

21-S3-C830A/P830A-032002

# USER'S MANUAL

**S3C830A/P830A**  
**8-Bit CMOS**  
**Microcontroller**  
**Revision 1**



# 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 four CPU clocks) can be assigned to specific interrupt levels.

## S3C830A MICROCONTROLLER

The S3C830A single-chip microcontroller are fabricated using the highly advanced CMOS process. Its design is based on the powerful SAM88RC CPU core. Stop and idle (power-down) modes were implemented to reduce power consumption.

The S3C830A is a microcontroller with a 48K-byte mask-programmable ROM embedded.  
The S3P830A is a microcontroller with a 48K-byte one-time-programmable ROM embedded.

Using the SAM88RC modular design approach, the following peripherals were integrated with the SAM88RC CPU core:

- Large number of programable I/O ports (Total 72 pins)
- PLL frequency synthesizer
- 16-bits intermediate frequency counter
- Two synchronous SIO modules
- Two 8-bit timer/counters
- One 16-bit timer/counter
- Low voltage reset
- A/D converter with 4 selectable input pins

## OTP

The S3C830A microcontroller is also available in OTP (One Time Programmable) version, S3P830A. The S3P830A microcontroller has an on-chip 48K-byte one-time-programmable EPROM instead of masked ROM. The S3P830A is comparable to S3C830A, both in function and in pin configuration.

## FEATURES

### CPU

- SAM88RC CPU core

### Memory

- 2064-byte internal register file (including LCD display RAM)
- 48K-byte internal program memory area

### Instruction Set

- 78 instructions
- Idle and Stop instructions

### 72 I/O Pins

- 32 normal I/O pins
- 40 pins sharing with LCD segment signals

### Interrupts

- 8 interrupt levels and 17 internal sources
- Fast interrupt processing feature

### 8-Bit Basic Timer

- Watchdog timer function
- 4 kinds of clock source

### Timer/Counter 0

- Programmable 8-bit internal timer
- External event counter function
- PWM and capture function

### Timer/Counter 1

- Programmable 8-bit interval timer
- External event counter function

### Timer/Counter 2

- Programmable 16-bit interval timer
- External event counter function

### Watch Timer

- Interval Time: 50ms, 0.5s, 1.0s at 4.5 MHz
- 1/1.5/3/6 kHz buzzer output selectable

### Analog to Digital Converter

- 4-channel analog input
- 8-bit conversion resolution

### Two 8-bit Serial I/O Interface

- 8-bit transmit/receive mode
- 8-bit receive mode
- Selectable baud rate or external clock source

### PLL Frequency Synthesizer

- $V_{IN}$  level: 300mVpp (minimum)
- AMVCO range: 0.5 MHz–30 MHz
- FMVCO range: 30 MHz–150 MHz

### 16-Bit Intermediate Frequency (IF) Counter

- $V_{IN}$  level: 300mVpp (minimum)
- AMIF range: 100 kHz–1 MHz
- FMIF range: 5 MHz–15 MHz

### LCD Controller/Driver

- 40 segments and 4 common terminals
- 4/3/2 common and static selectable
- Internal or external resistor circuit for LCD bias

### Low Voltage Reset (LVR)

- Low voltage check to make system reset
- $V_{LVR}$ : 3.5 V (typical)

### Two Power-Down Modes

- Idle mode: only CPU clock stops
- Stop mode: system clock and CPU clock stop

### Oscillation Source

- Crystal or ceramic for system clock (fx)

### Instruction Execution Time

- 890 ns at 4.5 MHz (minimum)

