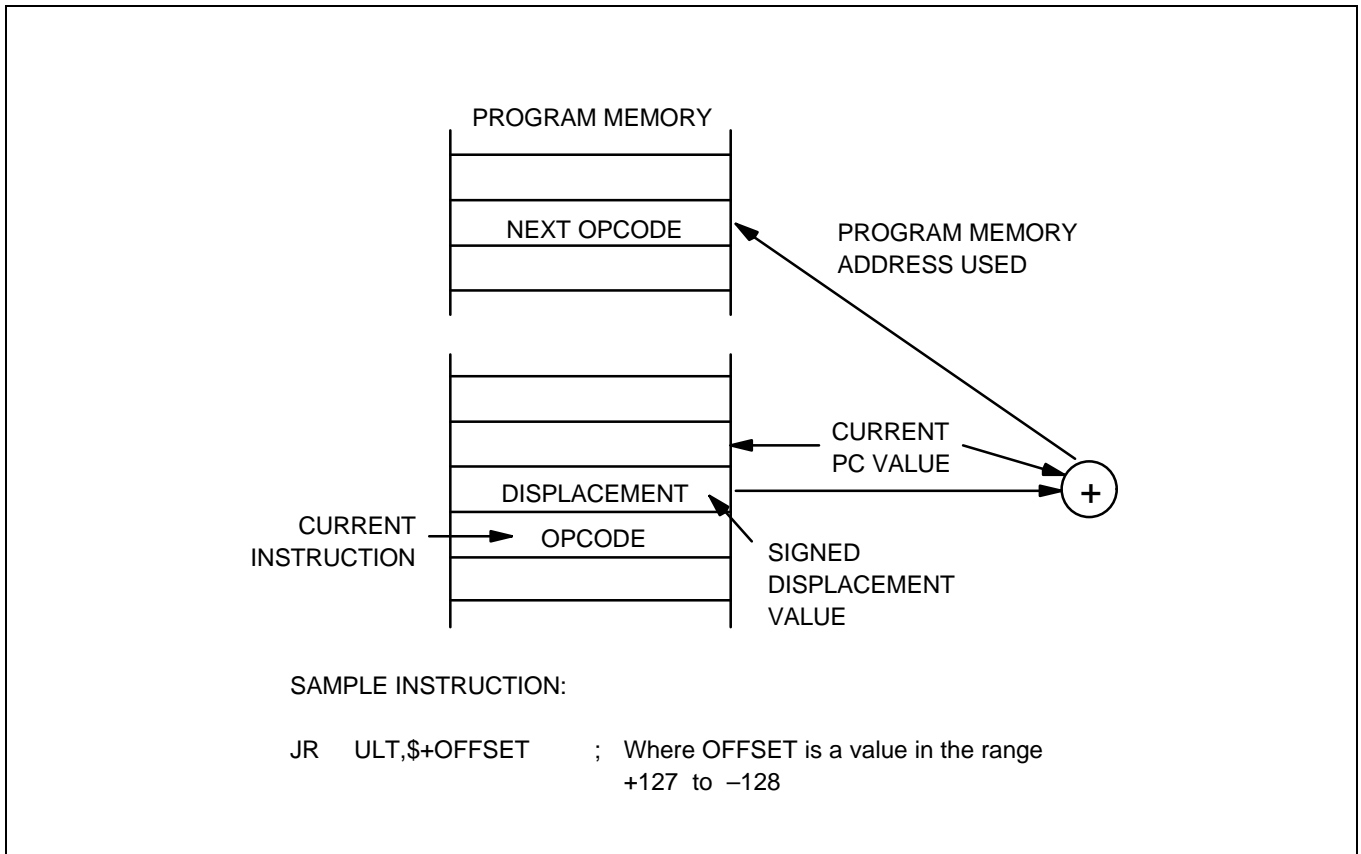


Figure 3-13. Relative Addressing

IMMEDIATE MODE (IM)

In Immediate (IM) mode, the operand value used in the instruction is the value supplied in the operand field itself. The operand may be one byte or one word in length, depending on the instruction used. Immediate addressing mode is useful for loading constant values into registers.

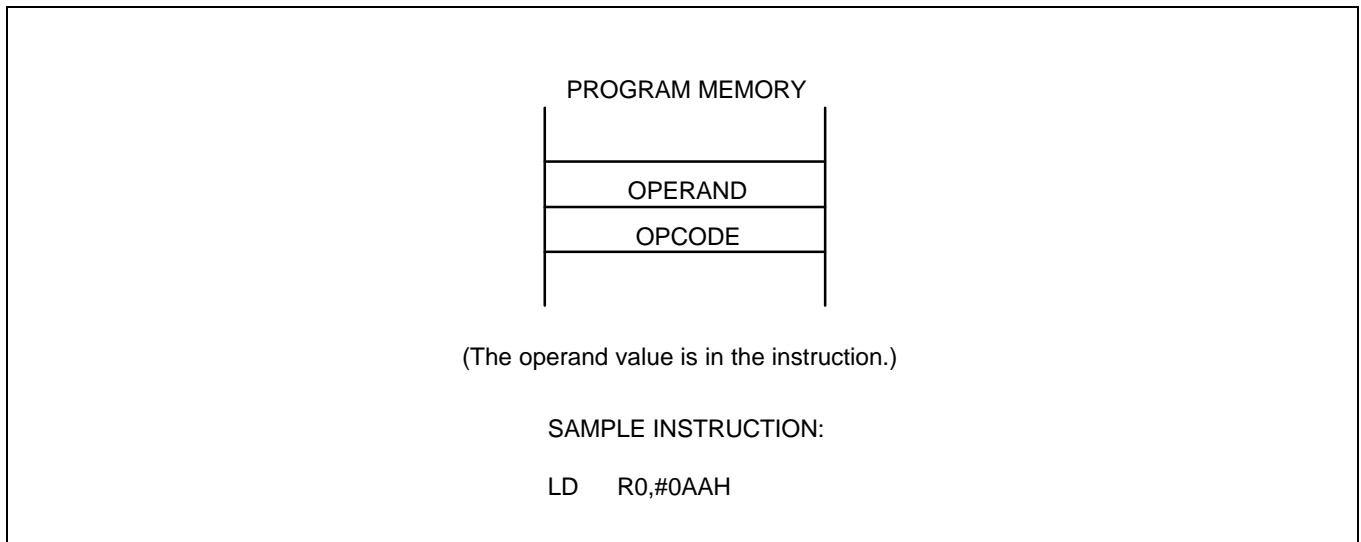


Figure 3-14. Immediate Addressing

4 CONTROL REGISTERS

OVERVIEW

In this chapter, detailed descriptions of the S3C821A control registers are presented in an easy-to-read format. You can use this chapter as a quick-reference source when writing application programs. 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 detailed information about control registers is presented in the context of the specific peripheral hardware descriptions in Part II of this manual.

Data and counter registers are not described in detail in this reference chapter. More information about all of the registers used by a specific peripheral is presented in the corresponding peripheral descriptions in Part II of this manual.

The locations and read/write characteristics of all mapped registers in the S3C821A register file are listed in Table 4-1. The hardware reset value for each mapped register is described in Chapter 8, "RESET and Power-Down."

Table 4-1. Set 1 Registers

Register Name	Mnemonic	Decimal	Hex	R/W
Timer 0 counter	T0CNT	208	D0H	R (note)
Timer 0 data register	T0DATA	209	D1H	R/W
Timer 0 control register	T0CON	210	D2H	R/W
Basic timer control register	BTCON	211	D3H	R/W
Clock control register	CLKCON	212	D4H	R/W
System flags register	FLAGS	213	D5H	R/W
Register pointer 0	RP0	214	D6H	R/W
Register pointer 1	RP1	215	D7H	R/W
Stack pointer (high byte)	SPH	216	D8H	R/W
Stack pointer (low byte)	SPL	217	D9H	R/W
Instruction pointer (high byte)	IPH	218	DAH	R/W
Instruction pointer (low byte)	IPL	219	DBH	R/W
Interrupt request register	IRQ	220	DCH	R (note)
Interrupt mask register	IMR	221	DDH	R/W
System mode register	SYM	222	DEH	R/W
Register page pointer	PP	223	DFH	R/W

NOTE: You cannot use a read-only register as a destination for the instruction OR, AND, LD, or LDB.

Table 4-2. Set 1, Bank 0 Registers

Register Name	Mnemonic	Decimal	Hex	R/W
Port 0 data register	P0	224	E0H	R/W
Port 1 data register	P1	225	E1H	R/W
Port 2 data register	P2	226	E2H	R/W
Port 3 data register	P3	227	E3H	R/W
Port 4 data register	P4	228	E4H	R/W
Port 5 data register	P5	229	E5H	R/W
Port 2 interrupt control register	P2INT	230	E6H	R/W
Port 4 interrupt control register	P4INT	231	E7H	R/W
Port 4 interrupt pending register	P4PND	232	E8H	R/W
Port 4 interrupt edge select register	P4EDGE	233	E9H	R/W
Locations EAH–EDH are not mapped.				
Timer B counter	TBCNT	238	EEH	R ⁽²⁾
Timer A counter	TACNT	239	EFH	R ⁽²⁾
Timer B data register	TBDATA	240	F0H	R/W
Timer A data register	TADATA	241	F1H	R/W
Timer B control register	TBCON	242	F2H	R/W
Timer 1/A control register	TACON	243	F3H	R/W
SIO data register	SIO	244	F4H	R/W
SIO control register	SIOCON	245	F5H	R/W
SIO prescaler register	SIOPS	246	F6H	R/W
ADC control register	ADCON	247	F7H	R/W ⁽¹⁾
ADC data register	ADDATA	248	F8H	R ⁽²⁾
Location F9H is not mapped.				
LCD control register	LCON	250	FAH	R/W
Watch timer control register	WTCON	251	FBH	R/W
Location FCH is not mapped.				
Basic timer counter	BTCNT	253	FDH	R ⁽²⁾
External memory timing register	EMT	254	FEH	R/W
Interrupt priority register	IPR	255	FFH	R/W

NOTES:

1. ADCON.3 is a read-only bit.
2. You cannot use a read-only register as a destination for the instructions OR, AND, LD, or LDB.

Table 4-3. Set 1, Bank 1 Registers

Register Name	Mnemonic	Decimal	Hex	R/W
Port 0 control register	P0CON	224	E0H	R/W
Location E1H is not mapped.				
Port 1 control register	P1CON	226	E2H	R/W
Location E3H is not mapped.				
Port 2 control register (high byte)	P2CONH	228	E4H	R/W
Port 2 control register (low byte)	P2CONL	229	E5H	R/W
Port 3 control register (high byte)	P3CONH	230	E6H	R/W
Port 3 control register (low byte)	P3CONL	231	E7H	R/W
Port 4 control register (high byte)	P4CONH	232	E8H	R/W
Port 4 control register (low byte)	P4CONL	233	E9H	R/W
Port 5 control register (high byte)	P5CONH	234	EAH	R/W
Port 5 control register (low byte)	P5CONL	235	EBH	R/W
Locations ECH–FEH are not mapped.				
Sub-clock control register	SCLKCON	255	FFH	R/W

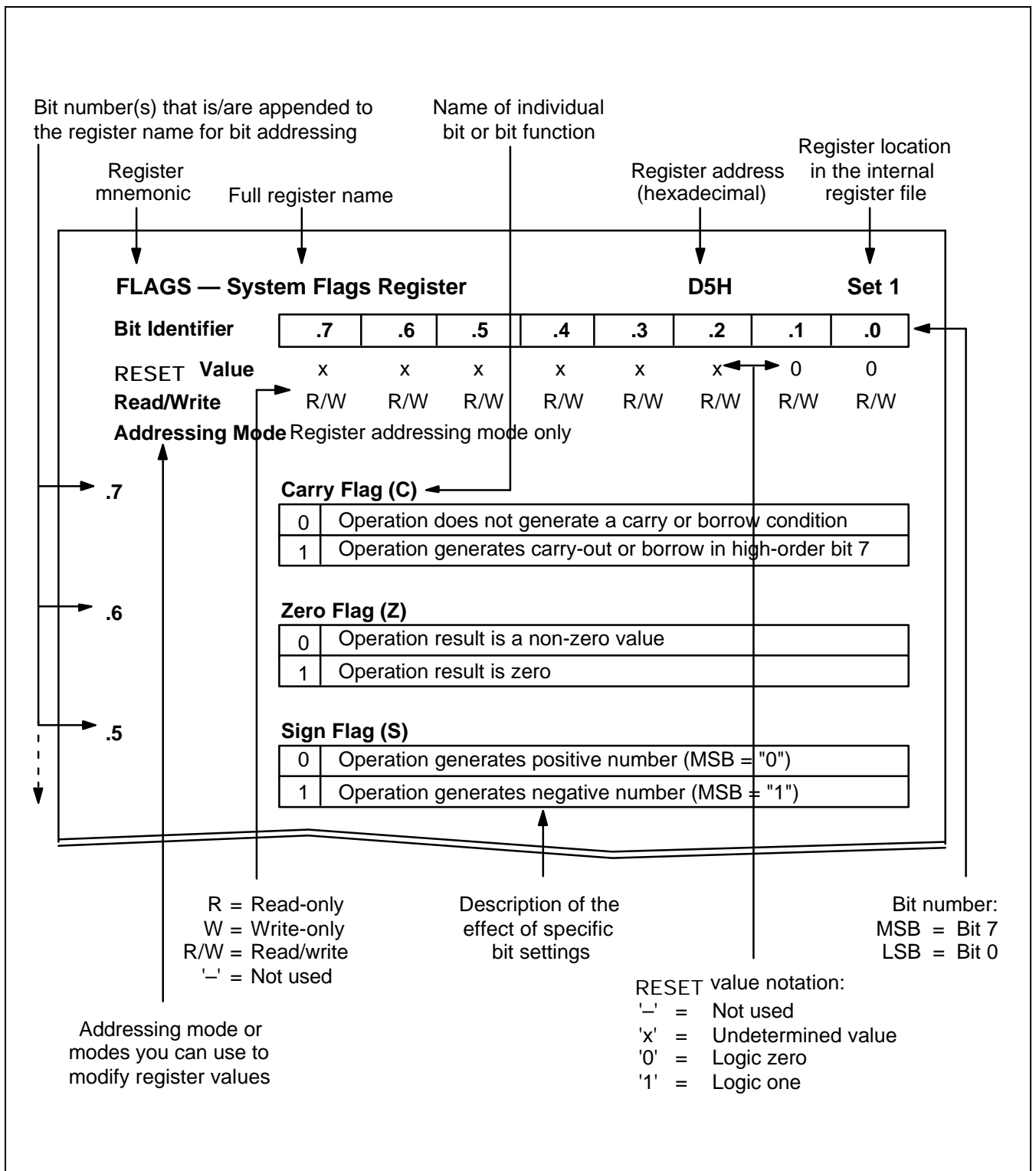


Figure 4-1. Register Description Format



ADCON — A/D Converter Control Register

F7H

Set 1, Bank 0

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	0	0	0	0	0	0	0
Read/Write	–	R/W	R/W	R/W	R	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7

Not used for the S3C821A

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

0	0	0	ADC0
0	0	1	ADC1
0	1	0	ADC2
0	1	1	ADC3

.3 **End-of-Conversion Bit (Read-only)**

0	Conversion not complete
1	Conversion complete

.2 and .1 **Clock Source Selection Bits**

0	0	f _{xx} /16
0	1	f _{xx} /8
1	0	f _{xx} /4
1	1	f _{xx}

.0 **Start or Enable Bit**

0	No effect (when write)
1	Start operation and auto cleared (when write)

BTCON — Basic Timer Control Register

D3H

Set 1

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
Addressing Mode	Register addressing mode only							

.7–.4**Watchdog Timer Function Disable Code (for System Reset)**

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

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

0	0	fxx/4096 ⁽³⁾
0	1	fxx/1024
1	0	fxx/128
1	1	fxx/16

.1**Basic Timer Counter Clear Bit ⁽¹⁾**

0	No effect
1	Clear the basic timer counter value

.0**Clock Frequency Divider Clear Bit for Basic Timer and Timer 0 ⁽²⁾**

0	No effect
1	Clear both clock frequency dividers

NOTES:

- When you write a "1" to BTCON.1, the basic timer counter value is cleared to "00H". Immediately following the write operation, the BTCON.1 value is automatically cleared to "0".
- When you write a "1" to BTCON.0, the corresponding frequency divider is cleared to "00H". Immediately following the write operation, the BTCON.0 value is automatically cleared to "0".
- The fxx is selected clock for peripheral hardware (main system clock or sub system clock).

CLKCON — System Clock Control Register

D4H

Set 1

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
Addressing Mode	Register addressing mode only							

.7 **Oscillator IRQ Wake-up Enable 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 **Main Oscillator Stop Control Bits ⁽³⁾**

0	0	No effect
0	1	No effect
1	0	STOP main oscillator (fx)
1	1	No effect

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

0	0	fx/16 or fxt
0	1	fx/8 or fxt
1	0	fx/2 or fxt
1	1	fx (non-divided) or fxt

.2–.0 **Subsystem Clock Selection Bits ⁽²⁾**

1	0	1	Select sub system clock (fxt)
Others			Select main system clock (fx)

NOTES:

- After a reset, the slowest clock (divided by 16) is selected as the CPU clock. To select faster clock speeds, load the appropriate values to CLKCON.3 and CLKCON.4.
- Although sub clock (fxt) is selected as CPU clock (fcpu), the selected clock (fxx) for basic timer, timer counter 0/1, watch timer, and A/D converter is main clock if the value of CLKCON.6-.5 are not "10B (value to stop main clock)."
- Although the value of CLKCON. 6-.5 are "10B (value to stop main clock)", main clock is not stopped if main system clock (fx) is selected as CPU clock (fcpu).

EMT — External Memory Timing Register (note) **FEH** **Set 1, Bank 0**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	–	–	0	–
Read/Write	–	–	–	–	–	–	R/W	–
Addressing Mode	Register addressing mode only							

.7–.2 Not used for the S3C821A

.1 **Stack Area Selection Bit**

0	Select internal register file area
1	Select external data memory area

.0 Not used for the S3C821A

NOTE: The EMT register is not used except EMT.1 register bit. The program initialization routine should clear the EMT register except the bit 1 to "00H" following a reset. Modification of EMT values during normal operation may cause a system malfunction.

FLAGS — System Flags Register

D5H

Set 1

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Addressing Mode	Register addressing mode only							

.7	Carry Flag (C)	
0	Operation does not generate a carry or borrow condition	
1	Operation generates a carry-out or borrow into high-order bit 7	
.6	Zero Flag (Z)	
0	Operation result is a non-zero value	
1	Operation result is zero	
.5	Sign Flag (S)	
0	Operation generates a positive number (MSB = "0")	
1	Operation generates a negative number (MSB = "1")	
.4	Overflow Flag (V)	
0	Operation result is $\leq +127$ or ≥ -128	
1	Operation result is $> +127$ or < -128	
.3	Decimal Adjust Flag (D)	
0	Add operation completed	
1	Subtraction operation completed	
.2	Half-Carry Flag (H)	
0	No carry-out of bit 3 or no borrow into bit 3 by addition or subtraction	
1	Addition generated carry-out of bit 3 or subtraction generated borrow into bit 3	
.1	Fast Interrupt Status Flag (FIS)	
0	Interrupt return (IRET) in progress (when read)	
1	Fast interrupt service routine in progress (when read)	
.0	Bank Address Selection Flag (BA)	
0	Bank 0 is selected	
1	Bank 1 is selected	

IMR — Interrupt Mask Register

DDH

Set 1

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	–	x	x	x
Read/Write	R/W	R/W	R/W	R/W	–	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7 Interrupt Level 7 (IRQ7) Enable Bit; External Interrupts P4.3–P4.0

0	Disable (mask)
1	Enable (un-mask)

.6 Interrupt Level 6 (IRQ6) Enable Bit; External Interrupts P4.7–P4.4

0	Disable (mask)
1	Enable (un-mask)

.5 Interrupt Level 5 (IRQ5) Enable Bit; External Interrupts P2.7–P2.4

0	Disable (mask)
1	Enable (un-mask)

.4 Interrupt Level 4 (IRQ4) Enable Bit; Watch Timer Overflow

0	Disable (mask)
1	Enable (un-mask)

.3 Not used for the S3C821A

.2 Interrupt Level 2 (IRQ2) Enable Bit; SIO Interrupt

0	Disable (mask)
1	Enable (un-mask)

.1 Interrupt Level 1 (IRQ1) Enable Bit; Timer 1 Match (Timer A and B)

0	Disable (mask)
1	Enable (un-mask)

.0 Interrupt Level 0 (IRQ0) Enable Bit; Timer 0 Match/Capture or Overflow

0	Disable (mask)
1	Enable (un-mask)

NOTES:

1. When an interrupt level is masked, any interrupt requests that may be issued are not recognized by the CPU.
2. Interrupt levels IRQ3 is not used in the S3C821A interrupt structure.

IPH — Instruction Pointer (High Byte)

DAH

Set 1

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	x	x
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7–.0

Instruction Pointer Address (High Byte)

The high-byte instruction pointer value is the upper eight bits of the 16-bit instruction pointer address (IP15–IP8). The lower byte of the IP address is located in the IPL register (DBH).

IPL — Instruction Pointer (Low Byte)

DBH

Set 1

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	x	x
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7–.0

Instruction Pointer Address (Low Byte)

The low-byte instruction pointer value is the lower eight bits of the 16-bit instruction pointer address (IP7/IP0). The upper byte of the IP address is located in the IPH register (DAH).

IPR — Interrupt Priority Register**FFH****Set 1, Bank 0**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	x	x
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7, .4, and .1**Priority Control Bits for Interrupt Groups A, B, and C**

0	0	0	Group priority undefined
0	0	1	B > C > A
0	1	0	A > B > C
0	1	1	B > A > C
1	0	0	C > A > B
1	0	1	C > B > A
1	1	0	A > C > B
1	1	1	Group priority undefined

.6**Interrupt Subgroup C Priority Control Bit**

0	IRQ6 > IRQ7
1	IRQ7 > IRQ6

.5**Interrupt Group C Priority Control Bit**

0	IRQ5 > (IRQ6, IRQ7)
1	(IRQ6, IRQ7) > IRQ5

.3**Interrupt Subgroup B Priority Control Bit ^(note)**

0	IRQ4
1	IRQ4

.2**Interrupt Group B Priority Control Bit ^(note)**

0	IRQ2 > IRQ4
1	IRQ4 > IRQ2

.0**Interrupt Group A Priority Control Bit**

0	IRQ0 > IRQ1
1	IRQ1 > IRQ0

NOTE: The S3C821A interrupt structure uses only the seven levels: IRQ0–IRQ2, and IRQ4–IRQ7. Because IRQ3 is not recognized, the interrupt B subgroup settings (IPR.2 and IPR.3) are not evaluated.

IRQ — Interrupt Request Register

DCH

Set 1

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R
Addressing Mode	Register addressing mode only							

.7	Level 7 (IRQ7) Request Pending Bit; External Interrupts P4.3–P4.0	
	0	Not pending
	1	Pending

.6	Level 6 (IRQ6) Request Pending Bit; External Interrupts P4.7–P4.4	
	0	Not pending
	1	Pending

.5	Level 5 (IRQ5) Request Pending Bit; External Interrupts P2.7–P2.4	
	0	Not pending
	1	Pending

.4	Level 4 (IRQ4) Request Pending Bit; Watch Timer Overflow	
	0	Not pending
	1	Pending

.3	Not used for the S3C821A	
-----------	--------------------------	--

.2	Level 2 (IRQ2) Request Pending Bit; SIO Interrupt	
	0	Not pending
	1	Pending

.1	Level 1 (IRQ1) Request Pending Bit; Timer 1 Match (Timer A and B)	
	0	Not pending
	1	Pending

.0	Level 0 (IRQ0) Request Pending Bit; Timer 0 Match/Capture or Overflow	
	0	Not pending
	1	Pending

NOTE: Interrupt level IRQ3 is not used in the S3C821A interrupt structure.

LCON — LCD Control Register

FAH

Set 1, Bank 0

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
Addressing Mode	Register addressing mode only							

.7 and .6**LCD Duty and Bias Selection Bits**

0	0	Display off, P-Tr off
0	1	Normal display (using V_{LC1} with external voltage), P-Tr off
1	0	All dots off, P-Tr on
1	1	Normal display (using V_{LC1} with internal voltage), P-Tr on

.5 and .4

Not used for the S3C821A.

.3 and .2**LCD Duty and Bias Selection Bits**

0	0	1/3 duty, 1/3 bias; COM0–COM2/SEG0–SEG31
0	1	1/4 duty, 1/3 bias; COM0–COM3/SEG0–SEG31
1	0	1/8 duty, 1/4 bias; COM0–COM7/SEG4–SEG31
1	1	1/8 duty, 1/5 bias; COM0–COM7/SEG4–SEG31

.1 and .0**LCD Clock Selection Bits**

0	0	$f_w/2^6$ (512 Hz when f_w is 32.768 kHz)
0	1	$f_w/2^5$ (1,024 Hz when f_w is 32.768 kHz)
1	0	$f_w/2^4$ (2,048 Hz when f_w is 32.768 kHz)
1	1	$f_w/2^3$ (4,096 Hz when f_w is 32.768 kHz)

P0CON — Port 0 Control Register

E0H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7–.4**Upper Nibble Port Configuration (P0.7/A15–P0.4/A12)**

0	0	x	0	Input mode
0	1	x	0	Pull-up input mode
0	0	0	1	Push-pull output mode
0	0	1	1	Open-drain output mode
0	1	x	1	LCD segment (SEG23–SEG20)
1	x	x	x	External interface (A15–A12)

.3–.0**Lower Nibble Port Configuration (P0.3/A11–P0.0/A8)**

0	0	x	0	Input mode
0	1	x	0	Pull-up input mode
0	0	0	1	Push-pull output mode
0	0	1	1	Open-drain output mode
0	1	x	1	LCD segment (SEG19–SEG16)
1	x	x	x	External interface (A11–A8)

NOTE: "x" means don't care.

P1CON — Port 1 Control Register

E2H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7–.4

Upper Nibble Port Configuration (P1.7/AD7–P1.4/AD4)

0	0	x	0	Input mode
0	1	x	0	Pull-up input mode
0	0	0	1	Push-pull output mode
0	0	1	1	Open-drain output mode
0	1	x	1	LCD segment (SEG31–SEG28)
1	x	x	x	External interface (AD7–AD4)

.3–.0

Lower Nibble Port Configuration (P1.3/AD3–P1.0/AD0)

0	0	x	0	Input mode
0	1	x	0	Pull-up input mode
0	0	0	1	Push-pull output mode
0	0	1	1	Open-drain output mode
0	1	x	1	LCD segment (SEG27–SEG24)
1	x	x	x	External interface (AD3–AD0)

NOTE: "x" means don't care.



P2CONH — Port 2 Control Register (High Byte)

E4H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6**P2.7/TB Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Select alternate function for P2.7 (TB clock)

.5 and .4**P2.6/TA Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Select alternate function for P2.6 (TA clock)

.3 and .2**P2.5/T1CK Mode Selection Bits**

0	0	Schmitt trigger input mode (T1CK)
0	1	Schmitt trigger input, pull-up mode (T1CK)
1	0	Push-pull output mode
1	1	P2.5 remains "0" state

.1 and .0**P2.4/T0CK Mode Selection Bits**

0	0	Schmitt trigger input mode (T0CK)
0	1	Schmitt trigger input, pull-up mode (T0CK)
1	0	Push-pull output mode
1	1	P2.4 remains "0" state

NOTE: Pins configured as input can be used as interrupt input with noise filter.

P2CONL — Port 2 Control Register (Low Byte)

E5H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6

P2.3/DM Mode Selection Bits

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	External memory interface line (DM)

.5 and .4

P2.2/DW Mode Selection Bits

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	External memory interface line (DW)

.3 and .2

P2.1/DR Mode Selection Bits

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	External memory interface line (DR)

.1 and .0

P2.0/AS Mode Selection Bits

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	External memory interface line (AS)



P2INT — Port 2 Interrupt Control Register

E6H

Set 1, Bank 0

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
Addressing Mode	Register addressing mode only							

.7 Port 2 Interrupt Request Pending Bit (P2.7/INT3)

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

.6 Interrupt Control Settings (P2.7/INT3)

0	Disable interrupt on P2.7
1	Enable interrupt at falling edge on P2.7

.5 Port 2 Interrupt Request Pending Bit (P2.6/INT2)

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

.4 Interrupt Control Settings (P2.6/INT2)

0	Disable interrupt on P2.6
1	Enable interrupt at falling edge on P2.6

.3 Port 2 Interrupt Request Pending Bit (P2.5/INT1)

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

.2 Interrupt Control Settings (P2.5/INT1)

0	Disable interrupt on P2.5
1	Enable interrupt at falling edge on P2.5

.1 Port 2 Interrupt Request Pending Bit (P2.4/INT0)

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

.0 Interrupt Control Settings (P2.4/INT0)

0	Disable interrupt on P2.4
1	Enable interrupt at falling edge on P2.4

P3CONH — Port 3 Control Register (High Byte)

E6H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6**P3.7/T0/T0PWM/T0CAP Mode Selection Bits**

0	0	Schmitt trigger input mode (T0CAP input)
0	1	Schmitt trigger input, pull-up mode (T0CAP input)
1	0	Push-pull output mode
1	1	Select alternate function (T0PWM/T0)

.5 and .4**P3.6 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output, high-impedance

.3 and .2**P3.5 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output, high-impedance

.1 and .0**P3.4 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output, high-impedance

P3CONL — Port 3 Control Register (Low Byte)

E7H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6**P3.3/ADC3 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	A/D converter input mode; schmitt trigger input off

.5 and .4**P3.2/ADC2 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	A/D converter input mode; schmitt trigger input off

.3 and .2**P3.1/ADC1 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	A/D converter input mode; schmitt trigger input off

.1 and .0**P3.0/ADC0 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	A/D converter input mode; schmitt trigger input off

P4CONH — Port 4 Control Register (High Byte)

E8H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6**P4.7/INT11 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

.5 and .4**P4.6/INT10 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

.3 and .2**P4.5/INT9 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

.1 and .0**P4.4/INT8 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

NOTE: Pins configured as input can be used as interrupt input with noise filter.

P4CONL — Port 4 Control Register (Low Byte)

E9H

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6**P4.3/INT7 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

.5 and .4**P4.2/INT6 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

.3 and .2**P4.1/INT5 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

.1 and .0**P4.0/INT4 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Open-drain output mode

NOTE: Pins configured as input can be used as interrupt input with noise filter.

P4INT — Port 4 Interrupt Control Register**E7H****Set 1, Bank 0**

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
Addressing Mode	Register addressing mode only							

.7 P4.7 External Interrupt (INT11) Enable Bit

0	Disable interrupt
1	Enable interrupt

.6 P4.6 External Interrupt (INT10) Enable Bit

0	Disable interrupt
1	Enable interrupt

.5 P4.5 External Interrupt (INT9) Enable Bit

0	Disable interrupt
1	Enable interrupt

.4 P4.4 External Interrupt (INT8) Enable Bit

0	Disable interrupt
1	Enable interrupt

.3 P4.3 External Interrupt (INT7) Enable Bit

0	Disable interrupt
1	Enable interrupt

.2 P4.2 External Interrupt (INT6) Enable Bit

0	Disable interrupt
1	Enable interrupt

.1 P4.1 External Interrupt (INT5) Enable Bit

0	Disable interrupt
1	Enable interrupt

.0 P4.0 External Interrupt (INT4) Enable Bit

0	Disable interrupt
1	Enable interrupt

P4PND — Port 4 Interrupt Pending Register**E8H****Set 1, Bank 0**

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
Addressing Mode	Register addressing mode only							

.7	P4.7 External Interrupt (INT11) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.7 interrupt request is pending (when read)
.6	P4.6 External Interrupt (INT10) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.6 interrupt request is pending (when read)
.5	P4.5 External Interrupt (INT9) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.5 interrupt request is pending (when read)
.4	P4.4 External Interrupt (INT8) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.4 interrupt request is pending (when read)
.3	P4.3 External Interrupt (INT7) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.3 interrupt request is pending (when read)
.2	P4.2 External Interrupt (INT6) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.2 interrupt request is pending (when read)
.1	P4.1 External Interrupt (INT5) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.1 interrupt request is pending (when read)
.0	P4.0 External Interrupt (INT4) Pending Flag	
	0	Clear pending bit (when write)
	1	P4.0 interrupt request is pending (when read)

NOTE: Writing a “1” to an interrupt pending flag (P4PND.0–.7) has no effect.

P4EDGE — Port 4 Interrupt Edge Selection Register**E9H****Set 1, Bank 0**

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
Addressing Mode	Register addressing mode only							

.7 P4.7 External Interrupt (INT11) State Bit

0	Falling edge detection
1	Rising edge detection

.6 P4.6 External Interrupt (INT10) State Bit

0	Falling edge detection
1	Rising edge detection

.5 P4.5 External Interrupt (INT9) State Bit

0	Falling edge detection
1	Rising edge detection

.4 P4.4 External Interrupt (INT8) State Bit

0	Falling edge detection
1	Rising edge detection

.3 P4.3 External Interrupt (INT7) State Bit

0	Falling edge detection
1	Rising edge detection

.2 P4.2 External Interrupt (INT6) State Bit

0	Falling edge detection
1	Rising edge detection

.1 P4.1 External Interrupt (INT5) State Bit

0	Falling edge detection
1	Rising edge detection

.0 P4.0 External Interrupt (INT4) State Bit

0	Falling edge detection
1	Rising edge detection

P5CONH — Port 5 Control Register (High Byte)

EAH

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6

Not used for the S3C821A

.5 and .4**P5.6 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output, high-impedance

.3 and .2**P5.5 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output, high-impedance

.1 and .0**P5.4 Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output, high-impedance

P5CONL — Port 5 Control Register (Low Byte)

EBH

Set 1, Bank 1

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
Addressing Mode	Register addressing mode only							

.7 and .6**P5.3/BUZ Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Select alternate function (BUZ signal)

.5 and .4**P5.2/SO Mode Selection Bits**

0	0	Schmitt trigger input mode
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Select alternate function (SO)

.3 and .2**P5.1/SI Mode Selection Bits**

0	0	Schmitt trigger input mode (SI input)
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Output high-impedance

.1 and .0**P5.0/SCK Mode Selection Bits**

0	0	Schmitt trigger input mode (SCK input)
0	1	Schmitt trigger input, pull-up mode
1	0	Push-pull output mode
1	1	Select alternate function (SCK output)

NOTE: SI and SCK inputs have noise filter.

PP — Register Page Pointer

DFH

Set 1

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
Addressing Mode	Register addressing mode only							

.7–4**Destination Register Page Selection Bits**

0	0	0	0	Destination: page 0
0	0	0	1	Destination: page 1
0	0	1	0	Destination: page 2
0	0	1	1	Destination: page 3
0	1	0	0	Destination: page 4

.3–0**Source Register Page Selection Bits**

0	0	0	0	Source: page 0
0	0	0	1	Source: page 1
0	0	1	0	Source: page 2
0	0	1	1	Source: page 3
0	1	0	0	Source: page 4

NOTES:

1. In the S3C821A microcontroller, the internal register file is configured as five pages (Pages 0–4). The pages 0–3 are used for general purpose register file, and page 4 is used for LCD data register or general purpose registers.
2. You should refer to page 6-39 and use DJNZ instruction properly when DJNZ instruction is used in your program.

RP0 — Register Pointer 0**D6H****Set 1**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	1	1	0	0	0	–	–	–
Read/Write	R/W	R/W	R/W	R/W	R/W	–	–	–
Addressing Mode	Register addressing only							

.7–3**Register Pointer 0 Address Value**

Register pointer 0 can independently point to one of the 256-byte working register areas in the register file. Using the register pointers RP0 and RP1, you can select two 8-byte register slices at one time as active working register space. After a reset, RP0 points to address C0H in register set 1, selecting the 8-byte working register slice C0H–C7H.

.2–0

Not used for the S3C821A

RP1 — Register Pointer 1**D7H****Set 1**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	1	1	0	0	1	–	–	–
Read/Write	R/W	R/W	R/W	R/W	R/W	–	–	–
Addressing Mode	Register addressing only							

.7–3**Register Pointer 1 Address Value**

Register pointer 1 can independently point to one of the 256-byte working register areas in the register file. Using the register pointers RP0 and RP1, you can select two 8-byte register slices at one time as active working register space. After a reset, RP1 points to address C8H in register set 1, selecting the 8-byte working register slice C8H–CFH.

.2–0

Not used for the S3C821A

SCLKCON — Sub-System Clock Control Register **FFH** **Set 1, Bank 1**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	–	–	–	–	–	–	–	0
Read/Write	–	–	–	–	–	–	–	R/W
Addressing Mode	Register addressing only							

.7–.1

Not used for the S3C821A

.0

Sub-System Clock Stop Enable Bit	
0	Enable sub-system clock
1	Disable sub-system clock

NOTES:

1. The sub oscillator can be halted only by SCLKCON.0. In sub-oscillation mode, sub-oscillation stop can be released only by a reset operation.
2. When sub and main oscillator are halted in main operating mode, stop mode can be released by an external interrupt or a reset operation.

SIOCON — SIO Control Register

F5H

Set 1, Bank 0

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	1	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
Addressing Mode	Register addressing mode only							

.7 SIO Shift Clock Selection Bit

0	External clock
1	CPU clock

.6 SIO Interrupt Pending Bit

0	No interrupt pending
0	<i>Clear pending condition (when write)</i>
1	Interrupt is pending

.5 SIO Interrupt Enable Bit

0	Disable SIO interrupt
1	Enable SIO interrupt

.4 SIO Start Edge Selection Bit

0	Falling edge start
1	Rising edge start

.3 SIO Counter Clear and Shifter Start Bit

0	No action (when write)
1	Clear 3-bit counter and start shifting

.2 SIO Shift Operation Enable Bit

0	Disable shifter and counter, retain IRQ status
1	Enable shifter and clock counter, set IRQ flag to "1"

.1 SIO Mode Selection Bit

0	Receive only mode
1	Transmit/receive mode

.0 Data Direction Control Bit

0	MSB first mode
1	LSB first mode

SPH — Stack Pointer (High Byte)**D8H****Set 1**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	x	x
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7–0**Stack Pointer Address (High Byte)**

The high-byte stack pointer value is the upper eight bits of the 16-bit stack pointer address (SP15–SP8). The lower byte of the stack pointer value is located in register SPL (D9H). The SP value is undefined following a reset.

SPL — Stack Pointer (Low Byte)**D9H****Set 1**

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	x	x	x	x	x	x	x	x
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7–0**Stack Pointer Address (Low Byte)**

The low-byte stack pointer value is the lower eight bits of the 16-bit stack pointer address (SP7–SP0). The upper byte of the stack pointer value is located in register SPH (D8H). The SP value is undefined following a reset.

SYM — System Mode Register

DEH

Set 1

Bit Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	–	–	x	x	x	0	0
Read/Write	R/W	–	–	R/W	R/W	R/W	R/W	R/W
Addressing Mode	Register addressing mode only							

.7**Tri-State External Interface Enable Bit**

0	Normal operation (disable tri-state operation)
1	Set external interface lines to high impedance (enable tri-state operation)

.6 and .5

Not used for the S3C821A

.4–.2**Fast Interrupt Level Selection Bits (1)**

0	0	0	IRQ0
0	0	1	IRQ1
0	1	0	IRQ2
0	1	1	Not used for the S3C821A
1	0	0	IRQ4
1	0	1	IRQ5
1	1	0	IRQ6
1	1	1	IRQ7

.1**Fast Interrupt Enable Bit (2)**

0	Disable fast interrupt processing
1	Enable fast interrupt processing

.0**Global Interrupt Enable Bit (3)**

0	Disable all interrupt processing
1	Enable all interrupt processing

NOTES:

1. You can select only one interrupt level at a time for fast interrupt processing.
2. Setting SYM.1 to "1" enables fast interrupt processing for the interrupt level currently selected by SYM.2-SYM.4.
3. Following a reset, you must enable global interrupt processing by executing an EI instruction (not by writing a "1" to SYM.0).

T0CON — Timer 0 Control Register

D2H

Set 1

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
Addressing Mode	Register addressing mode only							

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

0	0	f _{xx} /4,096
0	1	f _{xx} /256
1	0	f _{xx} /8
1	1	External clock input (T0CK)

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

0	0	Interval timer mode
0	1	Capture mode (rising edges, counter running, OVF interrupt can occur)
1	0	Capture mode (falling edges, counter running, OVF interrupt can occur)
1	1	PWM mode (OVF interrupt can occur)

.3**Timer 0 Counter Clear Bit**

0	No effect
1	Clear T0 counter (when write)

.2**Timer 0 Overflow Interrupt Enable Bit** *(note)*

0	Disable T0 overflow interrupt
1	Enable T0 overflow interrupt

.1**Timer 0 Interrupt Enable Bit**

0	Disable T0 interrupt
1	Enable T0 interrupt

.0**Timer 0 Interrupt Pending Bit**

0	No T0 interrupt pending (when read)
0	<i>Clear T0 interrupt pending condition (when write)</i>
1	T0 interrupt is pending

NOTE: A timer 0 overflow interrupt pending condition is automatically cleared by hardware. However, the timer 0 match/capture interrupt, IRQ0, vector FCH, must be cleared by the interrupt service routine.

TACON — Timer Counter 1/A Control Register

F3H

Set 1, Bank 0

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
Addressing Mode	Register addressing mode only							

.7 **Timer 1 Operating Mode Selection Bit**

0	Two 8-bit timers mode (Timer A/B)
1	One 16-bit timer mode (Timer 1)

.6–.4 **Timer 1/A Clock Selection Bits**

0	0	0	fxx/1,024
0	0	1	fxx/512
0	1	0	fxx/8
0	1	1	fxx
1	x	x	T1CK (External clock at P2.5)

.3 **Timer 1/A Counter Clear Bit**

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

.2 **Timer 1/A Counter Run Enable Bit**

0	Disable Counter Running
1	Enable Counter Running

.1 **Timer 1/A Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0 **Timer 1/A Interrupt Pending Bit**

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

NOTES:

- "x" means don't care.
- fxx is a selected clock for peripheral hardware.

TBCON — Timer B Control Register

F2H

Set 1, Bank 0

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
Addressing Mode	Register addressing mode only							

.7 and .6

Not used for the S3C821A

.5 and .4**Timer B Clock Selection Bits**

0	0	fxx/1,024
0	1	fxx/512
1	0	fxx/8
1	1	fxx

.3**Timer B Counter Clear Bit**

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

.2**Timer B Counter Run Enable Bit**

0	Disable Counter Running
1	Enable Counter Running

.1**Timer B Interrupt Enable Bit**

0	Disable interrupt
1	Enable interrupt

.0**Timer B Interrupt Pending Bit**

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

NOTE: fxx is a selected clock for peripheral hardware.

WTCON — Watch Timer Control Register

FBH

Set 1, Bank 0

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
Addressing Mode	Register addressing mode only							

.7 Buzzer Signal Enable Bit

0	Disable buzzer signal output
1	Enable buzzer signal output

.6 Watch Timer Enable Bit

0	Disable watch timer; clear frequency dividing circuit
1	Enable watch timer

.5 and .4 Buzzer Signal Selection Bits

0	0	0.5 kHz buzzer (BUZ) signal output
0	1	1 kHz buzzer (BUZ) signal output
1	0	2 kHz buzzer (BUZ) signal output
1	1	4 kHz buzzer (BUZ) signal output

.3 Watch Timer Speed Selection Bit

0	$f_w/2^{14}$ (0.5 sec) interval
1	$f_w/2^7$ (3.91 ms) interval

.2 Watch Timer Clock Selection Bit

0	Main system clock divided by 2^7 ($f_x/128$)
1	Sub system clock (f_{xt})

.1 Watch Timer Interrupt Pending Bit

0	No interrupt pending
0	<i>Clear pending bit (when write)</i>
1	Interrupt is pending

.0 Watch Timer Interrupt Enable Bit

0	Disable interrupt
1	Enable interrupt

5

INTERRUPT STRUCTURE

OVERVIEW

The S3C8-series interrupt structure has three basic components: levels, vectors, and sources. The SAM8 CPU recognizes up to eight interrupt levels and supports up to 128 interrupt vectors. When a specific interrupt level has more than one vector address, the vector priorities are established in hardware. A vector address can be assigned to one or more sources.

Levels

Interrupt levels are the main unit for interrupt priority assignment and recognition. All peripherals and I/O blocks can issue interrupt requests. In other words, peripheral and I/O operations are interrupt-driven. There are eight possible interrupt levels: IRQ0–IRQ7, also called level 0–level 7. Each interrupt level directly corresponds to an interrupt request number (IRQn). The total number of interrupt levels used in the interrupt structure varies from device to device. The S3C821A interrupt structure recognizes seven interrupt levels.

The interrupt level numbers 0 through 7 do not necessarily indicate the relative priority of the levels. They are just identifiers for the interrupt levels that are recognized by the CPU. The relative priority of different interrupt levels is determined by settings in the interrupt priority register, IPR. Interrupt group and subgroup logic controlled by IPR settings lets you define more complex priority relationships between different levels.

Vectors

Each interrupt level can have one or more interrupt vectors, or it may have no vector address assigned at all. The maximum number of vectors that can be supported for a given level is 128 (The actual number of vectors used for S3C8-series devices is always much smaller). If an interrupt level has more than one vector address, the vector priorities are set in hardware. S3C821A uses nineteen vectors.

Sources

A source is any peripheral that generates an interrupt. A source can be an external pin or a counter overflow. Each vector can have several interrupt sources. In the S3C821A interrupt structure, there are nineteen possible interrupt sources.

When a service routine starts, the respective pending bit should be either cleared automatically by hardware or cleared "manually" by program software. The characteristics of the source's pending mechanism determine which method would be used to clear its respective pending bit.

INTERRUPT TYPES

The three components of the S3C8 interrupt structure described before — levels, vectors, and sources — are combined to determine the interrupt structure of an individual device and to make full use of its available interrupt logic. There are three possible combinations of interrupt structure components, called interrupt types 1, 2, and 3. The types differ in the number of vectors and interrupt sources assigned to each level (see Figure 5-1):

- Type 1: One level (IRQ_n) + one vector (V₁) + one source (S₁)
- Type 2: One level (IRQ_n) + one vector (V₁) + multiple sources (S₁ – S_n)
- Type 3: One level (IRQ_n) + multiple vectors (V₁ – V_n) + multiple sources (S₁ – S_n, S_{n+1} – S_{n+m})

In the S3C821A microcontroller, two interrupt types are implemented.

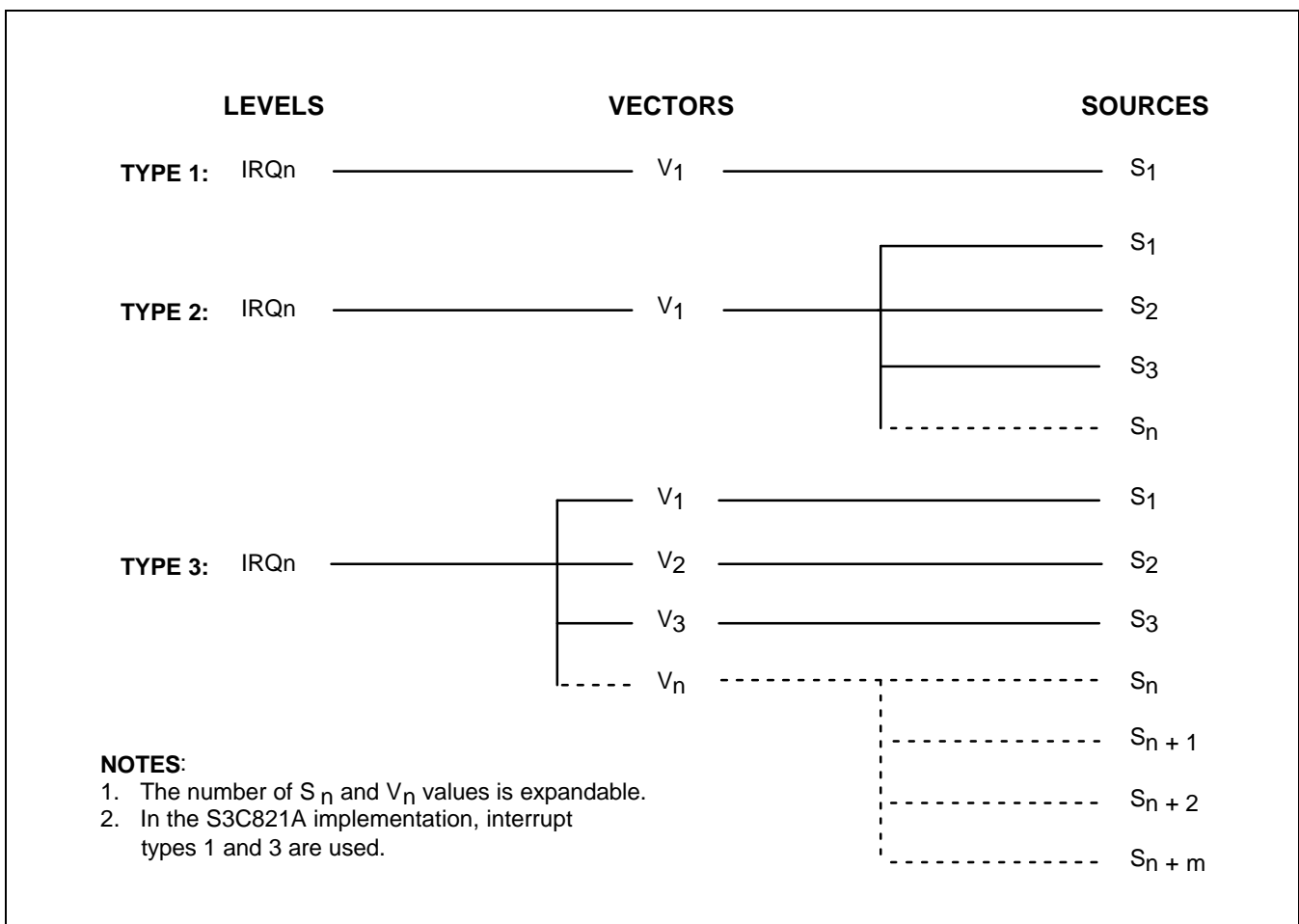


Figure 5-1. S3C8-Series Interrupt Types

S3C821A INTERRUPT STRUCTURE

The S3C821A microcontroller supports nineteen interrupt sources. All nineteen of the interrupt sources have a corresponding interrupt vector address. Seven interrupt levels are recognized by the CPU in this device-specific interrupt structure, as shown in Figure 5-2.

When multiple interrupt levels are active, the interrupt priority register (IPR) determines the order in which contending interrupts are to be serviced. If multiple interrupts occur within the same interrupt level, the interrupt with the lowest vector address is usually processed first (The relative priorities of multiple interrupts within a single level are fixed in hardware).

When the CPU grants an interrupt request, interrupt processing starts. All other interrupts are disabled and the program counter value and status flags are pushed to stack. The starting address of the service routine is fetched from the appropriate vector address (plus the next 8-bit value to concatenate the full 16-bit address) and the service routine is executed.

LEVEL	VECTOR		SOURCE	RESET/CLEAR
RESET	100H	_____	Basic timer overflow	H/W
IRQ0	FCH	_____ 1	Timer 0 match/capture	S/W
	FAH	_____ 0	Timer 0 overflow	H/W
IRQ1	F6H	_____ 1	Timer 1/A match	S/W
	F4H	_____ 0	Timer B match	S/W
IRQ2	F0H	_____	SIO interrupt	S/W
IRQ4	F2H	_____	Watch timer overflow	S/W
IRQ5	DEH	_____ 3	P2.7 external interrupt	S/W
	DCH	_____ 2	P2.6 external interrupt	S/W
	DAH	_____ 1	P2.5 external interrupt	S/W
	D8H	_____ 0	P2.4 external interrupt	S/W
IRQ6	D6H	_____ 3	P4.7 external interrupt	S/W
	D4H	_____ 2	P4.6 external interrupt	S/W
	D2H	_____ 1	P4.5 external interrupt	S/W
	D0H	_____ 0	P4.4 external interrupt	S/W
IRQ7	C6H	_____ 3	P4.3 external interrupt	S/W
	C4H	_____ 2	P4.2 external interrupt	S/W
	C2H	_____ 1	P4.1 external interrupt	S/W
	C0H	_____ 0	P4.0 external interrupt	S/W

NOTE: For interrupt levels with two or more vectors, the lowest vector address usually has the highest priority. For example, FAH has higher priority (0) than FCH (1) within the level IRQ0. These priorities are fixed in hardware.

Figure 5-2. S3C821A Interrupt Structure

INTERRUPT VECTOR ADDRESSES

All interrupt vector addresses for the S3C821A interrupt structure are stored in the vector address area of the internal 48-K byte ROM, 0H-BFFFH (see Figure 5-3).

You can allocate unused locations in the vector address area as normal program memory. If you do so, please be careful not to overwrite any of the stored vector addresses (Table 5-1 lists all vector addresses).

The program reset address in the ROM is 0100H.

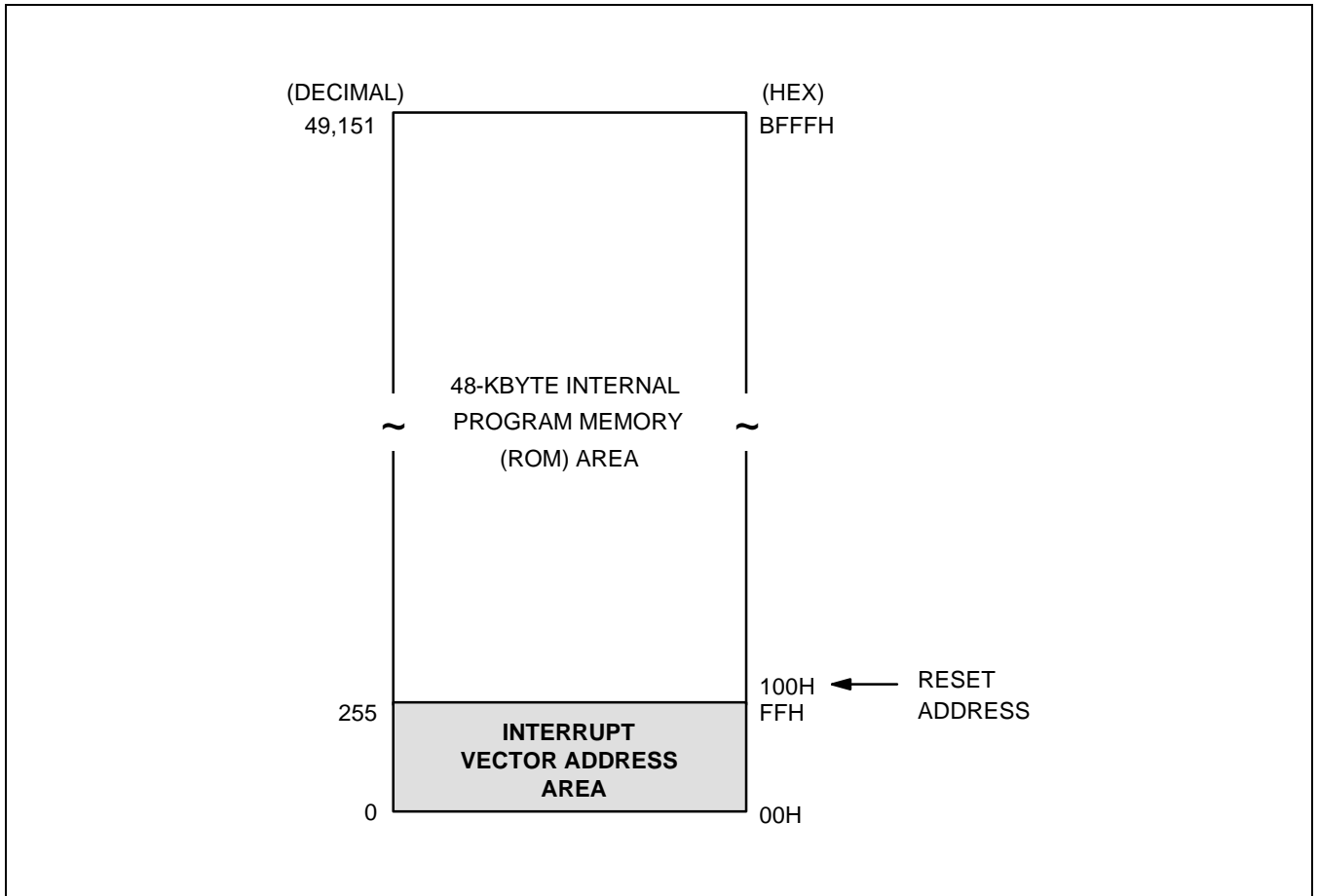


Figure 5-3. ROM Vector Address Area

Table 5-1. S3C821A Interrupt Vectors

Vector Address		Interrupt Source	Request		Reset/Clear	
Decimal Value	Hex Value		Interrupt Level	Priority in Level	H/W	S/W
256	100H	Basic timer overflow	RESET	–	√	
252	FCH	Timer 0 (match/capture)	IRQ0	1		√
250	FAH	Timer 0 overflow		0	√	
246	F6H	Timer 1/A match	IRQ1	1		√
244	F4H	Timer B match		0		√
240	F0H	SIO interrupt	IRQ2	–		√
242	F2H	Watch timer overflow	IRQ4	–		√
222	DEH	P2.7 external interrupt	IRQ5	3		√
220	DCH	P2.6 external interrupt		2		√
218	DAH	P2.5 external interrupt		1		√
216	D8H	P2.4 external interrupt		0		√
214	D6H	P4.7 external interrupt	IRQ6	3		√
212	D4H	P4.6 external interrupt		2		√
210	D2H	P4.5 external interrupt		1		√
208	D0H	P4.4 external interrupt		0		√
198	C6H	P4.3 external interrupt	IRQ7	3		√
196	C4H	P4.2 external interrupt		2		√
194	C2H	P4.1 external interrupt		1		√
192	C0H	P4.0 external interrupt		0		√

NOTES:

1. Interrupt priorities are identified in inverse order: "0" is the highest priority, "1" is the next highest, and so on.
2. If two or more interrupts within the same level contend, the interrupt with the lowest vector address usually has priority over one with a higher vector address. The priorities within a given level are fixed in hardware.

ENABLE/DISABLE INTERRUPT INSTRUCTIONS (EI, DI)

Executing the Enable Interrupts (EI) instruction globally enables the interrupt structure. All interrupts are then serviced as they occur according to the established priorities.

NOTE

The system initialization routine executed after a reset must always contain an EI instruction to globally enable the interrupt structure.

During the normal operation, you can execute the DI (Disable Interrupt) instruction at any time to globally disable interrupt processing. The EI and DI instructions change the value of bit 0 in the SYM register.

SYSTEM-LEVEL INTERRUPT CONTROL REGISTERS

In addition to the control registers for specific interrupt sources, four system-level registers control interrupt processing:

- The interrupt mask register, IMR, enables (un-masks) or disables (masks) interrupt levels.
- The interrupt priority register, IPR, controls the relative priorities of interrupt levels.
- The interrupt request register, IRQ, contains interrupt pending flags for each interrupt level (as opposed to each interrupt source).
- The system mode register, SYM, enables or disables global interrupt processing (SYM settings also enable fast interrupts and control the activity of external interface, if implemented).

Table 5-2. Interrupt Control Register Overview

Control Register	ID	R/W	Function Description
Interrupt mask register	IMR	R/W	Bit settings in the IMR register enable or disable interrupt processing for each of the seven interrupt levels: IRQ0–IRQ2, and IRQ4–IRQ7.
Interrupt priority register	IPR	R/W	Controls the relative processing priorities of the interrupt levels. The seven levels of S3C821A are organized into three groups: A, B, and C. Group A is IRQ0 and IRQ1, group B is IRQ2 and IRQ4, and group C is IRQ5, IRQ6, and IRQ7.
Interrupt request register	IRQ	R	This register contains a request pending bit for each interrupt level.
System mode register	SYM	R/W	This register enables/disables fast interrupt processing, dynamic global interrupt processing, and external interface control (An external memory interface is implemented in the S3C821A microcontroller).

INTERRUPT PROCESSING CONTROL POINTS

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

- Global interrupt enable and disable (by EI and DI instructions or by direct manipulation of SYM.0)
- Interrupt level enable/disable settings (IMR register)
- Interrupt level priority settings (IPR register)
- Interrupt source enable/disable settings in the corresponding peripheral control registers

NOTE

When writing an application program that handles interrupt processing, be sure to include the necessary register file address (register pointer) information.

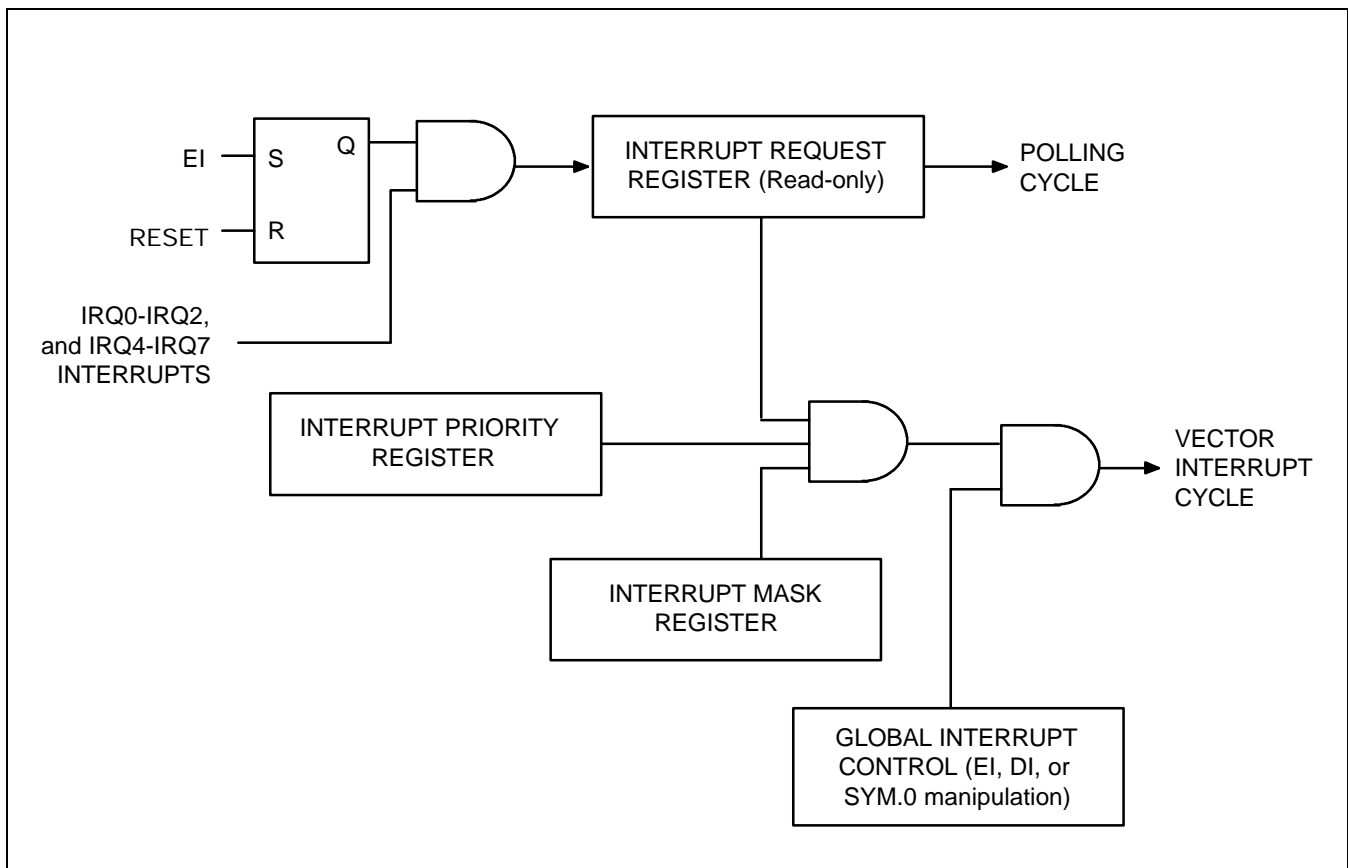


Figure 5-4. Interrupt Function Diagram

PERIPHERAL INTERRUPT CONTROL REGISTERS

For each interrupt source there is one or more corresponding peripheral control registers that let you control the interrupt generated by the related peripheral (see Table 5-3).

Table 5-3. Interrupt Source Control and Data Registers

Interrupt Source	Interrupt Level	Register(s)	Location(s) in Set 1
Timer 0 match/capture Timer 0 overflow	IRQ0	T0CON (note) T0DATA	D2H D1H
Timer 1/Timer A match Timer B match	IRQ1	TACON, TBCON TADATA, TBDATA	F3H, F2H, bank 0 F1H, F0H, bank 0
SIO interrupt	IRQ2	SIOPS, SIOCON SIO	F6H, F5H, bank 0 F4H, bank 0
Watch timer overflow	IRQ4	WTCON	FBH, bank 0
P2.7 external interrupt P2.6 external interrupt P2.5 external interrupt P2.4 external interrupt	IRQ5	P2CONH P2INT	E4H, bank 1 E6H, bank 0
P4.7 external interrupt P4.6 external interrupt P4.5 external interrupt P4.4 external interrupt	IRQ6	P4CONH P4INT P4PND P4EDGE	E8H, bank 1 E7H, bank 0 E8H, bank 0 E9H, bank 0
P4.3 external interrupt P4.2 external interrupt P4.1 external interrupt P4.0 external interrupt	IRQ7	P4CONL P4INT P4PND P4EDGE	E9H, bank 1 E7H, bank 0 E8H, bank 0 E9H, bank 0

NOTE: Because the timer 0 overflow interrupt is cleared by hardware, the T0CON register controls only the enable/disable functions. The T0CON register contains enable/disable and pending bits for the timer 0 match/capture interrupt.

SYSTEM MODE REGISTER (SYM)

The system mode register, SYM (set 1, DEH), is used to globally enable and disable interrupt processing and to control fast interrupt processing (see Figure 5-5).

A reset clears SYM.7, SYM.1, and SYM.0 to "0". The 3-bit value for fast interrupt level selection, SYM.4-SYM.2, is undetermined.

The instructions EI and DI enable and disable global interrupt processing, respectively, by modifying the bit 0 value of the SYM register. In order to enable interrupt processing, an Enable Interrupt (EI) instruction must be included in the initialization routine, which follows a reset operation. Although you can manipulate SYM.0 directly to enable and disable interrupts during the normal operation, it is recommended to use the EI and DI instructions for this purpose.

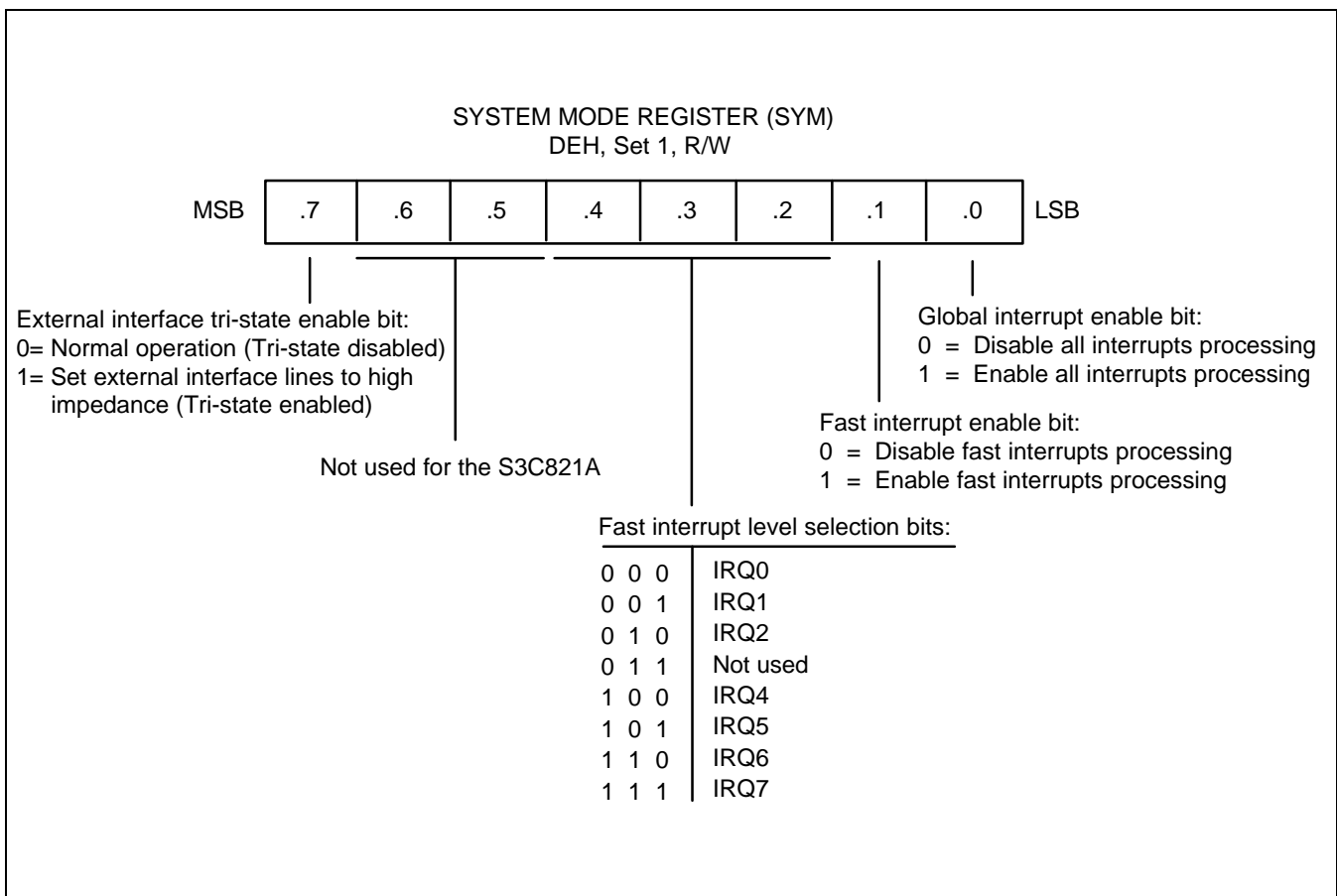


Figure 5-5. System Mode Register (SYM)

INTERRUPT MASK REGISTER (IMR)

The interrupt mask register, IMR (set 1, DDH) is used to enable or disable interrupt processing for individual interrupt levels. After a reset, all IMR bit values are undetermined and must therefore be written to their required settings by the initialization routine.

Each IMR bit corresponds to a specific interrupt level: bit 1 to IRQ1, bit 2 to IRQ2, and so on. When the IMR bit of an interrupt level is cleared to "0", interrupt processing for that level is disabled (masked). When you set a level's IMR bit to "1", interrupt processing for the level is enabled (not masked).

The IMR register is mapped to register location DDH in set 1. Bit values can be read and written by instructions using the Register addressing mode.

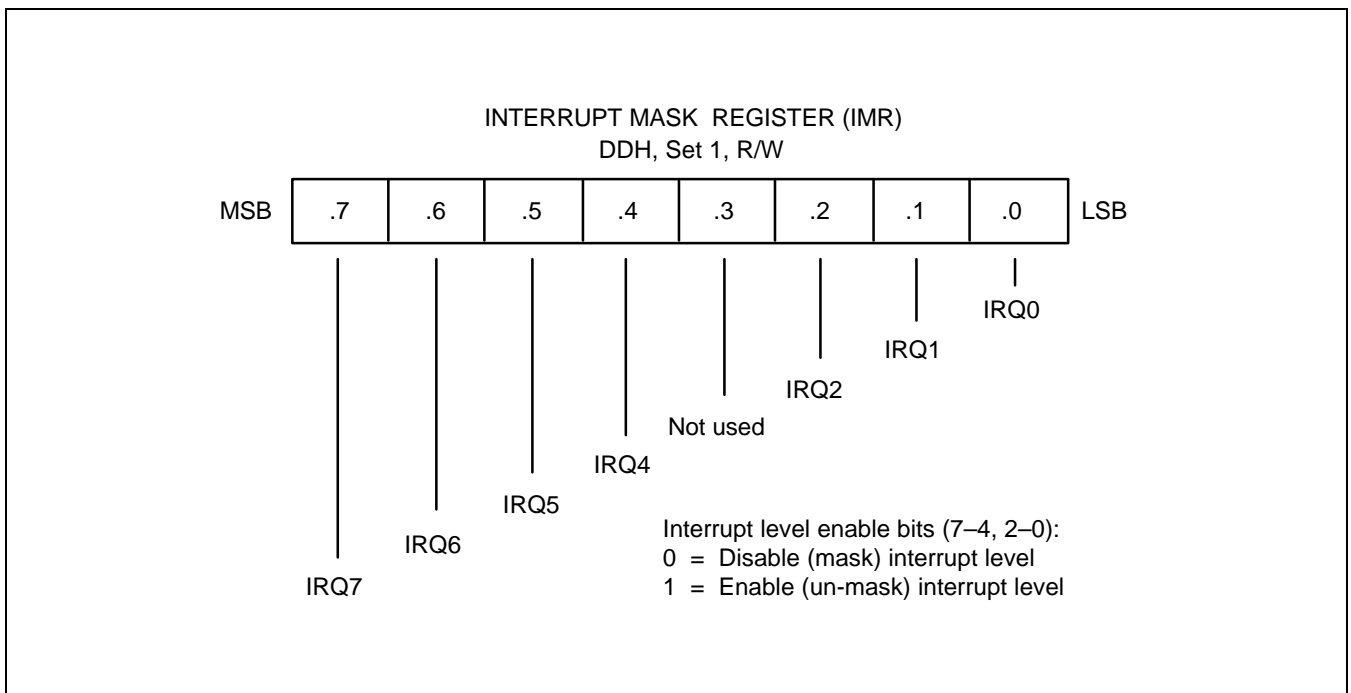


Figure 5-6. Interrupt Mask Register (IMR)

INTERRUPT PRIORITY REGISTER (IPR)

The interrupt priority register, IPR (set 1, bank 0, FFH), is used to set the relative priorities of the interrupt levels in the microcontroller’s interrupt structure. After a reset, all IPR bit values are undetermined and must therefore be written to their required settings by the initialization routine.

When more than one interrupt sources are active, the source with the highest priority level is serviced first. If two sources belong to the same interrupt level, the source with the lower vector address usually has the priority (This priority is fixed in hardware).

To support programming of the relative interrupt level priorities, they are organized into groups and subgroups by the interrupt logic. Please note that these groups (and subgroups) are used only by IPR logic for the IPR register priority definitions (see Figure 5-7):

- Group A IRQ0, IRQ1
- Group B IRQ2, IRQ4
- Group C IRQ5, IRQ6, IRQ7

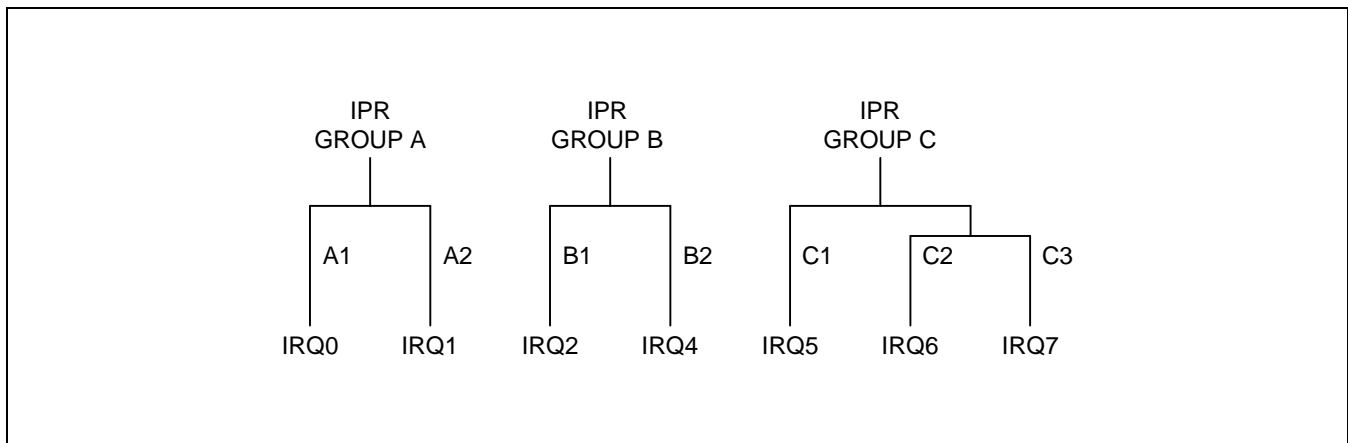


Figure 5-7. Interrupt Request Priority Groups

As you can see in Figure 5-8, IPR.7, IPR.4, and IPR.1 control the relative priority of interrupt groups A, B, and C. For example, the setting "001B" for these bits would select the group relationship B > C > A. The setting "101B" would select the relationship C > B > A.

The functions of the other IPR bit settings are as follows:

- IPR.5 controls the relative priorities of group C interrupts.
- Interrupt group C includes a subgroup that has an additional priority relationship among the interrupt levels 5, 6, and 7. IPR.6 defines the subgroup C relationship. IPR.5 controls the interrupt group C. In the S3C821A implementation, the interrupt level 3 is not used. Therefore, IPR.3 setting is not evaluated.
- IPR.0 controls the relative priority setting of IRQ0 and IRQ1 interrupts.

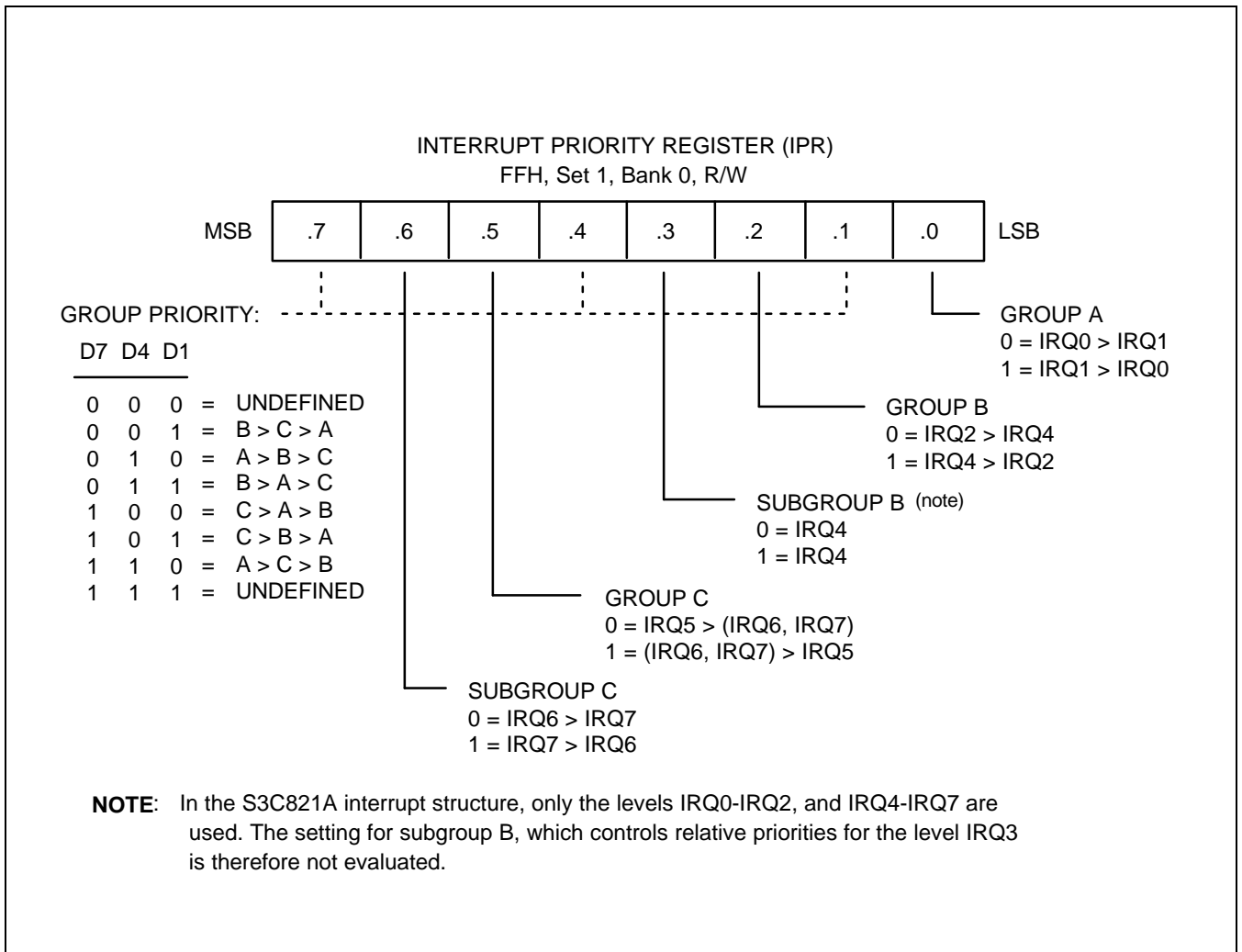


Figure 5-8. Interrupt Priority Register (IPR)

INTERRUPT PENDING FUNCTION TYPES

Overview

There are two types of interrupt pending bits: one type that is automatically cleared by hardware after the interrupt service routine is acknowledged and executed; the other that must be cleared in the interrupt service routine.

Pending Bits Cleared Automatically by Hardware

For interrupt pending bits that are cleared automatically by hardware, interrupt logic sets the corresponding pending bit to "1" when a request occurs. It then issues an IRQ pulse to inform the CPU that an interrupt is waiting to be serviced. The CPU acknowledges the interrupt source by sending an IACK, executes the service routine, and clears the pending bit to "0". This type of pending bit is not mapped and cannot, therefore, be read or written by application software.

In the S3C821A interrupt structure, the timer 0 overflow interrupt (IRQ0) belongs to this category of interrupts in which pending condition is cleared automatically by hardware.

Pending Bits Cleared by the Service Routine

The second type of pending bit is the one that should be cleared by program software. The service routine must clear the appropriate pending bit before a return-from-interrupt subroutine (IRET) occurs. To do this, a "0" must be written to the corresponding pending bit location in the source's mode or control register.

In the S3C821A interrupt structure, pending conditions for all interrupt sources, *except* the timer 0 overflow interrupt, must be cleared in the interrupt service routine.

INTERRUPT SOURCE POLLING SEQUENCE

The interrupt request polling and servicing sequence is as follows:

1. A source generates an interrupt request by setting the interrupt request bit to "1".
2. The CPU polling procedure identifies a pending condition for that source.
3. The CPU checks the source's interrupt level.
4. The CPU generates an interrupt acknowledge signal.
5. Interrupt logic determines the interrupt's vector address.
6. The service routine starts and the source's pending bit is cleared to "0" (by hardware or by software).
7. The CPU continues polling for interrupt requests.

INTERRUPT SERVICE ROUTINES

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

- Interrupt processing must be globally enabled (EI, SYM.0 = "1")
- The interrupt level must be enabled (IMR register)
- The interrupt level must have the highest priority if more than one levels are currently requesting service
- The interrupt must be enabled at the interrupt's source (peripheral control register)

When all 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 interrupt enable bit in the SYM register (SYM.0) to disable all subsequent interrupts.
2. Save the program counter (PC) and status flags to the system stack.
3. Branch to the interrupt vector to fetch the address of the service routine.
4. Pass control to the interrupt service routine.

When the interrupt service routine is completed, the CPU issues an Interrupt Return (IRET). The IRET restores the PC and status flags, setting SYM.0 to "1". It allows the CPU to process the next interrupt request.

GENERATING INTERRUPT VECTOR ADDRESSES

The interrupt vector area in the ROM (00H–FFH) contains the addresses of interrupt service routines that correspond to each level in the interrupt structure. Vectored interrupt processing follows this sequence:

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

NOTE

A 16-bit vector address always begins at an even-numbered ROM address within the range of 00H–FFH.

NESTING OF VECTORED INTERRUPTS

It is possible to nest a higher-priority interrupt request while a lower-priority request is being serviced. To do this, you must follow these steps:

1. Push the current 8-bit interrupt mask register (IMR) value to the stack (PUSH IMR).
2. Load the IMR register with a new mask value that enables only the higher priority interrupt.
3. Execute an EI instruction to enable interrupt processing (a higher priority interrupt will be processed if it occurs).
4. When the lower-priority interrupt service routine ends, restore the IMR to its original value by returning the previous mask value from the stack (POP IMR).
5. Execute an IRET.

Depending on the application, you may be able to simplify the procedure above to some extent.

INSTRUCTION POINTER (IP)

The instruction pointer (IP) is adopted by all the S3C8-series microcontrollers to control the optional high-speed interrupt processing feature called *fast interrupts*. The IP consists of register pair DAH and DBH. The names of IP registers are IPH (high byte, IP15–IP8) and IPL (low byte, IP7–IP0).

FAST INTERRUPT PROCESSING

The feature called *fast interrupt processing* allows an interrupt within a given level to be completed in approximately six clock cycles rather than the usual 22 clock cycles. To select a specific interrupt level for fast interrupt processing, you write the appropriate 3-bit value to SYM.4–SYM.2. Then, to enable fast interrupt processing for the selected level, you set SYM.1 to “1”.

FAST INTERRUPT PROCESSING (Continued)

Two other system registers support fast interrupt processing:

- The instruction pointer (IP) contains the starting address of the service routine (and is later used to swap the program counter values), and
- When a fast interrupt occurs, the contents of the FLAGS register is stored in an unmapped, dedicated register called FLAGS' ("FLAGS prime").

NOTE

For the S3C821A microcontroller, the service routine for any one of the seven interrupt levels: IRQ0–IRQ2, or IRQ4–IRQ7, can be selected for fast interrupt processing.

Procedure for Initiating Fast Interrupts

To initiate fast interrupt processing, follow these steps:

1. Load the start address of the service routine into the instruction pointer (IP).
2. Load the interrupt level number (IRQn) into the fast interrupt selection field (SYM.4–SYM.2)
3. Write a "1" to the fast interrupt enable bit in the SYM register.

Fast Interrupt Service Routine

When an interrupt occurs in the level selected for fast interrupt processing, the following events occur:

1. The contents of the instruction pointer and the PC are swapped.
2. The FLAG register values are written to the FLAGS' ("FLAGS prime") register.
3. The fast interrupt status bit in the FLAGS register is set.
4. The interrupt is serviced.
5. Assuming that the fast interrupt status bit is set, when the fast interrupt service routine ends, the instruction pointer and PC values are swapped back.
6. The content of FLAGS' ("FLAGS prime") is copied automatically back to the FLAGS register.
7. The fast interrupt status bit in FLAGS is cleared automatically.

Relationship to Interrupt Pending Bit Types

As described previously, there are two types of interrupt pending bits: One type that is automatically cleared by hardware after the interrupt service routine is acknowledged and executed; the other that must be cleared by the application program's interrupt service routine. You can select fast interrupt processing for interrupts with either type of pending condition clear function — by hardware or by software.

Programming Guidelines

Remember that the only way to enable/disable a fast interrupt is to set/clear the fast interrupt enable bit in the SYM register, SYM.1. Executing an EI or DI instruction globally enables or disables all interrupt processing, including fast interrupts. If you use fast interrupts, remember to load the IP with a new start address when the fast interrupt service routine ends.

7

CLOCK CIRCUITS

OVERVIEW

The S3C821A microcontroller has two oscillator circuits: a main system clock, and a subsystem clock circuit. The CPU and peripheral hardware operate on the system clock frequency supplied through these circuits. The maximum CPU clock frequency is determined by CLKCON register settings.

SYSTEM CLOCK CIRCUIT

The system clock circuit has the following components:

- External crystal, ceramic resonator, an external clock, or RC oscillation source
- Oscillator stop and wake-up functions
- Programmable frequency divider for the CPU clock (fx divided by 1, 2, 8, or 16 or fxt)
- Clock circuit control register, CLKCON

System Clock Notation

In this document, the following notation is used for descriptions of the system clock:

fx main system clock

fxt subsystem clock

fx selected system clock for peripheral hardware

fcpu CPU clock

MAIN SYSTEM OSCILLATOR CIRCUITS

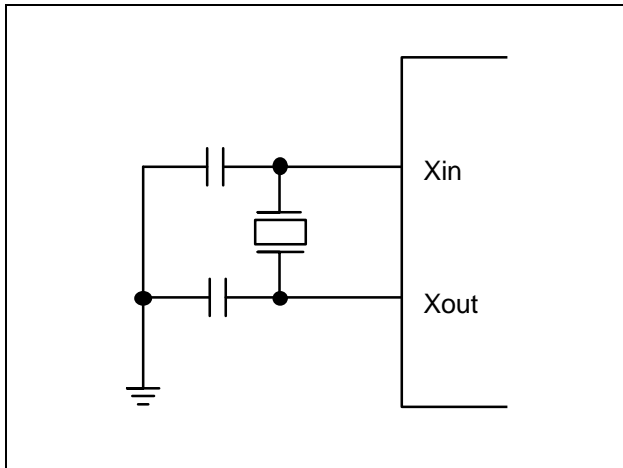


Figure 7-1. Crystal/Ceramic Oscillator

SUBSYSTEM OSCILLATOR CIRCUITS

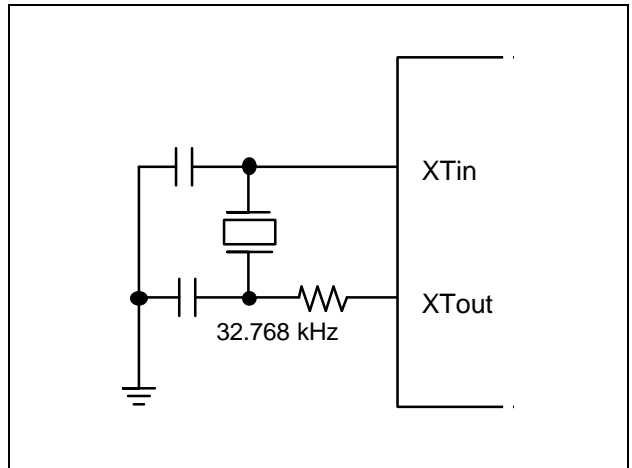


Figure 7-4. Crystal/Ceramic Oscillator

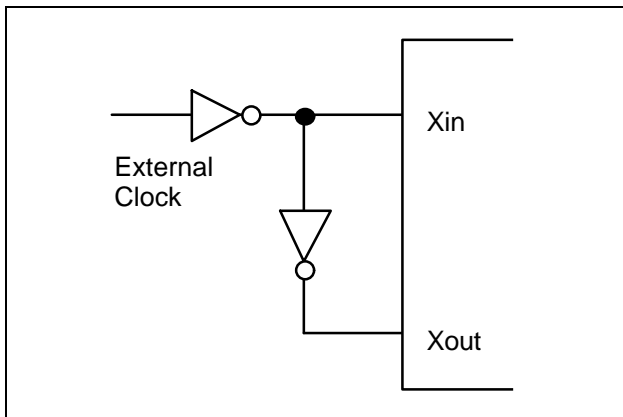


Figure 7-2. External Oscillator

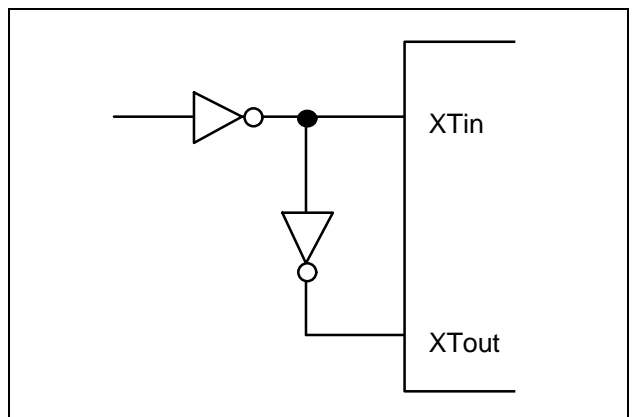


Figure 7-5. External Oscillator

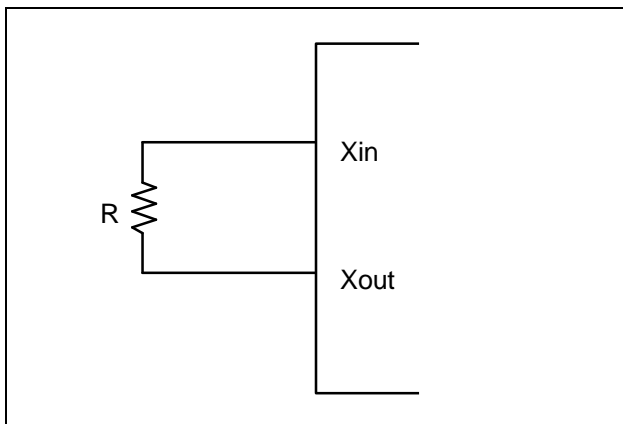


Figure 7-3. RC Oscillator

SYSTEM CLOCK CONTROL REGISTER (CLKCON)

The system clock control register, CLKCON, is located in set 1, address D4H. It is read/write addressable and has the following functions:

- Oscillator IRQ wake-up function enable/disable
- Oscillator frequency divide-by value
- System clock signal selection

CLKCON register settings control 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 $f_x/16$ (the slowest clock speed) is selected as the CPU clock. If necessary, you can raise the CPU clock speed to f_x , $f_x/2$, $f_x/8$, or f_{xt} (subsystem clock).

For the S3C821A microcontroller, the CLKCON.2-CLKCON.0 system clock signature code can be any value (The "101B" setting selects subsystem clock as CPU clock). The reset value for the clock signature code is "000B".

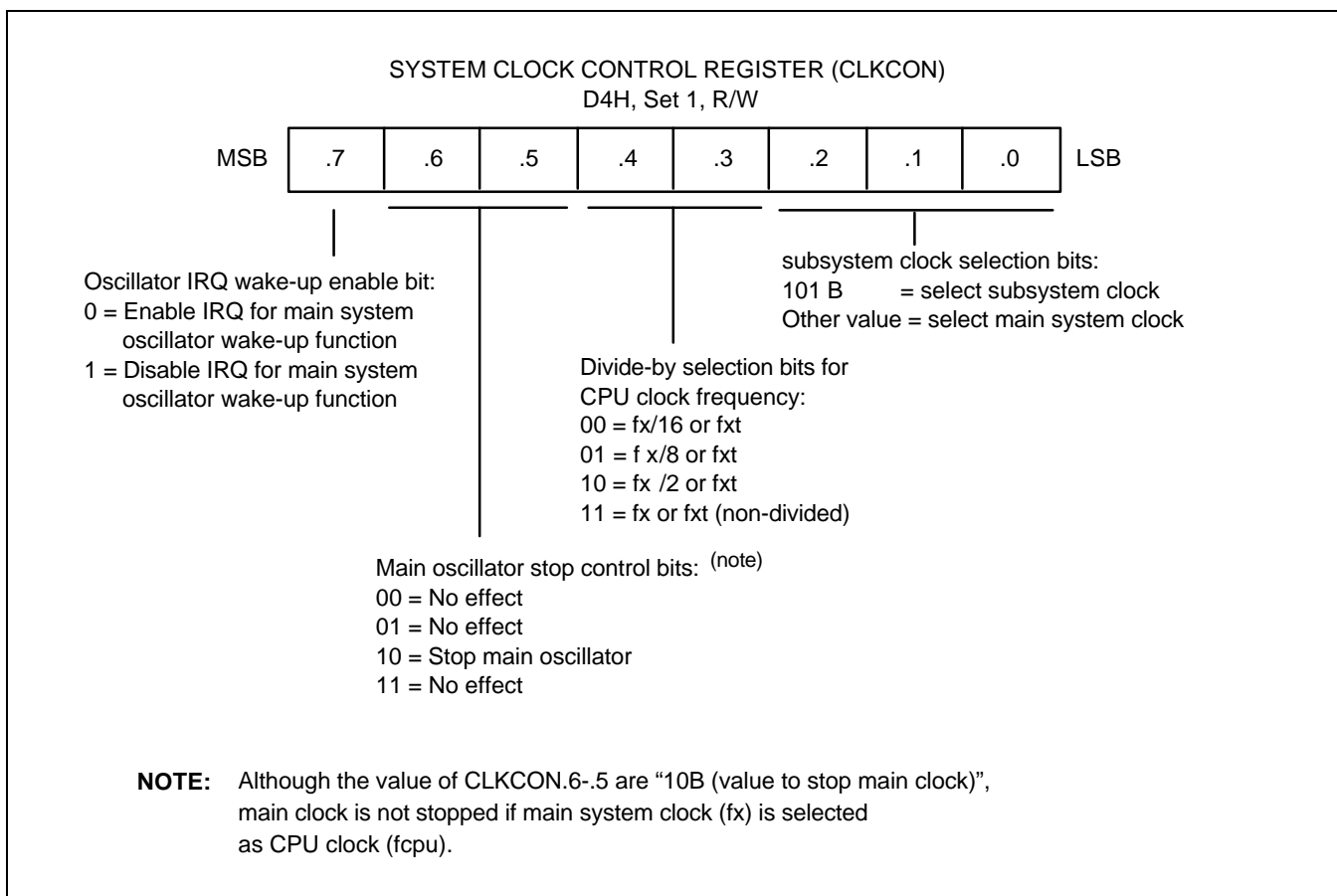


Figure 7-7. System Clock Control Register (CLKCON)

SWITCHING THE CPU CLOCK

Data loadings in the system clock control register, CLKCON, determine whether a main or a sub clock is selected as the CPU clock, the system clock for peripheral hardware, and also how this frequency is to be divided. This makes it possible to switch dynamically between main and sub clocks and to modify operating frequencies.

CLKCON.2–.0 select the main clock (fx) or a sub clock (fxt). CLKCON.6–.5 start or stop main clock oscillation. CLKCON.4–.3 control the frequency divider circuit, and divide the selected fx clock by 1, 2, 8, or 16, or fxt clock by 1.

Let's say that, you are using the default CPU clock (normal operating mode and a main clock of fx/16) and you want to switch from the fx clock to a sub clock and to stop the main clock. To do this, you need to set CLKCON.2–.0 to "101B" and CLKCON.6–.5 to "10B" simultaneously. This switches the clock from fx to fxt and stops main clock oscillation.

The following steps must be taken to switch from a sub clock to the main clock: first, set CLKCON.6–.5 to any value except "10B" to enable main system clock oscillation. Then, after a certain number of machine cycles has elapsed, select the main clock by setting CLKCON.2–.0 to any value except "101B". You must remember that the selected clock (fxx) for the basic timer, timer counter 0/1, watch timer, and A/D converter is the main clock during the interval time. Refer to "Figure 7-6. System Clock Circuit Diagram."

PROGRAMMING TIP — Switching the CPU Clock

1. This example shows how to change from the main clock to the sub clock:

```
MA2SUB    LD          CLKCON,#5DH          ; Switches to the sub clock
          ; Stop the main clock oscillation
          RET
```

2. This example shows how to change from the sub clock to the main clock:

```
SUB2MA    AND          CLKCON,#9FH          ; Start the main clock oscillation
          CALL         DLY20                ; Delay 20 ms
          AND          CLKCON,#98H          ; Switch to the main clock
          RET
DLY20     SRP          #0C0H
          LD           R0,#1BH
DEL       NOP
          NOP
          DJNZ        R0,DEL
          RET
```

SUB-SYSTEM CLOCK CONTROL REGISTER (SCLKCON)

The sub-system clock control register, SCLKCON, is located in set 1, bank 1, address FFH. It is read/write addressable and has sub-system clock stop function. SCLKCON register setting controls whether sub-system clock is halted or not. After a reset, all CLKCON register and SCLKCON register values are cleared to logic zero, both main-system clock and sub-system clock oscillation start, but main-system clock oscillation is selected as the CPU clock.

Main-system clock oscillation or sub-system clock oscillation can be halted by manipulating CLKCON.6–.5 or SCLKCON.0 respectively. At that time, oscillation stop can't be restarted by any interrupt but manipulating CLKCON or SCLKCON.