### Operating Temperature Range

- $-25^{\circ}\text{C}$  to  $+85^{\circ}\text{C}$

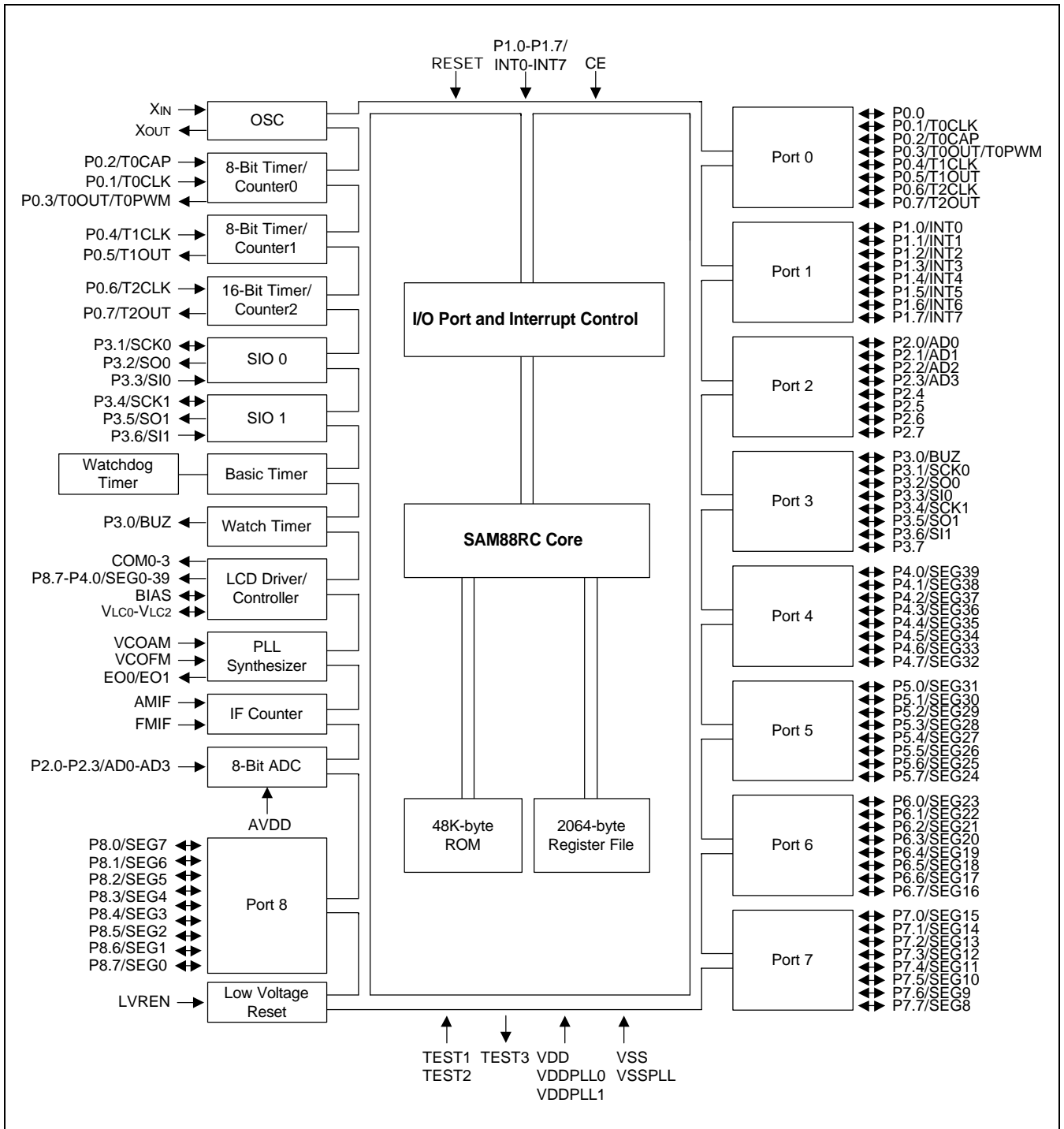
### Operating Voltage Range

- 3.0 V to 5.5 V at 0.4 MHz–4.5 MHz
- 4.5 V to 5.5 V in PLL/IFC block

### Package Type

- 100-pin QFP package

**BLOCK DIAGRAM**



**Figure 1-1. Block Diagram**

PIN ASSIGNMENT

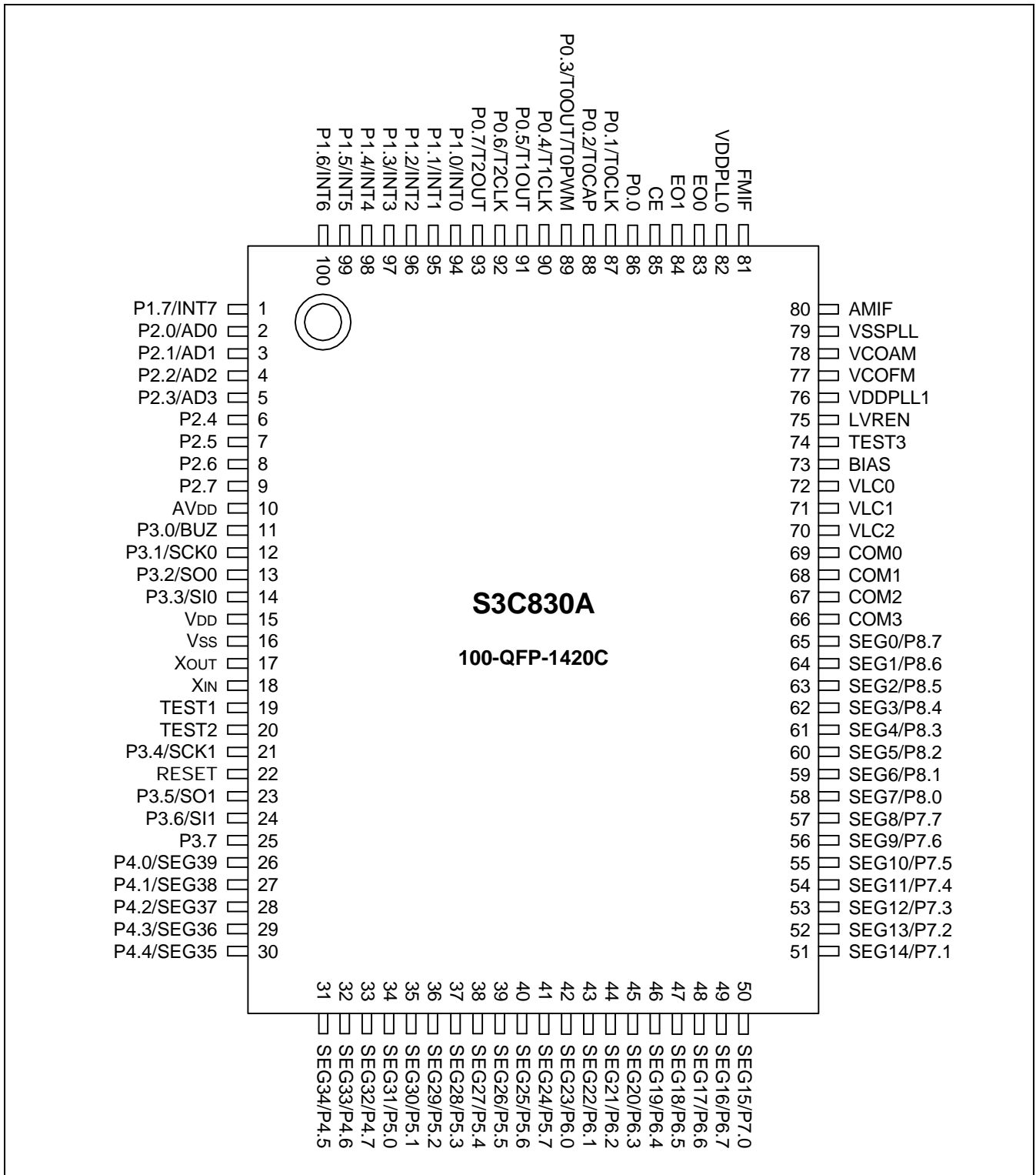


Figure 1-2. S3C830A Pin Assignments (100-QFP)

## PIN DESCRIPTIONS

Table 1-1. S3C830A Pin Descriptions

Pin Names	Pin Type	Pin Description	Circuit Type	Pin No.	Share Pins
P0.0 P0.1 P0.2 P0.3 P0.4 P0.5 P0.6 P0.7	I/O	I/O port with bit programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	E-4	86 87 88 89 90 91 92 93	– T0CLK T0CAP T0OUT/T0PWM T1CLK TOUT T2CLK T2OUT
P1.0-P1.7	I/O	I/O port with bit programmable pins; Schmitt trigger Input or push-pull output and software assignable pull-ups; Alternately used for external interrupt input (noise filters, interrupt enable and pending control).	D-7	94-1	INT0-INT7
P2.0-P2.3 P2.4-P2.7	I/O	I/O port with bit programmable pins; Schmitt trigger input or push-pull output and software assignable pull-ups.	F-16 D-4	2-5 6-9	AD0-AD3 –
P3.0 P3.1 P3.2 P3.3 P3.4 P3.5 P3.6 P3.7	I/O	I/O port with bit programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	E-4	11 12 13 14 21 23 24 25	BUZ SCLK0 SO0 SI0 SCK1 SO1 SI1 –
P4.0-P4.7	I/O	I/O port with nibble programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-41	26-33	SEG39-SEG32
P5.0-P5.7	I/O	I/O port with nibble programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-41	34-41	SEG31-SEG24
P6.0-P6.7	I/O	I/O port with nibble programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-41	42-49	SEG23-SEG16
P7.0-P7.7	I/O	I/O port with nibble programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-41	50-57	SEG15-SEG8
P8.0-P8.7	I/O	I/O port with nibble programmable pins; Schmitt trigger input or push-pull, open-drain output and software assignable pull-ups.	H-41	58-65	SEG7-SEG0

Table 1-1. S3C830A Pin Descriptions (Continued)

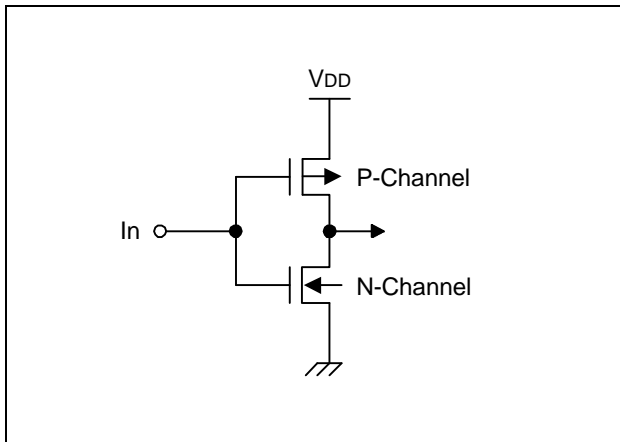
Pin Names	Pin Type	Pin Description	Circuit Type	Pin No.	Share Pins
COM0-COM3	O	Common signal output for LCD display	H	69-66	–
SEG0-SEG39	I/O	LCD segment signal output	H-41	65-26	P8-P4
BIAS	I	LCD power control	–	73	–
VLC0 VLC1 VLC2	I	LCD power supply Voltage dividing resistors are assignable by software	–	72-70	–
V <sub>DD</sub>	–	Main power supply	–	15	–
V <sub>SS</sub>	–	Main ground	–	16	–
VDDPLL0-1	–	PLL/IFC power supply	–	82, 76	–
VSSPLL	–	PLL/IFC ground	–	79	–
AV <sub>DD</sub>	–	A/D converter power supply	–	10	–
X <sub>OUT</sub> , X <sub>IN</sub>	–	Main oscillator pins for CPU oscillation	–	17, 18	–
TEST1, TEST2	I	Test signal input pin (Must be connected to V <sub>SS</sub> )	–	19, 20	–
TEST3	O	Test signal output pin (Must be remained to open)	–	74	–
LVREN	I	LVR enable pin (Must be connected to V <sub>DD</sub> or V <sub>SS</sub> )	A	75	–
RESET	I	System reset pin	B	22	–
CE	I	Input pin for checking device power Normal operation is high level and PLL/IFC Operation is stopped at low power	B-5	85	–
EO0	O	PLL's phase error output0	A-2	83	–
EO1	O	PLL's phase error output1	A-2	84	–
VCOAM VCOFM	I	External VCOAM/VCOFM signal inputs	B-4	78, 77	–

Table 1-1. S3C830A Pin Descriptions (Continued)