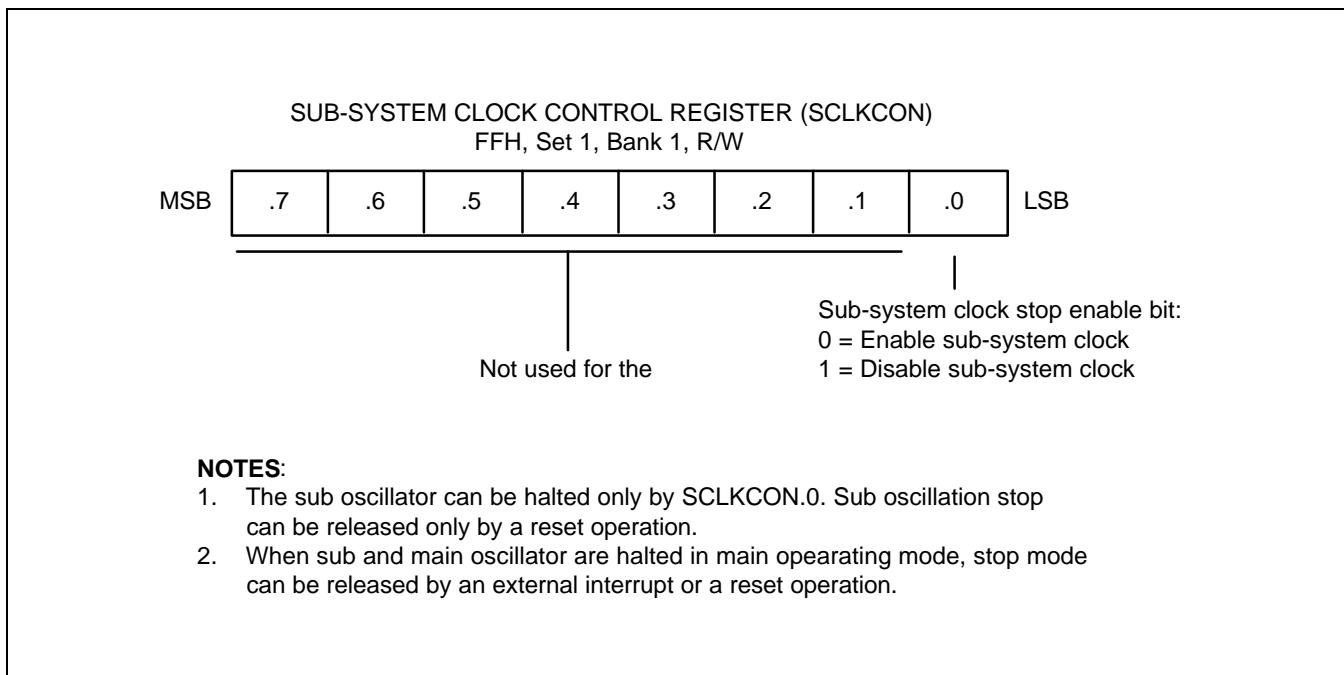


Figure 7-8. Sub-System Clock Control Register (SCLKCON)

8 RESET and POWER-DOWN

SYSTEM RESET

OVERVIEW

During a power-on reset, the voltage at V_{DD} goes to High level and the RESET pin is forced to Low level. The RESET signal is input through a schmitt trigger circuit where it is then synchronized with the CPU clock. This procedure brings the S3C821A into a known operating status.

To allow time for internal CPU clock oscillation to stabilize, the RESET pin must be held to Low level for a minimum time interval after the power supply comes within tolerance. The minimum required time of a reset operation for oscillation stabilization is 1 millisecond.

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

In summary, the following sequence of events occurs during a reset operation:

- All interrupts are disabled.
- The watchdog function (basic timer) is enabled.
- Ports 0, 1, 2, 3, 4, and 5 are set to input mode and all pull-up resistors are disabled for the I/O port pin circuits.
- Peripheral control and data register settings are disabled and reset to their default hardware values (see Table 8-1).
- The program counter (PC) is loaded with the program reset address in the ROM, 0100H.
- When the programmed oscillation stabilization time interval has elapsed, the instruction stored in ROM location 0100H (and 0101H) is fetched and executed.

NOTE

To program the duration of the oscillation stabilization interval, you 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.

HARDWARE RESET VALUES

Table 8-1 list the reset values for CPU and system registers, peripheral control registers, and peripheral data registers following a reset operation. The following notation is used to represent 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 after a reset.
- A dash ("–") means that the bit is either not used or not mapped (but a "0" is read from the bit position).

Table 8-1. Set 1 Register Values After Reset

Register Name	Mnemonic	Address		Bit Values After Reset								
		Dec	Hex	7	6	5	4	3	2	1	0	
Timer 0 counter	T0CNT	208	D0H	0	0	0	0	0	0	0	0	0
Timer 0 data register	T0DATA	209	D1H	1	1	1	1	1	1	1	1	1
Timer 0 control register	T0CON	210	D2H	0	0	0	0	0	0	0	0	0
Basic timer control register	BTCON	211	D3H	0	0	0	0	0	0	0	0	0
Clock control register	CLKCON	212	D4H	0	0	0	0	0	0	0	0	0
System flags register	FLAGS	213	D5H	x	x	x	x	x	x	x	0	0
Register pointer 0	RP0	214	D6H	1	1	0	0	0	–	–	–	–
Register pointer 1	RP1	215	D7H	1	1	0	0	1	–	–	–	–
Stack pointer (high byte)	SPH	216	D8H	x	x	x	x	x	x	x	x	x
Stack pointer (low byte)	SPL	217	D9H	x	x	x	x	x	x	x	x	x
Instruction pointer (high byte)	IPH	218	DAH	x	x	x	x	x	x	x	x	x
Instruction pointer (low byte)	IPL	219	DBH	x	x	x	x	x	x	x	x	x
Interrupt request register	IRQ	220	DCH	0	0	0	0	0	0	0	0	0
Interrupt mask register	IMR	221	DDH	x	x	x	x	–	x	x	x	x
System mode register	SYM	222	DEH	0	–	–	x	x	x	0	0	0
Register page pointer	PP	223	DFH	0	0	0	0	0	0	0	0	0

Table 8-2. Set 1, Bank 0 Register Values After Reset

Register Name	Mnemonic	Address		Bit Values After Reset								
		Dec	Hex	7	6	5	4	3	2	1	0	
Port 0 data register	P0	224	E0H	0	0	0	0	0	0	0	0	0
Port 1 data register	P1	225	E1H	0	0	0	0	0	0	0	0	0
Port 2 data register	P2	226	E2H	0	0	0	0	0	0	0	0	0
Port 3 data register	P3	227	E3H	0	0	0	0	0	0	0	0	0
Port 4 data register	P4	228	E4H	0	0	0	0	0	0	0	0	0
Port 5 data register	P5	229	E5H	–	0	0	0	0	0	0	0	0
Port 2 interrupt control register	P2INT	230	E6H	0	0	0	0	0	0	0	0	0
Port 4 interrupt control register	P4INT	231	E7H	0	0	0	0	0	0	0	0	0
Port 4 interrupt pending register	P4PND	232	E8H	0	0	0	0	0	0	0	0	0
Port 4 interrupt edge select register	P4EDGE	233	E9H	0	0	0	0	0	0	0	0	0
Locations EAH–EDH are not mapped.												
Timer B counter	TBCNT	238	EEH	0	0	0	0	0	0	0	0	0
Timer A counter	TACNT	239	EFH	0	0	0	0	0	0	0	0	0
Timer B data register	TBDATA	240	F0H	1	1	1	1	1	1	1	1	1
Timer A data register	TADATA	241	F1H	1	1	1	1	1	1	1	1	1
Timer B control register	TBCON	242	F2H	–	–	0	0	0	0	0	0	0
Timer A control register	TACON	243	F3H	0	0	0	0	0	0	0	0	0
SIO data register	SIO	244	F4H	0	0	0	0	0	0	0	0	0
SIO control register	SIOCON	245	F5H	1	0	0	0	0	0	0	0	0
SIO prescaler register	SIOPS	246	F6H	0	0	0	0	0	0	0	0	0
ADC control register	ADCON	247	F7H	–	0	0	0	0	0	0	0	0
ADC data register	ADDATA	248	F8H	x	x	x	x	x	x	x	x	x
Location F9H is not mapped.												
LCD control register	LCON	250	FAH	0	0	–	–	0	0	0	0	0
Watch timer mode register	WTCON	251	FBH	0	0	0	0	0	0	0	0	0
Location FCH is not mapped.												
Basic timer counter	BTCNT	253	FDH	x	x	x	x	x	x	x	x	x
External memory timing register	EMT	254	FEH	–	–	–	–	–	–	0	–	–
Interrupt priority register	IPR	255	FFH	x	x	x	x	x	x	x	x	x

NOTES:

1. Except for T0CNT, IRQ, TACNT, TBCNT, ADCON.3, ADDATA, BTCNT, and FIS flag which are read-only, all registers in set 1 are read/write addressable.
2. You cannot use a read-only register as a destination field for the instructions OR, AND, LD, and LDB.

Table 8-3. Set 1, Bank 1 Register Values After Reset

Register Name	Mnemonic	Address		Bit Values After Reset								
		Dec	Hex	7	6	5	4	3	2	1	0	
Port 0 control register	P0CON	224	E0H	0	0	0	0	0	0	0	0	0
Location E1H is not mapped.												
Port 1 control register	P1CON	226	E2H	0	0	0	0	0	0	0	0	0
Location E3H is not mapped.												
Port 2 control register (high byte)	P2CONH	228	E4H	0	0	0	0	0	0	0	0	0
Port 2 control register (low byte)	P2CONL	229	E5H	0	0	0	0	0	0	0	0	0
Port 3 control register (high byte)	P3CONH	230	E6H	0	0	0	0	0	0	0	0	0
Port 3 control register (low byte)	P3CONL	231	E7H	0	0	0	0	0	0	0	0	0
Port 4 control register (high byte)	P4CONH	232	E8H	0	0	0	0	0	0	0	0	0
Port 4 control register (low byte)	P4CONL	233	E9H	0	0	0	0	0	0	0	0	0
Port 5 control register (high byte)	P5CONH	234	EAH	–	–	0	0	0	0	0	0	0
Port 5 control register (low byte)	P5CONL	235	EBH	0	0	0	0	0	0	0	0	0
Locations ECH–FEH are not mapped.												
Sub-clock control register	SCLKCON	255	FFH	–	–	–	–	–	–	–	–	0

NOTE: You cannot use a read-only register as a destination for the instructions OR, AND, LD, or LDB.

POWER-DOWN MODES

STOP MODE

Stop mode is invoked by the instruction STOP. In Stop mode, the operation of the CPU and main oscillator is halted. All peripherals which the main oscillator is selected as a clock source stop also because main oscillator stops. That is, the watch timer and LCD controller will not halted in stop mode if the subsystem clock is selected as watch timer clock source. The data stored in the internal register file are retained in stop mode. Stop mode can be released in one of three ways: by a system reset, by an internal watch timer interrupt (when subsystem clock is selected as clock source of watch timer), or by an external interrupt.

Example: STOP
 NOP
 NOP
 NOP

Using RESET to Release Stop Mode

Stop mode is released when the RESET signal goes active (Low level): all system and peripheral control registers are reset to their default hardware values and the contents of all data registers are retained. When the programmed oscillation stabilization interval has elapsed, the CPU starts the system initialization routine by fetching the program instruction stored in ROM location 0100H.

Using an External Interrupt to Release Stop Mode

External interrupts can be used to release stop mode. For the S3C821A microcontroller, we recommend using the INT0–INT11 interrupt though P2.4–P2.7, P4.0–P4.7.

Using an Internal Interrupt to Release Stop Mode

An internal interrupt, watch timer, can be used to release stop mode because, the watch timer operates in stop mode if the clock source of watch timer is subsystem clock. If system clock is subsystem clock, you can't use any interrupts to release stop mode. That is, you had better use the idle instruction instead of stop one when subsystem clock is selected as the system clock.

Please note the following conditions for Stop mode release:

- If you release stop mode using an internal or external interrupt, the current values in system and peripheral control registers are unchanged.
- If you use an internal or 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.
- If you use an interrupt to release stop mode, the bit-pair setting for CLKCON.4/CLKCON.3 remains unchanged and the currently selected clock value is used.

The internal or 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 stop mode if you are using an external clock source because X_{IN} input must be cleared internally to V_{SS} to reduce current leakage, and do not configure any pins to floating node in stop mode to reduce power consumption.

IDLE MODE

Idle mode is invoked by the instruction IDLE. In Idle mode, CPU operations are halted while some peripherals remain active. During Idle mode, the internal clock signal is gated away from the CPU and from all but the following peripherals, which remain active:

- Interrupt logic
- Basic timer
- Timer 0
- Timer 1 (Timer A and B)
- Watch timer
- LCD controller
- A/D converter

I/O port pins retain the mode (input or output) they had at the time Idle mode was entered. External interface pins are halted by high or low level, in the idle mode.

Idle Mode Release

You can release Idle mode in one of two ways:

1. Execute a reset. All system and peripheral control registers are reset to their default values and the contents of all data registers are retained. The reset automatically selects the *slowest clock (1/16)* because of the hardware reset value for the CLKCON register. If all external interrupts are masked in the IMR register, a reset is the only way you can release Idle mode.
2. Activate any enabled interrupt — internal or external. When you use an interrupt to release Idle mode, the 2-bit CLKCON.4/CLKCON.3 value remains unchanged, and the *currently selected clock* value is used. The interrupt is then serviced. When the return-from-interrupt condition (IRET) occurs, the instruction immediately following the one which initiated Idle mode is executed.

9 I/O PORTS

OVERVIEW

The S3C821A microcontroller has two nibble-programmable and four bit-programmable I/O ports, P0-P5. The ports from P0 to P4 are 8-bit ports and port 5 is a 7-bit port. This gives a total of 47 I/O pins. Each port can be flexibly configured to meet application design requirements. The CPU accesses ports by directly writing or reading port registers. No special I/O instructions are required.

Table 9-1 gives you a general overview of the S3C821A I/O port functions.

Table 9-1. S3C821A Port Configuration Overview

Port	Configuration Options
0	4-bit-programmable I/O port. Pull-up resistors and open-drain outputs are software assignable. Pull-up resistors are automatically disabled for output pins. Configurable as LCD segments/external interface address lines.
1	4-bit-programmable I/O port. Pull-up resistors and open-drain outputs are software assignable. Pull-up resistors are automatically disabled for output pins. Configurable as LCD segment/external interface address and data lines.
2	1-bit-programmable I/O port. Pull-up resistors are software assignable, and automatically disabled for output pins. P2.0–P2.3 can alternately be used as external interface lines. P2.4–P2.7 are configurable as alternate functions or external interrupts at falling edge with noise filters.
3	1-bit-programmable I/O port. Pull-up resistors are software assignable, and automatically disabled for output pins. P3.0–P3.3 can alternately be used as ADC. P3.7 is configurable as an alternate function.
4	1-bit-programmable I/O port. Pull-up resistors and open-drain outputs are software assignable. Pull-up resistors are automatically disabled for output pins. P4.0–P4.7 are configurable as external interrupts at a selectable edge with noise filters.
5	1-bit-programmable I/O port. Pull-up resistors are software assignable, and automatically disabled for output pins. P5.0–P5.3 are configurable as alternate functions. When SCK and SI are used as input, these pins have noise filters.

PORT DATA REGISTERS

Table 9-2 gives you an overview of the register locations of all four S3C821A I/O port data registers. Data registers for ports 0, 1, 2, 3, 4, and 5 have the general format shown in Figure 9-1.

Table 9-2. Port Data Register Summary

Register Name	Mnemonic	Decimal	Hex	Location	R/W
Port 0 data register	P0	224	E0H	Set 1, Bank 0	R/W
Port 1 data register	P1	225	E1H	Set 1, Bank 0	R/W
Port 2 data register	P2	226	E2H	Set 1, Bank 0	R/W
Port 3 data register	P3	227	E3H	Set 1, Bank 0	R/W
Port 4 data register	P4	228	E4H	Set 1, Bank 0	R/W
Port 5 data register	P5	229	E5H	Set 1, Bank 0	R/W

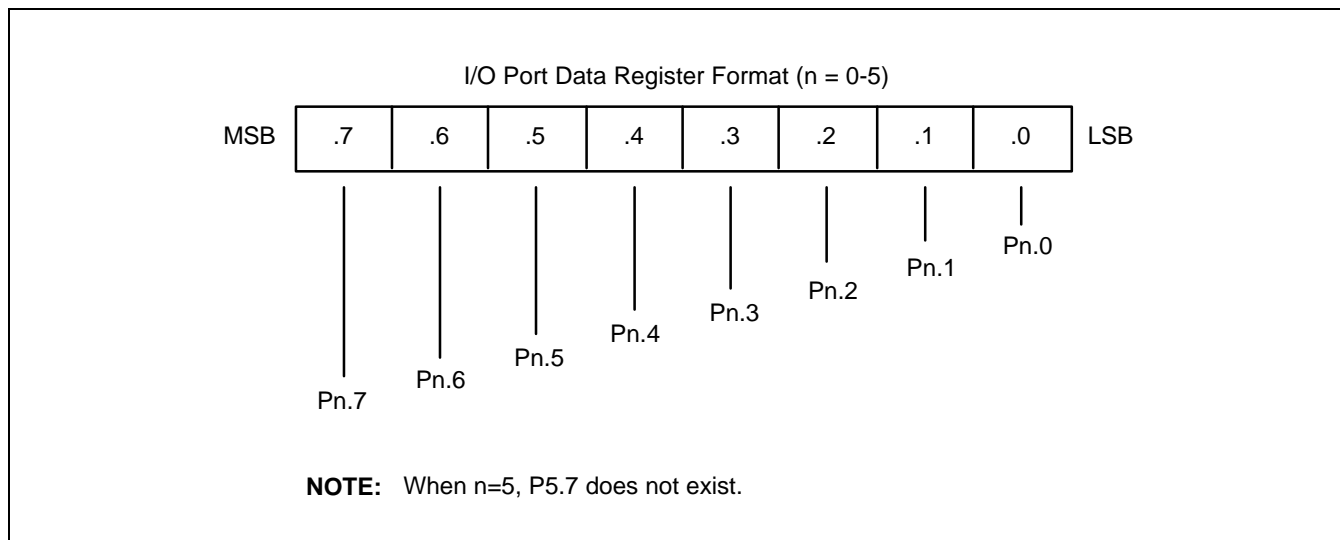


Figure 9-1. S3C821A I/O Port Data Register Format

PORT 0

Port 0 pins P0.0–P0.7 can be configured on a nibble basis for general data input or output. When configured as outputs, the pins in each nibble may optionally be set to open-drain. You can alternately configure port 0 as additional address lines (A8–A15) for the external peripheral interface. It is possible to configure the lower nibble as external interface address lines A8–A11, and to use the upper nibble pins for general I/O.

To access port 0, you should write or read the port 0 data register, P0 (R224, E0H, Bank 0) in set 1. The port 0 data register can't be written, however, when port 0 bits are configured as address lines for the external interface: writing has no effect and reading only loads P0 data register with the state of the pin.

PORT 0 CONTROL REGISTER

The port 0 control register, P0CON (R224, E0H, set 1, Bank 1), controls the direction of the I/O lines and allows an optional selection of pull-up resistor in input mode, and that of open-drain, or push-pull in output modes. The P0CON setting "1xxB" for each nibble configures the pins as external interface lines. The bits 4-7 control the upper nibble pins, P0.4–P0.7, and the bits 0-3 control the lower nibble pins, P0.0–P0.3.

In normal operating mode, a reset operation clears all P0CON register values to "0". If you want to configure an external memory area, you can use a routine to set the P0CON value to "1xx1xxB". This setting correctly configures the address lines A8–A11 (lower nibble) and A12–A15 (upper nibble).

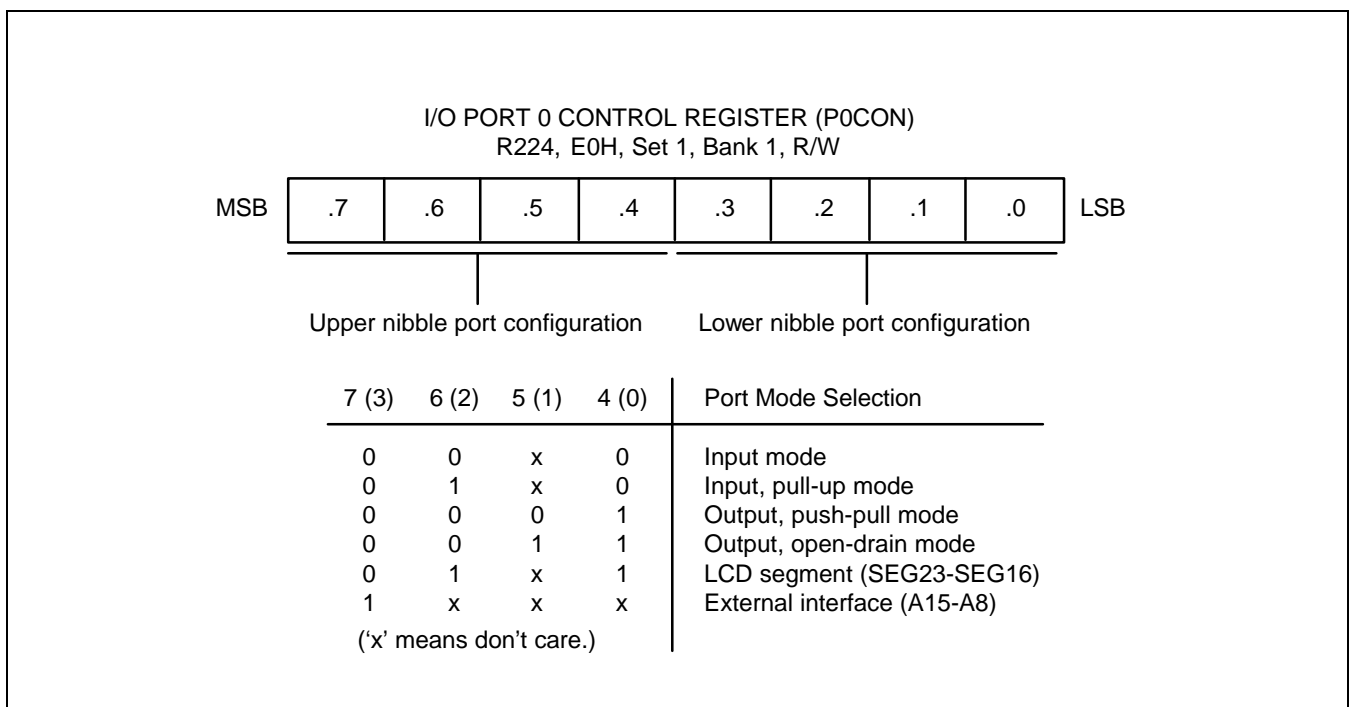


Figure 9-2. Port 0 Control Register (P0CON)

PORT 1

Port 1 is basically identical to port 0, except for its alternate use as multiplexed address/data lines for the external interface. (Port 0 can alternately be configured as additional address lines A8–A15.)

Port 1 pins, P1.0–P1.7, can be configured on a nibble basis for general data input or output. When configured as outputs, the pins in each nibble may optionally be set to open-drain. You can alternately configure port 0 as additional address/data lines (AD0–AD7) for external peripheral interface.

To access port 1, you should write or read the port 1 data register, P1 (R225, E1H, Bank 0) in set 1. The port 1 data register cannot be written, however, when the port 1 bits are configured as address lines for external interface: writing has no effect and reading only loads P1 data register with the state of the pin.

PORT 1 CONTROL REGISTER

The port 1 control register, P1CON (R226, E2H, set 1, Bank 1), controls the direction of the I/O lines and allows an optional selection of pull-up resistor in input mode, and that of open-drain, or push-pull in output mode. The P1CON setting "1xxxB" for each nibble configures the pins as external interface lines. The bits 4–7 control the upper nibble pins, P1.4–P1.7, and the bits 0–3 control the lower nibble pins, P1.0–P1.3.

In normal operating mode a reset operation clears all P1CON register values to "0". If you want to configure an external memory area, you can use a routine to set the P1CON value to "1xxx1xxxB". This setting correctly configures the address/data lines AD0–AD7.

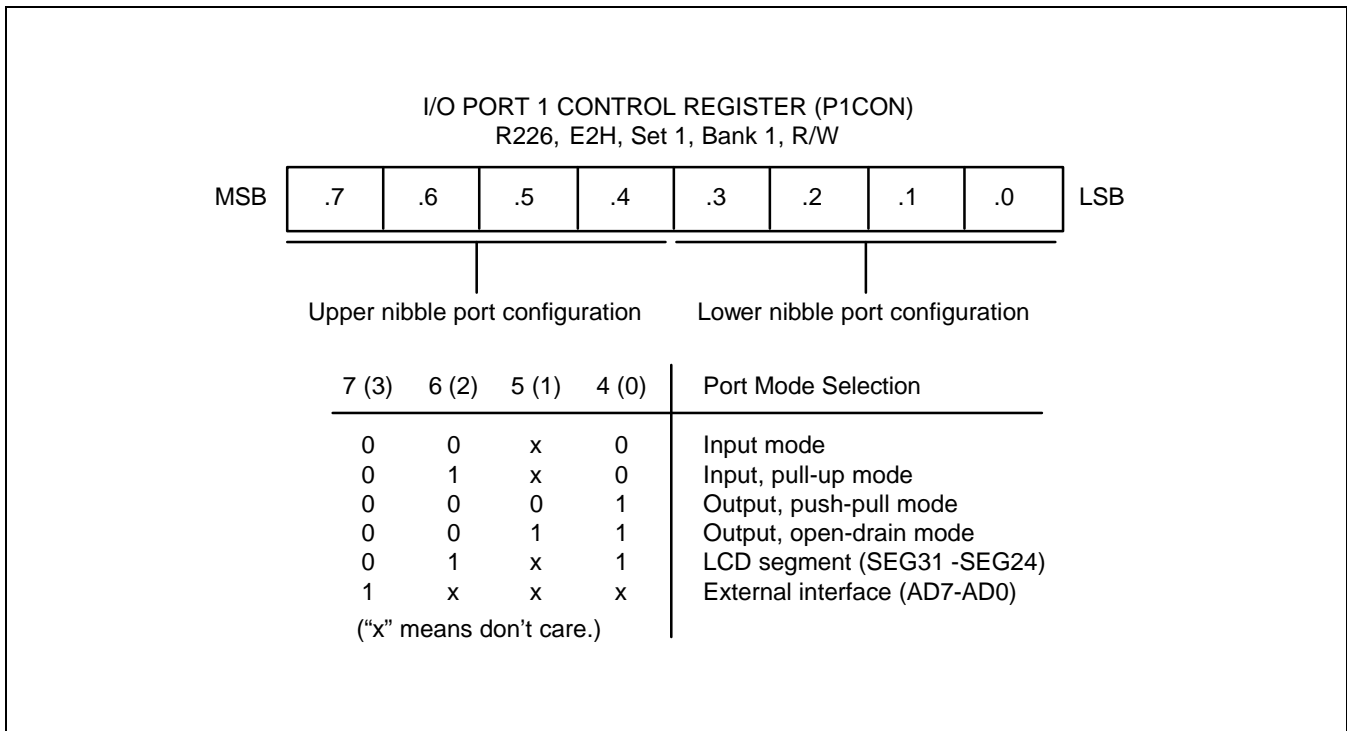


Figure 9-3. Port 1 Control Register (P1CON)

PORT 2

Port 2 is an 8-bit I/O port with individually configurable pins. It is accessed directly by writing or reading the port 2 data register, P2 (R226, E2H, Bank 0) in set 1. You can use port 2 for general I/O, or for the following alternative functions:

- P2.0–P2.3 can be configured as multiplexed external interface bus control lines for AS (address strobe) signals, DR (data read), DW (data write), and DM (data memory).
- P2.4–P2.7 can be configured, respectively, as T0CK (T0 clock input), T1CK (T1 clock input), TA, and TB output.

The special functions that you can program using the port 2 high byte control register must also be enabled in the associated peripheral. Also, when using port 2 pins for functions other than general I/O, you must still set the corresponding port 2 control register value to configure each bit to input or output mode.

PORT 2 CONTROL REGISTERS

Two 8-bit control registers are used to configure port 2 pins: P2CONH (E4H, set 1, Bank 1) for pins P2.4–P2.7 and P2CONL (E5H, set 1, Bank 1) for pins P2.0–P2.3. Each byte contains four bit-pairs and each bit-pair configures one port 2 pin. The P2CONH and the P2CONL registers also control the alternative functions described above.

Port 2 High-Byte Control Register (P2CONH)

Four bit-pairs in the port 2 control register (P2CONH) configure the port 2 pins, P2.4–P2.7, to schmitt trigger input, schmitt trigger input with pull-up resistor, or push-pull output mode.

Table 9-3. Port 2 Data Register Summary (High Nibbles)

P2CONH Bit-Pair	Corresponding Port 2 Pin	Alternate Pin Function
Bits 0 and 1	P2.4	T0 output or capture input (T0CK)
Bits 2 and 3	P2.5	T0 clock input (T1CK)
Bits 4 and 5	P2.6	Clock output (TA)
Bits 6 and 7	P2.7	Clock output (TB)

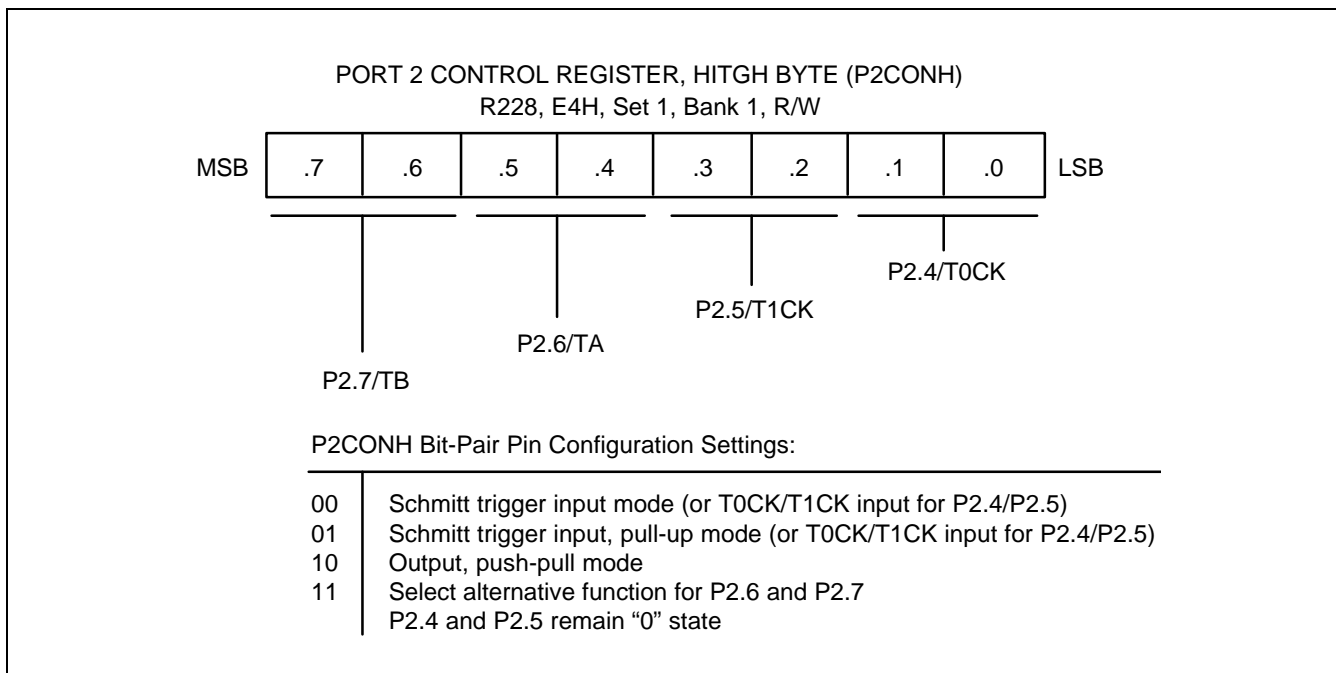


Figure 9-4. Port 2 High-Byte Control Register (P2CONH)

Port 2 Low-Byte Control Register (P2CONL)

The low-byte port 2 pins, P2.0–P2.3, can be configured individually as schmitt trigger inputs, schmitt trigger input with pull-up resistor, or as push-pull outputs. You can alternately configure these pins as multiplexed bus control signal lines for external interface. To select the bus signal function, you must set the related bit-pairs to "11B".

Table 9-4. Port 2 Data Register Summary (Low Nibbles)

P2CONL Bit-Pair	Corresponding Port 2 Pin	Alternate Pin Function
Bits 0 and 1	P2.0	Address strobe (AS)
Bits 2 and 3	P2.1	Data read (DR)
Bits 4 and 5	P2.2	Data write (DW)
Bits 6 and 7	P2.3	Data memory (DM)

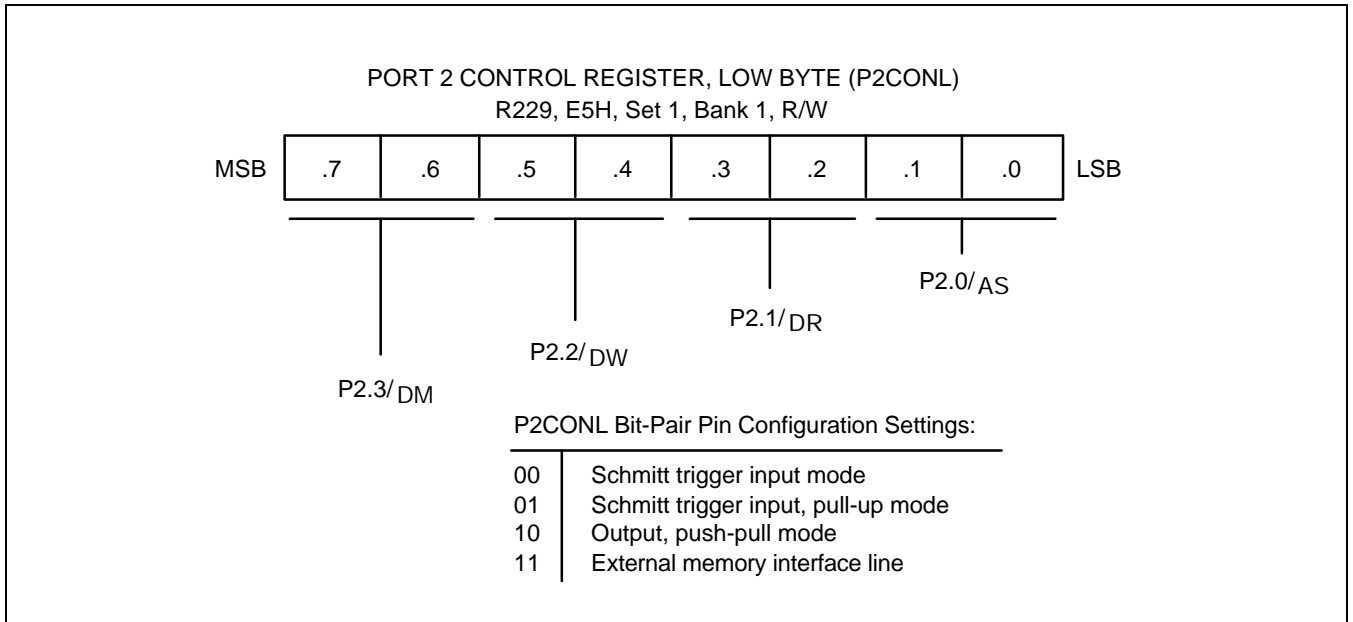


Figure 9-5. Port 2 Low-Byte Control Register (P2CONL)

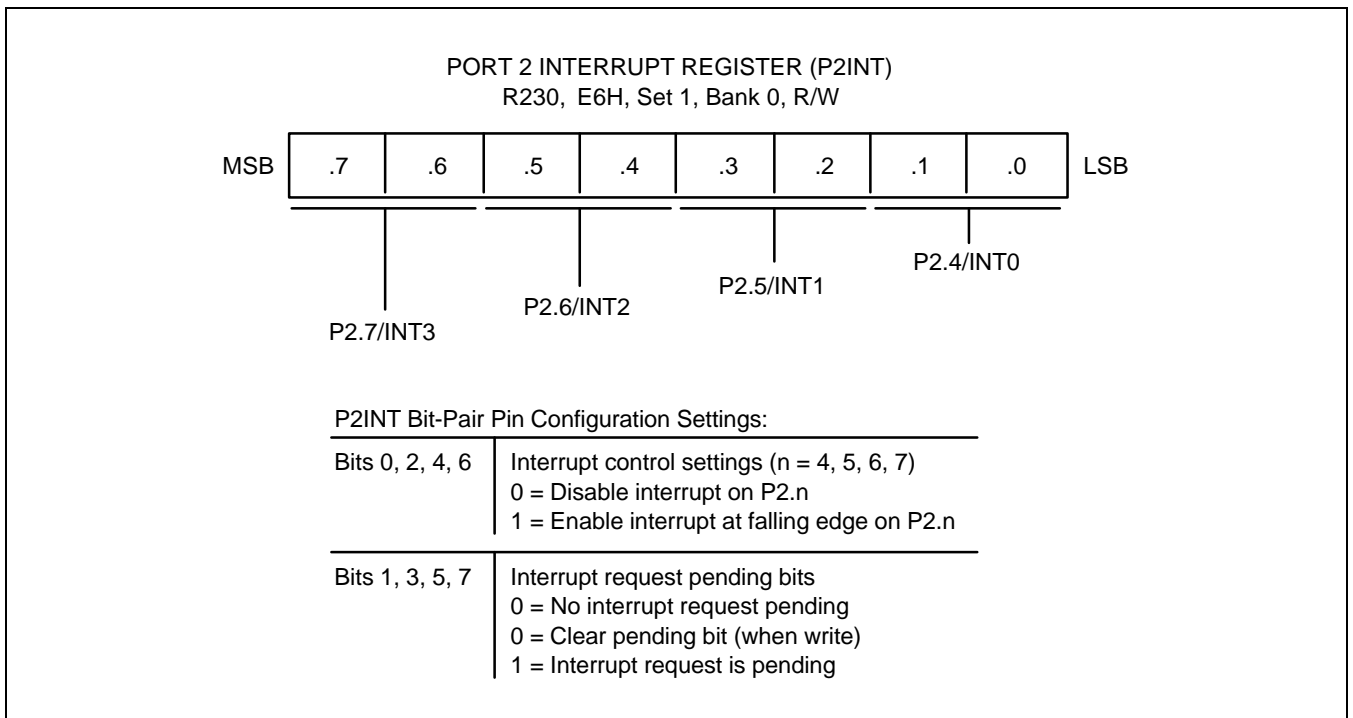


Figure 9-6. Port 2 Interrupt Register (P2INT)

PORT 3

Port 3 can serve as a general-purpose 8-bit I/O port, or support alternative functions (T0CAP input or T0PWM/T0 output for P3.7 and A/D converter inputs for P3.0–P3.3). Port 3 is accessed directly by writing or reading the Port 3 data register, P3 (R227, E3H, Bank 0) in set 1.

- P3.0–P3.3 can be configured, respectively, as ADC0–ADC3 input.
- P3.7 can be configured as T0CAP input, T0PWM, or T0 output.

PORT 3 CONTROL REGISTERS

The direction of each port pin is configured by bit-pair settings in two control registers: P3CONH (high byte, E6H, set 1, Bank 1) and P3CONL (low byte, E7H, set 1, Bank 1). P3CONH controls the pins, P3.4–P3.7, and P3CONL controls the pins, P3.0–P3.3. Both registers are read-write addressable using 1-bit or 8-bit instructions.

There are two input modes: schmitt trigger input or schmitt trigger input with pull-up resistor.

A reset clears all P3CONH and P3CONL bits to logic zero. This configures Port 3 pins to schmitt trigger input.

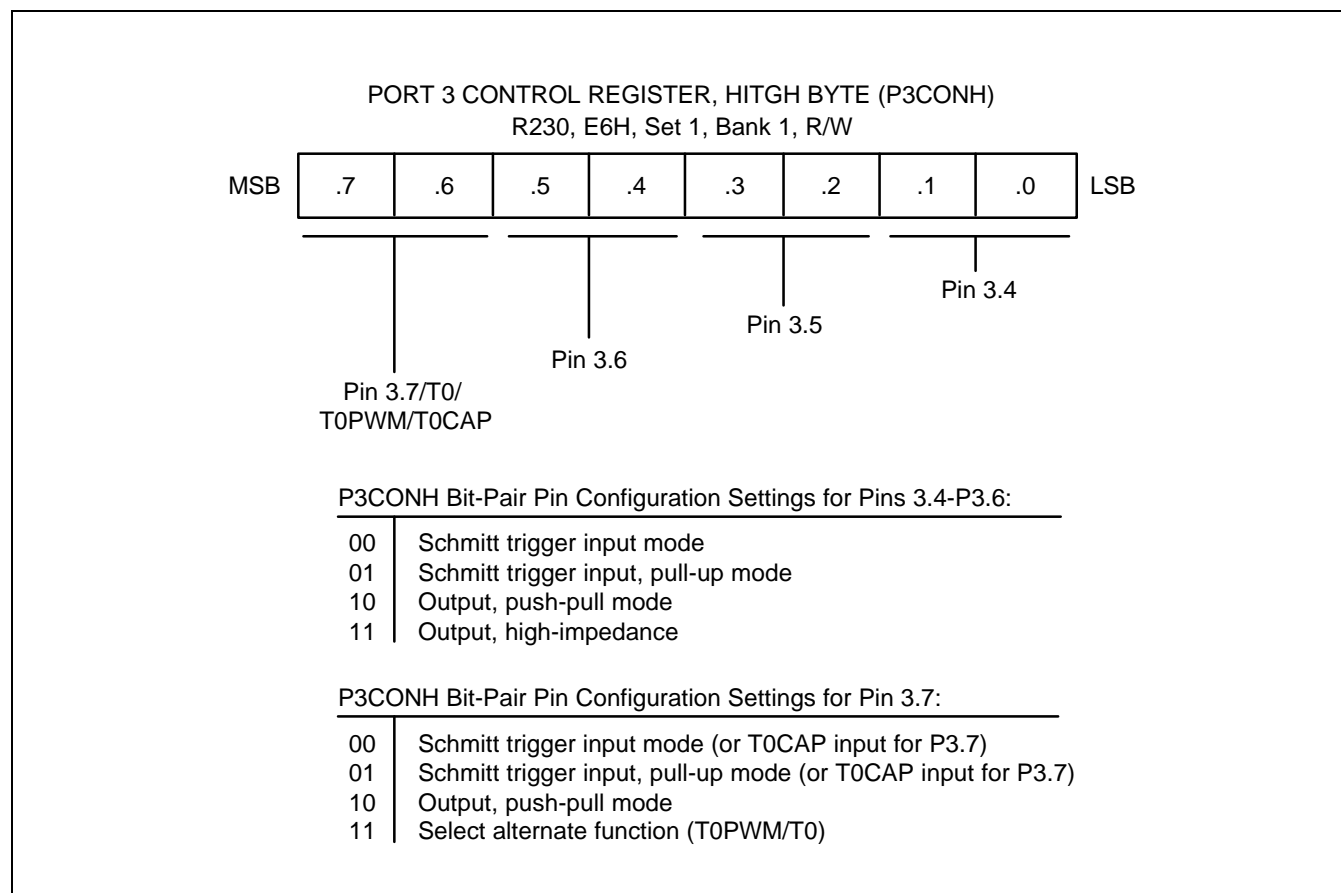


Figure 9-7. Port 3 High-Byte Control Register (P3CONH)

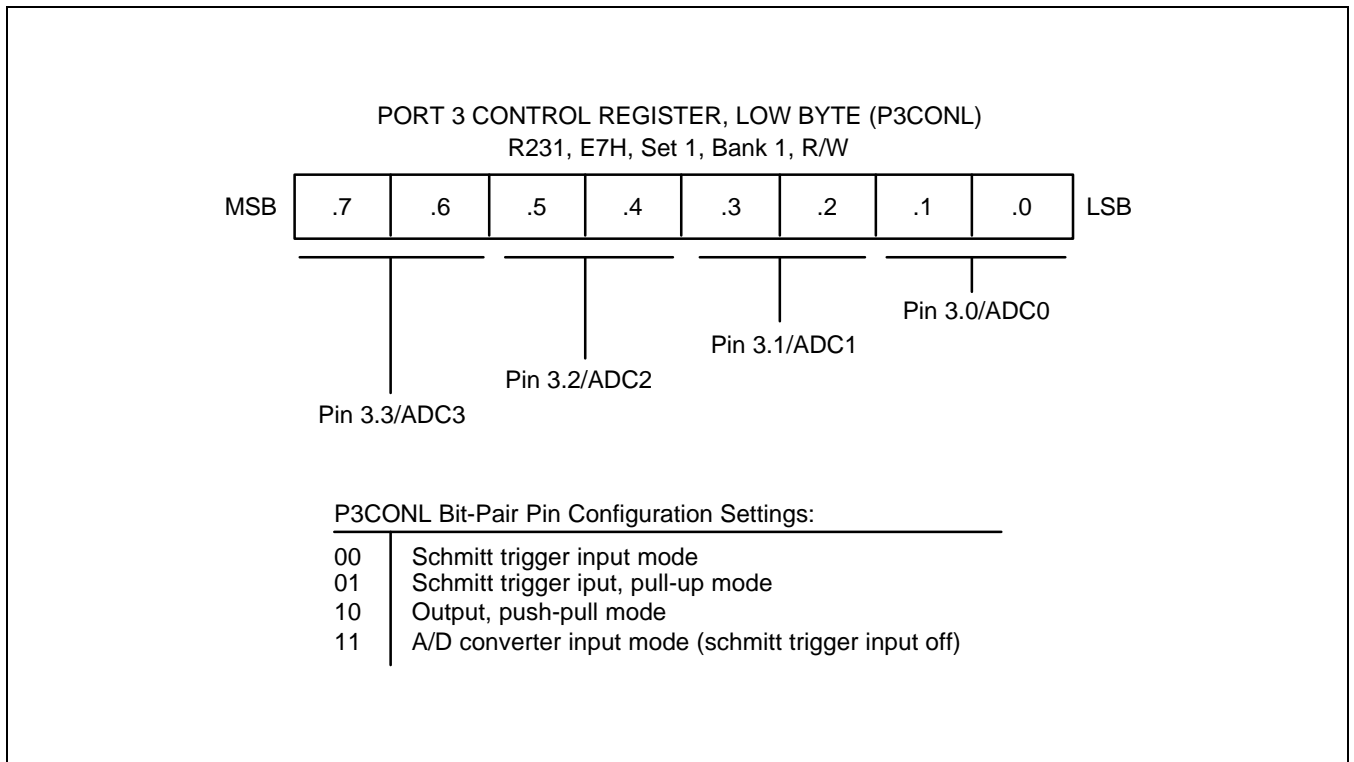


Figure 9-8. Port 3 Low-Byte Control Register (P3CONL)

PORT 4

Port 4 pins P4.0–P4.7 can be configured on a bit-pair setting basis for general data input or output. When configured as outputs, the pins may optionally be set to open-drain. All inputs are schmitt triggered. Port 4 is accessed directly by writing or reading the Port 4 data register, P4 (R228, E4H, Bank 0) in set 1.

PORT 4 CONTROL REGISTERS

The direction of each port pin is configured by bit-pair settings in two control registers: P4CONH (high byte, E8H, set 1, Bank 1) and P4CONL (low byte, E9H, set 1, Bank 1). P4CONH controls the pins, P4.4–P4.7, and P4CONL controls the pins, P4.0–P4.3. Both registers are read-write addressable using 1-bit or 8-bit instructions.

There are two input modes: schmitt trigger input or schmitt trigger input with pull-up resistor.

A reset clears all P4CONH and P4CONL bits to logic zero. This configures Port 4 pins to schmitt trigger input.