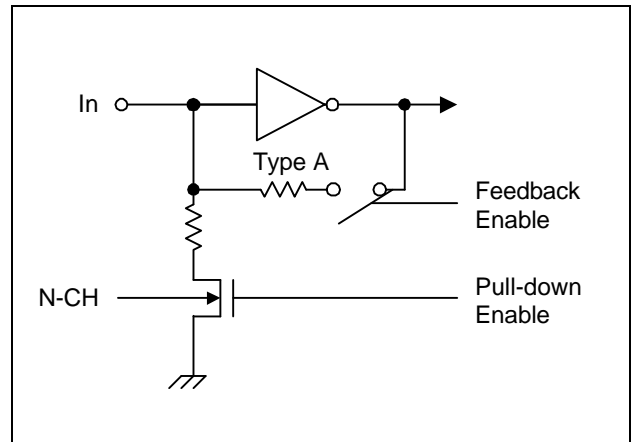
Pin Names	Pin Type	Pin Description	Circuit Type	Pin No.	Share Pins
FMIF, AMIF	I	FM/AM intermediate frequency signal inputs	B-4	81, 80	–
AD0-AD3	I/O	ADC input pins	F-16	2-5	P2.0-P2.3
BUZ	I/O	1, 1.5, 3 or 6 kHz frequency output for buzzer sound at 4.5 MHz clock	E-4	11	P3.0
SCK0	I/O	SIO0 interface signal	E-4	12	P3.1
SO0	I/O	SIO0 interface data output signal	E-4	13	P3.2
SI0	I/O	SIO0 interface data input signal	E-4	14	P3.3
SCK1	I/O	SIO1 interface signal	E-4	21	P3.4
SO1	I/O	SIO1 interface data output signal	E-4	23	P3.5
SI1	I/O	SIO1 interface data input signal	E-4	24	P3.6
T0CLK	I/O	Timer 0 clock input	E-4	87	P0.1
T0CAP	I/O	Timer 0 capture input	E-4	88	P0.2
T0OUT	I/O	Timer 0 clock output	E-4	89	P0.3
T0PWM	I/O	Timer 0 PWM output	E-4	89	P0.3
T1CLK	I/O	Timer 1 clock input	E-4	90	P0.4
T1OUT	I/O	Timer 1 clock output	E-4	91	P0.5
T2CLK	I/O	Timer 2 clock input	E-4	92	P0.6
T2OUT	I/O	Timer 2 clock output	E-4	93	P0.7
INT0-INT7	I/O	External interrupt input pins	D-7	94-1	P1.0-P1.7