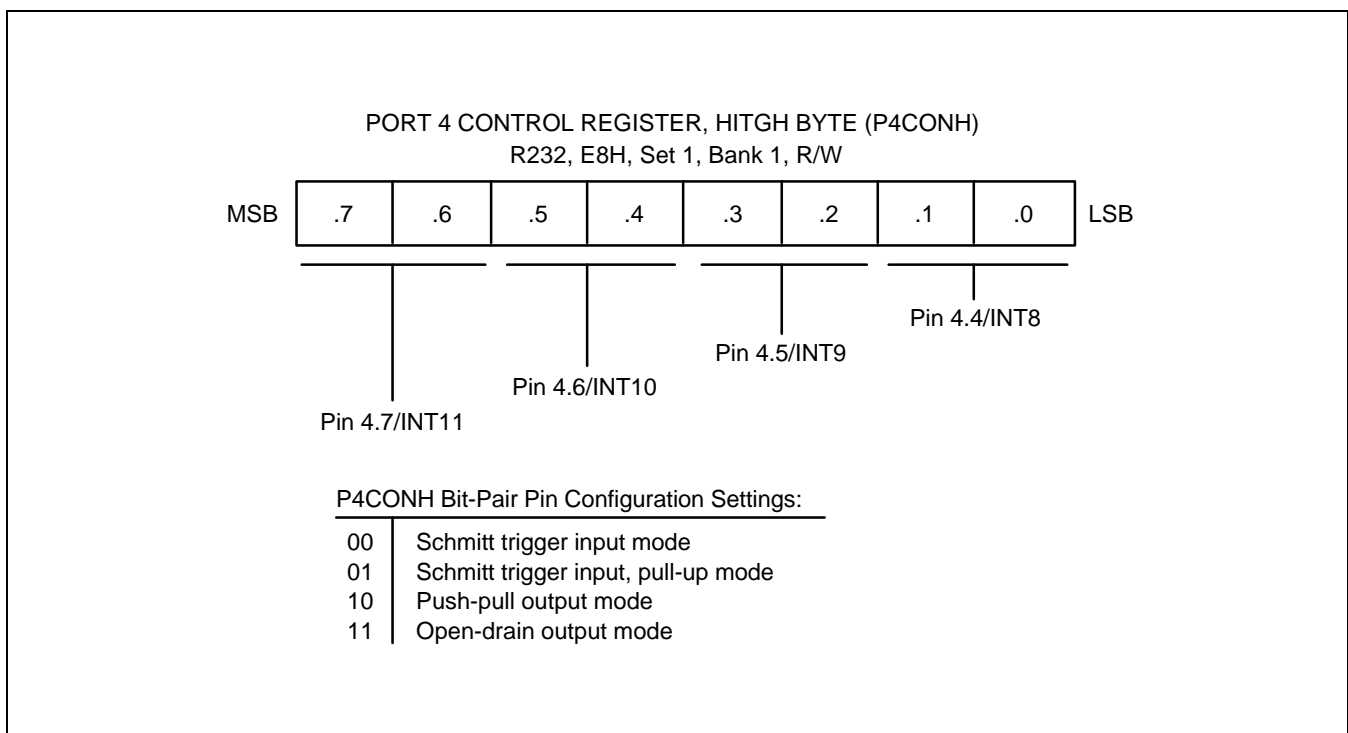


Figure 9-9. Port 4 High-Byte Control Register (P4CONH)

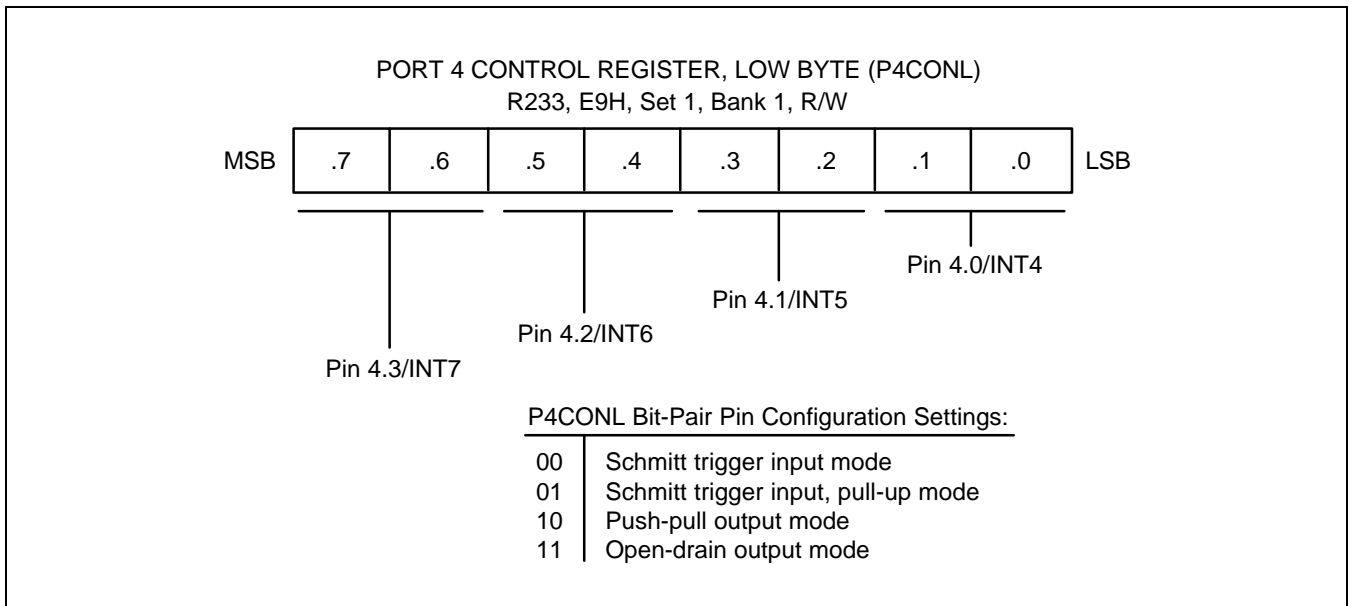


Figure 9-10. Port 4 Low-Byte Control Register (P4CONL)

Port 4 Interrupt Enable and Pending Registers (P4INT, P4PND)

To process external interrupts, two additional control registers are provided: the Port 4 interrupt enable register, P4INT (R231, E7H, set 1, Bank 0) and the Port 4 interrupt pending register, P4PND (R232, E8H, set 1, Bank 0).

By setting bits in the Port 4 interrupt enable register P4INT to "1", you can use specific Port 4 pins to generate interrupt requests when specific signal edges are detected. The interrupt names INT4–INT11 correspond to the pins, P4.0–P4.7. After a reset, P4INT bits are cleared to "00H", disabling all external interrupts.

The Port 4 interrupt pending register P4PND lets you check for interrupt pending conditions and clear the pending condition when the interrupt request has been serviced. Incoming interrupt requests are detected by polling the P4PND bit values.

When the interrupt enable bit of any Port 4 pin is set to "1", a rising or falling signal edge at that pin generates an interrupt request. (Remember that the Port 4 interrupt pins must first be configured by setting them to input mode in the corresponding P4CONH or P4CONL register.)

The corresponding P4PND bit is then set to "1" and the IRQ pulse goes high to signal the CPU that an interrupt request is waiting.

When a Port 4 interrupt request has been serviced, the application program must clear the related interrupt pending register bit by writing a "0" to the pending bit in the P4PND register. Please note that writing a "1" value has no effect.

Port 4 Interrupt Edge Selection Register (P4EDGE)

P4 interrupt can be generated in falling edge or rising edge, depending on the value of the P4 interrupt edge selection register (R233, E9H, set 1, Bank 0), P4EDGE. If the value is set to "1", P4 interrupt is generated in rising edge. If the value is set to "0", P4 interrupt is generated in falling edge.

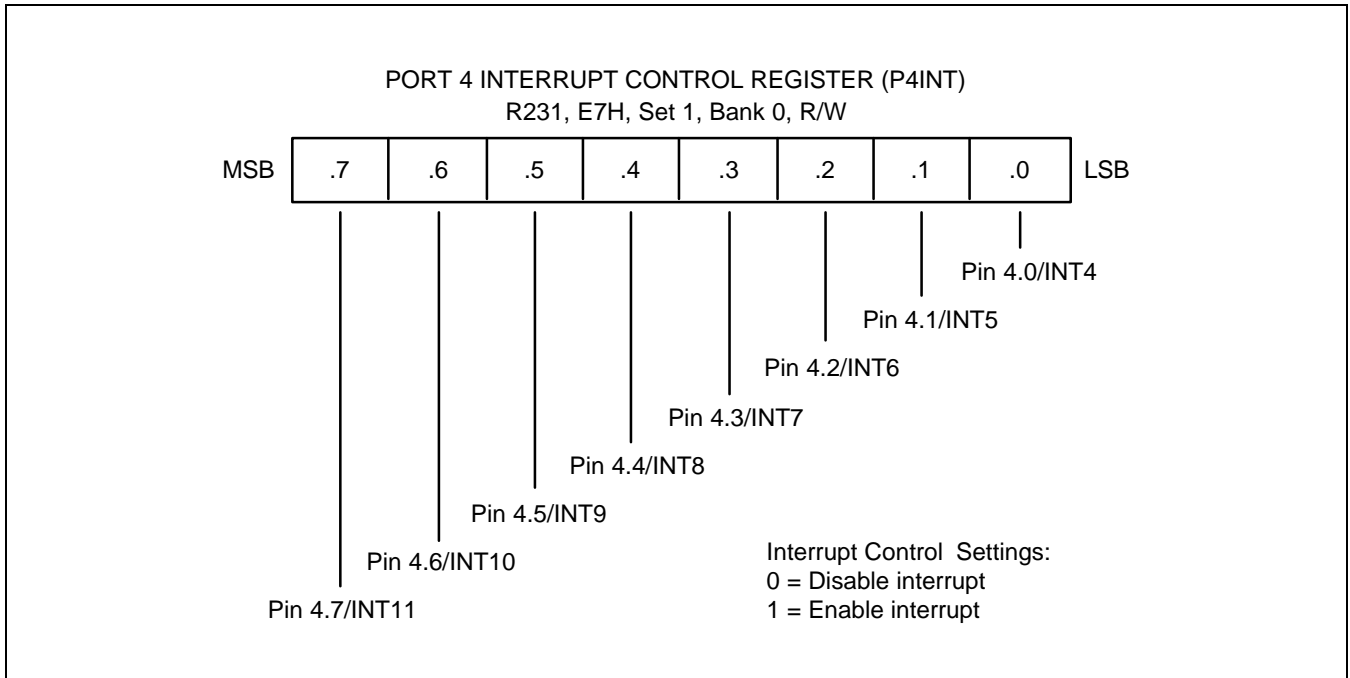


Figure 9-11. Port 4 Interrupt Control Register (P4INT)

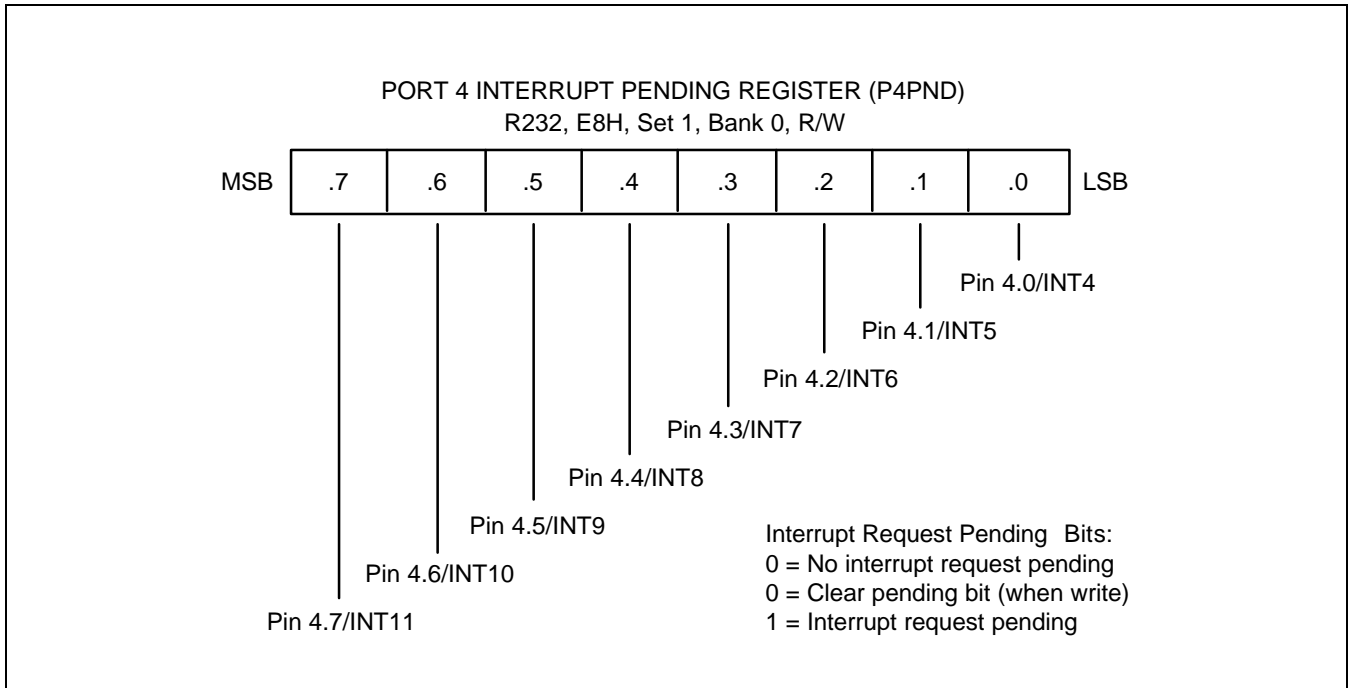


Figure 9-12. Port 4 Interrupt Pending Register (P4PND)

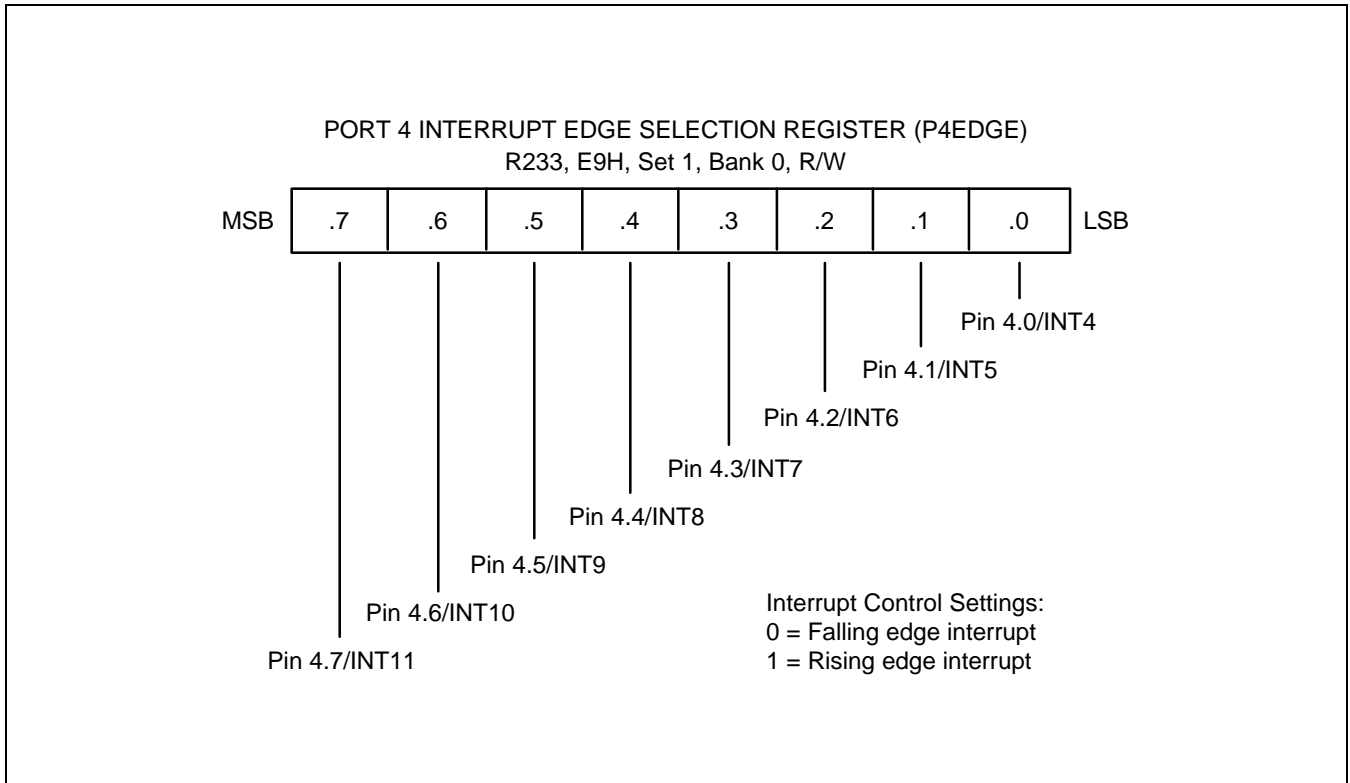


Figure 9-13. Port 4 Edge Selection Register (P4EDGE)

PORT 5

Port 5 is a 7-bit I/O port with individually configurable pins. It is accessed directly by writing or reading the port 5 data register, P5 (R229, E5H, Bank 0) in set 1. You can use port 5 for general I/O, or for the following alternative functions:

- P5.0–P5.3 can be configured, respectively, as SCK, output high impedance, SO, and BUZ output.

The special functions that you can program using the port 5 control register must also be enabled in the associated peripheral. Also, when using port 5 pins for functions other than general I/O, you must still set the corresponding port 5 control register value to configure each bit to input or output mode.

PORT 5 CONTROL REGISTERS

Two 8-bit control registers are used to configure port 5 pins: P5CONH (high byte, EAH, Set 1, Bank 1) for the pins P5.4–P5.6 and P5CONL (EBH, set 1, Bank 1) for the pins P5.0–P5.3. Each byte contains four or three bit-pairs and each bit-pair configures one of port 5 pins. The P5CONL register also controls the alternative functions described below.

Port 5 High-Byte Control Register (P5CONH)

Three bit-pairs in the port 5 control register (P5CONH) configure the port 5 pins, P5.4–P5.6 to schmitt trigger input, schmitt trigger input with pull-up resistor, or push-pull output mode.

Port 5 Low-Byte Control Register (P5CONL)

The low-byte port 5 pins, P5.0–P5.3, can be configured individually as schmitt trigger inputs, schmitt trigger input with pull-up resistor, or as push-pull outputs. You can alternately configure these pins. To select the alternative functions, you must set the related bit-pairs — SCK output, output high impedance, SO, and BUZ output — to "11B".

Table 9-5. Port 5 Data Register Summary (Low Nibbles)

P5CONL Bit-Pair	Corresponding Port 5 Pin	Alternate Pin Function
Bits 0 and 1	P5.0	SCK output
Bits 2 and 3	P5.1	High impedance
Bits 4 and 5	P5.2	SO
Bits 6 and 7	P5.3	BUZ

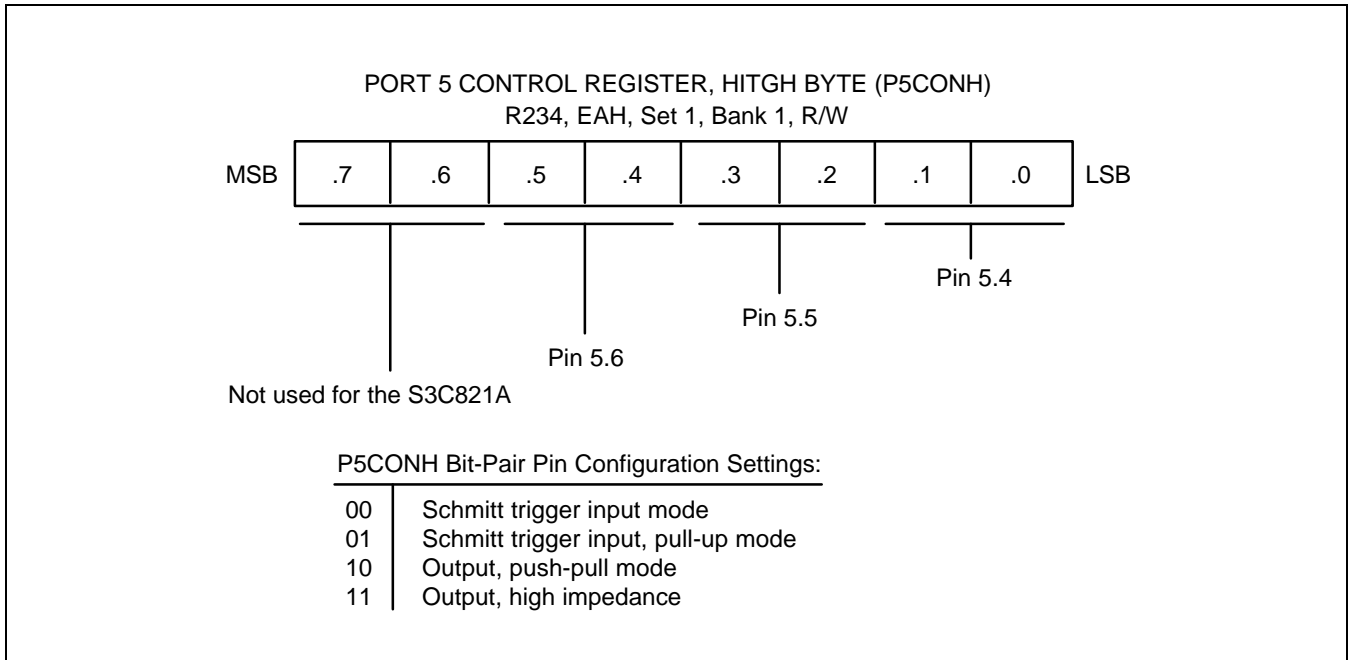


Figure 9-14. Port 5 High-Byte Control Register (P5CONH)

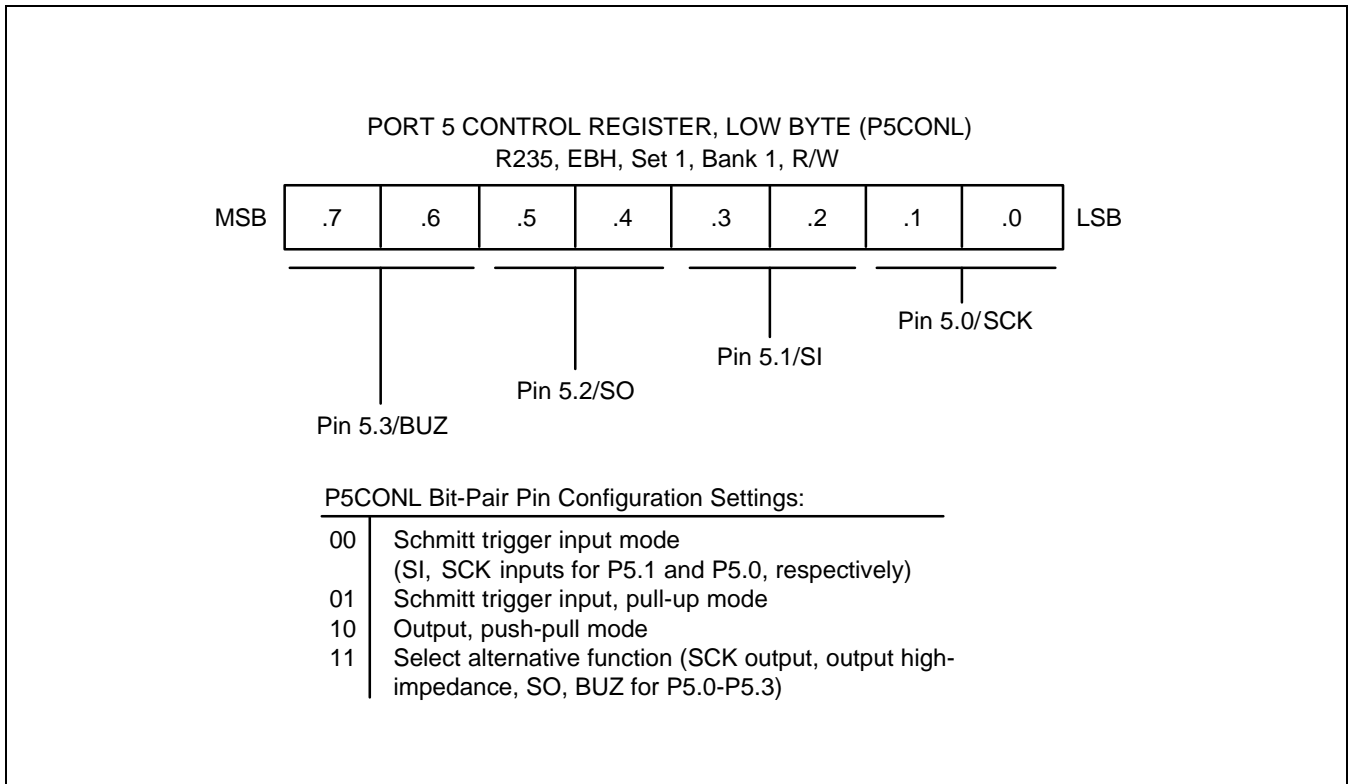


Figure 9-15. Port 5 Low-Byte Control Register (P5CONL)

10

BASIC TIMER and TIMER 0

OVERVIEW

S3C821A has two default timers: an 8-bit *basic timer* and an 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, or
- 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_{xx} divided by 4096, 1024, 128, or 16) with multiplexer
- 8-bit basic timer counter, BTCNT (set 1, Bank 0, FDH, read-only)
- Basic timer control register, BTCON (set 1, D3H, read/write)

Timer 0

Timer 0 offers three operating modes, which you can select by setting T0CON appropriately:

- Interval timer mode
- Capture input mode with a rising or falling edge trigger at the P3.7 pin
- PWM mode

Timer 0 has the following functional components:

- Clock frequency divider (f_{xx} divided by 4096, 256, or 8) with multiplexer
- External clock input pin (P2.4, T0CK)
- 8-bit counter (T0CNT), 8-bit comparator, and 8-bit reference data register (T0DATA)
- I/O pins for capture input (P3.7) or match output
- Timer 0 overflow interrupt (IRQ0, vector FAH) and match/capture interrupt (IRQ0, vector FCH) generation
- Timer 0 control register, T0CON (set 1, D2H, read/write)

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. It is located in set 1, address D3H, and is read/write addressable using Register addressing mode.

A reset clears BTCON to "00H". This enables the watchdog function and selects a basic timer clock frequency of $f_x/4096$. 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 (set 1, bank 0, FDH), can be cleared at any time during the 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 (unless timer 0 uses an external clock source), you should write a "1" to BTCON.0.

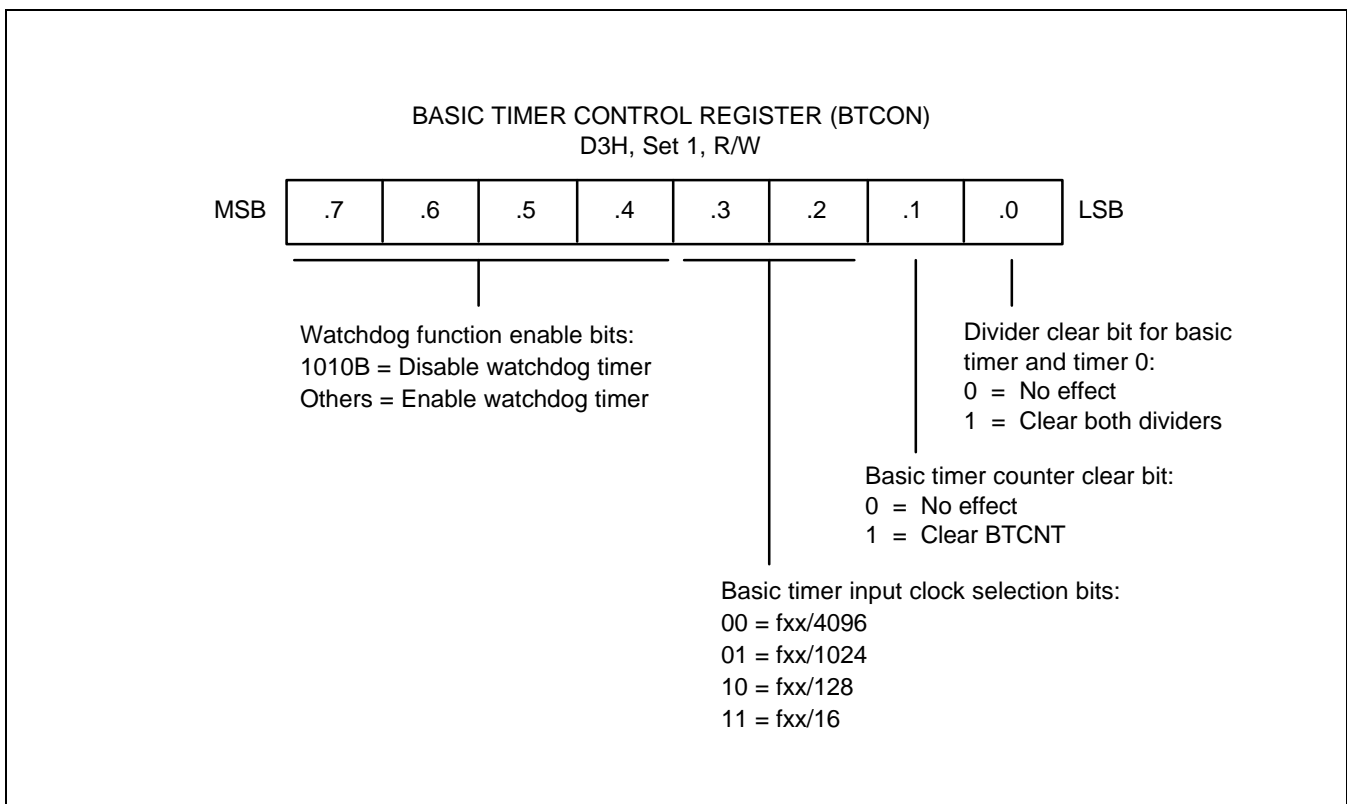


Figure 10-1. Basic Timer Control Register (BTCON)

BASIC TIMER FUNCTION DESCRIPTION

Watchdog Timer Function

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

Reset occurs whenever a basic timer counter overflows. During the normal operation, the application program must prevent the overflow, which causes a 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 the 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 after a reset or when stop mode has been released by an external interrupt.

In stop mode, whenever a reset or an internal and an external interrupt occurs, the oscillator starts. The BTCNT value then starts increasing at the rate of $f_x/4096$ (for reset), or at the rate of the preset clock source (for an internal and an external interrupt). When BTCNT.3 overflows, 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 the normal operation.

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

1. During the 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 would increase at the rate of $f_x/4096$. If an internal or 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 3 of the basic timer counter overflows.
4. When a BTCNT.3 overflow occurs, the normal CPU operation resumes.

TIMER 0 CONTROL REGISTER (T0CON)

You use the timer 0 control register, T0CON, to

- Select the timer 0 operating mode (interval timer, capture mode, or PWM mode)
- Select the timer 0 input clock frequency
- Clear the timer 0 counter, T0CNT
- Enable the timer 0 overflow interrupt or timer 0 match/capture interrupt
- Clear timer 0 match/capture interrupt pending conditions

T0CON is located in set 1, at address D2H, and is read/write addressable using Register addressing mode.

A reset clears T0CON to "00H". This sets timer 0 to normal interval timer mode, selects an input clock frequency of $f_x/4096$, and disables all timer 0 interrupts. You can clear the timer 0 counter at any time during the normal operation by writing a "1" to T0CON.3.

The timer 0 overflow interrupt (T0OVF) has interrupt level IRQ0 and the vector address FAH. When a timer 0 overflow interrupt occurs and is serviced by the CPU, the pending condition is cleared automatically by hardware.

To enable the timer 0 match/capture interrupt (IRQ0, vector FCH), you must write T0CON.1 to "1". To detect a match/capture interrupt pending condition, the application program polls T0CON.0. When a "1" is detected, a timer 0 match or capture interrupt is pending. After the interrupt request is serviced, the pending condition must be cleared by software by writing a "0" to the timer 0 interrupt pending bit, T0CON.0.

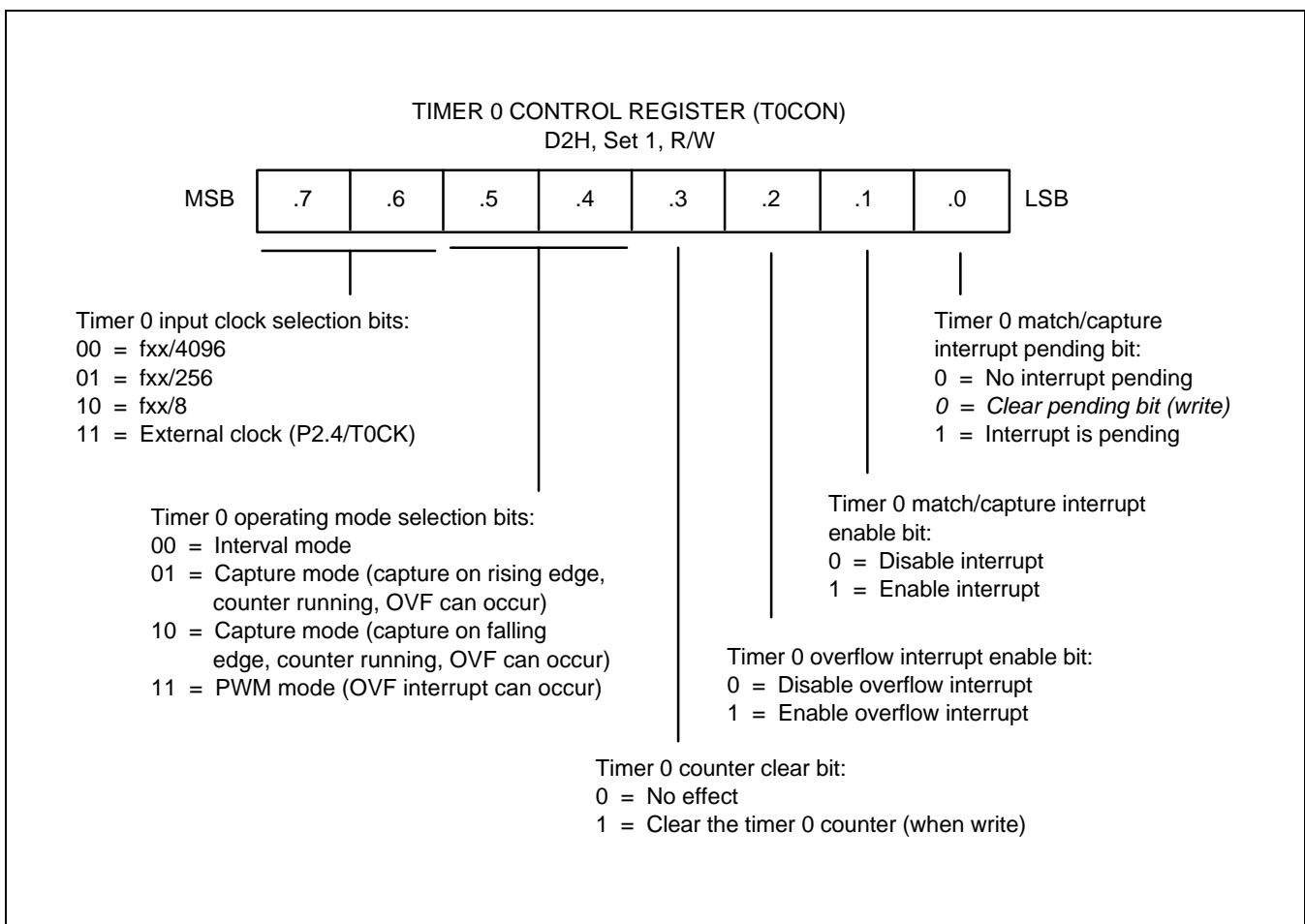


Figure 10-2. Timer 0 Control Register (T0CON)

TIMER 0 FUNCTION DESCRIPTION

Timer 0 Interrupts (IRQ0, Vectors FAH and FCH)

The timer 0 module can generate two interrupts: the timer 0 overflow interrupt (T0OVF), and the timer 0 match/capture interrupt (T0INT). T0OVF is assigned the interrupt level IRQ0, vector FAH. T0INT also belongs to the interrupt level IRQ0, but is assigned a separate vector address, FCH.

A timer 0 overflow interrupt pending condition is automatically cleared by hardware after it is serviced. The T0INT pending condition must, however, be cleared by the application's interrupt service routine by writing a "0" to the T0CON.0 interrupt pending bit.

Interval Timer Mode

In interval timer mode, a match signal is generated when the counter value is identical to the value written to the T0 reference data register, T0DATA. The match signal generates a timer 0 match interrupt (T0INT, vector FCH) and clears the counter.

If, for example, you write the value "10H" to T0DATA and "0AH" to T0CON, the counter will increment until it reaches "10H". At this point, the T0 interrupt request is generated, the counter value is reset, and counting resumes. With each match, the level of the signal at the timer 0 output pin is inverted (see Figure 10-3).

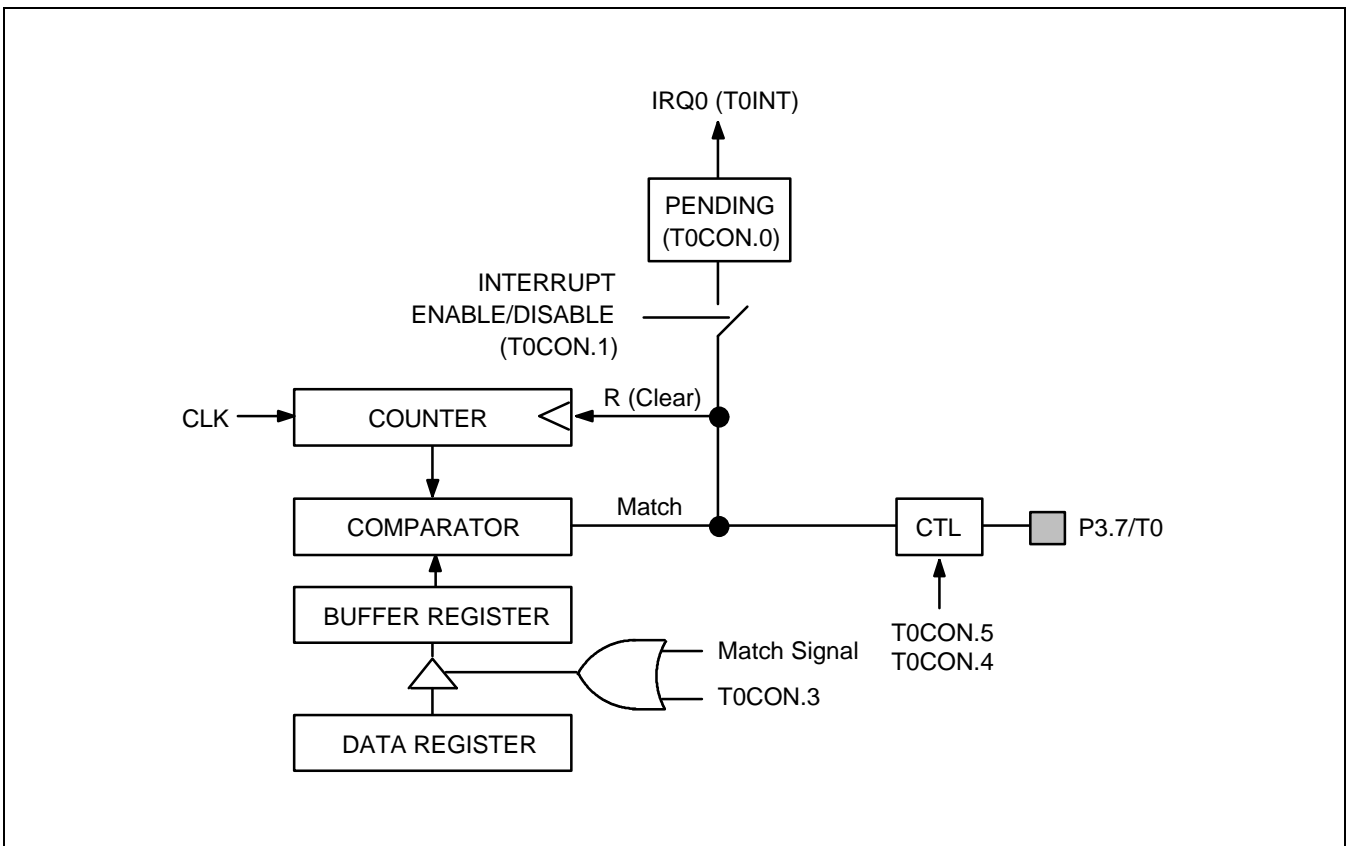


Figure 10-3. Simplified Timer 0 Function Diagram: Interval Timer Mode

Pulse Width Modulation Mode

Pulse width modulation (PWM) mode lets you program the width (duration) of the pulse that is output at the TOPWM pin. As in interval timer mode, a match signal is generated when the counter value is identical to the value written to the timer 0 data register. In PWM mode, however, the match signal does not clear the counter. Instead, it runs continuously, overflowing at "FFH", and then continues incrementing from "00H".

Although you can use the match signal to generate a timer 0 overflow interrupt, interrupts are not typically used in PWM-type applications. Instead, the pulse at the TOPWM pin is held to Low level as long as the reference data value is *less than or equal to* (\leq) the counter value and then the pulse is held to High level for as long as the data value is *greater than* ($>$) the counter value. The time period is equal to $t_{CLK} \times 256$ (see Figure 10-4).

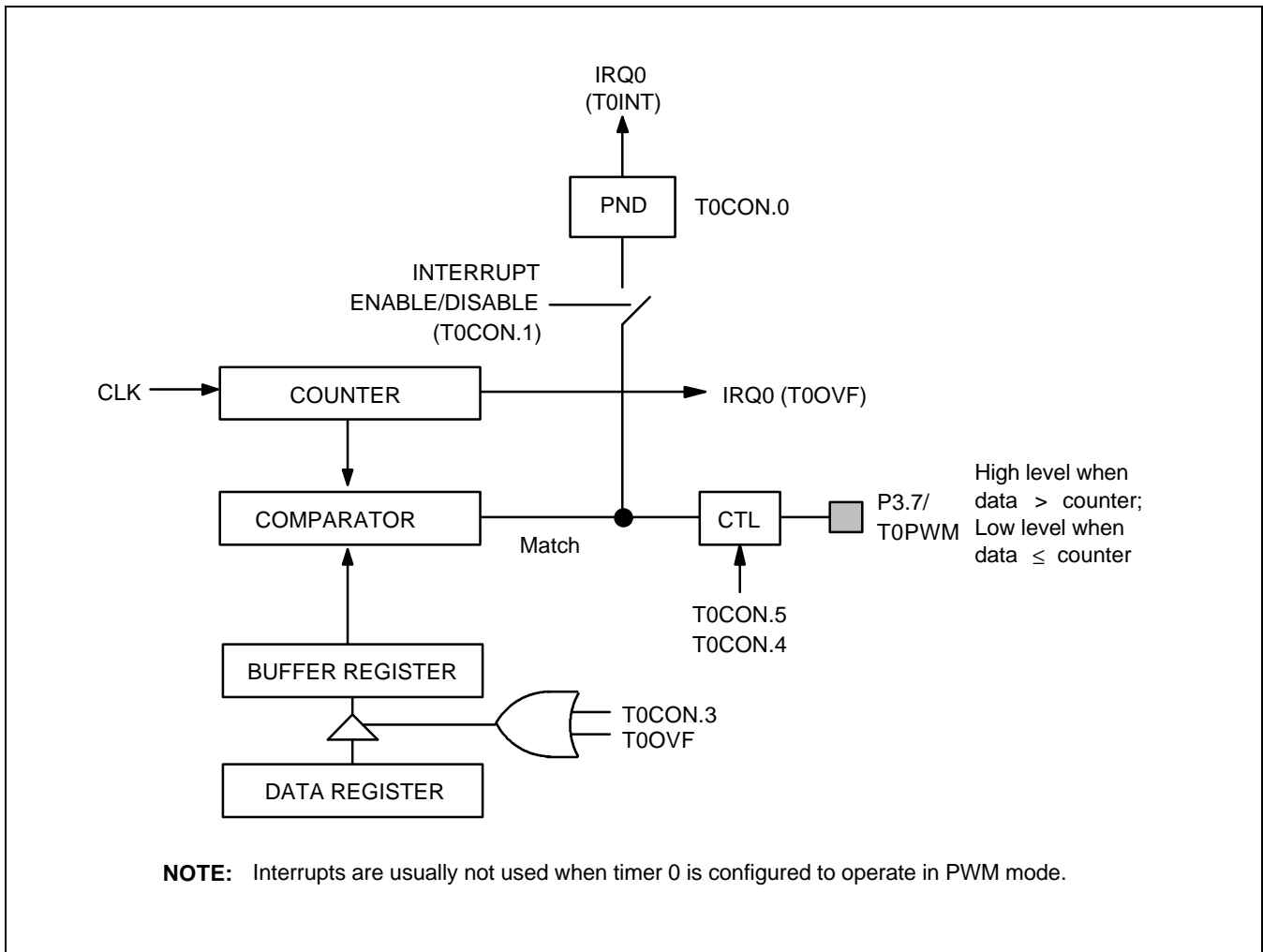


Figure 10-4. Simplified Timer 0 Function Diagram: PWM Mode

Capture Mode

In capture mode, a signal edge that is detected at the T0CAP pin opens a gate and loads the current counter value into the T0 data register. You can select rising or falling edge to trigger this operation.

Timer 0 also gives you capture input source: the signal edge at the T0CAP pin. You select the capture input by setting the values of the timer 0 capture input selection bits in the port 3 control register, P3CONH.7–6, (set 1, bank 1, E6H). When P3CONH.7–6 is "00" or "01", the T0CAP input is selected.

Both kinds of timer 0 interrupts can be used in capture mode. The timer 0 overflow interrupt is generated whenever a counter overflow occurs. The timer 0 match/capture interrupt is generated whenever the counter value is loaded into the T0 data register.

By reading the captured data value in T0DATA, and assuming a specific value for the timer 0 clock frequency, you can calculate the pulse width (duration) of the signal being input from the T0CAP pin (see Figure 10-5).

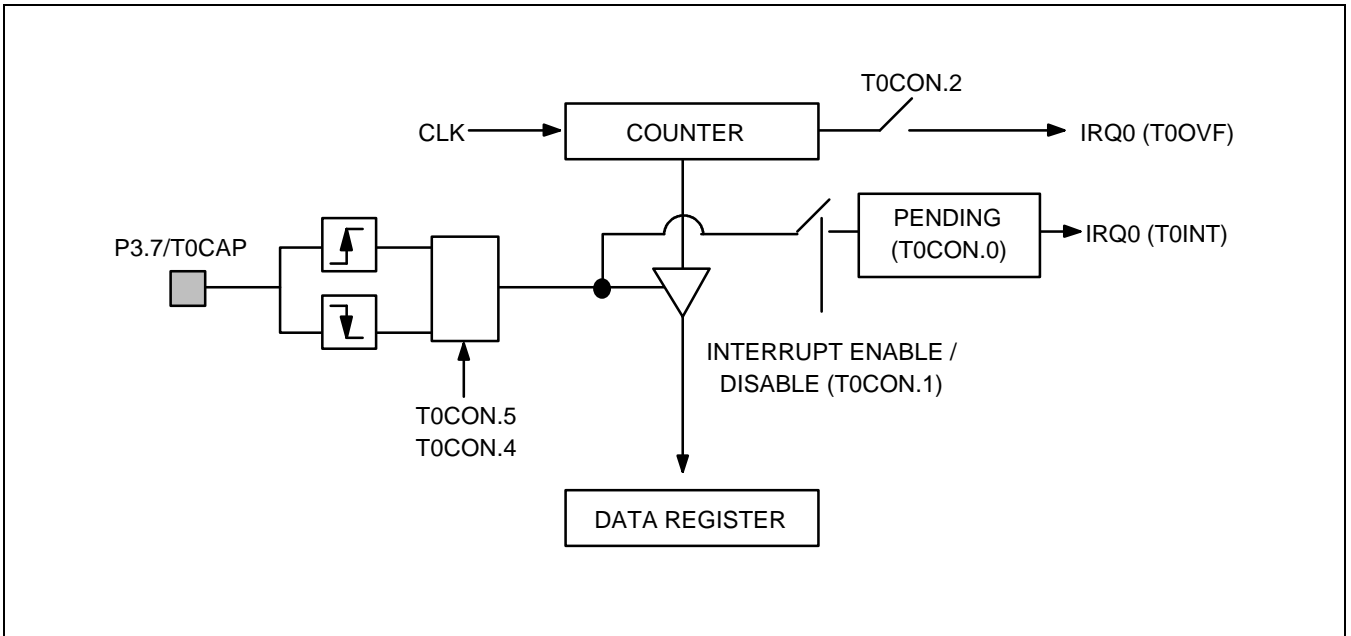


Figure 10-5. Simplified Timer 0 Function Diagram: Capture Mode

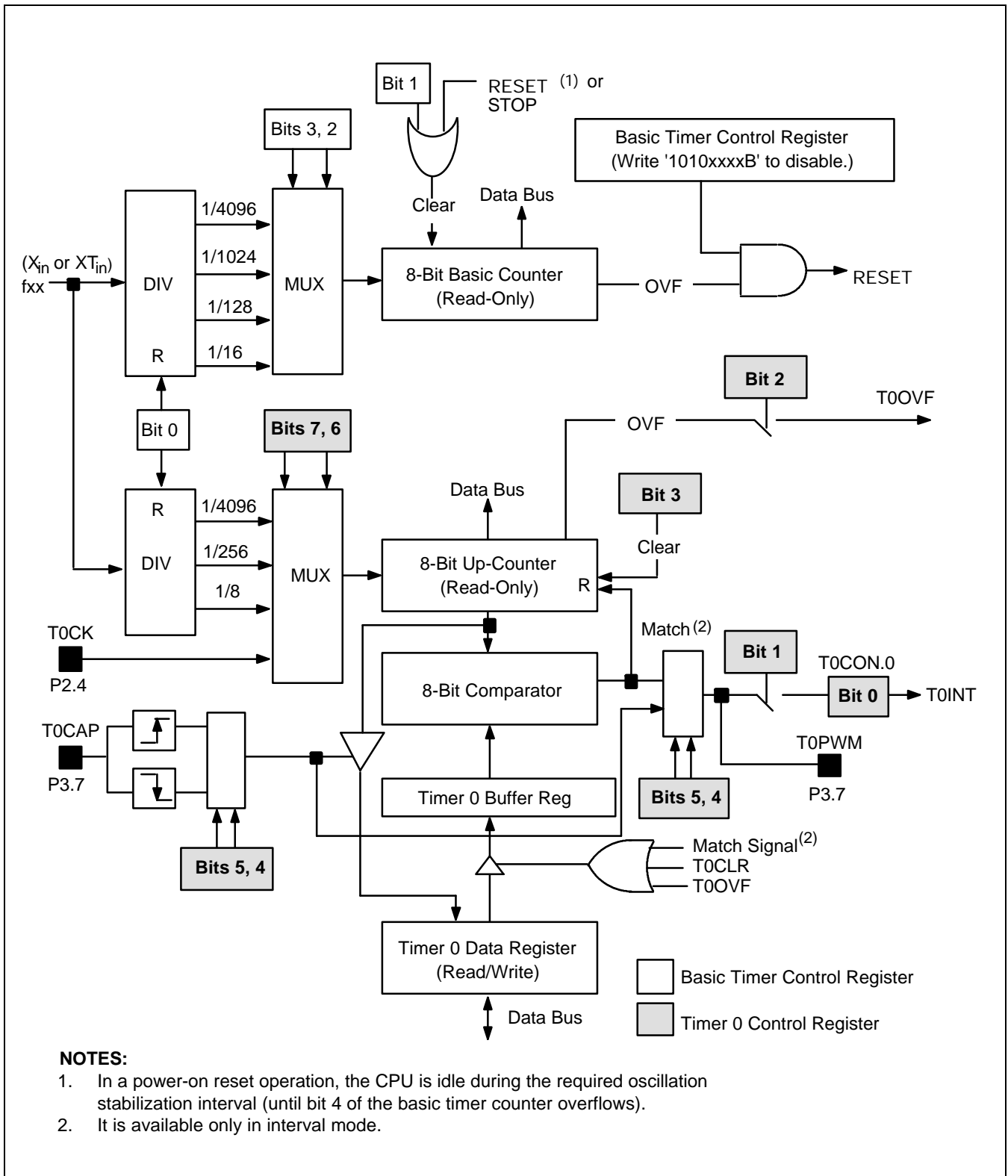


Figure 10-6. Basic Timer and Timer 0 Block Diagram

PROGRAMMING TIP — Configuring the Basic Timer

This example shows how to configure the basic timer to sample specifications:

```

                ORG      0100H

RESET          DI                ; Disable all interrupts
               LD        BTCON,#0ABH ; Disable the watchdog timer
               LD        CLKCON,#18H ; Non-divided clock
               CLR       SYM       ; Disable global and fast interrupts
               CLR       SPL       ; Stack pointer low byte ← "0"
                                   ; Stack area starts at 0FFH
               .
               .
               .
               SRP        #0C0H    ; Set register pointer ← 0C0H
               EI                ; Enable interrupts
               .
               .
               .
MAIN           LD        BTCON,#53H ; Enable the watchdog timer
                                   ; Basic timer clock: fxx/4096
                                   ; Clear basic timer counter
               NOP
               NOP
               .
               .
               .
               JP        T,MAIN
               .
               .
               .

```

PROGRAMMING TIP — Programming Timer 0

This sample program sets timer 0 to interval timer mode and the frequency of the oscillator clock, determining the execution sequence which follows a timer 0 interrupt. The program parameters are as follows:

- Timer 0 is used in interval mode. The timer interval is set to 4 milliseconds
- Oscillation frequency is 6 MHz
- General register 64H (page 0) ← 60H (page 0) + 62H (page 0) + 63H (page 0) + 64H (page 0) is executed after a timer 0 interrupt

```

                ORG      0FAH                ; Timer 0 overflow interrupt
                VECTOR   T0OVER
                ORG      0FCH                ; Timer 0 match/capture interrupt
                VECTOR   T0INT
                ORG      0100H

RESET          DI                ; Disable all interrupts
               LD         BTCON,#0ABH      ; Disable the watchdog timer
               LD         CLKCON,#18H      ; Select non-divided clock
               CLR        SYM              ; Disable global and fast interrupts
               CLR        SPL              ; Stack pointer low byte ← "0"
               ; Stack area starts at 0FFH
               .
               .
               .
               LD         T0CON,#4AH       ; Write "01001010B"
               ; Input clock is fxx/256
               ; Interval timer mode
               ; Enable the timer 0 interrupt
               ; Disable the timer 0 overflow interrupt
               LD         T0DATA,#5DH      ; Set timer interval to 4 milliseconds
               ; Clear pending bit
               ; (6 MHz/256) + (93 + 1) = 0.25 kHz (4 ms)
               SRP        #60H             ; RP ← 60H
               CLR        PP               ; Destination ← 0, Source ← 0
               CLR        R0               ; Register 60H (page 0) = 00H
               BITR       R1.2             ; Bit reset (61.2H)

               SRP        #0C0H            ; Set register pointer ← 0C0H
               EI                ; Enable interrupts
               .
               .
               .
(Continued on next page)

```

 **PROGRAMMING TIP — Programming Timer 0 (Continued)**

```

TOINT    PUSH    RP0           ; Save RP0 to stack
         PUSH    PP           ; Save PP to stack
         SRP0    #60H         ; RP0 ← 60H
         LD     PP,#00H       ; Destination ← 0, Source ← 0
         INC    R0           ; R0 ← R0 + 1
         ADD    R2,R0         ; R2 ← R2 + R0
         ADC    R3,R2         ; R3 ← R3 + R2 + Carry
         ADC    R4,R3         ; R4 ← R4 + R3 + Carry

CP       R0,#32H             ; 50 × 4 = 200 ms
         JR     ULT,NO_200MS_SET
         BITS   R1.2         ; Bit setting (61.2H)

NO_200MS_SET:
         LD     T0CON,#4AH   ; Clear pending bit
         POP    PP           ; Restore page pointer value
         POP    RP0         ; Restore register pointer 0 value

TOOVER   IRET               ; Return from interrupt service routine

```

11

TIMER 1

ONE 16-BIT TIMER MODE (TIMER 1)

The 16-bit timer 1 is used in one 16-bit timer or two 8-bit timers mode. When TACON.7 is set to "1", it is in one 16-bit timer mode. When TACON.7 is set to "0", the timer 1 is used as two 8-bit timers.

- One 16-bit timer mode (Timer 1)
- Two 8-bit timers mode (Timer A and B)

OVERVIEW

The 16-bit timer 1 is an 16-bit general-purpose timer. Timer 1 includes interval timer mode using appropriate TACON setting.

Timer 1 has the following functional components:

- Clock frequency divider (f_{clk} divided by 1024, 512, 8, or 1 and T1CK: External clock) with multiplexer
- 16-bit counter (TACNT, TBCNT), 16-bit comparator, and 16-bit reference data register (TADATA, TBADATA)
- Timer 1 match interrupt (IRQ1, vector F6H) generation
- Timer 1 control register, TACON (set 1, bank 0, F3H, read/write)

FUNCTION DESCRIPTION

Interval Timer Function

The timer 1 module can generate an interrupt, the timer 1 match interrupt (T1INT). T1INT belongs to the interrupt level IRQ1, and is assigned a separate vector address, F6H.

The T1INT pending condition should be cleared by software after IRQ1 is serviced. The T1INT pending bit must be cleared by the application sub-routine by writing a "0" to the TACON.0 pending bit.

In interval timer mode, a match signal is generated when the counter value is identical to the values written to the T1 reference data registers, TADATA and TBADATA. The match signal generates a timer 1 match interrupt (T1INT, vector F6H) and clears the counter.

If, for example, you write the value 10H and 32H to TADATA and TBADATA, respectively, and 8EH to TACON, the counter will increment until it reaches 3210H. At this point, the T1 interrupt request is generated, the counter value is reset, and counting resumes.

Timer 1 Control Register (TACON)

You use the timer 1 control register, TACON, to

- Enable the timer 1 operating (interval timer)
- Select the timer 1 input clock frequency
- Clear the timer 1 counter, TACNT and TBCNT
- Enable the timer 1 interrupt
- Clear timer 1 interrupt pending conditions

TACON is located in set 1, bank 0, at address F3H, and is read/write addressable using register addressing mode.

A reset clears TACON to "00H". This sets timer 1 to disable interval timer mode, selects an input clock frequency of fxx/1024, and disables timer 1 interrupt. You can clear the timer 1 counter at any time during the normal operation by writing a "1" to TACON.3.

To enable the timer 1 interrupt (IRQ1, vector F6H), you must write TACON.7, TACON.2, and TACON.1 to "1". To generate the exact time interval, you should set TACON.3 and TACON.0 to "10B", which clear counter and interrupt pending bit. When the T1INT sub-routine is serviced, the pending condition must be cleared by software by writing a "0" to the timer 1 interrupt pending bit, TACON.0.

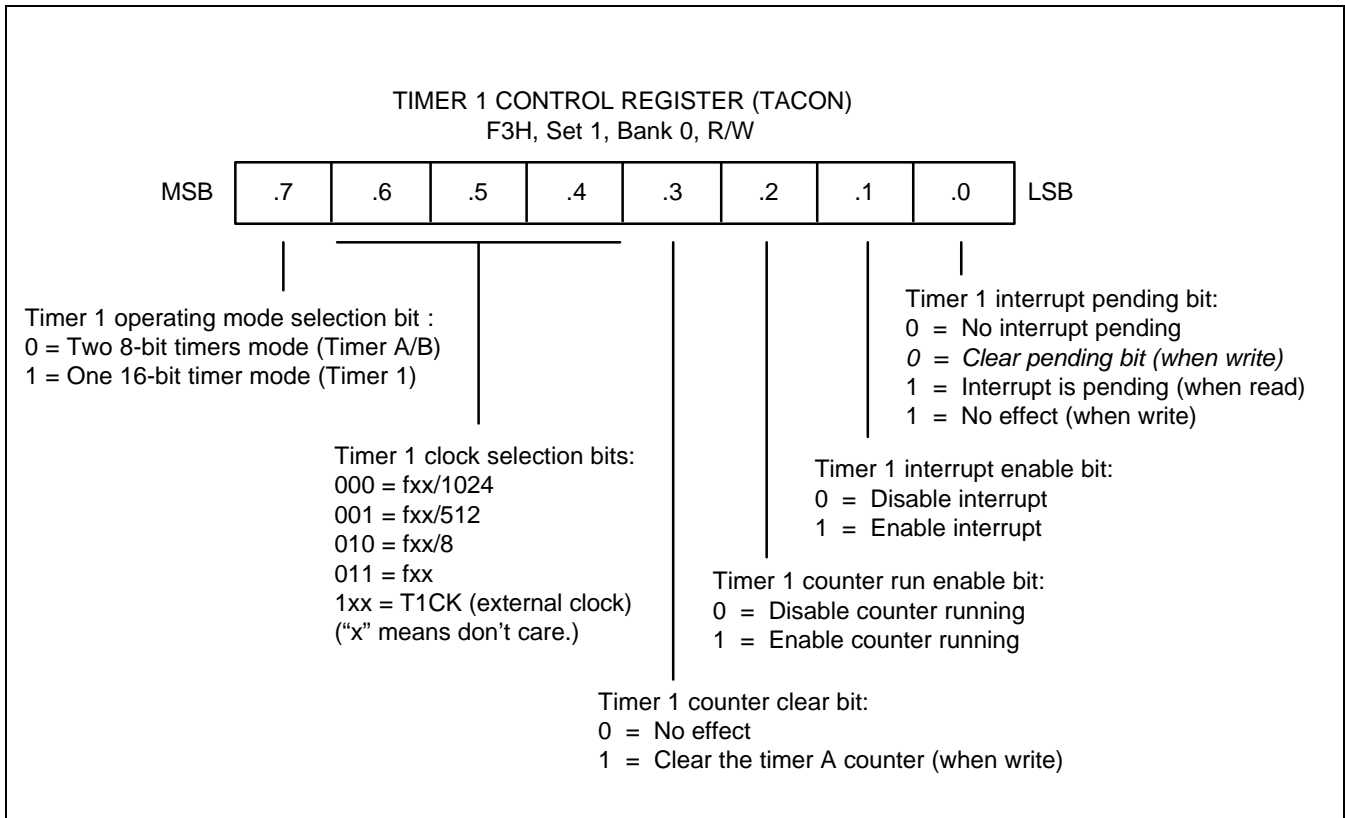


Figure 11-1. Timer 1 Control Register (TACON)

BLOCK DIAGRAM

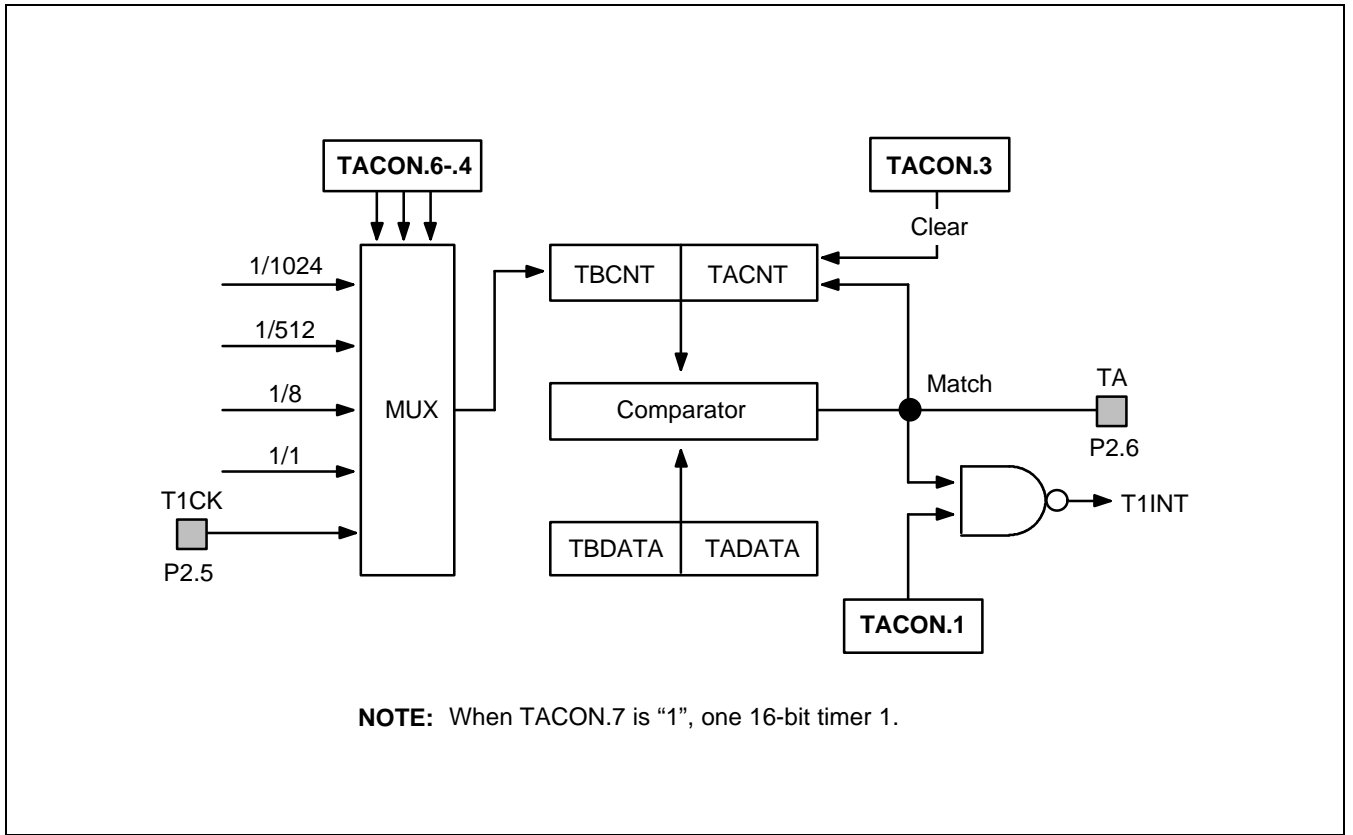


Figure 11-2. Timer 1 Functional Block Diagram

TWO 8-BIT TIMERS MODE (TIMER A and B)

OVERVIEW

The 8-bit timer A and B are the 8-bit general-purpose timers. Timer A and B support interval timer mode using appropriate TACON and TBCON setting, respectively.

Timer A and B have the following functional components:

- Clock frequency divider with multiplexer
 - fxx divided by 1024, 512, 8, or 1 and T1CK (External clock) for timer A
 - fxx divided by 1024, 512, 8, or 1 for timer B
- 8-bit counter (TACNT, TBCNT), 8-bit comparator, and 8-bit reference data register (TADATA, TBDATA)
- Timer A match interrupt (IRQ1, vector F6H) generation
- Timer A control register, TACON (set 1, bank 0, F3H, read/write)
- Timer B match interrupt (IRQ1, vector F4H) generation
- Timer B control register, TBCON (set 1, bank 0, F2H, read/write)

FUNCTION DESCRIPTION

Interval Timer Function

The timer A and B module can generate an interrupt: the timer A match interrupt (TAINT) and the timer B match interrupt (TBINT). TAINT belongs to the interrupt level IRQ1, and is assigned a separate vector address, F6H. TBINT belongs to the interrupt level IRQ1 and is assigned a separate vector address, F4H.

The TAINT and TBINT pending condition should be cleared by software after they are serviced.

In interval timer mode, a match signal is generated when the counter value is identical to the values written to the TA or TB reference data registers, TADATA or TBDATA. The match signal generates corresponding match interrupt (TAINT, vector F6H; TBINT, vector F4H) and clears the counter.

If, for example, you write the value 10H to TBDATA, "0" to TACON.7, and 0EH to TBCON, the counter will increment until it reaches 10H. At this point, the TB interrupt request is generated, the counter value is reset, and counting resumes.

Timer A and B Control Register (TACON, TBCON)

You use the timer A and B control register, TACON and TBCON, to

- Enable the timer A and B operating (interval timer)
- Select the timer A and B input clock frequency
- Clear the timer A and B counter, TACNT and TBCNT
- Enable the timer A and B interrupt
- Clear timer A and B interrupt pending conditions

TACON and TBCON are located in set 1, bank 0, at address F3H and F2H, and is read/write addressable using register addressing mode.

A reset clears TACON and TBCON to "00H". This sets timer A and B to disable interval timer mode, selects an input clock frequency of $f_{xx}/1024$, and disables timer A and B interrupt. You can clear the timer A and B counter at any time during normal operation by writing a "1" to TACON.3 and TBCON.3.

To enable the timer A and B interrupt (IRQ1, vector F6H, F4H), you must write TACON.7 to "0", TACON.2 (TBCON.2) and TACON.1 (TBCON.1) to "1". To generate the exact time interval, you should set TACON.3 (TBCON.3) and TACON.0 (TBCON.0) to "10B", which clear counter and interrupt pending bit, respectively. When the TAIN or TBINT sub-routine is serviced, the pending condition must be cleared by software by writing a "0" to the timer A or B interrupt pending bits, TACON.0 or TBCON.0.

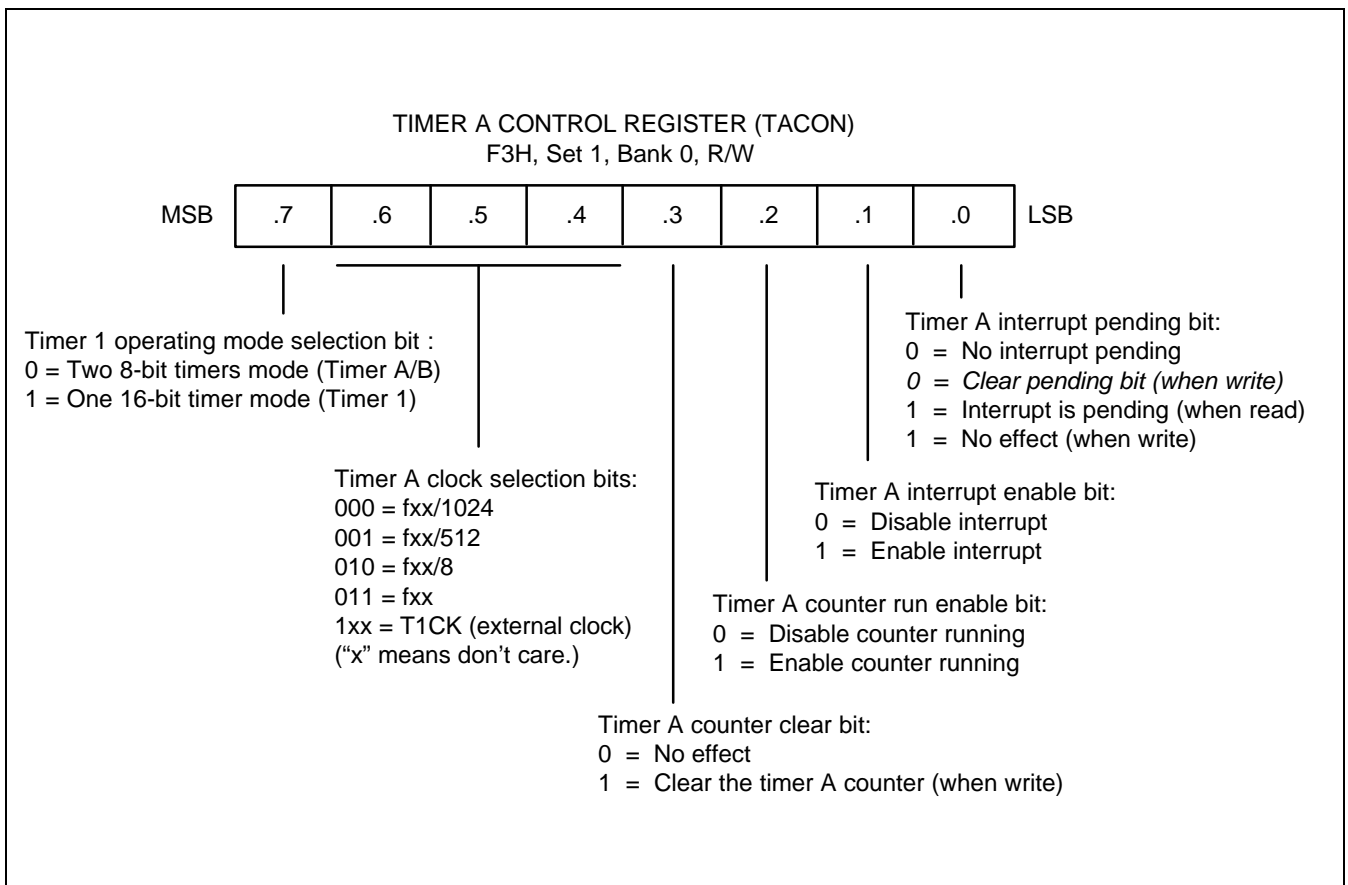


Figure 11-3. Timer A Control Register (TACON)

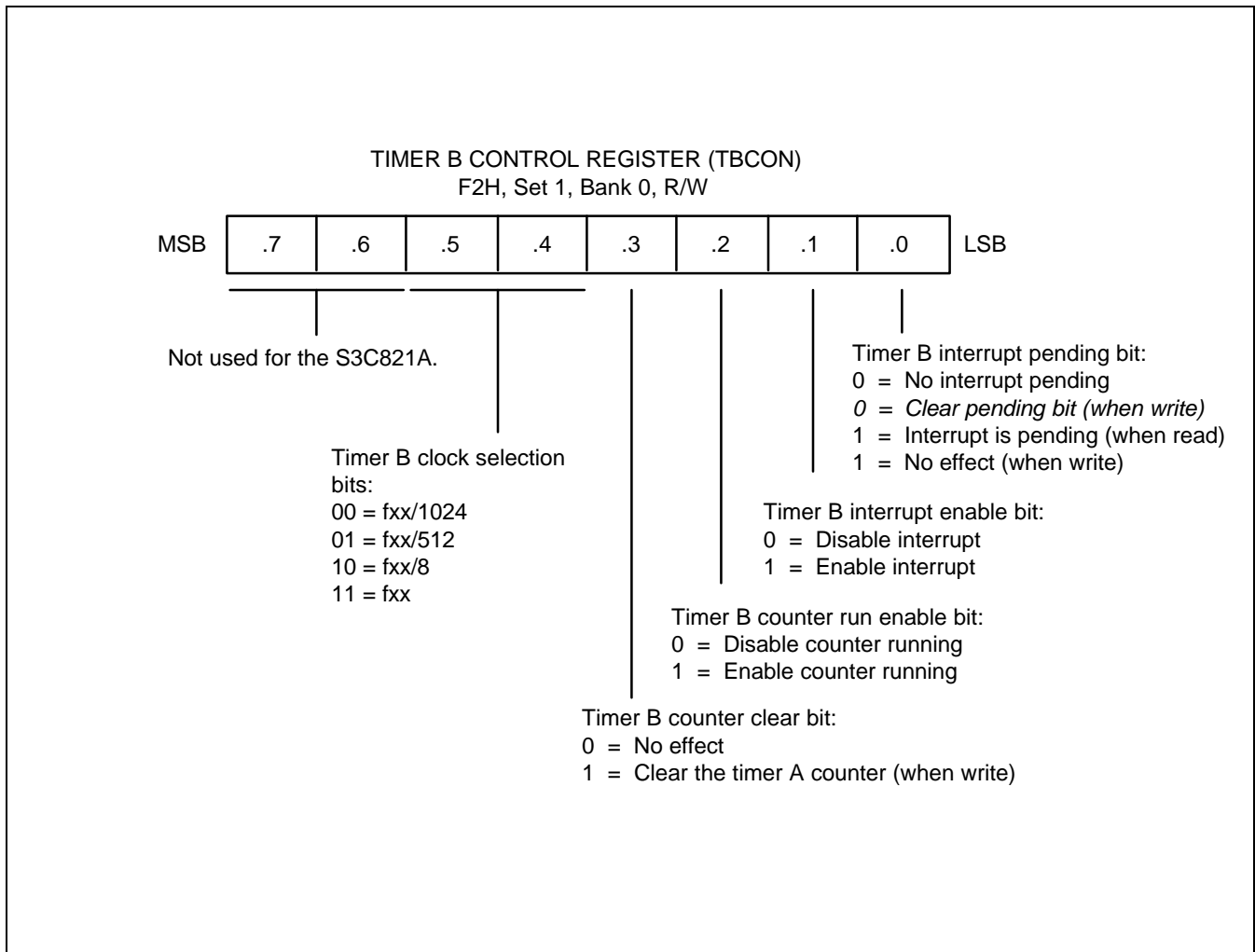


Figure 11-4. Timer B Control Register (TBCON)

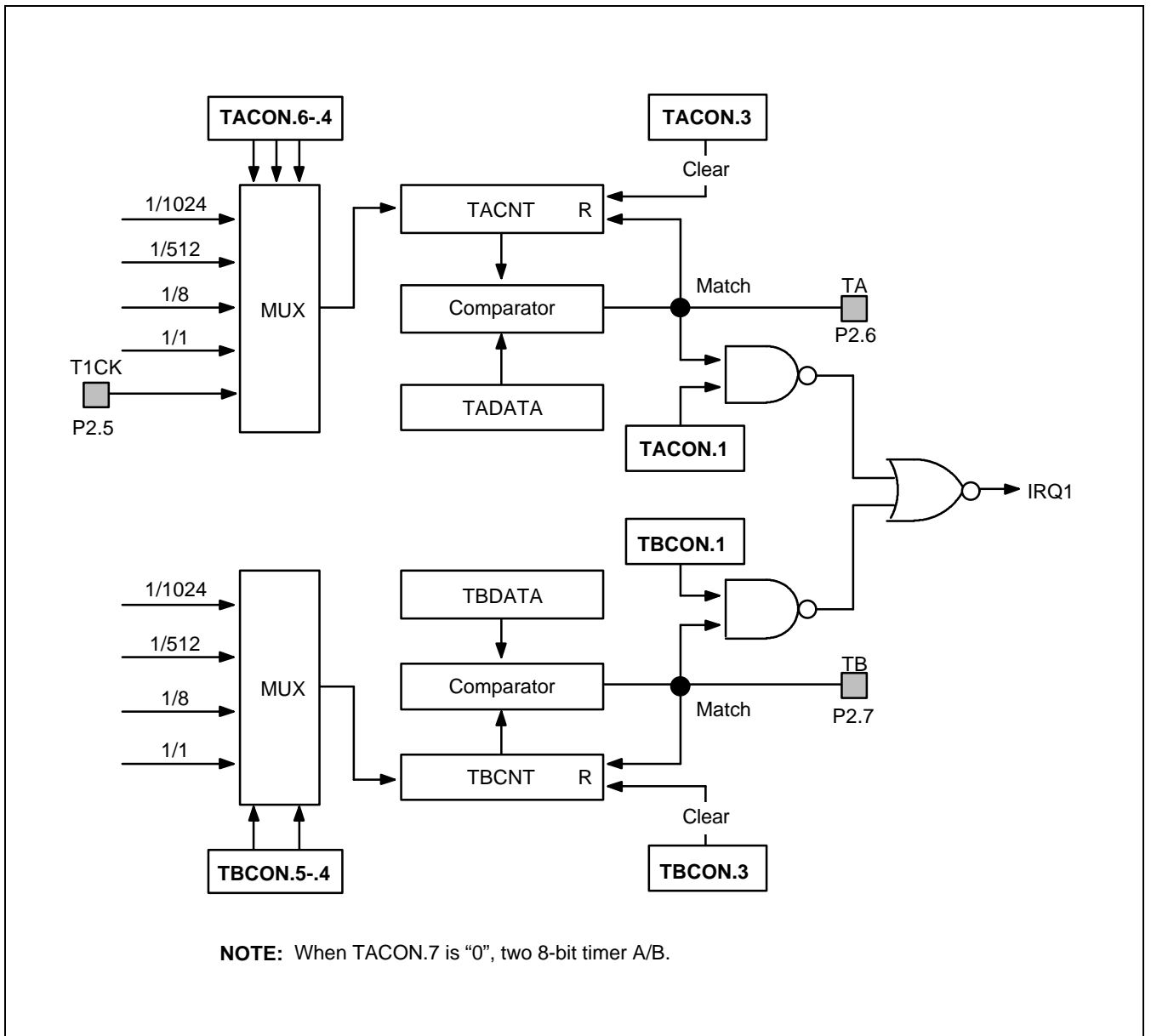


Figure 11-5. Timer A and B Function Block Diagram

12 WATCH TIMER

OVERVIEW

Watch timer functions include real-time and watch-time measurement and interval timing for the system clock. After the watch timer starts and elapses a time, the watch timer interrupt is automatically set to "1", and interrupt requests commence in 3.91 ms or 0.5 second.

The watch timer can generate a steady 0.5 kHz, 1 kHz, 2 kHz, or 4 kHz signal to the BUZ output. By setting WTCON.3 and WTCON.0 to "11b", the watch timer will function in high-speed mode, generating an interrupt every 3.91 ms. High-speed mode is useful for timing events for program debugging sequences. The watch timer supplies the clock frequency for the LCD controller (f_{LCD}). Therefore, if the watch timer is disabled, the LCD controller does not operate.

- Real-time and Watch-time measurement
- Using a main system or subsystem clock source
- Clock source generation for LCD controller
- Buzzer output frequency generator
- Timing tests in high-speed mode

WATCH TIMER CONTROL REGISTER (WTCON: R/W)

FBH	WTCON.7	WTCON.6	WTCON.5	WTCON.4	WTCON.3	WTCON.2	WTCON.1	WTCON.0
RESET	"0"	"0"	"0"	"0"	"0"	"0"	"0"	"0"

Table 12-1. Watch Timer Control Register (WTCON): 8-bit R/W

Bit Name	Values	Function	Address	
WTCON.7	0	Disable buzzer (BUZ) signal output	FBH, Bank 0	
	1	Enable buzzer (BUZ) signal output		
WTCON.6	0	Disable watch timer (clear frequency dividing circuits)		
	1	Enable watch timer		
WTCON.5-.4	0	0		0.5 kHz buzzer (BUZ) signal output
	0	1		1 kHz buzzer (BUZ) signal output
	1	0		2 kHz buzzer (BUZ) signal output
	1	1		4 kHz buzzer (BUZ) signal output
WTCON.3	0	$fw/2^{14}$ (0.5 sec) interval		
	1	$fw/2^7$ (3.91 ms) interval		
WTCON.2	0	Main system clock divided by 2^7 (fx/128)		
	1	Sub system clock (fxt)		
WTCON.1	0	No interrupt pending		
	0	<i>Clear pending bit (when write)</i>		
	1	Interrupt is pending		
WTCON.0	0	Disable interrupt		
	1	Enable interrupt		

WATCH TIMER CIRCUIT DIAGRAM

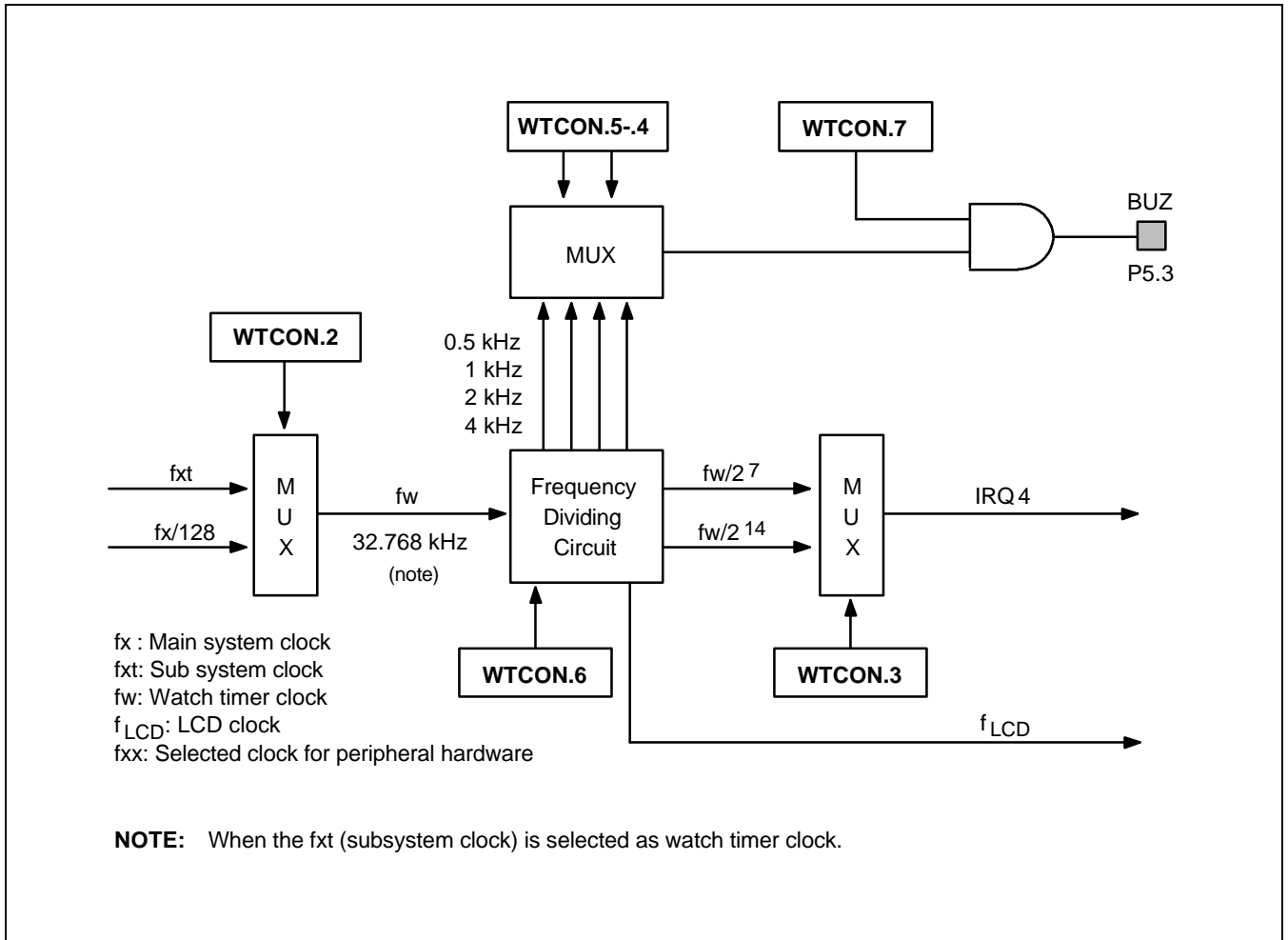


Figure 12-1. Watch Timer Circuit Diagram

13

LCD CONTROLLER/DRIVER

OVERVIEW

The S3C821A microcontroller can directly drive an up-to-224-dot (28 segments x 8 commons) LCD panel. Its LCD block has the following components:

- LCD controller/driver
- Display RAM for storing display data
- 4 common/segment output pins (COM4/SEG0–COM7/SEG3)
- 28 segment output pins (SEG4–SEG31)
- 4 common output pins (COM0–COM3)
- Internal resistor circuit for LCD bias
- V_{LC1} pin for controlling the driver and bias voltage

The LCD control register, LCON, is used to turn the LCD display on and off, switch the current to the dividing resistors for the LCD display, and frame frequency. Data written to the LCD display RAM can be automatically transferred to the segment signal pins without any program control.

When a subsystem clock is selected as the LCD clock source, the LCD display is enabled even in the main clock stop or idle mode.

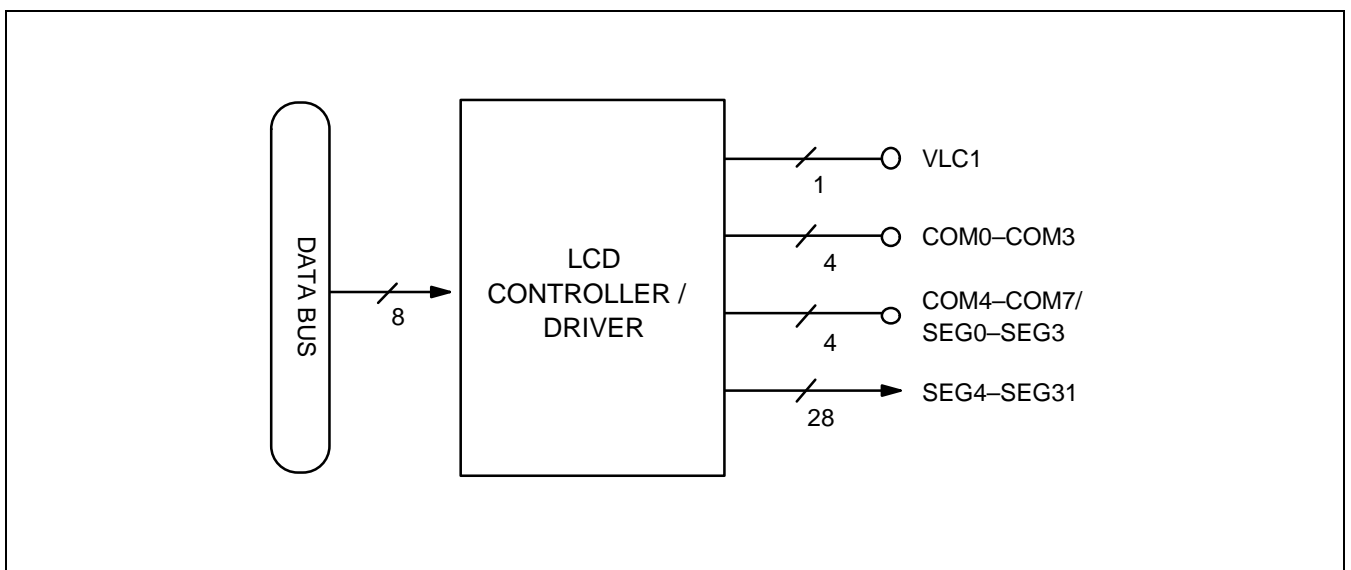


Figure 13-1. LCD Function Diagram

LCD CIRCUIT DIAGRAM

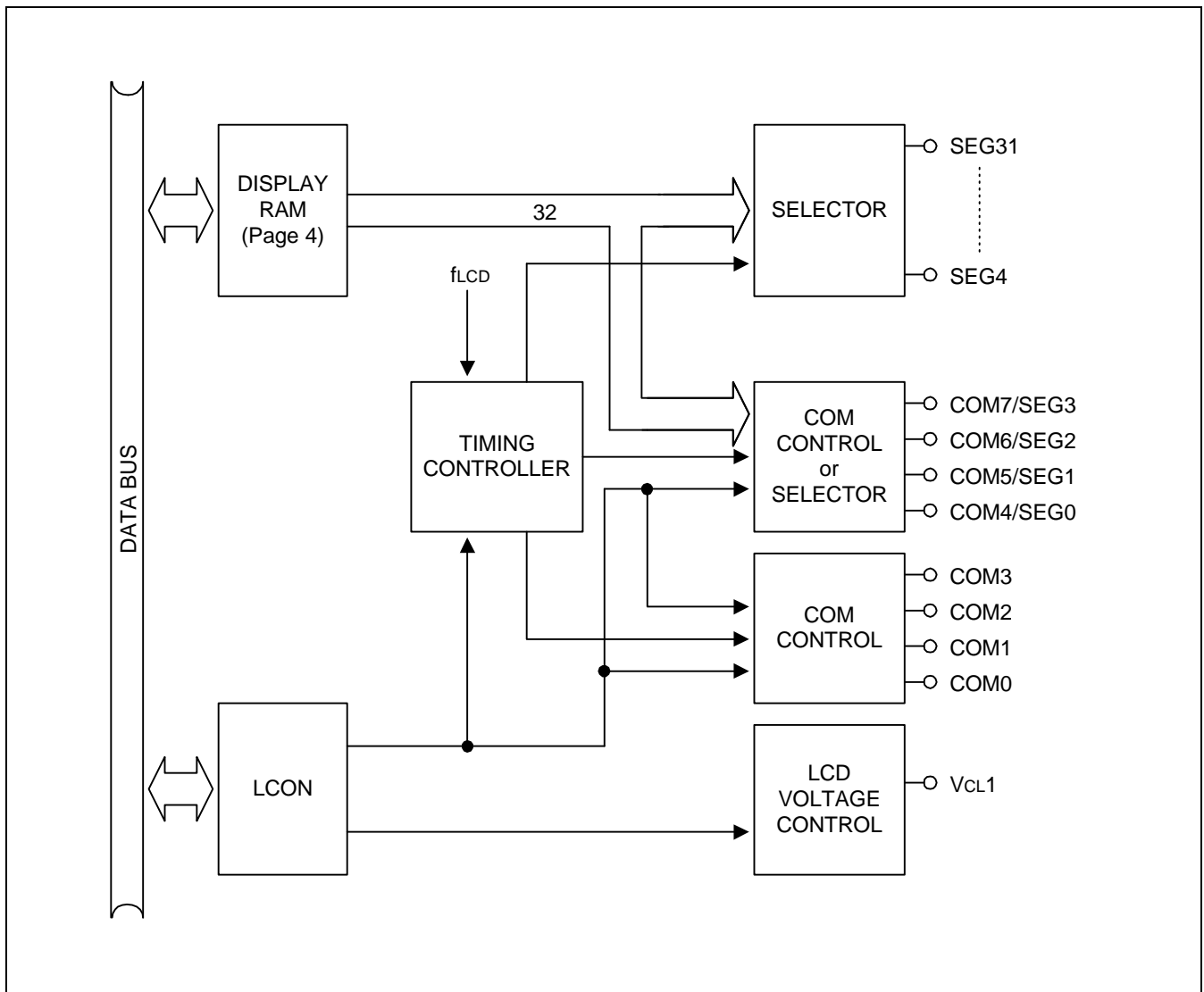


Figure 13-2. LCD Circuit Diagram

LCD RAM ADDRESS AREA

RAM addresses of page 4 are used as LCD data memory. These locations can be addressed by 1-bit or 8-bit instructions. If the bit value of a display segment is "1", the LCD display is turned on. If the bit value is "0", the display is turned off.

Display RAM data are sent out through the segment pins, SEG0–SEG31, using the direct memory access (DMA) method that is synchronized with the f_{LCD} signal. RAM addresses in this location that are not used for LCD display can be allocated to general-purpose use.

COM	Bit	SEG0	SEG1	SEG2	SEG3	SEG4	SEG30	SEG31
COM0	.0								
COM1	.1								
COM2	.2								
COM3	.3								
COM4	.4	00H	01H	02H	03H	04H	30H	31H
COM5	.5								
COM6	.6								
COM7	.7								

Figure 13-3. LCD Display Data RAM Organization

LCD CONTROL REGISTER (LCON)

The LCD control register (LCON) is used to turn the LCD display on and off, LCD frame frequency, and control the flow of the current to the dividing resistors in the LCD circuit. After a RESET, all LCON values are cleared to "0". This turns the LCD display off and stops the flow of the current to the dividing resistors.

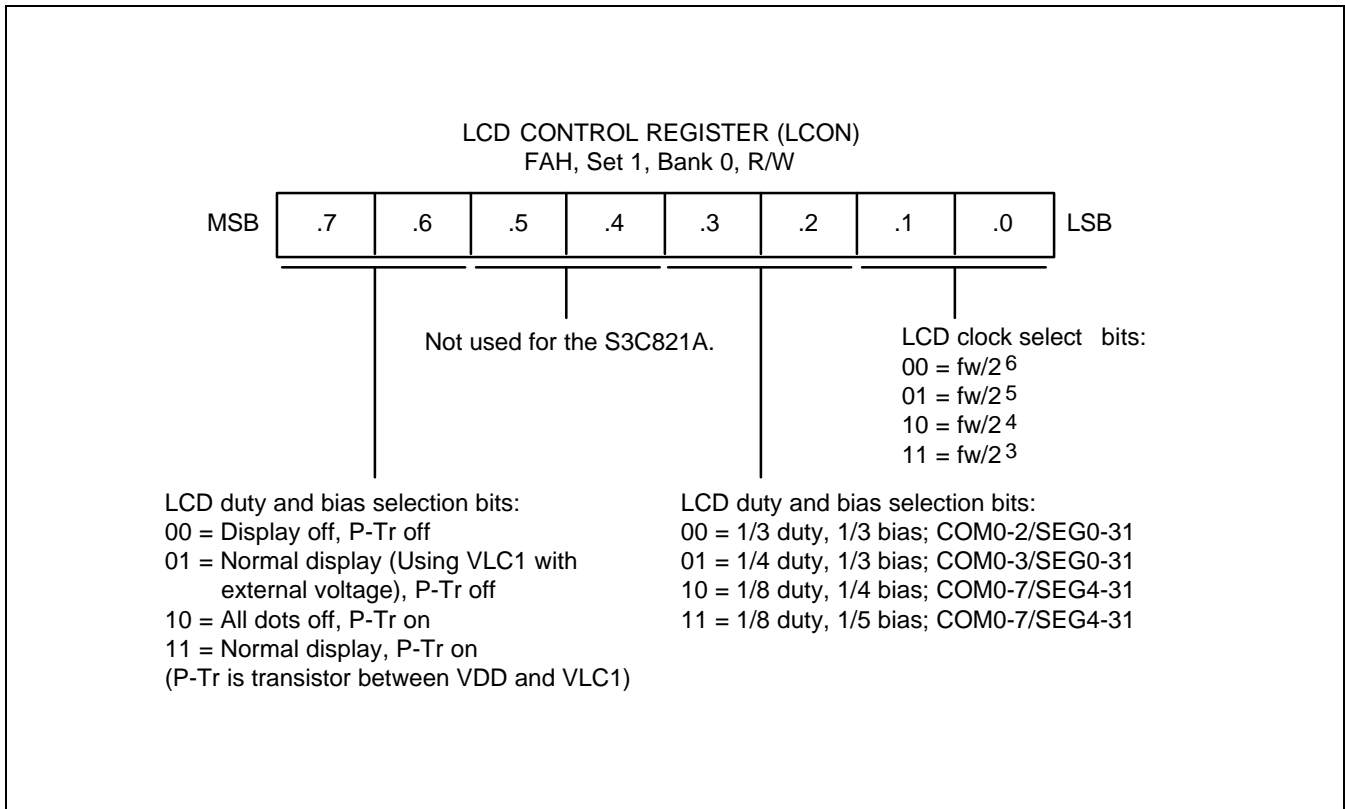


Figure 13-4. LCD Control Register (LCON)

LCD VOLTAGE DIVIDING RESISTORS

On-chip voltage dividing resistors for the LCD drive power supply are fixed to the V_{LC1} - V_{LC5} pins. Figure 13-5 shows the bias connections for the S3C821A LCD drive power supply. To cut off the flow of current through the dividing resistor, manipulate bits 7 and 6 of the LCON register.

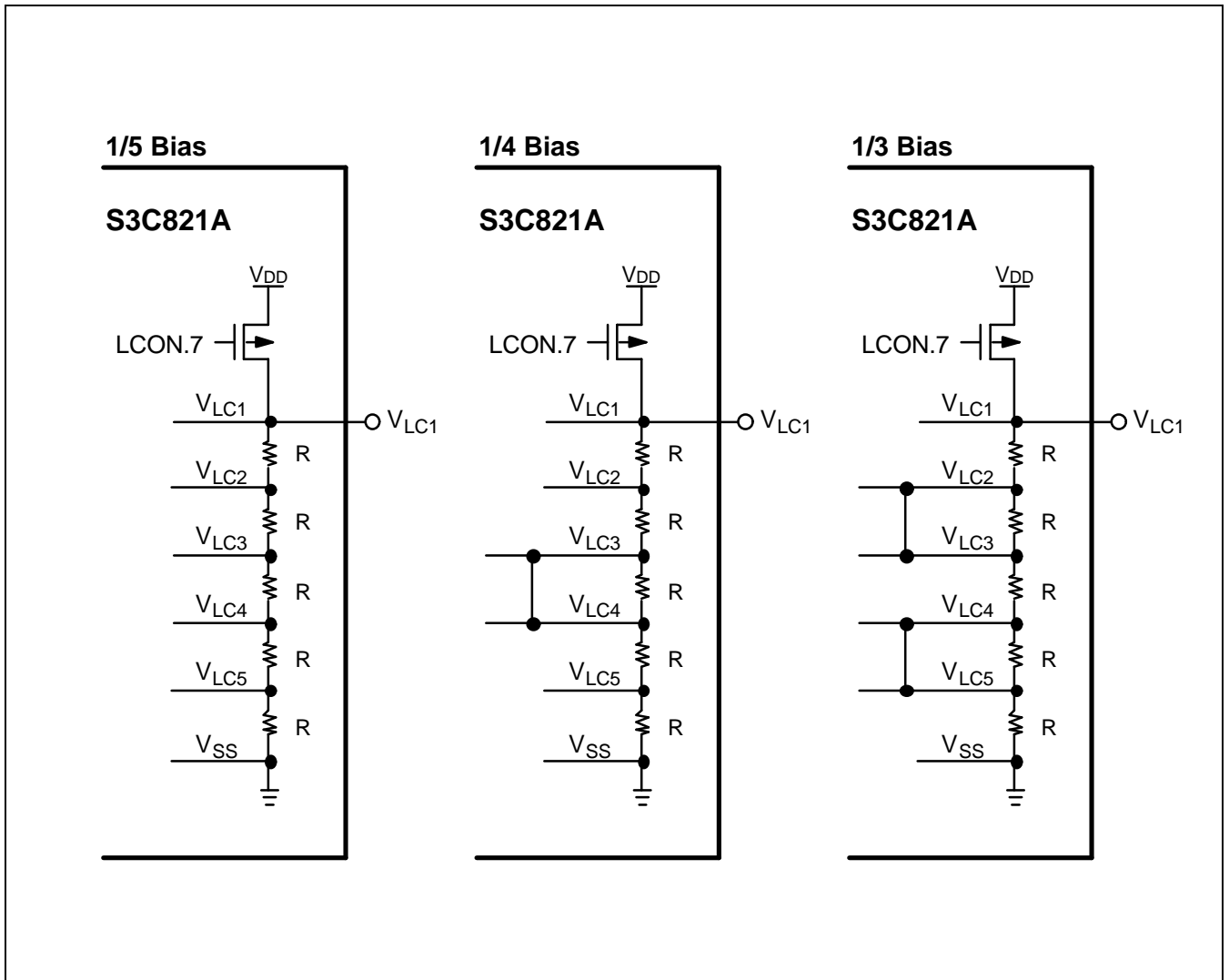


Figure 13-5. LCD Bias Circuit Connection

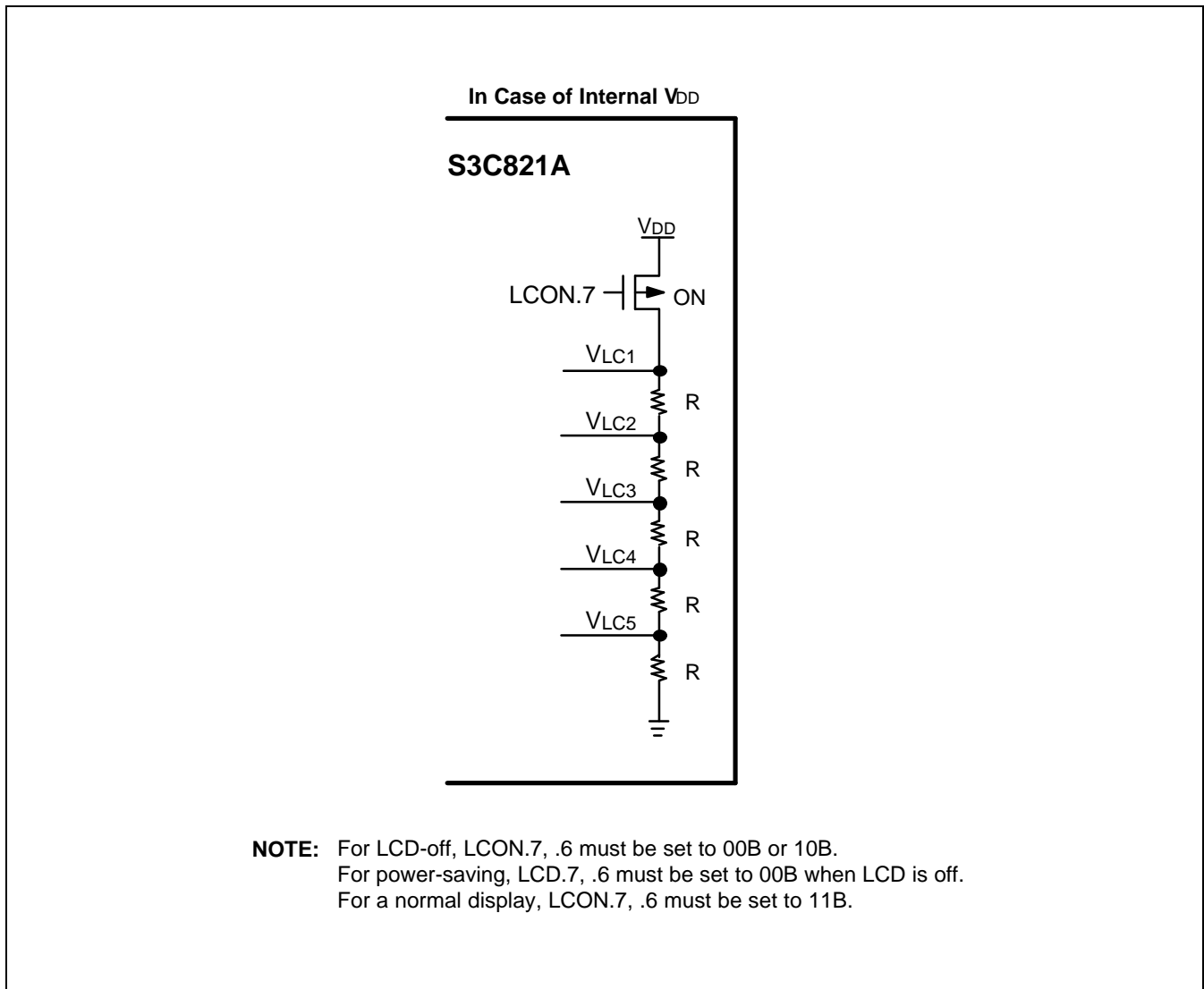


Figure 13-6. Example 1 for the Usage of LCON.7, LCON.6

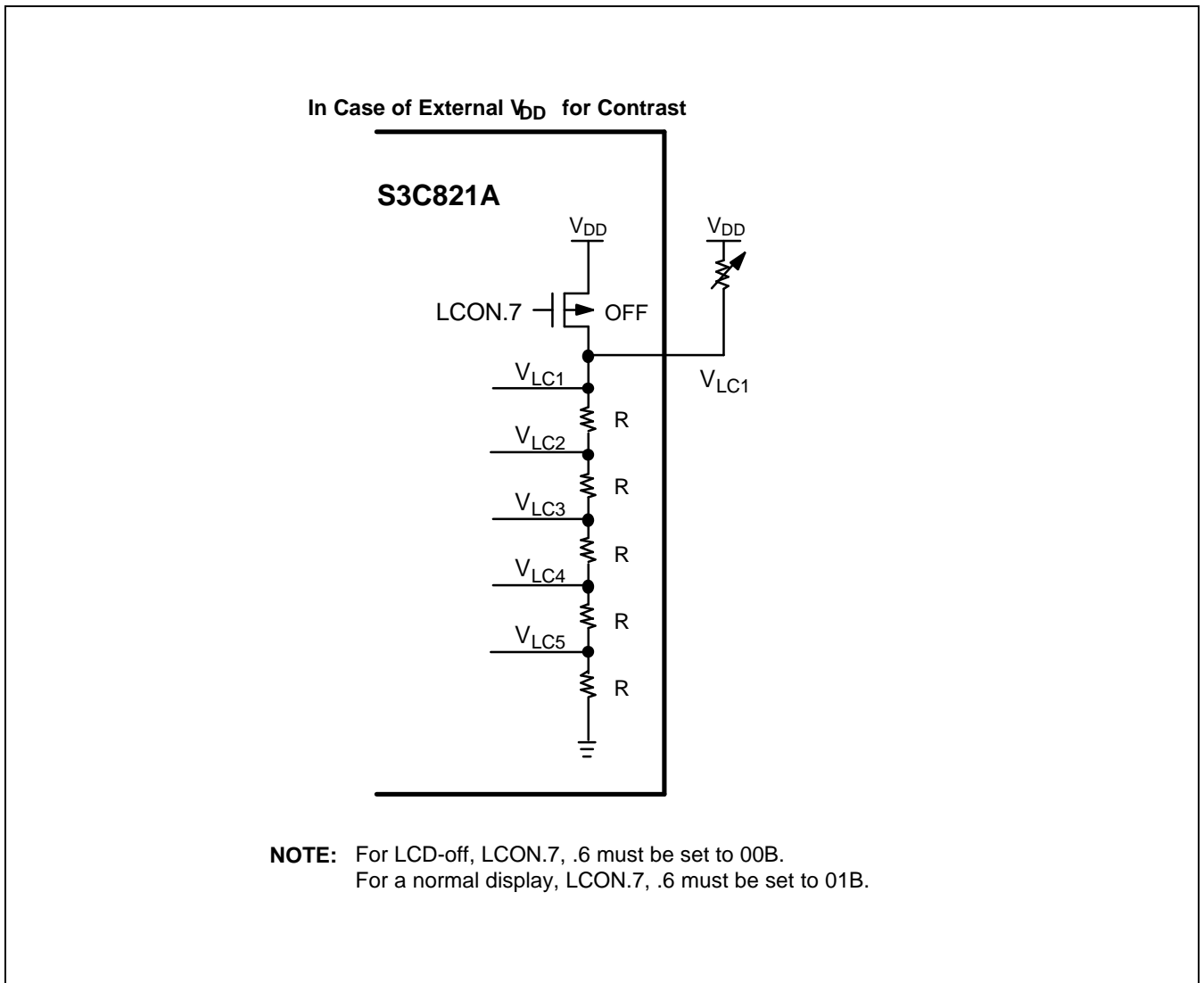
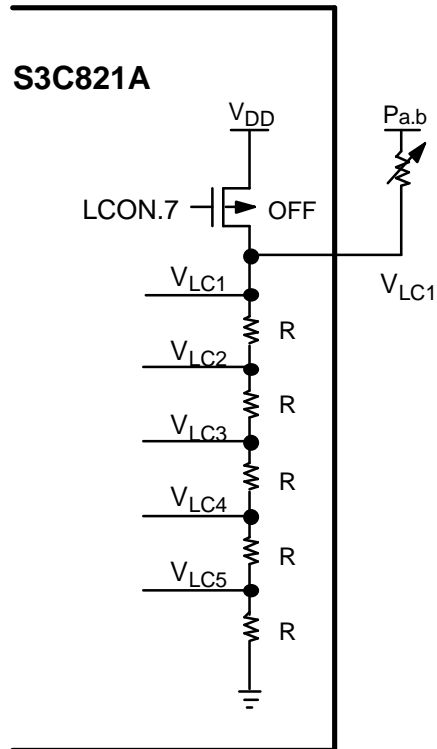


Figure 13-7. Example 2 for the Usage of LCON.7, LCON.6

In Case of Using a Port for Contrast



NOTE: For LCD-off, LCON.7, .6 must be set to 00B.
 For power-saving, Pa.b must be in low level when LCS is off.
 For a normal display, Pa.b must be in high level and LCON.7, .6 must be set to 01B.

Figure 13-8. Example 3 for the Usage of LCON.7, LCON.6

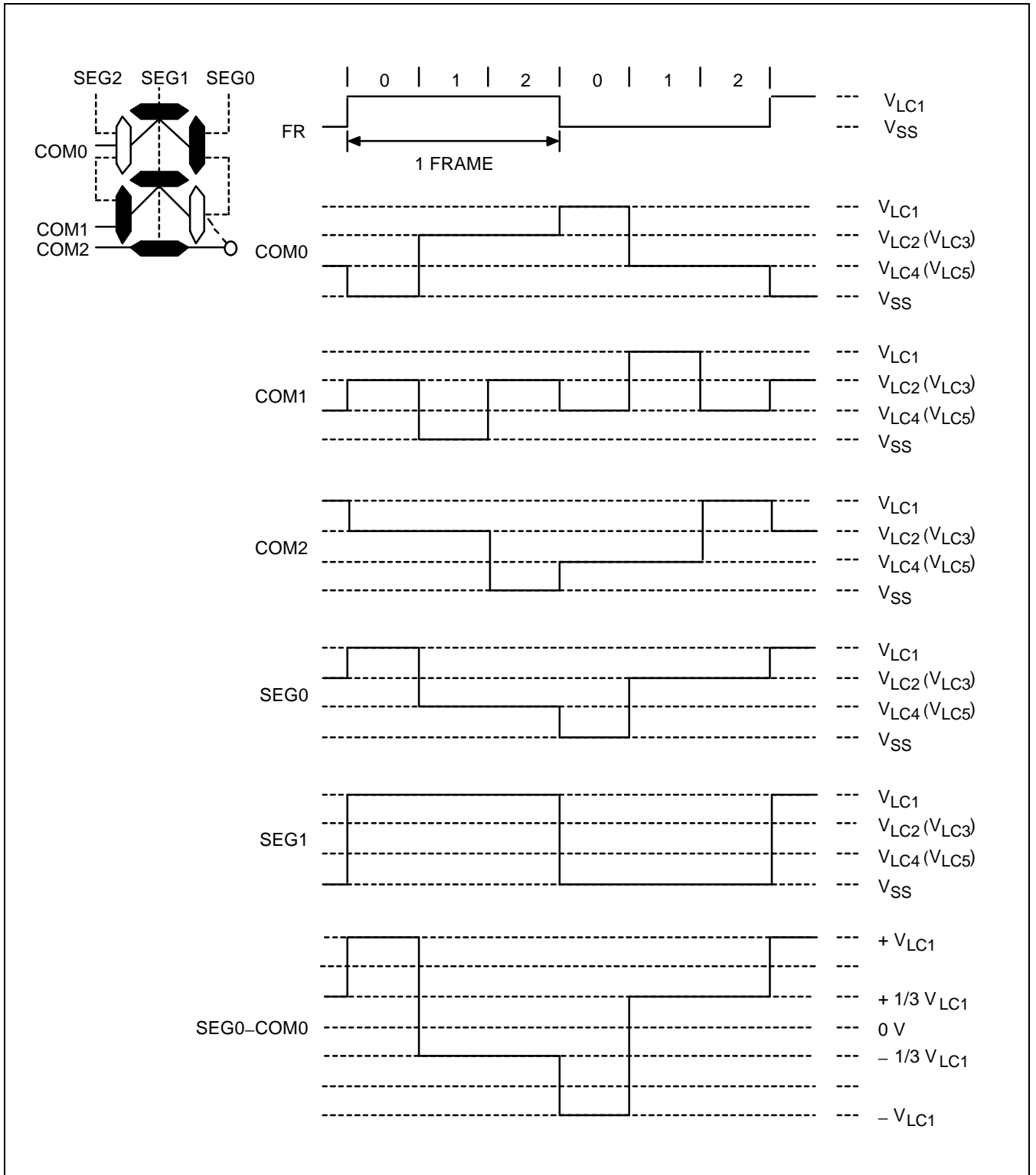


Figure 13-9. LCD Signal Waveforms (1/3 Duty, 1/3 Bias)

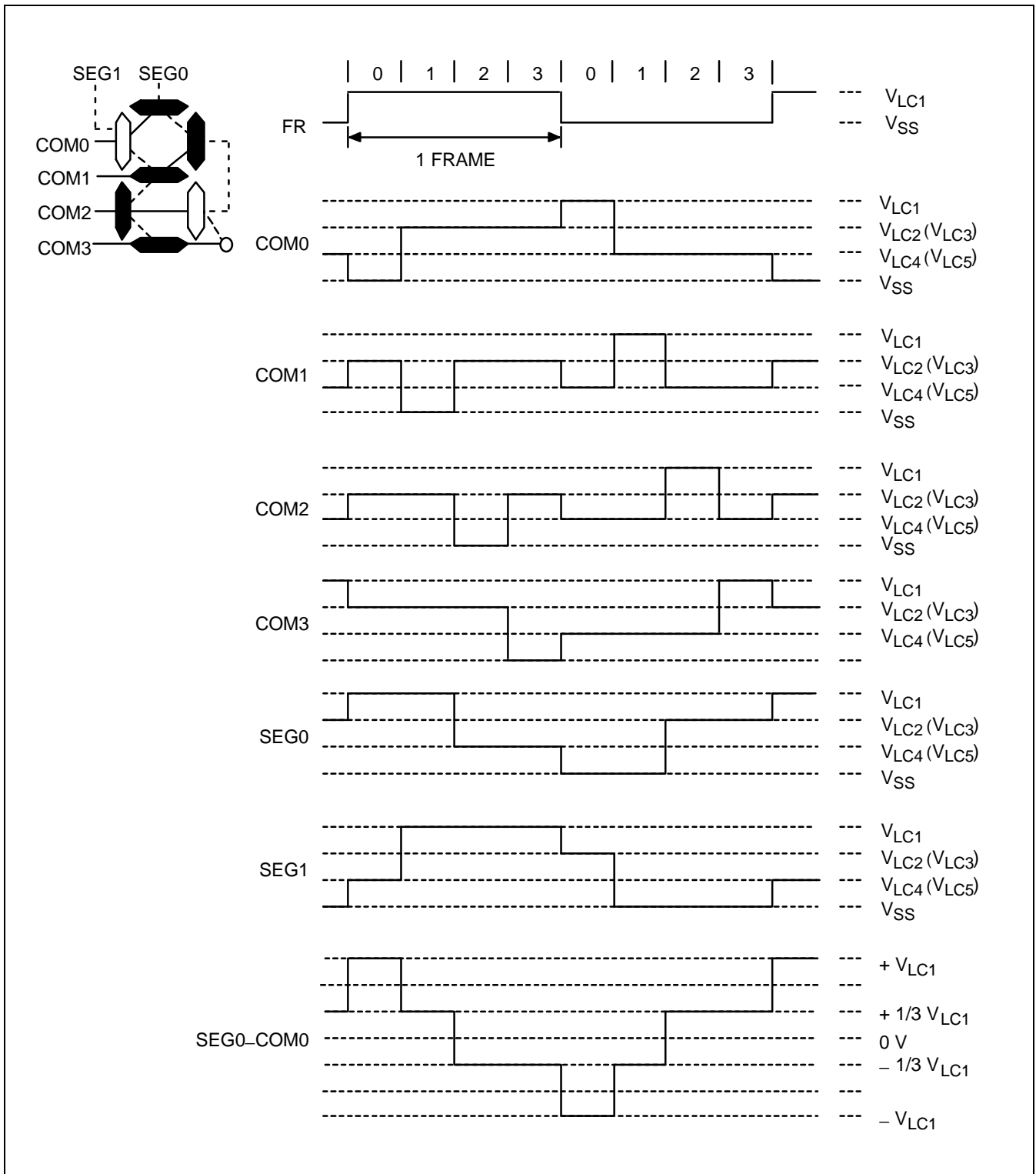


Figure 13-10. LCD Signal Waveforms (1/4 Duty, 1/3 Bias)

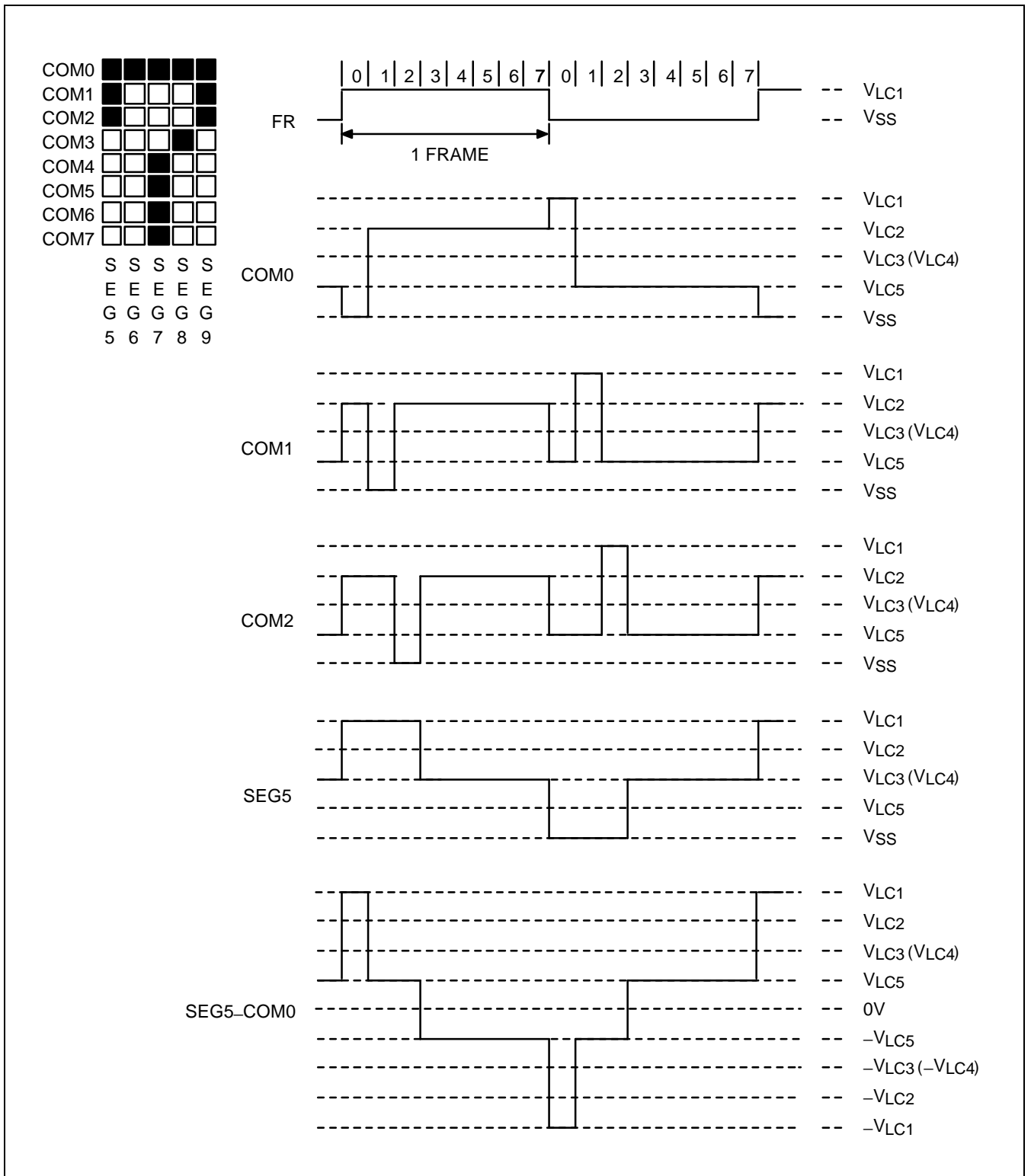


Figure 13-11. LCD Signal Waveforms (1/8 Duty, 1/4 Bias)

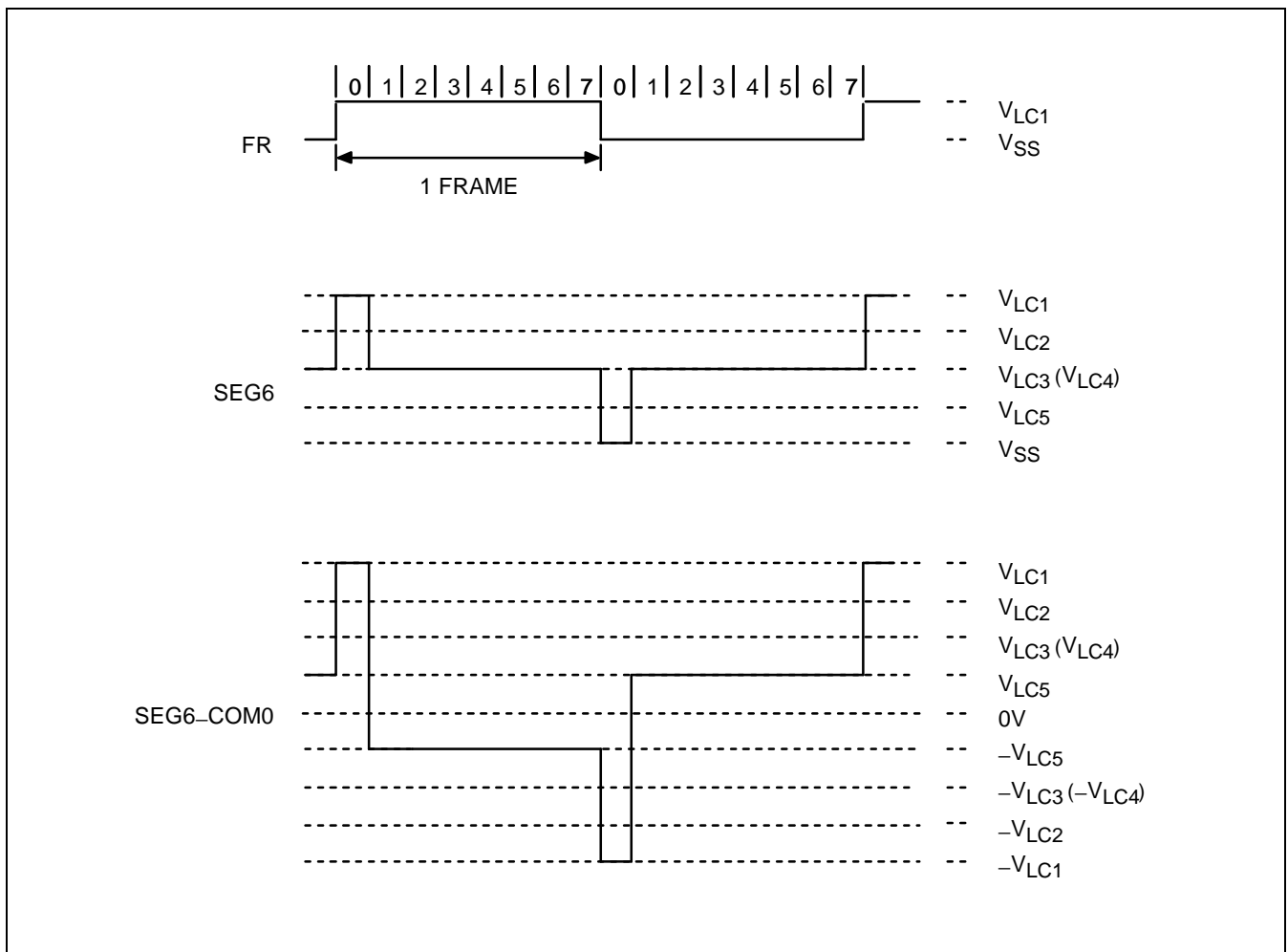


Figure 13-11. LCD Signal Waveforms (1/8 Duty, 1/4 Bias) (Continued)

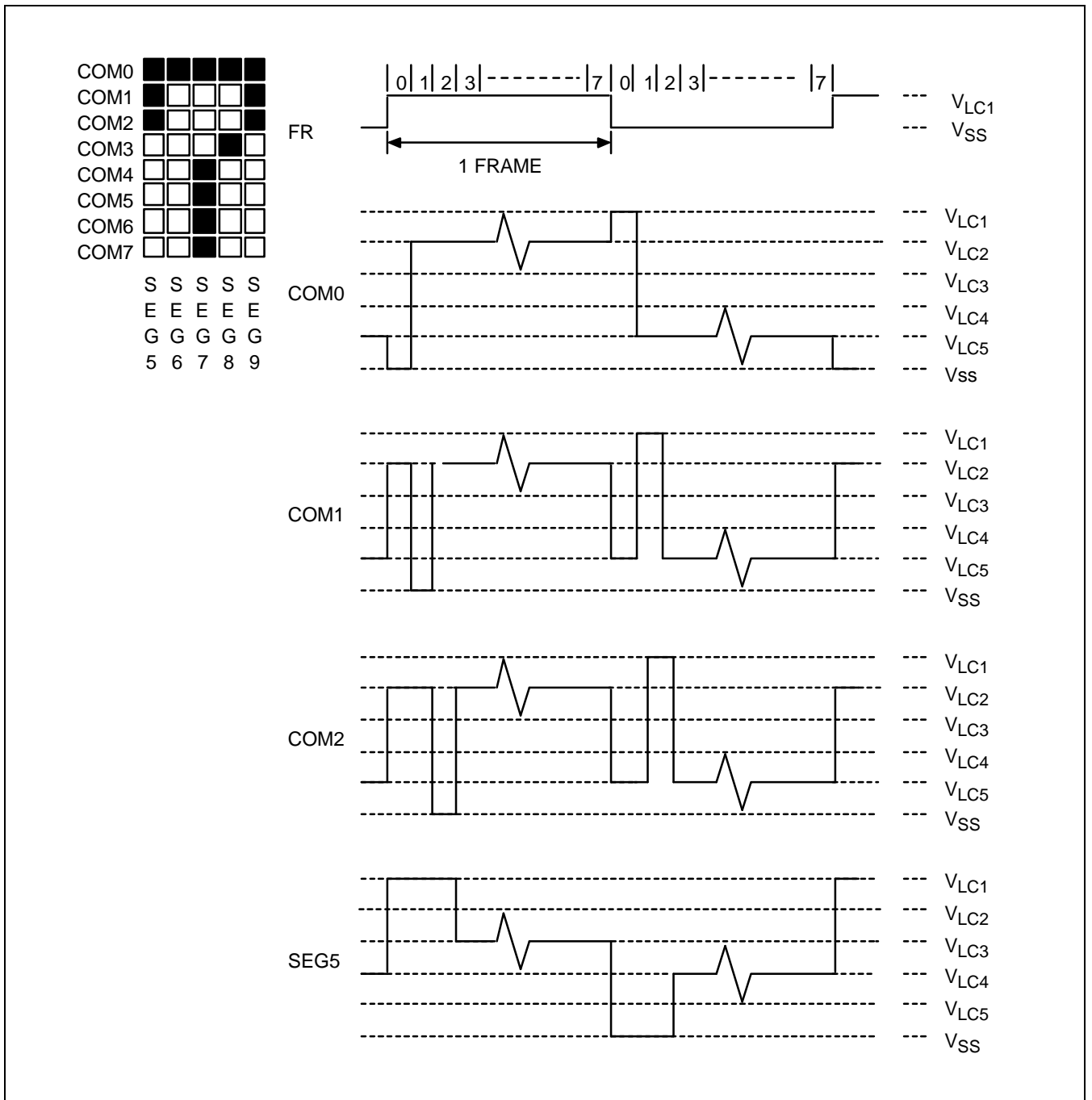


Figure 13-12. LCD Signal Waveforms (1/8 Duty, 1/5 Bias)

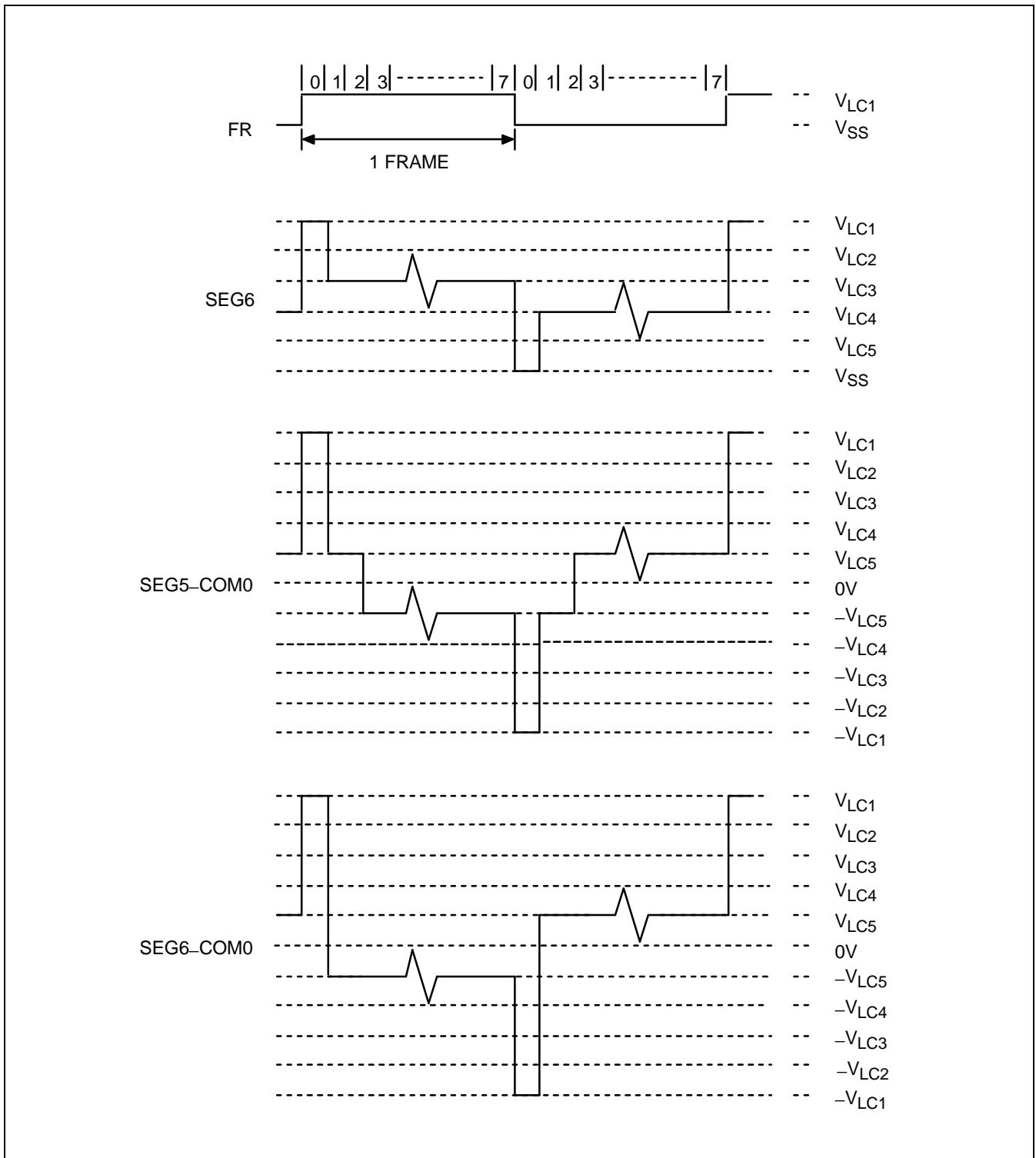


Figure 13-12. LCD Signal Waveforms (1/8 Duty, 1/5 Bias) (Continued)

14 ANALOG-TO-DIGITAL CONVERTER

OVERVIEW

The 8-bit A/D converter (ADC) module uses successive approximation logic to convert analog levels entering at one of the four input channels to equivalent 8-bit digital values. The analog input level must lie between the AV_{REF} and AV_{SS} values. The A/D converter has the following components:

- Analog comparator with successive approximation logic
- D/A converter logic (resistor string type)
- ADC control register (ADCON)
- Four multiplexed analog data input pins (ADC0–ADC3)
- 8-bit A/D conversion data output register (ADDATA)
- AV_{REF} and AV_{SS} input pins

To initiate an analog-to-digital conversion procedure, you should load a value in analog input pin selection bits in the A/D converter control register ADCON to select one of the four analog input pins (ADCn, n = 0–3) and set the conversion start or enable bit, ADCON.0. The read-write ADCON register is located in set 1, bank 0, at address F7H.

During a normal conversion, ADC block initially sets the successive approximation register to 80H (the approximate half-way point of an 8-bit register). This register is then updated automatically during each conversion step. The successive approximation block performs 8-bit conversions for one input channel at a time. You can dynamically select different channels by manipulating the channel selection bit value (ADCON.6–ADCON.4) in the ADCON register.

To start the A/D conversion, you should set the enable bit, ADCON.0. When a conversion is completed, ADCON.3, the end-of-conversion (EOC) bit is automatically set to 1 and the result is dumped into the ADDATA register where it can be read. The A/D converter then enters an idle state. Remember to read the contents of ADDATA before another conversion starts. Otherwise, the previous result will be overwritten by the next conversion result.

NOTE

As the A/D converter has no sample-and-hold circuitry, it is very important that fluctuation in the analog level at the ADC0–ADC3 input pins during a conversion procedure be kept to an absolute minimum. Any change in the input level, perhaps due to noise, will invalidate the result. If the chip enters to STOP or IDLE mode in conversion process, there will be a leakage current path in A/D block. You must use STOP or IDLE mode after ADC operation is finished.

CONVERSION TIMING

The A/D conversion process requires 4 clocks to convert each bit. Therefore, a total of 34 clocks are required to complete an 8-bit conversion (17 μ s–170 μ s). With an 4 MHz f_{xx} clock frequency and f_{xx}/4 clock source selected, one clock cycle is 1 μ s. If each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

$$(start\ 1\ clock + (4\ clock/bit \times 8\ bits) + EOC\ clock) = 34\ clocks, 1\ \mu s \times 34 = 34\ \mu s\ at\ 4\ MHz\ (f_{xx}/4)$$

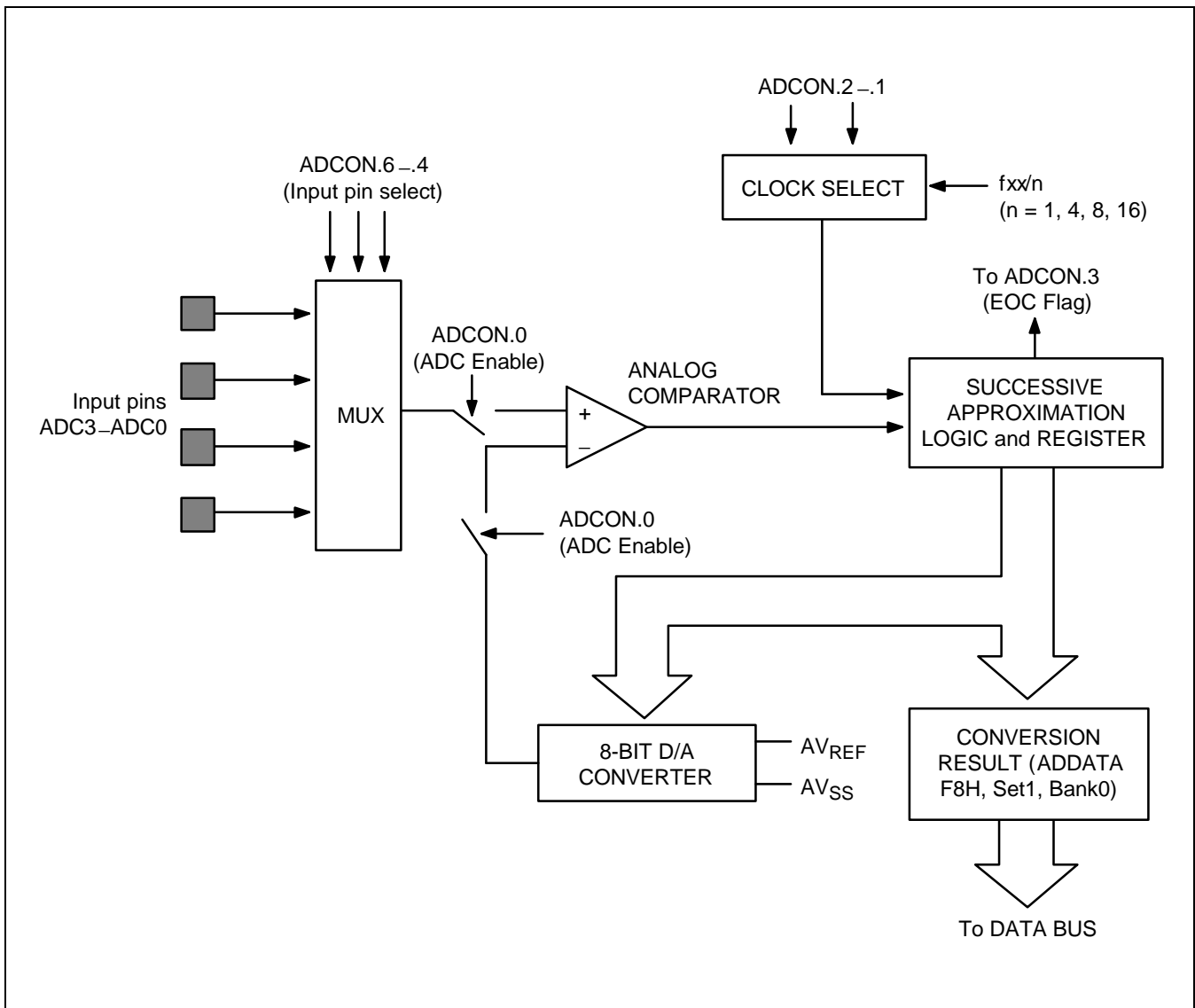


Figure 14-1. A/D Converter Functional Block Diagram

A/D CONVERTER CONTROL REGISTER (ADCON)

The A/D converter control register, ADCON, is located at address F7H in set 1, bank 0. It has four functions:

- Analog input pin selection (bits 4, 5, and 6)
- End-of-conversion status detection (bit 3)
- Clock source selection (bits 2 and 1)
- A/D operation start or enable (bit 0)

After a reset, the ADC0 pin is automatically selected as the analog data input pin, and the start bit is turned off. You can select only one analog input channel at a time. Other analog input pins (ADC0–ADC3) can be selected dynamically by manipulating the ADCON.4–.6 bits.

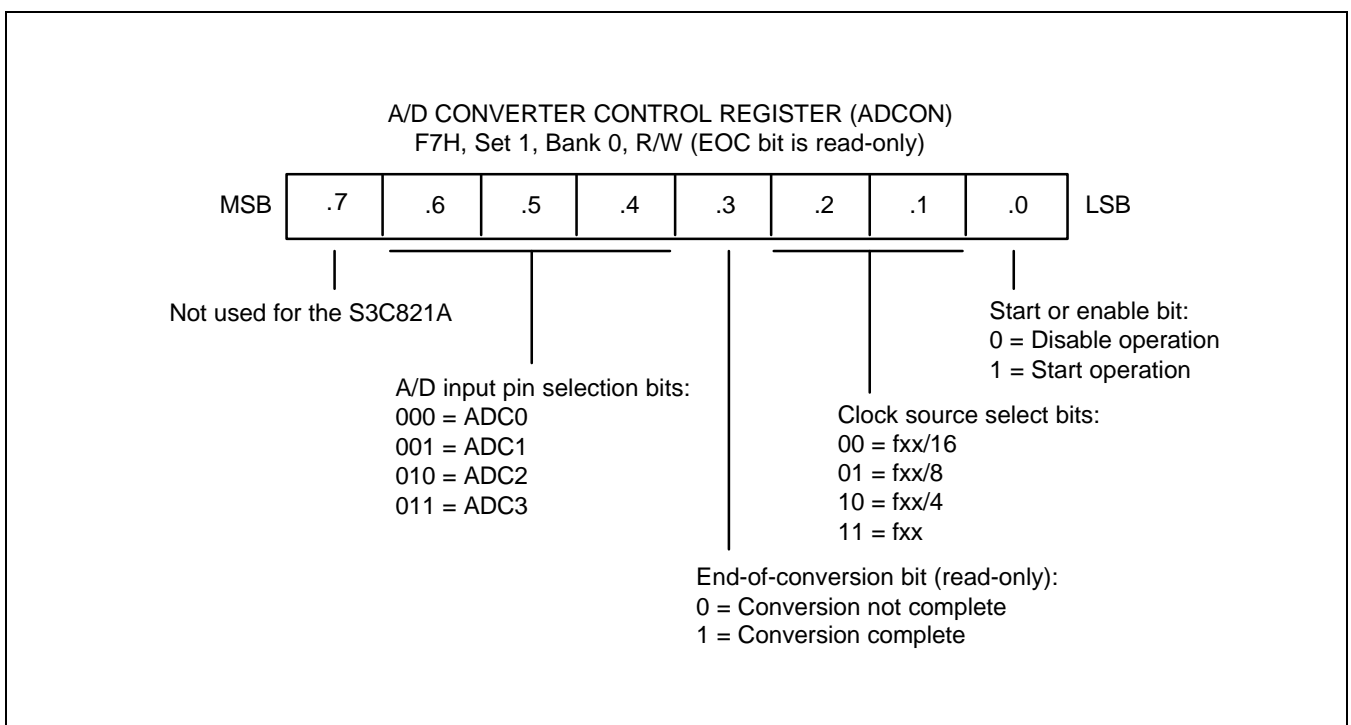


Figure 14-2. A/D Converter Control Register (ADCON)

INTERNAL REFERENCE VOLTAGE LEVELS

In the ADC function block, the analog input voltage level is logically compared to a reference voltage. For the S3C821A, the analog input level must be within the range of AV_{SS} to AV_{REF} , where $AV_{REF} = V_{DD}$. Different reference voltage levels are generated internally along the resistor tree during the analog conversion process for each conversion step. The reference voltage level for the first bit conversion is always $1/2 AV_{REF}$.

15 SERIAL I/O INTERFACE

OVERVIEW

Serial I/O module, SIO can interface with various types of external device that require serial data transfer. The components of each SIO function block are:

- 8-bit control register (SIOCON)
- Clock selector logic
- 8-bit data buffer (SIODATA)
- 8-bit prescaler (SIOPS)
- 3-bit serial clock counter
- Serial data I/O pins (SI, SO)
- External clock input/output pins (SCK)

The SIO module can transmit or receive 8-bit serial data at a frequency determined by its corresponding control register settings. To ensure flexible data transmission rates, you can select an internal or external clock source.

PROGRAMMING PROCEDURE

To program the SIO modules, follow these basic steps:

1. Configure the I/O pins at port (SO, SCK, SI) by loading the appropriate value to the P5CONL register if necessary.
2. Load an 8-bit value to the SIOCON control register to properly configure the serial I/O module. In this operation, SIOCON.2 must be set to "1" to enable the data shifter.
3. For interrupt generation, set the serial I/O interrupt enable bit (SIOCON.5) to "1".
4. When you transmit data to the serial buffer, write data to SIODATA and set SIOCON.3 to 1, the shift operation starts.
5. When the shift operation (transmit/receive) is completed, the SIO pending bit (SIOCON.6) is set to "1" and an SIO interrupt request is generated.

SIO CONTROL REGISTER (SIOCON)

The control register for serial I/O interface module, SIOCON, is located at F5H in set 1, bank 0. It has the control settings for SIO module.

- Clock source selection (internal or external) for shift clock
- Interrupt enable
- Edge selection for shift operation
- Clear 3-bit counter and start shift operation
- Shift operation (transmit) enable
- Mode selection (transmit/receive or receive-only)
- Data direction selection (MSB first or LSB first)

A reset sets the SIOCON value to "80H". This configures the corresponding module with a CPU clock source at the SCK, selects receive-only operating mode, and falling edge start. The data shift operation and the interrupt are disabled. The selected data direction is MSB-first.

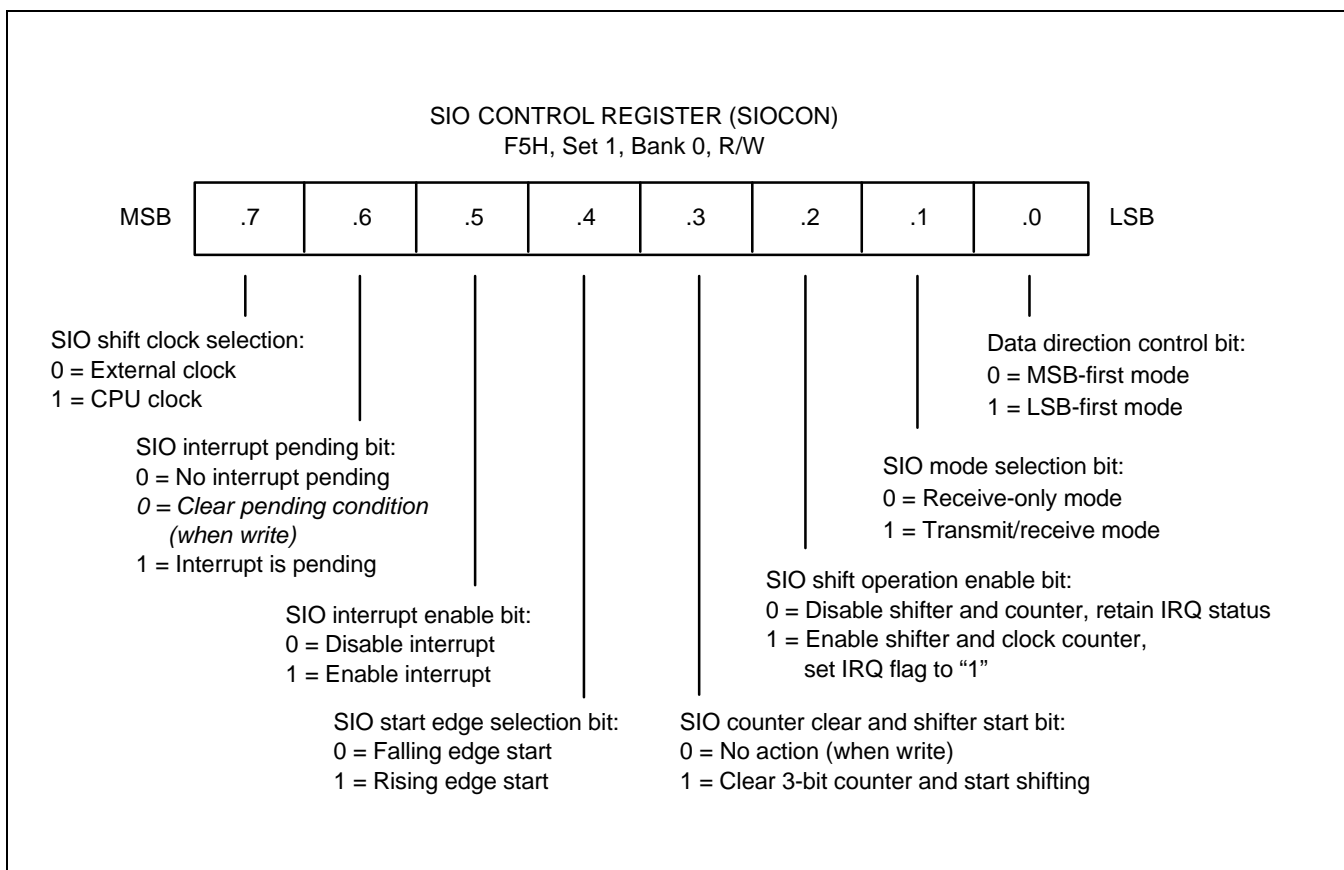


Figure 15-1. Serial I/O Port Control Register (SIOCON)

SIO PRE-SCALER REGISTER (SIOPS)

SIO prescaler register, SIOPS, is located at F6H in set 1, bank 0. The value stored in the SIO prescaler registers, SIOPS, lets you determine the SIO clock rate (baud rate) as follows:

$$\text{Baud rate} = \text{Input clock (CPU clock)} / 2 \times (\text{Prescaler value} + 1), \text{ or SCK input clock}$$

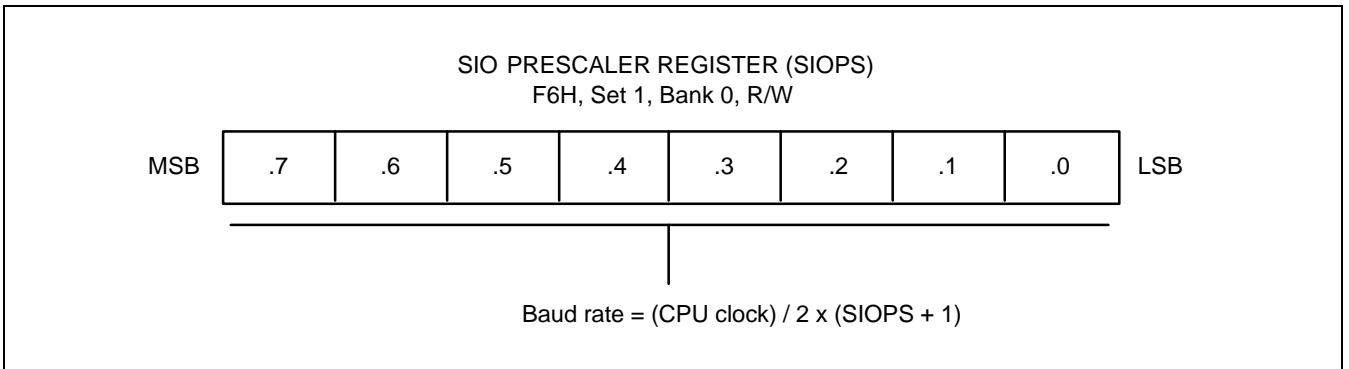


Figure 15-2. SIO Prescaler Register (SIOPS)

BLOCK DIAGRAM

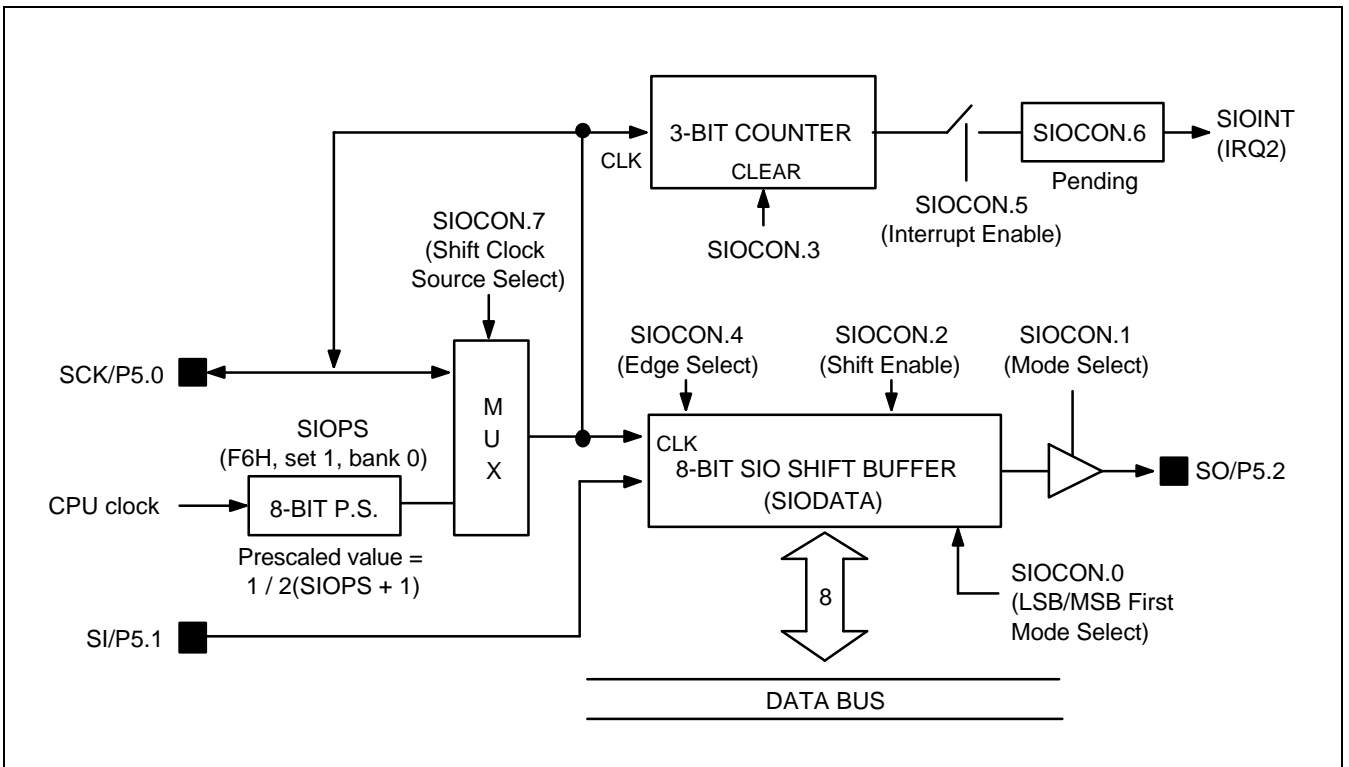


Figure 15-3. SIO Functional Block Diagram

SERIAL I/O TIMING DIAGRAMS

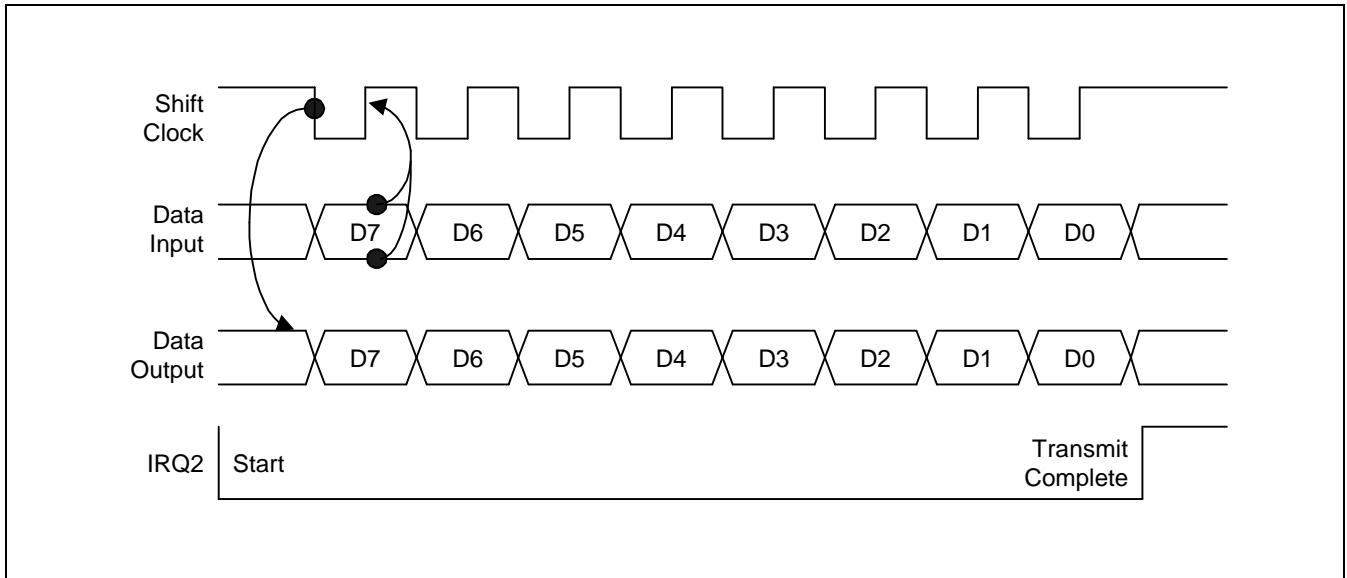


Figure 15-4. SIO Timing in Transmit/Receive Mode (Falling Edge Start)

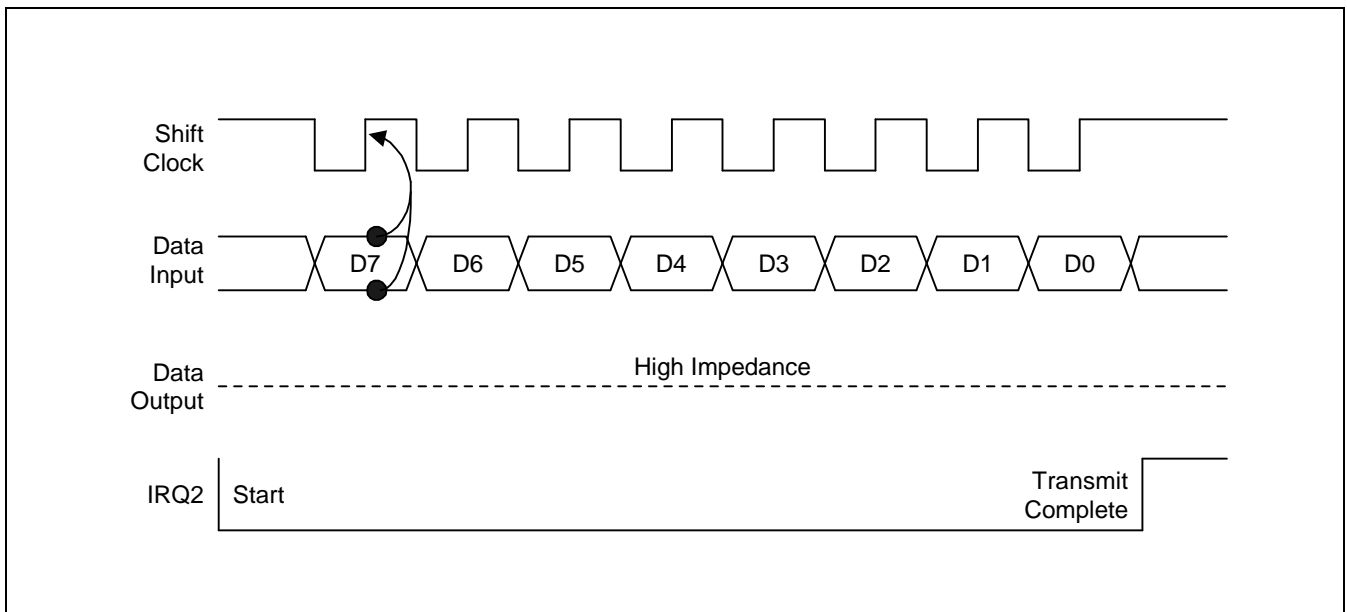


Figure 15-5. SIO Timing in Receive-Only Mode (Falling Edge Start)

SERIAL I/O TIMING DIAGRAMS (Continued)

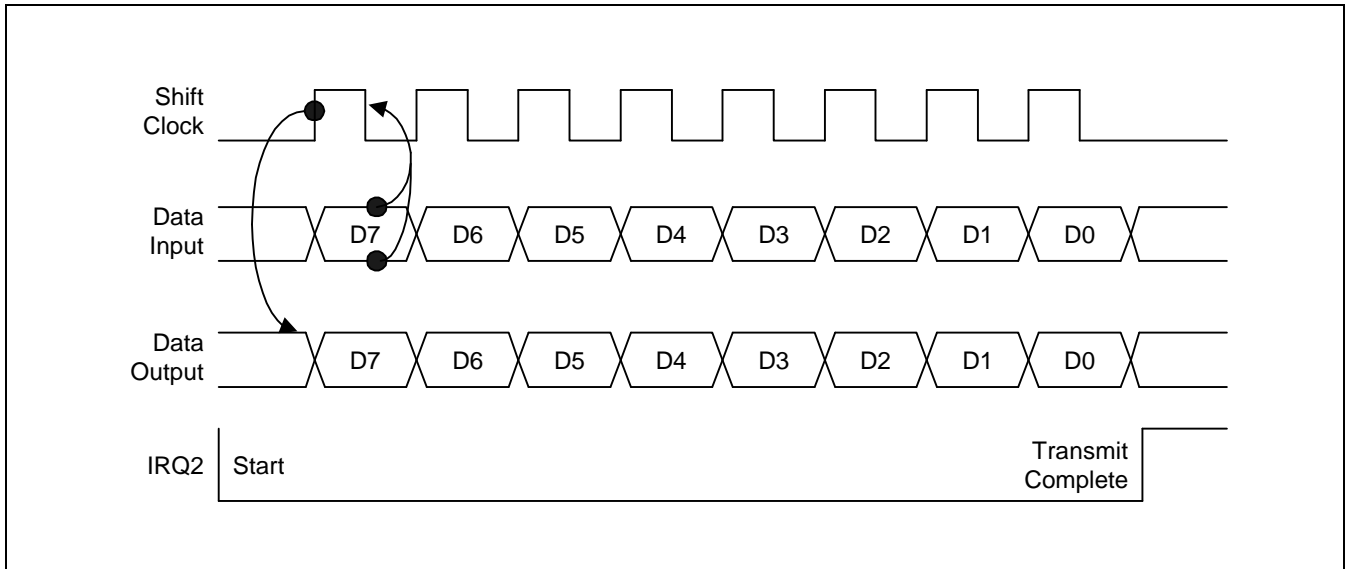


Figure 15-6. Timing in Transmit/Receive Mode (Rising Edge Start)

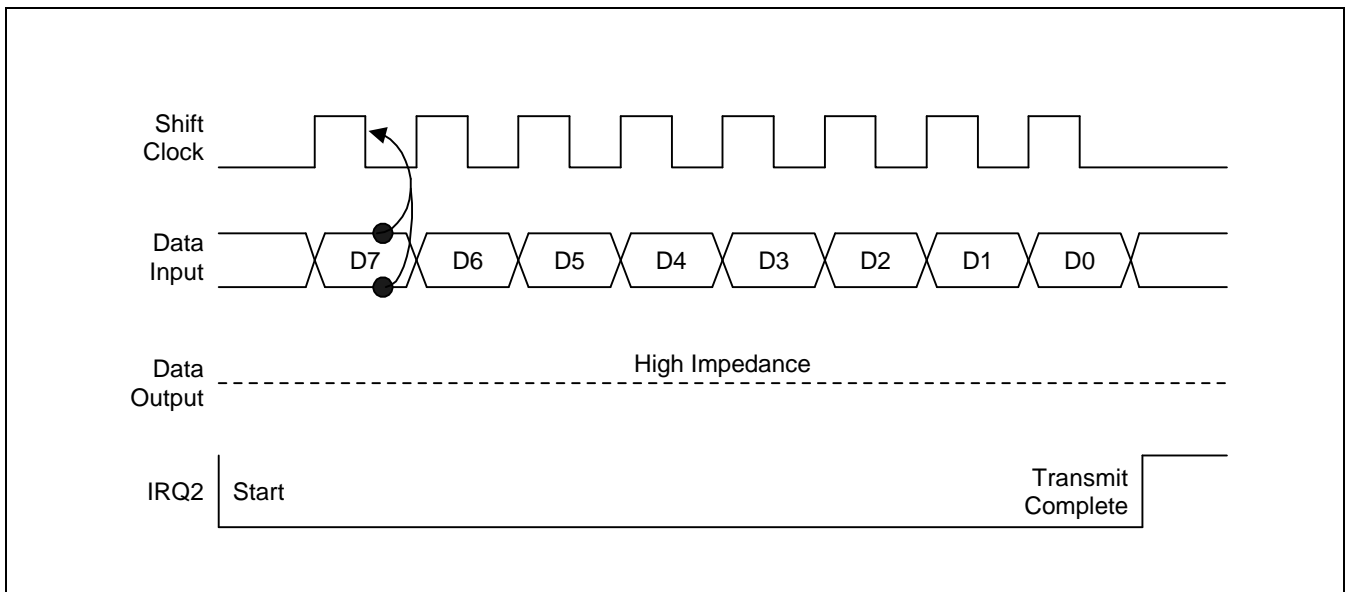


Figure 15-7. Timing in Receive-Only Mode (Rising Edge Start)

16

EXTERNAL INTERFACE

OVERVIEW

The SAM8 architecture supports an external memory interface. Data memory areas in external devices can be accessed over the 16-bit multiplexed address/data bus.

The S3C821A microcontroller has a total of 80 pins, 47 of which are for programmable I/O. Of these 47 I/O pins, up to 20 can be configured as an external interface that supports access to external memory and other peripheral devices.

Since memory addresses that are carried over the address/data bus are 16 bits long, up to 64 K bytes of memory space can be addressed.

EXTERNAL INTERFACE CONTROL REGISTERS

This subsection presents an overview of the S3C821A system registers which are used to configure and control the external peripheral interface:

- System mode register (SYM, R222, DEH, set 1)
- Port 0 control register (P0CON, R224, E0H, set 1, bank 1)
- Port 1 control register (P1CON, R226, E2H, set 1, bank 1)
- Port 2 low-byte control register (P2CONL, R229, E5H, set 1, bank 1)

Detailed descriptions of each of these registers can be found in Part I, Chapter 4, "Control Registers."

PORT 0 CONTROL REGISTER (P0CON)

The port 0 control register P0CON (E0H, set 1, bank 1) controls the upper and lower nibble configuration for port 0 pins. When P0CON bit 7 = "1", the upper nibble pins P0.7–P0.4 are configured as lines for the external memory interface. When P0CON bit 3 = "1", the lower nibble pins P0.3–P0.0 are configured for external memory. After a reset, P0CON is cleared to 00H. Bits 7 and/or 3 must then be set to "1" by program software to enable the external memory interface function for port 0.

PORT 1 CONTROL REGISTER (P1CON)

The port 1 control register P1CON (E2H, set 1, bank 1) functions identically to the P0CON register, except that it controls the upper and lower nibble configuration for port 1 pins: P1.7–P1.4 and P1.3–P1.0, respectively. P1CON is also cleared to 00H by a reset.

PORT 2 CONTROL REGISTER (P2CONL)

The pins of I/O port 2, P2.3–P2.0, can alternately be used as lines for the signal outputs that are required to control the activity of the multiplexed external memory interface bus. You manipulate the bit-pairs in the control register, P2CONL (E5H, set 1, bank 1) to configure the pins individually for general-purpose use or as external memory bus control lines. If the external memory interface is implemented, you must configure all four pins as memory lines. When bit pairs 7/6, 5/4, 3/2, and 1/0 are set to "11B", the corresponding memory signal outputs are activated:

Bit-pair	Pin	Symbol	Function
7/6	P2.3	DM	Data memory pin
5/4	P2.2	DW	Data write pin
3/2	P2.1	DR	Data read pin
1/0	P2.0	AS	Address strobe pin

In normal operating mode, a reset operation clears P2CONL to 00H, configuring P2.3–P2.0 as normal input pins.

HOW TO CONFIGURE THE EXTERNAL INTERFACE

The 3-state external memory interface is enabled or disabled by manipulating bit 7 of the system mode register SYM (R222, DEH). A reset clears SYM.7 to logic zero, disabling the high impedance levels of the interface bus lines and enabling the external interface.

To access the external memory, its port must be selected to external interface lines.

Table 16-1. Control Register Overview for the External Memory Interface

Register	Location	Bit (s)	Description
SYM	DEH	7	External 3-state interface enable bit
P0CON	E0H, Bank 1	3	If "1", enable port 0 pins P0.0–P0.3 for external memory interface
		7	If "1", enable port 0 pins P0.4–P0.7 for external memory interface
P1CON	E2H, Bank 1	3	If "1", enable port 1 low nibble pins (P1.0–P1.3) for external memory interface
		7	If "1", external memory interface enable for port 1 high nibble pins (P1.4–P1.7)
P2CONL	E5H, Bank 1	1, 0	If both "1", address strobe (AS) enabled at P2.0
		3, 2	If both "1", data read signal (DR) enabled at P2.1
		5, 4	If both "1", data write signal (DW) enabled at P2.2
		7, 6	If both "1", data memory signal (DM) enabled at P2.3

Table 16-2. External Interface Control Register Values After a RESET (Normal Mode)

Register Name	Mnemonic	Address		Bit Values After RESET							
		Dec	Hex	7	6	5	4	3	2	1	0
System Mode Register	SYM	R222	DEH	0	–	–	x	x	x	0	0
Port 0 Control Register	P0CON	R224	E0H, Bank 1	0	0	0	0	0	0	0	0
Port 1 Control Register	P1CON	R226	E2H, Bank 1	0	0	0	0	0	0	0	0
Port 2 Control Register (Low Byte)	P2CONL	R229	E5H, Bank 1	0	0	0	0	0	0	0	0

NOTE: A dash (–) indicates that the bit is not mapped. An "x" means that the value is undefined after a RESET.

USING AN EXTERNAL SYSTEM STACK

SAM8 microcontrollers use the system stack to implement subroutine calls and returns for interrupt processing and for dynamic data storage. Stack operations are supported in either the internal register file or in externally configured data memory.

The PUSH and POP instructions support external system stack operations. The instructions PUSHUI, PUSHUD, POPUI, and POPUD support user-defined stack operations in the register file only.

After a reset, the stack pointer value is undetermined. For external stack operations, a 16-bit stack pointer value (in other words, both SPL and SPH) must be used.

An external stack holds return addresses for procedure calls and interrupts, as well as dynamically-generated data. The contents of the PC are saved on the external stack during a CALL instruction and restored during a RET instruction. During interrupts, the contents of the PC and the FLAGS register are saved on the external stack and then restored by the IRET instruction.

To select the external stack area option, bit 1 in the external memory timing register (EMT, FEH, set 1, bank 0) must be set to logic one. The instruction used to change the stack selection bit in the EMT register should not be immediately followed by an instruction that uses the stack, since this will cause indeterminate program flow. Also, interrupts should be disabled with a DI instruction before changing the stack selection bit.

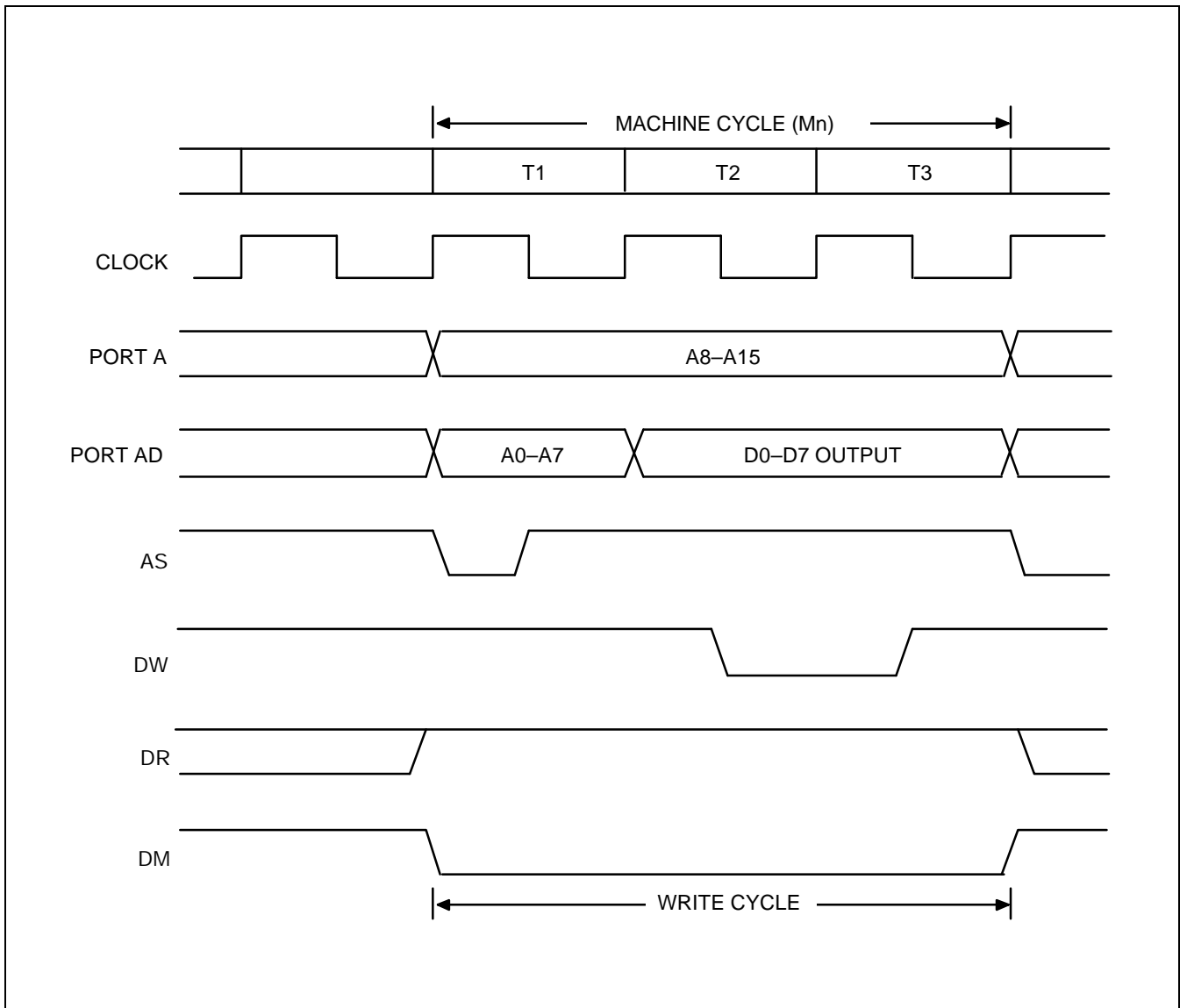


Figure 16-1. S3C821A External Bus Write Cycle Timing Diagram

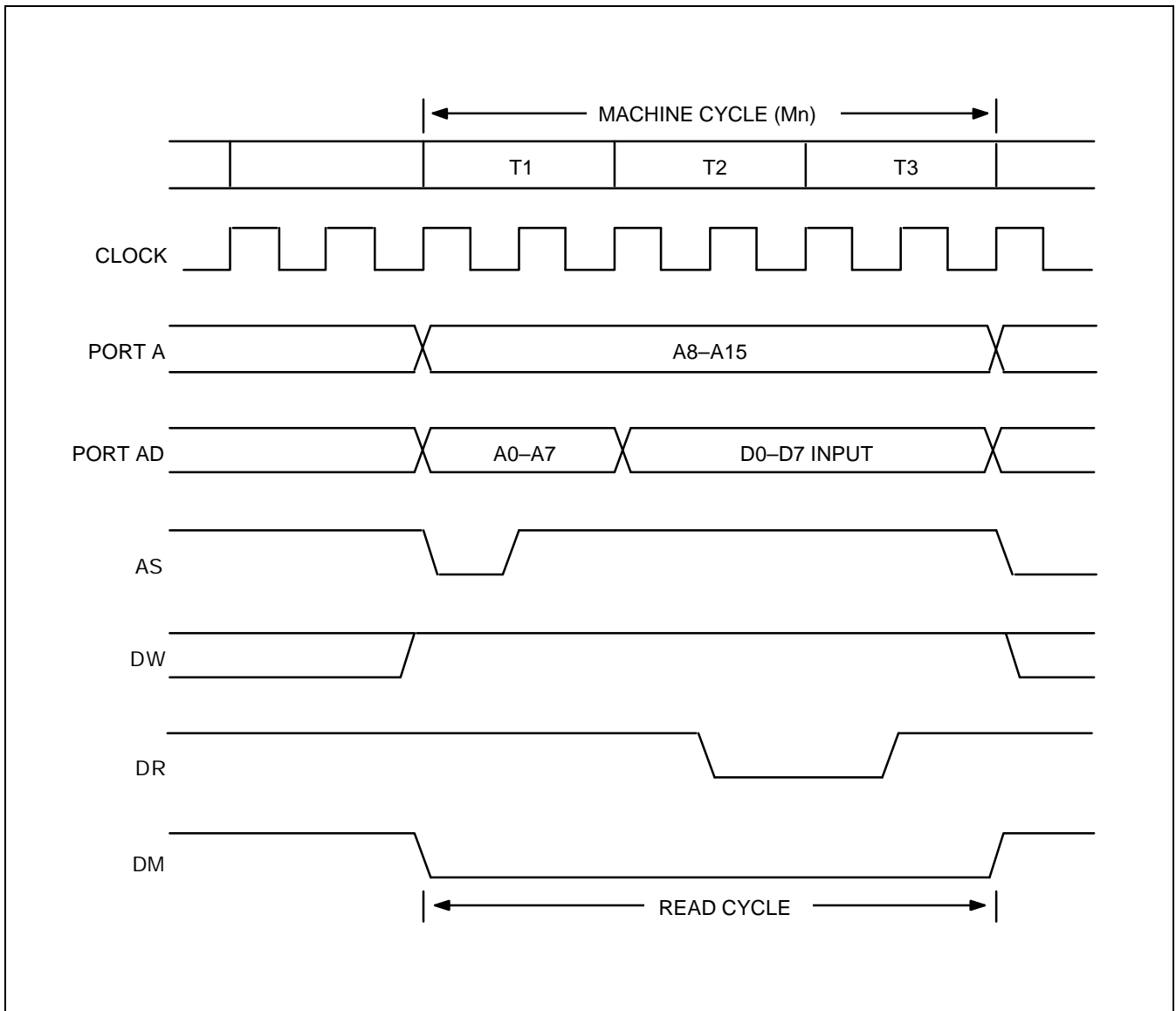


Figure 16-2. S3C821A External Bus Read Cycle Timing Diagram

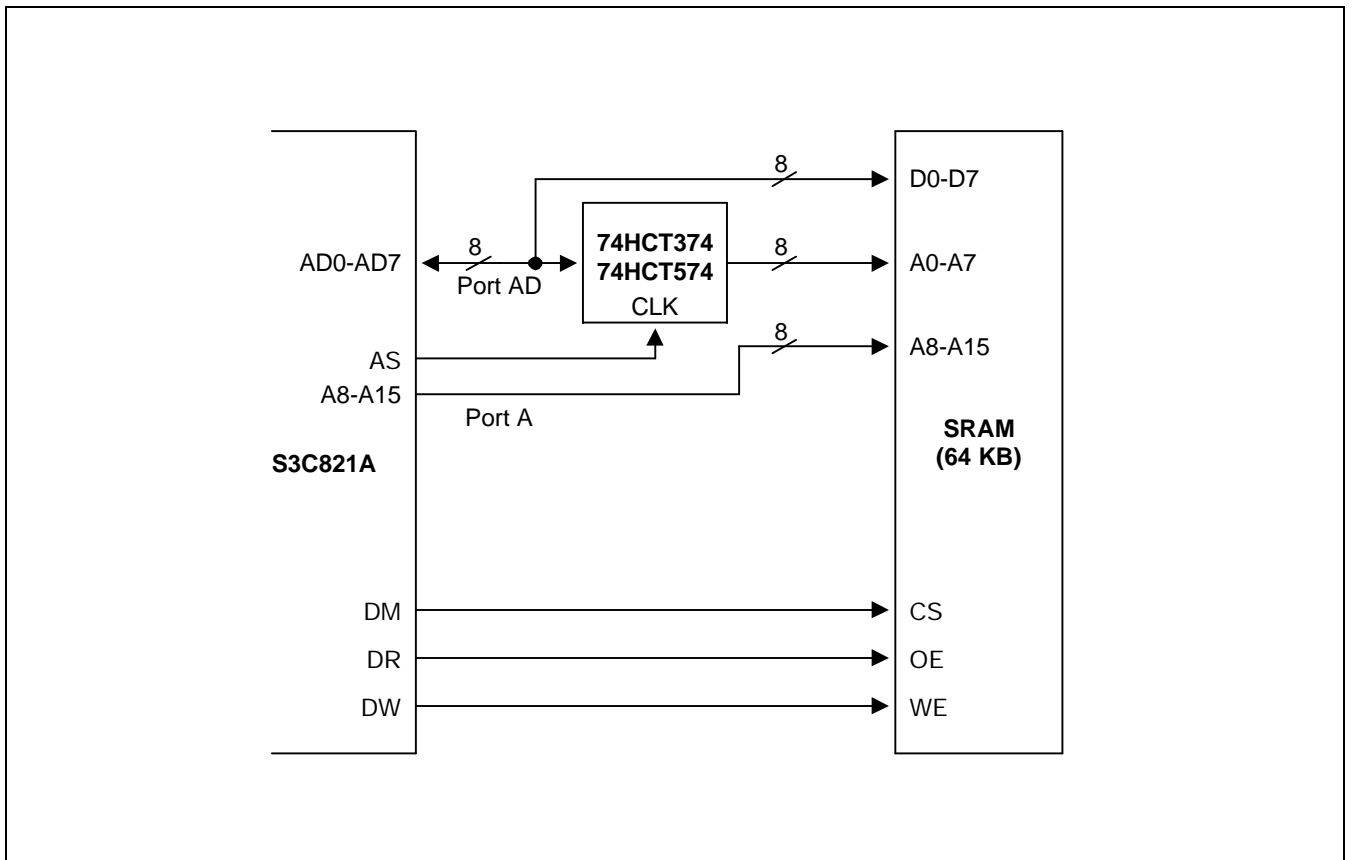


Figure 16-3. External Interface Function Diagram (S3C821A, SRAM, EPROM, EEPROM)

17 ELECTRICAL DATA

OVERVIEW

In this section, S3C821A electrical characteristics are presented in tables and graphs. The information is arranged in the following order:

- Absolute maximum ratings
- D.C. electrical characteristics
- Data retention supply voltage in Stop mode
- Stop mode release timing when initiated by an external interrupt
- Stop mode release timing when initiated by a Reset
- I/O capacitance
- A.C. electrical characteristics
- A/D converter electrical characteristics
- Input timing for external interrupts (P4, P2.4–P2.7)
- Input timing for RESET
- Serial data transfer timing
- Oscillation characteristics
- Oscillation stabilization time
- Operating voltage range

Table 17-1. Absolute Maximum Ratings

 $(T_A = 25\text{ }^\circ\text{C})$

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}	–	– 0.3 to + 6.5	V
Input voltage	V_{IN}	All I/O ports	– 0.3 to $V_{DD} + 0.3$	V
Output voltage	V_O	–	– 0.3 to $V_{DD} + 0.3$	V
Output current High	I_{OH}	One I/O port active	– 18	mA
		All I/O ports active	– 60	
Output current Low	I_{OL}	One I/O port active	+ 30 (peak value)	mA
			+ 15 (note)	
		Ports 0, 1, 2, and 3	+ 100 (peak value)	
			+ 60 (note)	
		Ports 4 and 5	+ 100 (peak value)	
			+ 60 (note)	
Operating temperature	T_A	–	– 40 to + 85	$^\circ\text{C}$
Storage temperature	T_{STG}	–	– 65 to + 150	$^\circ\text{C}$

NOTE: The values for Output Current Low (I_{OL}) are calculated as Peak Value $\times \sqrt{\text{Duty}}$.

Table 17-2. D.C. Electrical Characteristics

 $(T_A = -40\text{ }^\circ\text{C to } +85\text{ }^\circ\text{C, } V_{DD} = 2.0\text{ V to } 5.5\text{ V})$

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Operating Voltage	V_{DD}	$f_{OSC} = 8\text{ MHz}$ (Instruction clock = 1.33 MHz)	2.2	–	5.5	V
		$f_{OSC} = 6\text{ MHz}$ (Instruction clock = 1 MHz)	2.0			
Input High voltage	V_{IH1}	P0 and P1	$0.7 V_{DD}$	–	V_{DD}	V
	V_{IH2}	RESET, P2, P3, P4, and P5	$0.8 V_{DD}$		V_{DD}	
	V_{IH3}	X_{IN} , XT_{IN}	$V_{DD} - 0.1$		V_{DD}	
Input Low voltage	V_{IL1}	P0 and P1	0	–	$0.3 V_{DD}$	
	V_{IL2}	RESET, P2, P3, P4, and P5			$0.2 V_{DD}$	
	V_{IL3}	X_{IN} , XT_{IN}			0.1	
Output High voltage	V_{OH}	$V_{DD} = 3\text{ V}$; $I_{OH} = -200\text{ }\mu\text{A}$ All output pins	$V_{DD} - 1.0$	–	–	
Output Low voltage	V_{OL}	$V_{DD} = 3\text{ V}$; $I_{OL} = 1\text{ mA}$ All output pins	–	0.4	1.0	
Input High leakage current	I_{LIH1}	$V_{IN} = V_{DD}$ All input pins except those specified below for I_{LIH2}	–	–	1	μA
	I_{LIH2}	$V_{IN} = V_{DD}$ X_{IN} , X_{OUT} , XT_{IN} , and XT_{OUT}			20	
Input Low leakage current	I_{LIL1}	$V_{IN} = 0\text{ V}$ All input pins except those specified below for I_{LIL2} and RESET	–	–	–1	
	I_{LIL2}	$V_{IN} = 0\text{ V}$ X_{IN} , X_{OUT} , XT_{IN} , and XT_{OUT}			–20	
Output High leakage current	I_{LOH}	$V_{OUT} = V_{DD}$ All output pins	–	–	1	
Output Low leakage current	I_{LOL}	$V_{OUT} = 0\text{ V}$ All output pins	–	–	–1	
$ V_{DD-COMi} $ voltage drop ($i = 0-7$)	V_{DC}	$V_{DD} = 2.7\text{ V to } 5.5\text{ V}$ – $15\text{ }\mu\text{A}$ per common pin	–	–	120	mV
$ V_{DD-SEGx} $ voltage drop ($x = 0-31$)	V_{DS}	$V_{LCD} = 2.7\text{ V to } 5.5\text{ V}$ – $15\text{ }\mu\text{A}$ per segment pin	–	–	120	

Table 17-2. D.C. Electrical Characteristics (Continued)

(T_A = -40 °C to +85 °C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit		
V _{LC2} output voltage	V _{LC2}	V _{DD} = 2.7 V to 5.5 V LCD clock = 0 Hz	0.8 V _{DD} - 0.15	0.8 V _{DD}	0.8 V _{DD} + 0.15	V		
V _{LC3} output voltage	V _{LC3}	V _{LC1} = V _{DD}	0.6 V _{DD} - 0.15	0.6 V _{DD}	0.6 V _{DD} + 0.15	V		
V _{LC4} output voltage	V _{LC4}		0.4 V _{DD} - 0.15	0.4 V _{DD}	0.4 V _{DD} + 0.15			
V _{LC5} output voltage	V _{LC5}		0.2 V _{DD} - 0.15	0.2 V _{DD}	0.2 V _{DD} + 0.15			
Pull-up resistors	R _{L1}		V _{IN} = 0 V; T _A = 25 °C V _{DD} = 3.0 ± 10 %; Ports 0-5	30	80		200	kΩ
	R _{L2}	V _{IN} = 0 V; T _A = 25 °C V _{DD} = 3.0 ± 10 % RESET only	200	450	800			
LCD voltage dividing resistor	R _{LCD}	V _{LCD} = 2.7 V to 5.5 V T _A = 25 °C	45	65	80	kΩ		
Supply current (note)	I _{DD1}	Run mode; V _{DD} = 5.0 V ± 10 % Crystal oscillator C1 = C2 = 22 pF	6.0 MHz	-	6.0	12	mA	
			4.19 MHz		4.5	9.0		
		V _{DD} = 3.0 V ± 10 %	6.0 MHz		2.9	5.8		
			4.19 MHz		2.0	4.0		
	I _{DD2}	Idle mode; V _{DD} = 5.0 V ± 0 % Crystal oscillator C1 = C2 = 22 pF	6.0 MHz		1.3	2.6		
			4.19 MHz		1.2	2.4		
		V _{DD} = 3.0 V ± 10 %	6.0 MHz		0.6	1.2		
			4.19 MHz		0.4	0.8		
	I _{DD3}	Run mode; V _{DD} = 3.0 V ± 10 % 32 kHz crystal oscillator			20	40		μA
	I _{DD4}	Idle mode; V _{DD} = 3.0 V ± 10 % 32 kHz crystal oscillator			7	14		
I _{DD5}	Stop mode; V _{DD} = 5.0 V ± 10 %		0.5	3				
	Stop mode; V _{DD} = 3.0 V ± 10 %		0.3	2				

NOTES:

- Supply current does not include current drawn through internal pull-up resistors, LCD voltage dividing resistors, and ADC.
- I_{DD1} and I_{DD2} include power consumption for subsystem clock oscillation.
- I_{DD3} and I_{DD4} are current when main system clock oscillation stops and the subsystem clock is used.
- I_{DD5} is current when main system clock and subsystem clock oscillation stops.

Table 17-3. Data Retention Supply Voltage in Stop Mode

($T_A = -40\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V_{DDDR}	–	2.2	–	3.4	V
Data retention supply current	I_{DDDR}	$V_{DDDR} = 1.0\text{ V}$ Stop mode	–	–	1	μA
Oscillator stabilization wait time	t_{WAIT}	Released by RESET	–	$2^{16}/f_x$ (1)	–	ms
		Released by interrupt	–	(2)	–	

NOTES:

1. f_x is the main oscillator frequency.
2. The duration of the oscillation stabilization time (t_{WAIT}) when it is released by an interrupt is determined by the setting in the basic timer control register, BTCON.

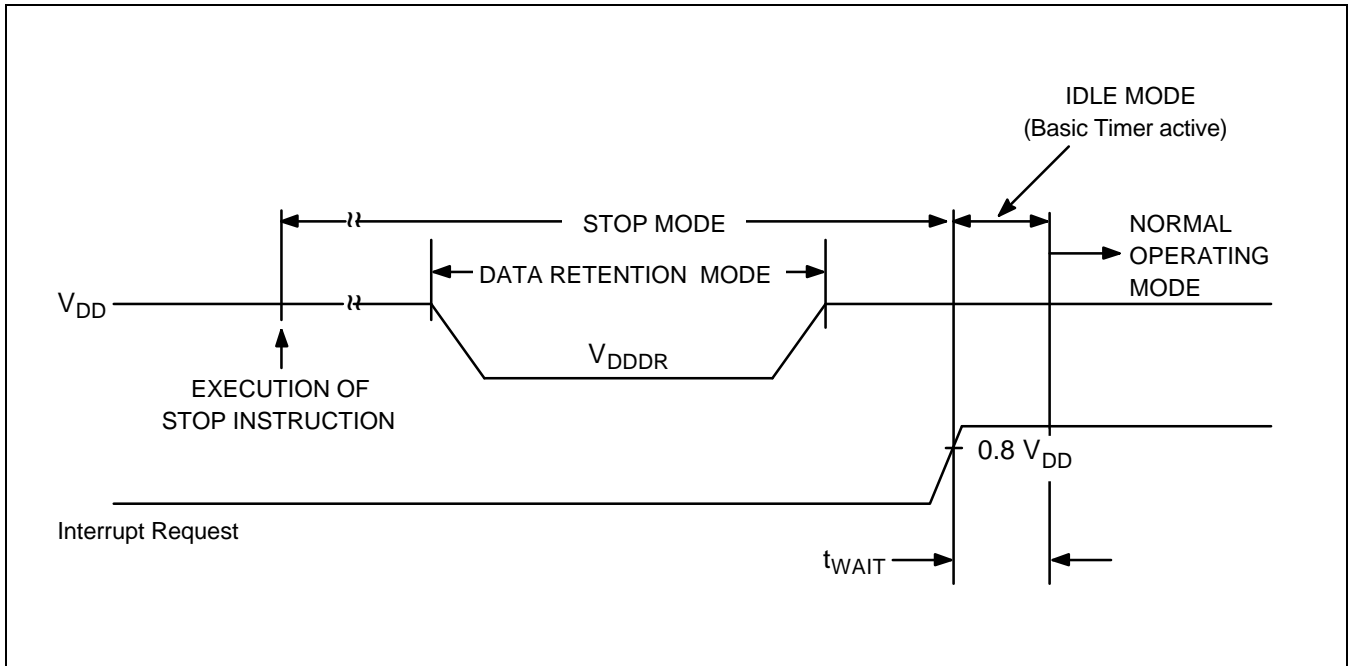


Figure 17-1. Stop Mode Release Timing When Initiated by an External Interrupt

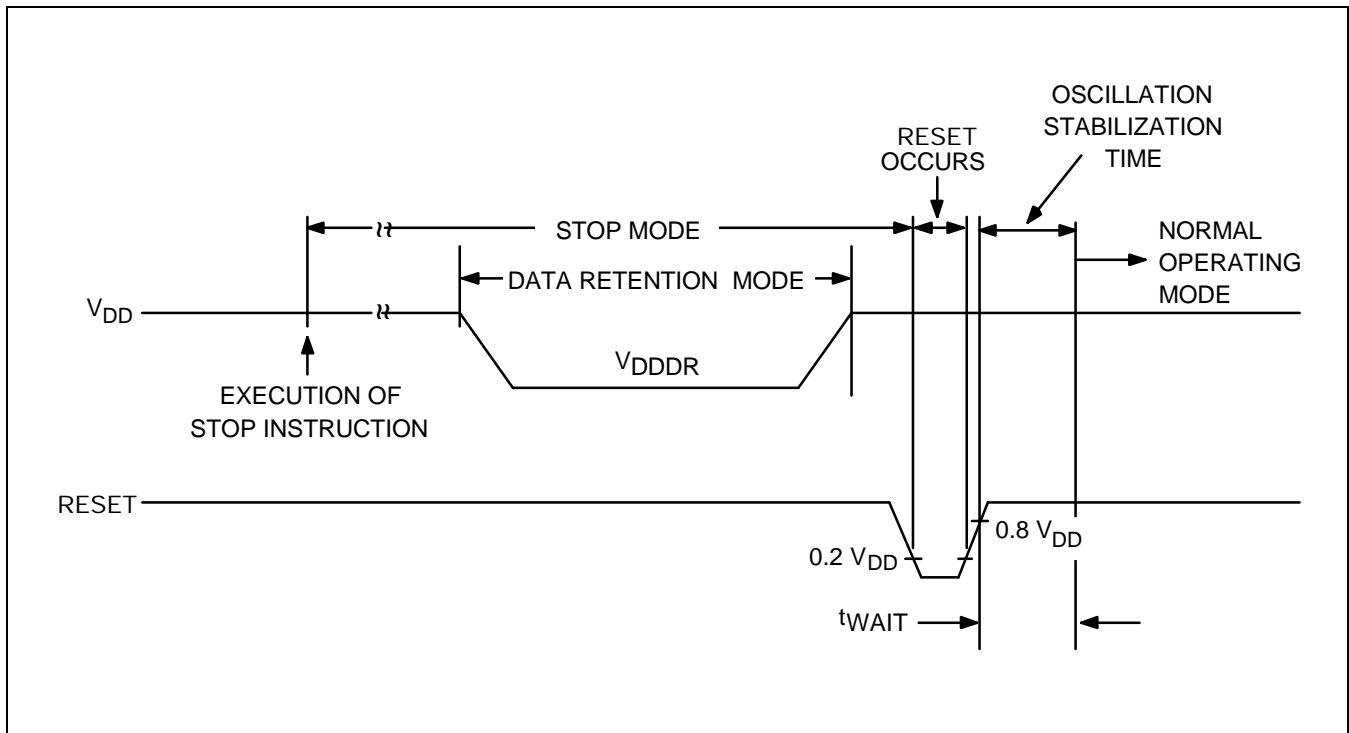


Figure 17-2. Stop Mode Release Timing When Initiated by a RESET

Table 17-4. Input/output Capacitance

(T_A = -25 °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 17-5. A.C. Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
SCK cycle time	t _{KCY}	External SCK source	1,000	-	-	ns
		Internal SCK source	1,000			
SCK high, low width	t _{KH} , t _{KL}	External SCK source	500	-	-	-
		Internal SCK source	t _{KCY} /2-50			
SI setup time to SCK high	t _{SIK}	External SCK source	250	-	-	-
		Internal SCK source	250			
SI hold time to SCK high	t _{KSI}	External SCK source	400	-	-	-
		Internal SCK source	400			
Output delay for SCK to SO	t _{KSO}	External SCK source	-	-	300	ns
		Internal SCK source			250	
Interrupt input, high, low width	t _{INTH} , t _{INTL}	All interrupt V _{DD} = 3 V	500	700	-	ns
RESET input low width	t _{RSL}	Input V _{DD} = 3 V	2,000	-	-	

Table 17-6. A/D Converter Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 2.7 V to 5.5 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Resolution			–	8	–	bit
Total accuracy		V _{DD} = 5.12 V AV _{REF} = 5.12 V AV _{SS} = 0 V	–	–	± 2	LSB
Conversion time ⁽¹⁾	t _{CON}	8 bit conversion 34 x n/f _{XX} ⁽²⁾ , n=1,4,8,16	17	–	170	μs
Analog input voltage	V _{IAN}	–	AV _{SS}	–	AV _{REF}	V
Analog input impedance	R _{AN}	–	2	1,000	–	MΩ
Analog reference voltage	AV _{REF}	–	2.5	–	V _{DD}	V
Analog ground	AV _{SS}	–	V _{SS}	–	V _{SS} + 0.3	V
Analog input current	I _{ADIN}	AV _{REF} = V _{DD} = 5V	–	–	10	μA

NOTES:

- "Conversion time" is the time required from the moment a conversion operation starts until it ends.
- f_{XX} is a selected system clock for peripheral hardware.

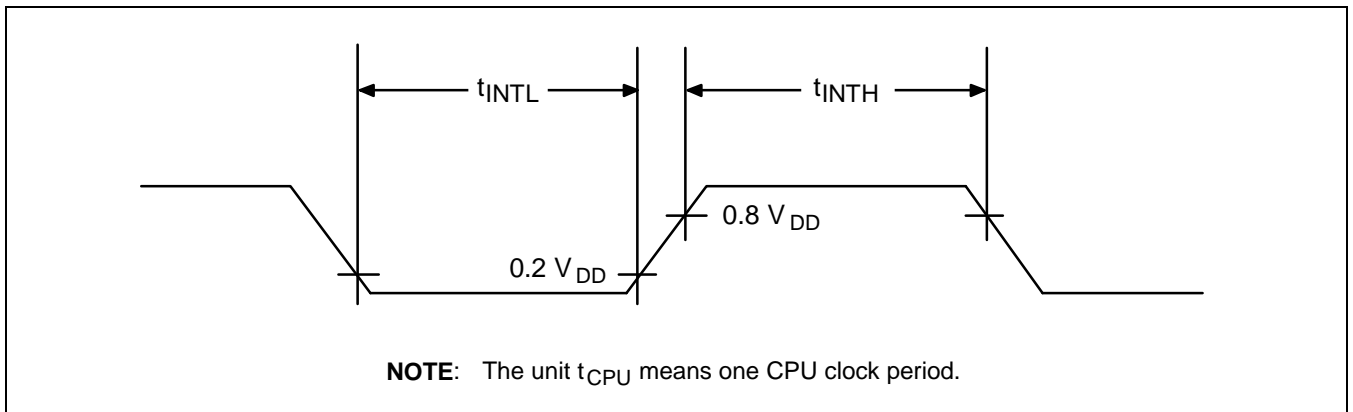


Figure 17-3. Input Timing for External Interrupts

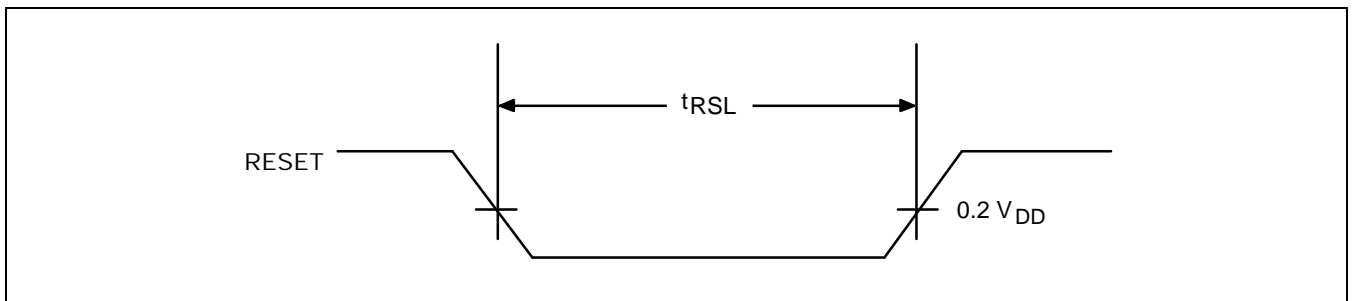


Figure 17-4. Input Timing for RESET

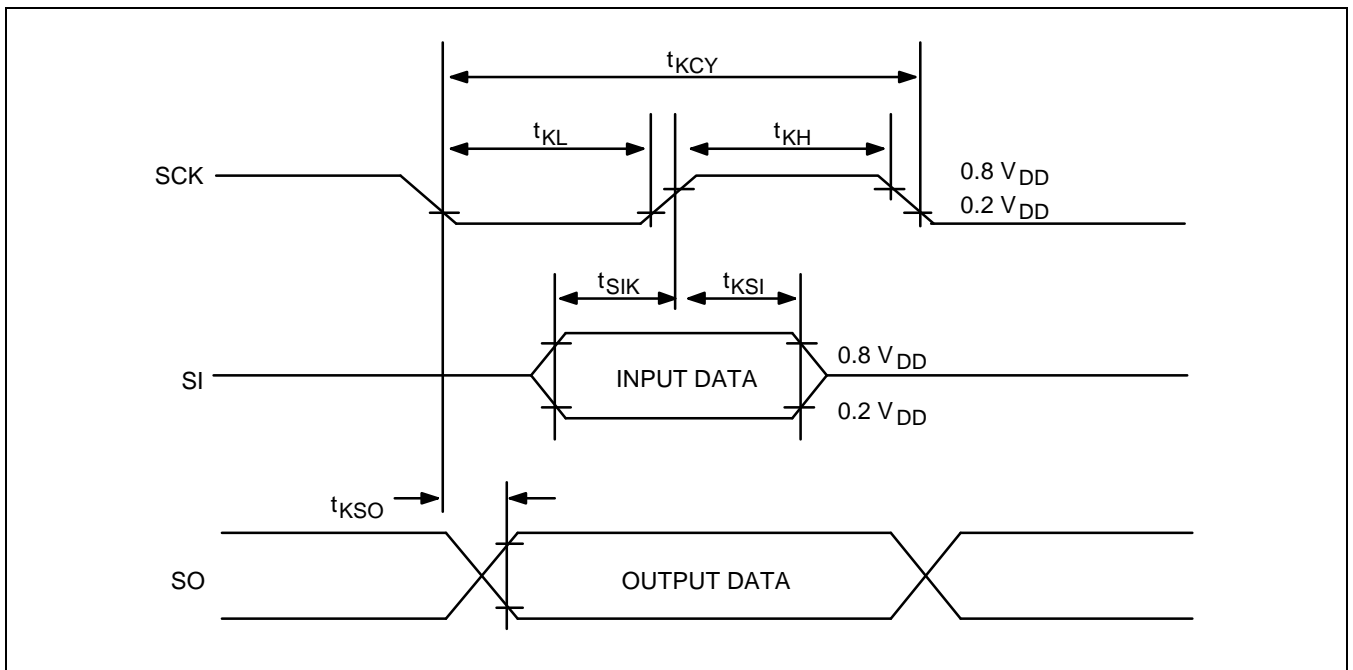


Figure 17-5. Serial Data Transfer Timing

Table 17-7. Main System Oscillation Characteristics

 $(T_A = -40\text{ }^\circ\text{C} + 85\text{ }^\circ\text{C})$

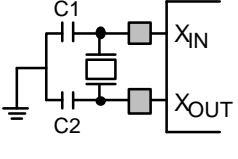
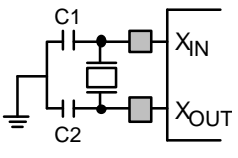
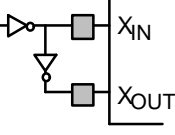
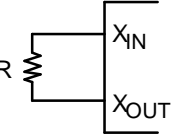
Oscillator	Clock Circuit	Parameter	Condition (V_{DD})	Min	Typ	Max	Unit
Crystal		Main oscillation frequency	2.2 V–5.5 V	0.4	–	8	MHz
			2.0 V–5.5 V	0.4	–	6	
Ceramic		Main oscillation frequency	2.2 V–5.5 V	0.4	–	8	
			2.0 V–5.5 V	0.4	–	6	
External clock		X_{1IN} input frequency	2.2 V–5.5 V	0.4	–	8	
			2.0 V–5.5 V	0.4	–	6	
RC		Frequency	3.0 V	0.4	–	2	

Table 17-8. Subsystem Oscillation Characteristics

 $(T_A = -40\text{ }^\circ\text{C} + 85\text{ }^\circ\text{C})$

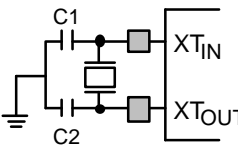
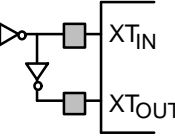
Oscillator	Clock Circuit	Parameter	Condition (V_{DD})	Min	Typ	Max	Unit
Crystal		Sub oscillation frequency	2.0 V–5.5 V	32	32.768	35	kHz
External clock		XT_{1IN} input frequency	2.0 V–5.5 V	32	–	500	kHz

Table 17-9. Main Oscillation Stabilization Time

($T_A = -40\text{ }^\circ\text{C} + 85\text{ }^\circ\text{C}$, $V_{DD} = 2.0\text{ V to } 5.5\text{ V}$)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	$f_x > 400\text{ kHz}$	–	–	20	ms
Ceramic	Oscillation stabilization occurs when V_{DD} is equal to the minimum oscillator voltage range.	–	–	10	ms
External clock	X_{IN} input High and Low width (t_{XH} , t_{XL})	25	–	500	ns

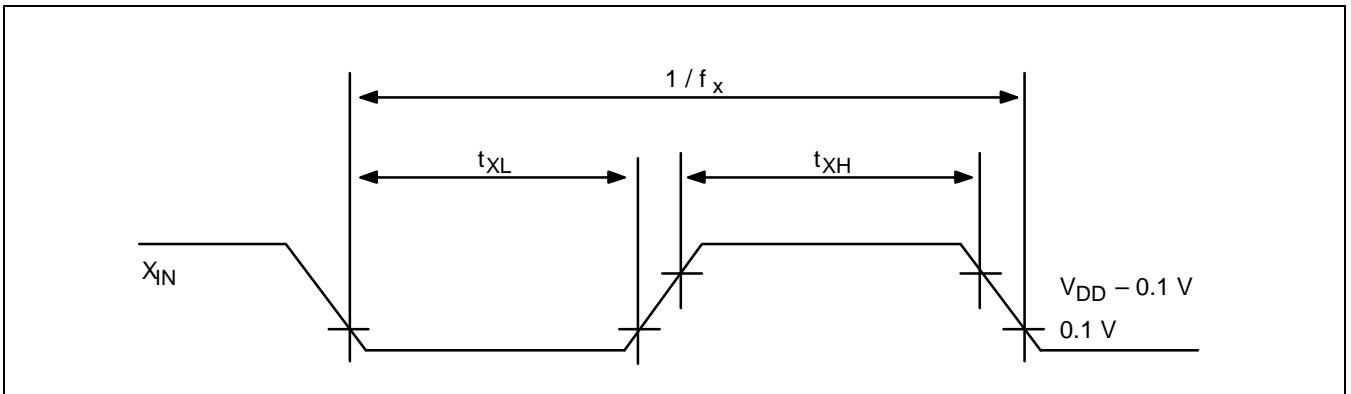


Figure 17-6. Clock Timing Measurement at X_{IN}

Table 17-10. Sub Oscillation Stabilization Time

($T_A = -40\text{ }^\circ\text{C} + 85\text{ }^\circ\text{C}$, $V_{DD} = 2.0\text{ V to } 5.5\text{ V}$)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	–	–	–	10	s
External clock	XT_{IN} input High and Low width (t_{XTH} , t_{XTL})	1	–	18	μs

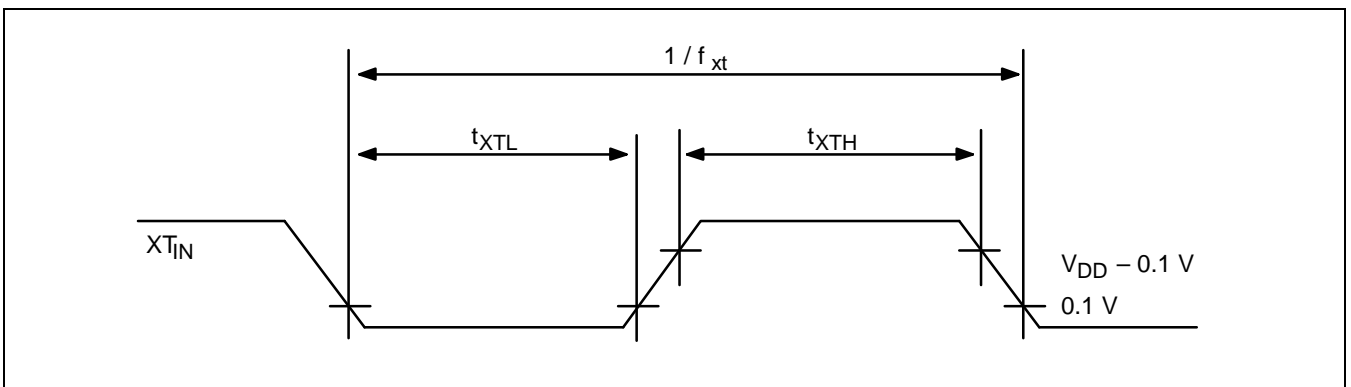


Figure 17-7. Clock Timing Measurement at XT_{IN}

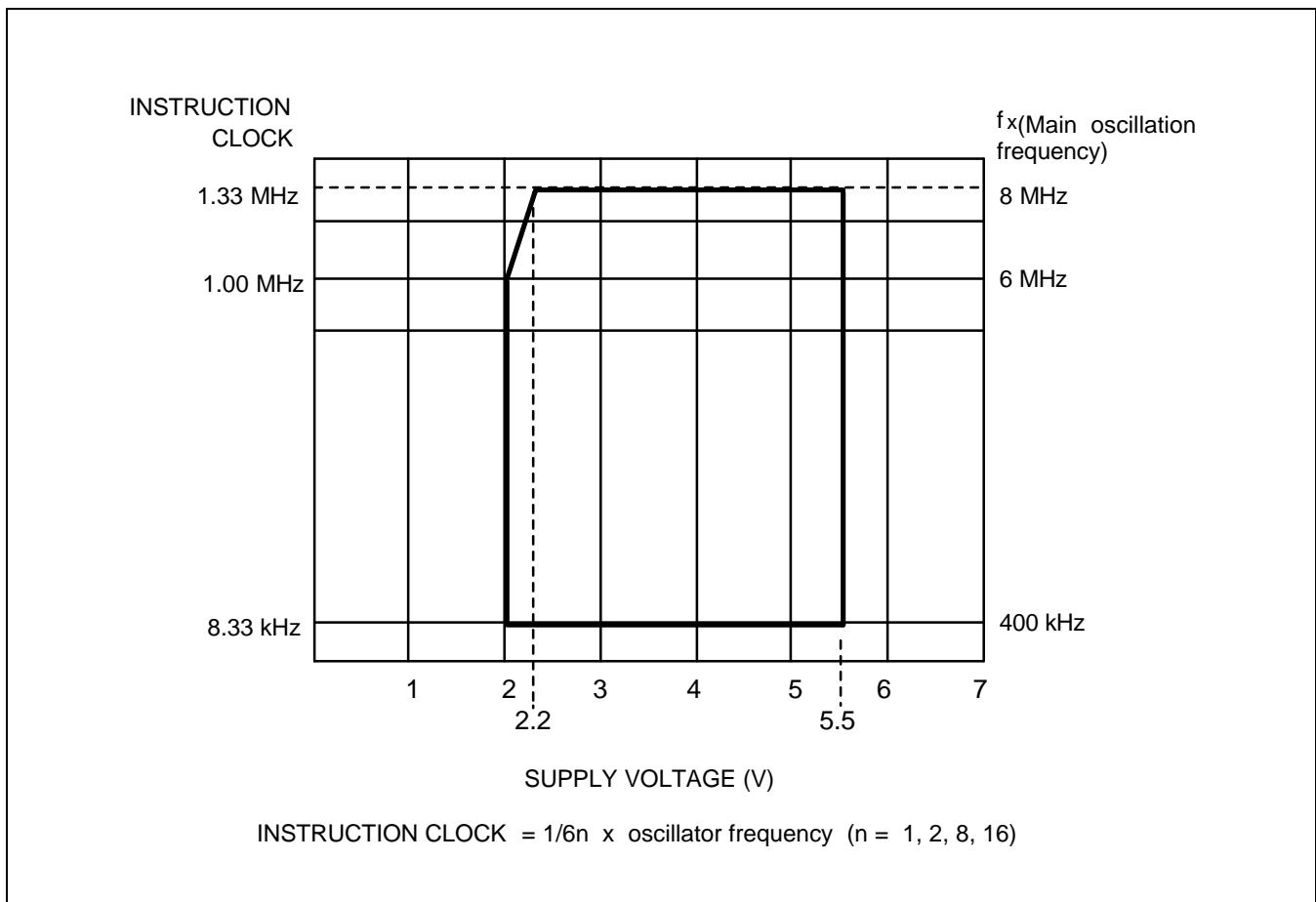


Figure 17-8. Operating Voltage Range

19

CUSTOMER NOTICE

DNJZ INSTRUCTION

When DJNZ instruction is used in a program, the working register being used as a counter should be set at the one of location 0C0H to 0CFH with SRP, SRP0, or SRP1 instruction.

PROGRAMMING TIP — Using DJNZ instruction to transport RAM data from page 0 to page 1

```

LD      PP,#10H           ; Destination ← 1, Source ← 0
SRP     #0C0H
LD      R0,#0FFH        ; Transportation starts
RAMTRN LD      R1,@R0
LD      @R0,R1
DJNZ   R0,RAMTRN
LD      R1,@R0           ; R0 = 00H
LD      @R0,R1

```

20

S3P821A OTP

OVERVIEW

The S3P821A single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the S3C821A microcontroller. It has an on-chip OTP ROM instead of a masked ROM. The EPROM is accessed by serial data format.

The S3P821A is fully compatible with the S3C821A, both in function and in pin configuration. Because of its simple programming requirements, the S3P821A is ideal as an evaluation chip for the S3C821A.

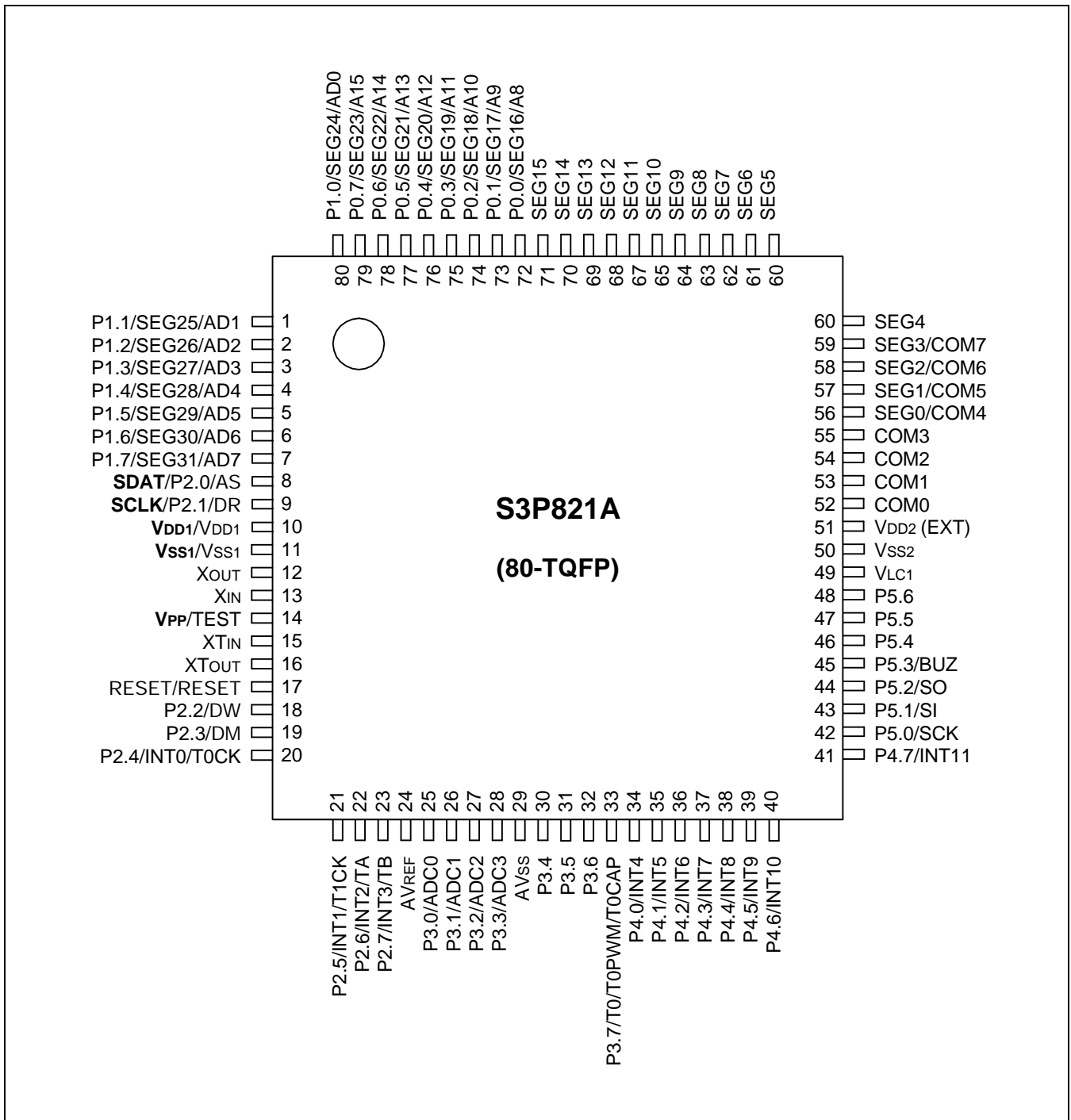


Figure 20-1. S3P821A Pin Assignments (80-TQFP-1212 Package)

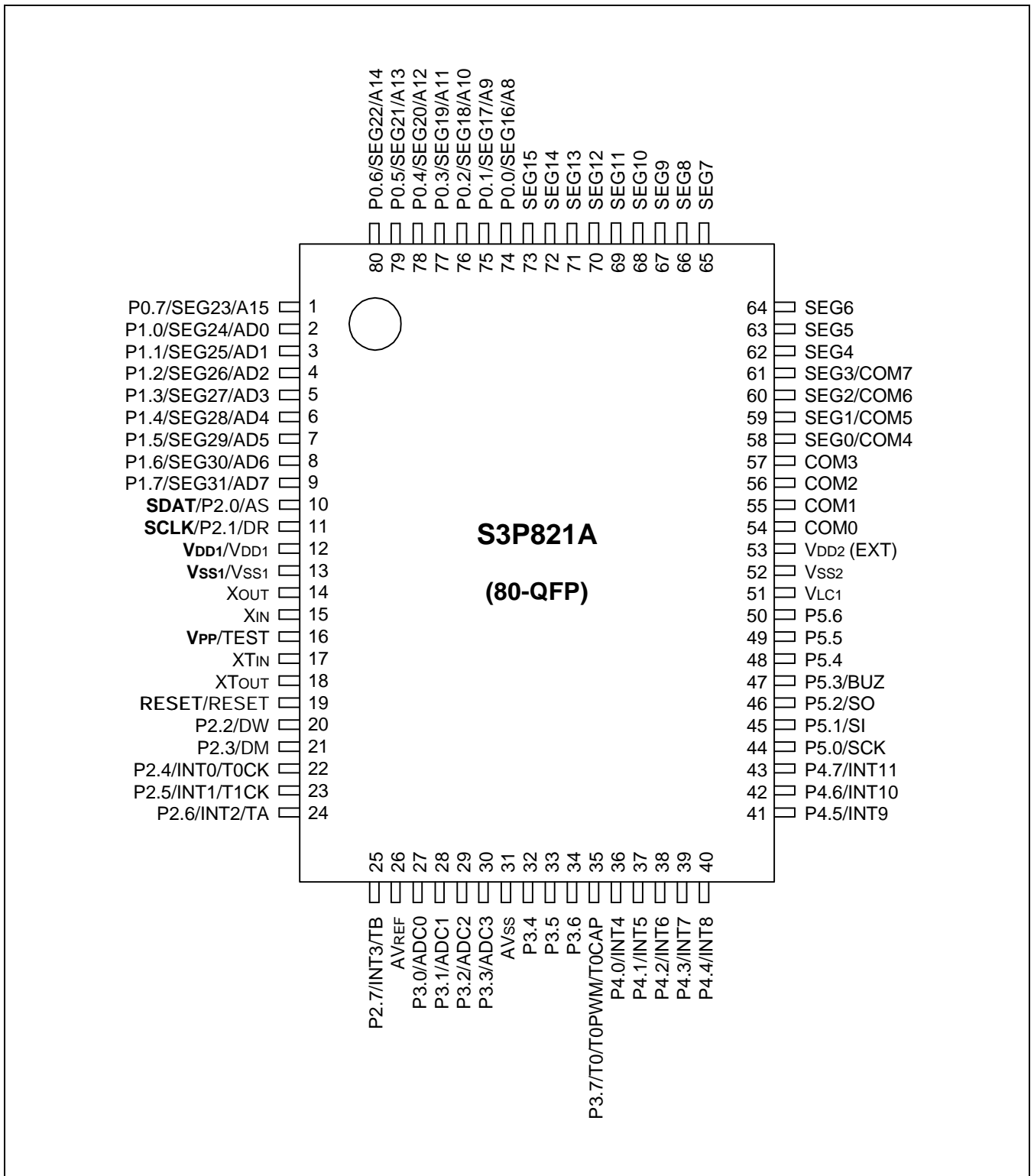


Figure 20-2. S3P821A Pin Assignments (80-QFP-1420C Package)

Table 20-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip Pin Name	During Programming			
	Pin Name	Pin No.	I/O	Function
P2.0	SDAT	8 (10)	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input/push-pull output port.
P2.1	SCLK	9 (11)	I/O	Serial clock pin. Input only pin.
V _{PP}	TEST	14 (16)	I	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.5 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode. (Option)
RESET	RESET	17 (19)	I	Chip Initialization
V _{DD1} /V _{SS1}	V _{DD1} /V _{SS1}	10 (12)/11 (13)	–	Logic power supply pin. V _{DD} should be tied to + 5 V during programming.

NOTE: () means 80 QFP package.

Table 20-2. Comparison of S3P821A and S3C821A Features

Characteristic	S3P821A	S3C821A
Program Memory	48-K byte EPROM	48-K byte mask ROM
Operating Voltage (V _{DD})	2.0 V to 5.5 V	2.0 V to 5.5 V
OTP Programming Mode	V _{DD} = 5 V, V _{PP} (TEST) = 12.5 V	
Pin Configuration	80 QFP/80 TQFP	80 QFP/80 TQFP
EPROM Programmability	User Program 1 time	Programmed at the factory

OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V_{PP} (TEST) pin of the S3P821A, 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 20-3 below.

Table 20-3. Operating Mode Selection Criteria

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

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

Table 20-4. D.C. Electrical Characteristics

(T_A = -40 °C to +85 °C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Operating Voltage	V _{DD}	f _{OSC} = 8 MHz (Instruction clock = 1.33 MHz)	2.2	–	5.5	V
		f _{OSC} = 6 MHz (Instruction clock = 1 MHz)	2.0			
Input High voltage	V _{IH1}	P0 and P1	0.7 V _{DD}	–	V _{DD}	V
	V _{IH2}	RESET, P2, P3, P4, and P5	0.8 V _{DD}		V _{DD}	
	V _{IH3}	X _{IN} , XT _{IN}	V _{DD} - 0.1		V _{DD}	
Input Low voltage	V _{IL1}	P0 and P1	0	–	0.3 V _{DD}	
	V _{IL2}	RESET, P2, P3, P4, and P5			0.2 V _{DD}	
	V _{IL3}	X _{IN} , XT _{IN}			0.1	
Output High voltage	V _{OH}	V _{DD} = 3 V; I _{OH} = -200 μA All output pins	V _{DD} - 1.0	–	–	
Output Low voltage	V _{OL}	V _{DD} = 3 V; I _{OL} = 1 mA All output pins	–	0.4	1.0	
Input High leakage current	I _{LIH1}	V _{IN} = V _{DD} All input pins except those specified below for I _{LIH2}	–	–	1	μA
	I _{LIH2}	V _{IN} = V _{DD} X _{IN} , X _{OUT} , XT _{IN} , and XT _{OUT}			20	
Input Low leakage current	I _{LIL1}	V _{IN} = 0 V All input pins except those specified below for I _{LIL2} and RESET	–	–	-1	
	I _{LIL2}	V _{IN} = 0 V X _{IN} , X _{OUT} , XT _{IN} , and XT _{OUT}			-20	
Output High leakage current	I _{LOH}	V _{OUT} = V _{DD} All output pins	–	–	1	
Output Low leakage current	I _{LOL}	V _{OUT} = 0 V All output pins	–	–	-1	
V _{DD-COMi} voltage drop (i = 0-7)	V _{DC}	V _{DD} = 2.7 V to 5.5 V - 15 μA per common pin	–	–	120	mV
V _{DD-SEGx} voltage drop (x = 0-31)	V _{DS}	V _{LCD} = 2.7 V to 5.5 V - 15 μA per segment pin	–	–	120	

Table 20-4. D.C. Electrical Characteristics (Continued)

(T_A = -40 °C to +85 °C, V_{DD} = 2.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit		
V _{LC2} output voltage	V _{LC2}	V _{DD} = 2.7 V to 5.5 V LCD clock = 0 Hz V _{LC1} = V _{DD}	0.8 V _{DD} - 0.15	0.8 V _{DD}	0.8 V _{DD} + 0.15	V		
V _{LC3} output voltage	V _{LC3}		0.6 V _{DD} - 0.15	0.6 V _{DD}	0.6 V _{DD} + 0.15			
V _{LC4} output voltage	V _{LC4}		0.4 V _{DD} - 0.15	0.4 V _{DD}	0.4 V _{DD} + 0.15			
V _{LC5} output voltage	V _{LC5}		0.2 V _{DD} - 0.15	0.2 V _{DD}	0.2 V _{DD} + 0.15			
Pull-up resistors	R _{L1}	V _{IN} = 0 V; T _A = 25 °C V _{DD} = 3.0 ± 10%; Ports 0-5	30	80	200	kΩ		
	R _{L2}	V _{IN} = 0 V; T _A = 25 °C V _{DD} = 3.0 ± 10 % RESET only	300	500	800			
LCD voltage dividing resistor	R _{LCD}	V _{LCD} = 2.7 V to 5.5 V T _A = 25 °C	45	65	80	kΩ		
Supply current (note)	I _{DD1}	Run mode; V _{DD} =5.0V±10% Crystal oscillator C1 = C2 = 22 pF	6.0 MHz	-	6.0	12	mA	
			4.19 MHz		4.5	9.0		
		V _{DD} = 3.0 V ± 10 %	6.0 MHz		2.9	5.8		
			4.19 MHz		2.0	4.0		
	I _{DD2}	Idle mode; V _{DD} =5.0 V± 0% Crystal oscillator C1 = C2 = 22 pF	6.0 MHz		1.3	2.6		
			4.19 MHz		1.2	2.4		
		V _{DD} = 3.0 V ± 10 %	6.0 MHz		0.6	1.2		
			4.19 MHz		0.4	0.8		
	I _{DD3}	Run mode; V _{DD} = 3.0 V ± 10 % 32 kHz crystal oscillator			20	40		μA
	I _{DD4}	Idle mode; V _{DD} = 3.0 V ± 10 % 32 kHz crystal oscillator			7	14		
I _{DD5}	Stop mode; V _{DD} = 5.0 V ± 10 %		0.5	3				
	Stop mode; V _{DD} = 3.0 V ± 10 %		0.3	2				

NOTES:

- Supply current does not include current drawn through internal pull-up resistors, LCD voltage dividing resistors, and ADC.
- I_{DD1} and I_{DD2} include power consumption for subsystem clock oscillation.
- I_{DD3} and I_{DD4} are current when main system clock oscillation stops and the subsystem clock is used.
- I_{DD5} is current when main system clock and subsystem clock oscillation stops.

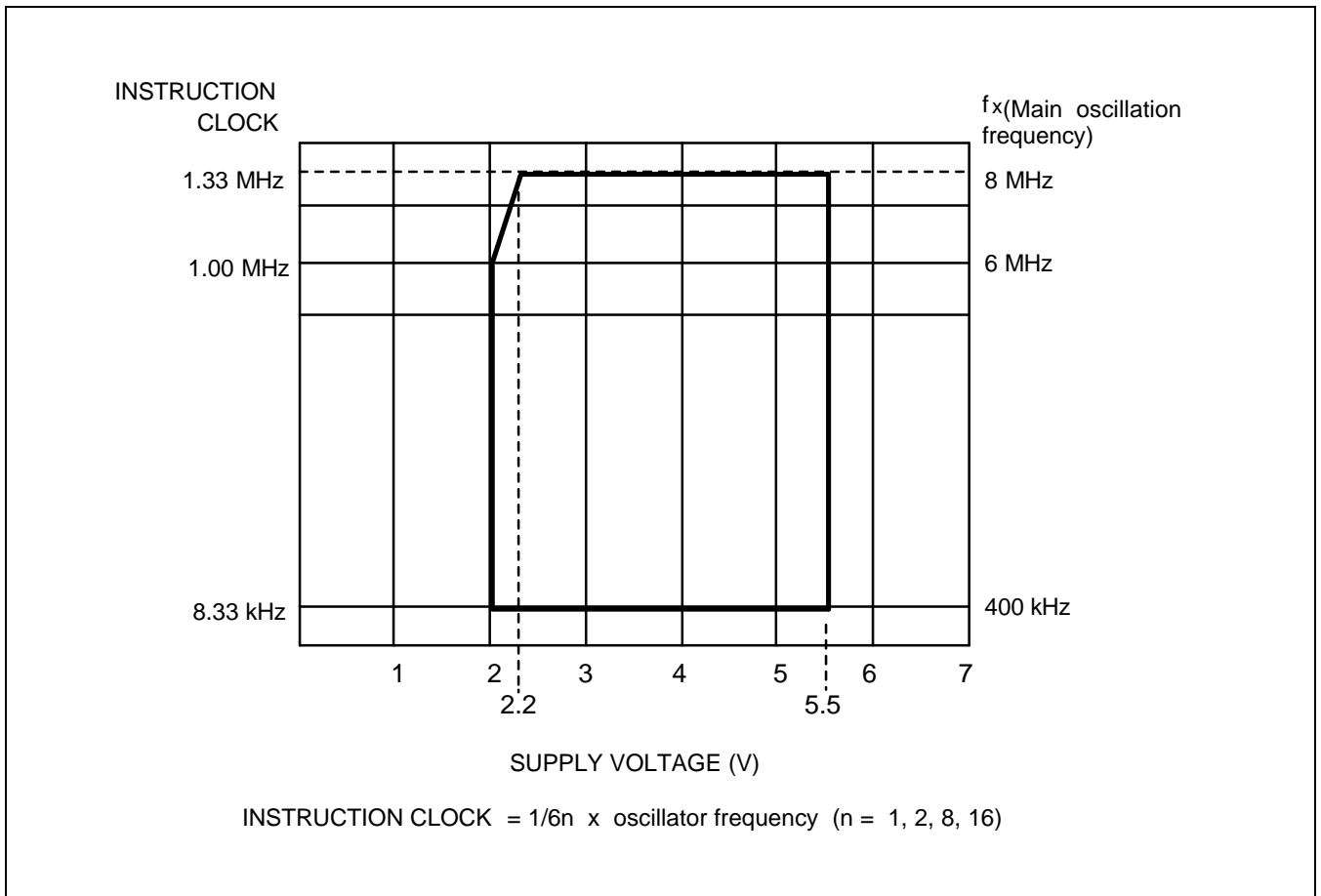


Figure 20-3. Operating Voltage Range

21

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 S3C7, S3C6, S3C8 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.

SASM88

The SASM88 is a relocatable assembler for Samsung's S3C8-series microcontrollers. The SASM88 takes a source file containing assembly language statements and translates into a corresponding source code, object code and comments. The SASM88 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 up to the maximum ROM size of the target device automatically.

TARGET BOARDS

Target boards are available for all S3C8-series microcontrollers. All required target system cables and adapters are included with the device-specific target board.



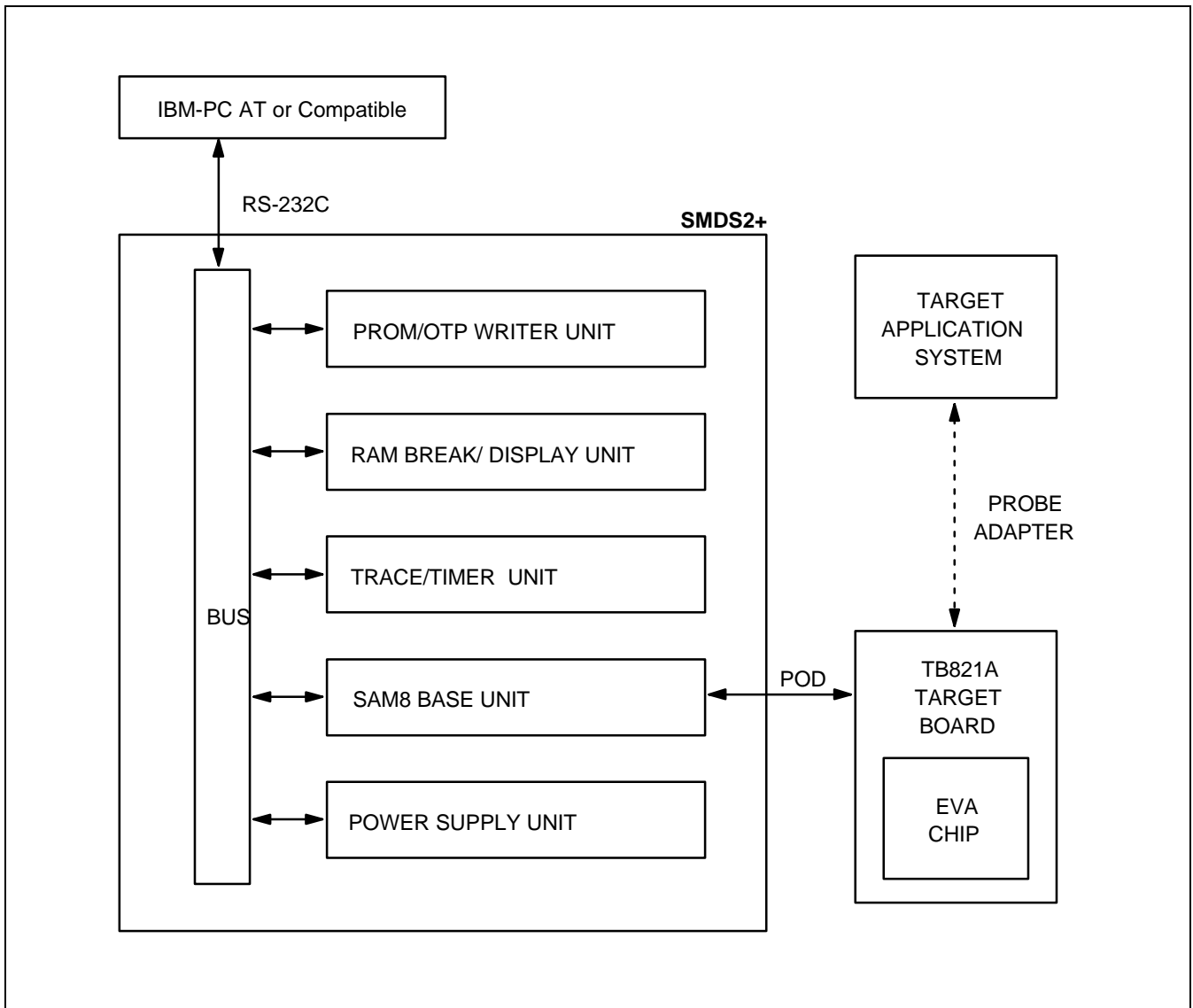


Figure 21-1. SMDS Product Configuration (SMDS2+)

TB821A TARGET BOARD

The TB821A target board is used for the S3C821A microcontroller. It is supported with the SMDS2+.

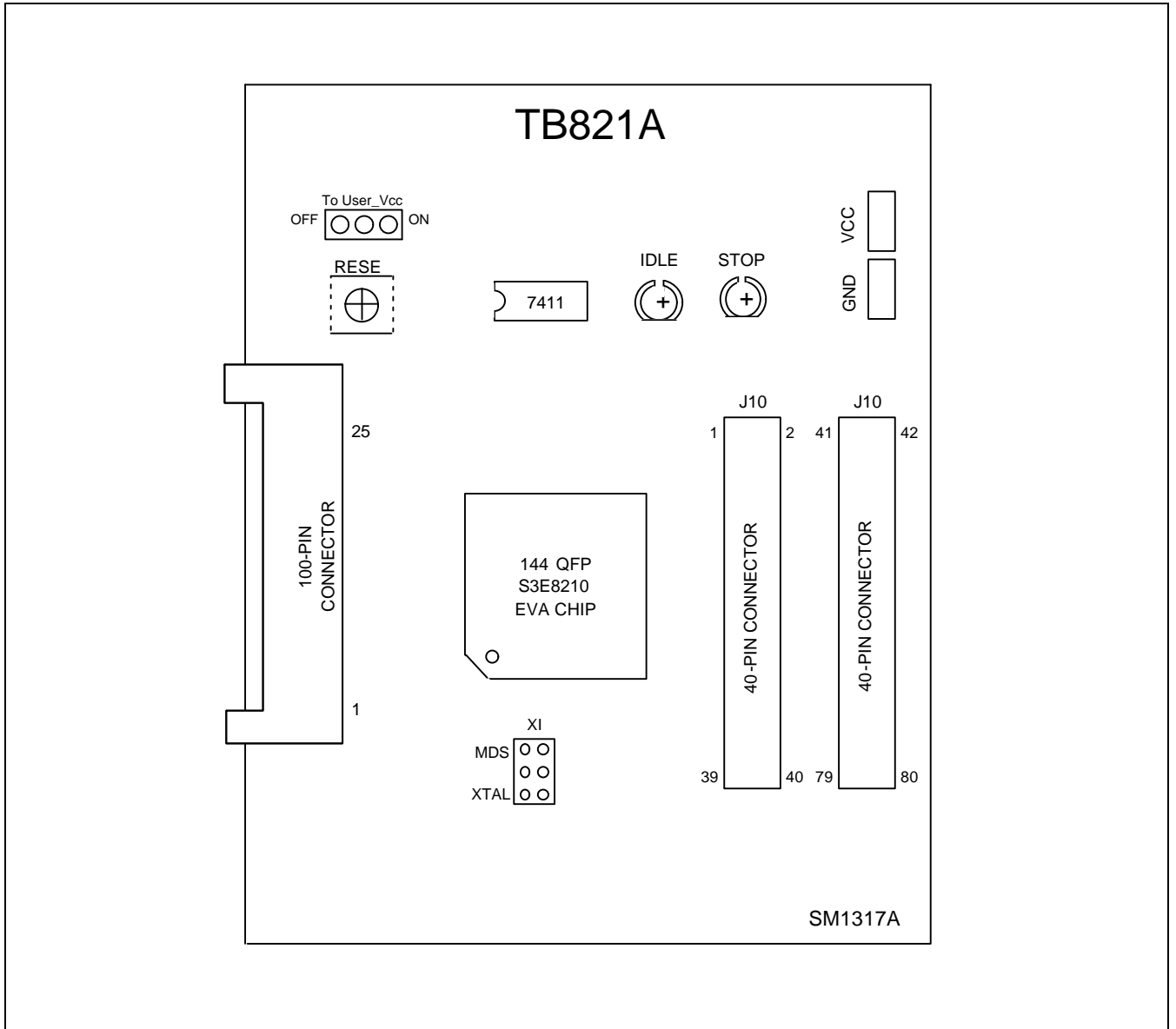

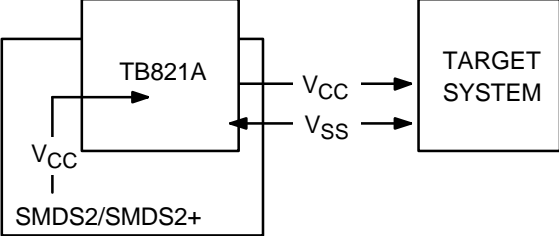

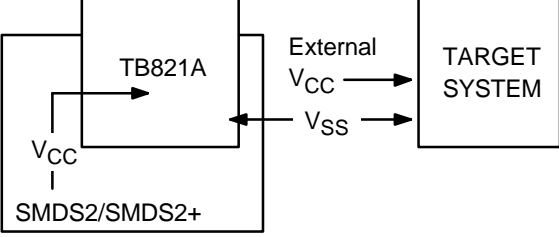


Figure 21-2. TB821A Target Board Configuration

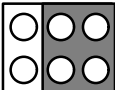
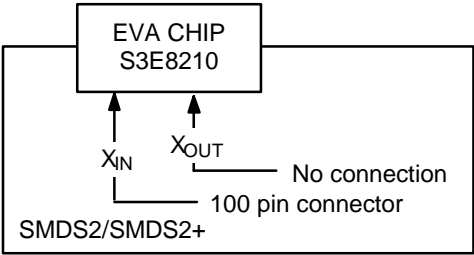
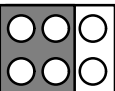
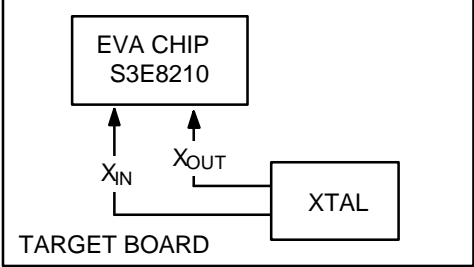
Table 21-1. Power Selection Settings for TB821A

"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_Vcc" Setting column indicates the electrical short (off) configuration:




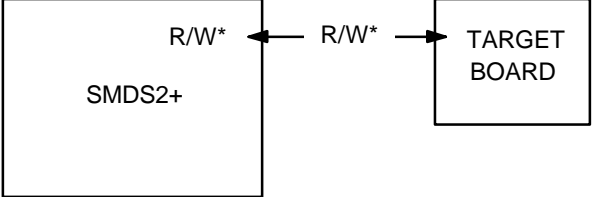
Table 21-2. Main-clock Selection Settings for TB821A

Sub Clock Settings	Operating Mode	Comments
XTAL  MDS		Set the XI switch to "MDS" when the target board is connected to the SMDS2/SMDS2+.
XTAL  MDS		Set the XI switch to "XTAL" when the target board is used as a standalone unit, and is not connected to the SMDS2/SMDS2+.

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 21-3. The SMDS2+ Tool Selection Setting

"SW1" Setting	Operating Mode
SMDS  SMDS2+	

IDLE LED

The Yellow LED is ON when the evaluation chip (S3E8210) is in idle mode.

STOP LED

The Red LED is ON when the evaluation chip (S3E8210) is in stop mode.

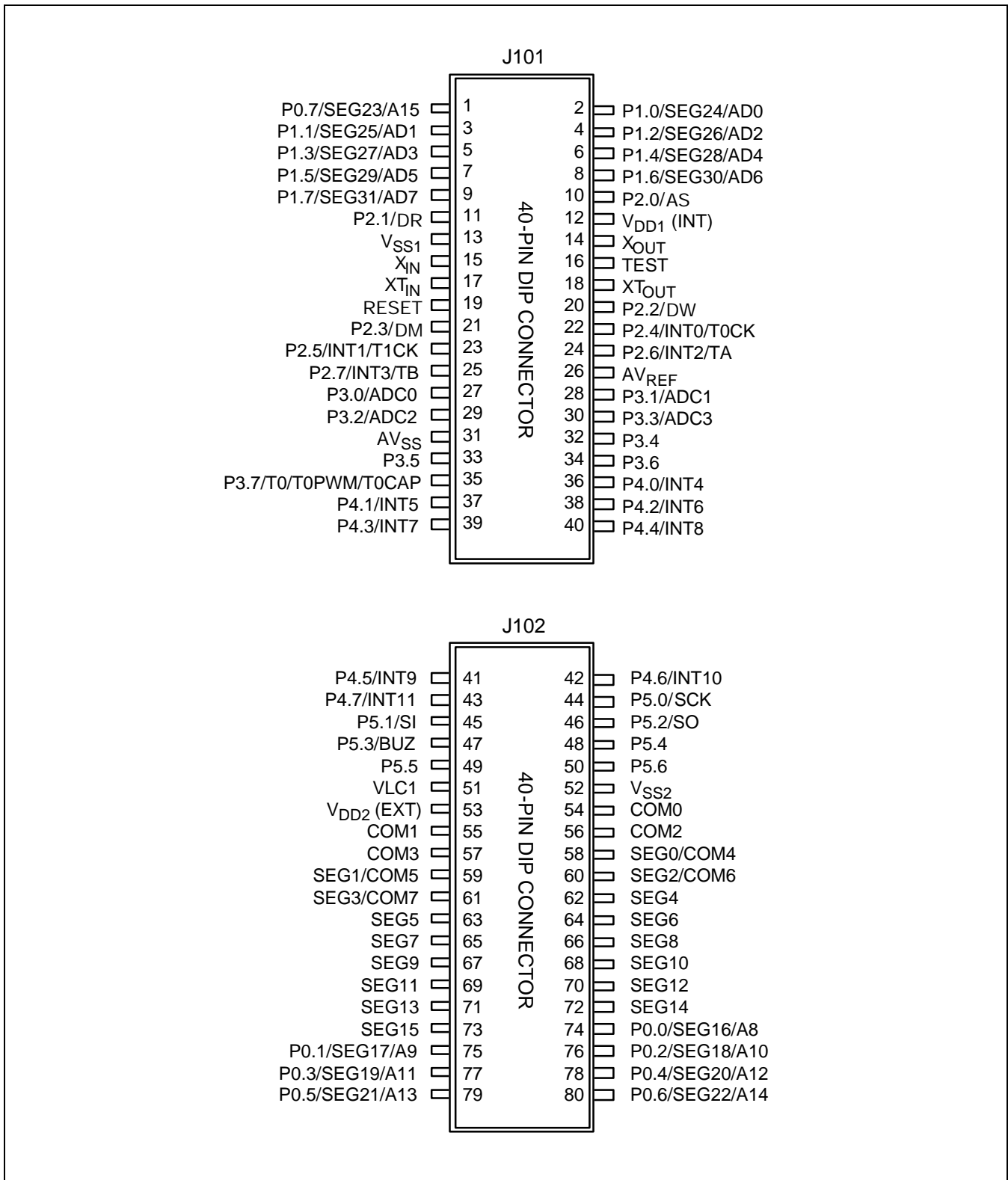


Figure 21-3. 40-Pin Connectors (J101, J102) for TB821A

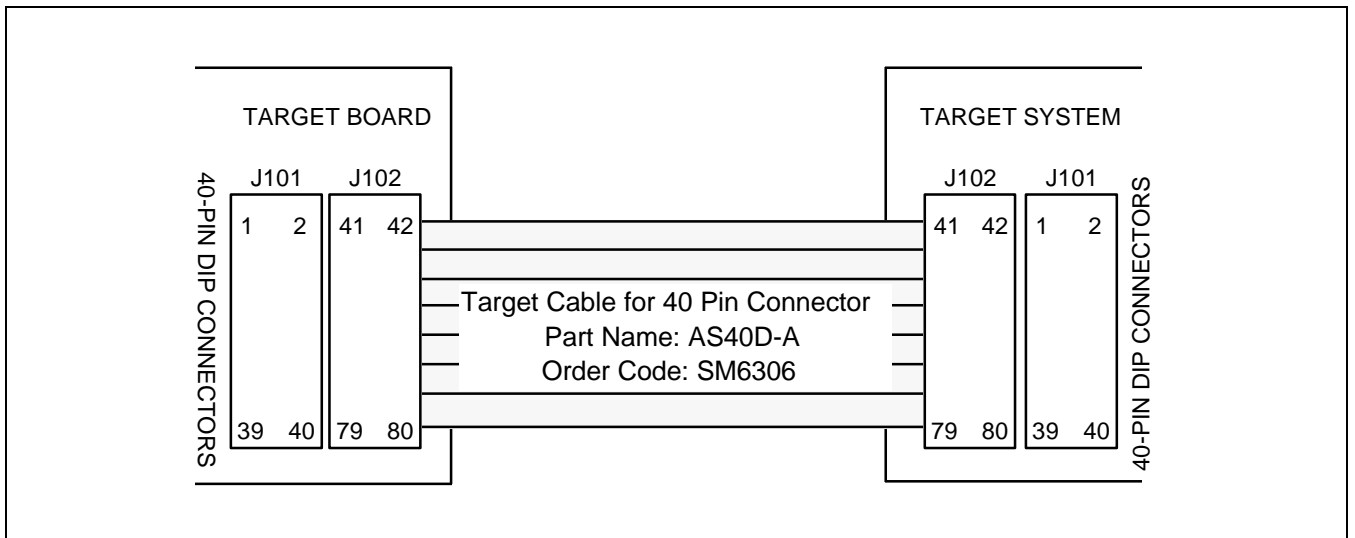


Figure 21-4. S3C821A Cables for 80-QFP Package