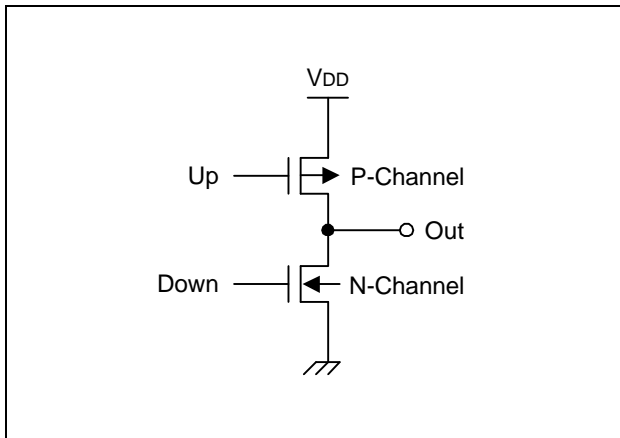
**PIN CIRCUITS**



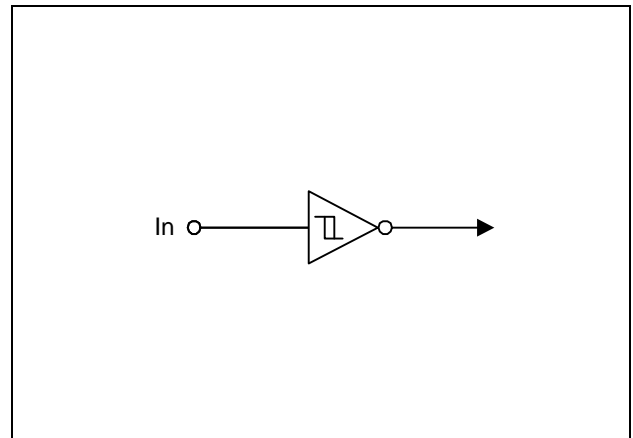
**Figure 1-3. Pin Circuit Type A**



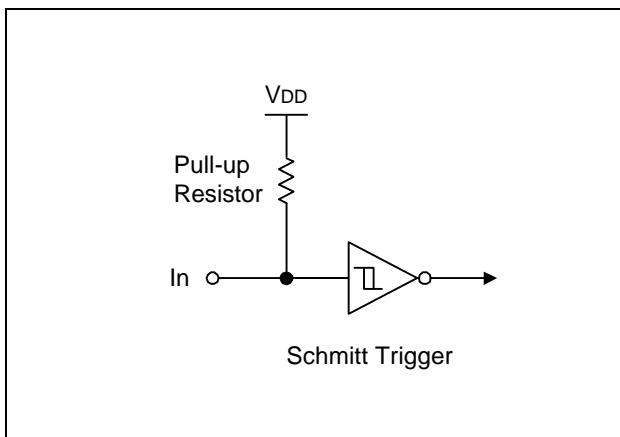
**Figure 1-6. Pin Circuit Type B-4**



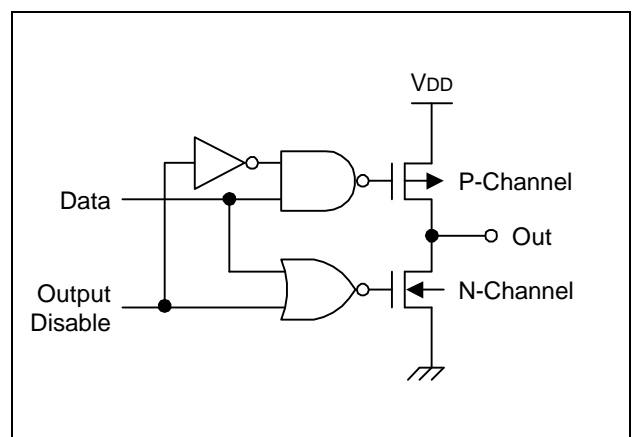
**Figure 1-4. Pin Circuit Type A-2 (EO)**



**Figure 1-7. Pin Circuit Type B-5 (CE)**



**Figure 1-5. Pin Circuit Type B (RESET)**



**Figure 1-8. Pin Circuit Type C**

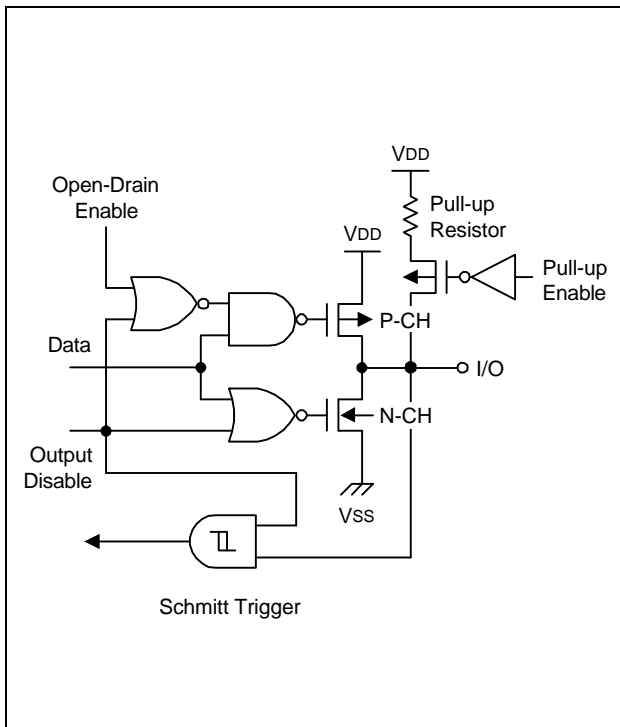


Figure 1-9. Pin Circuit Type E-4 (P0, P3)

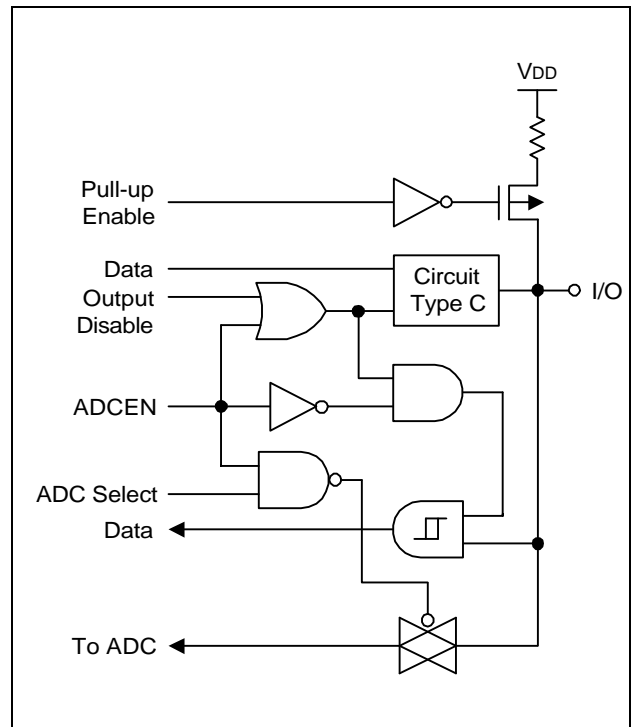


Figure 1-11. Pin Circuit Type F-16 (P2.0-P2.3)

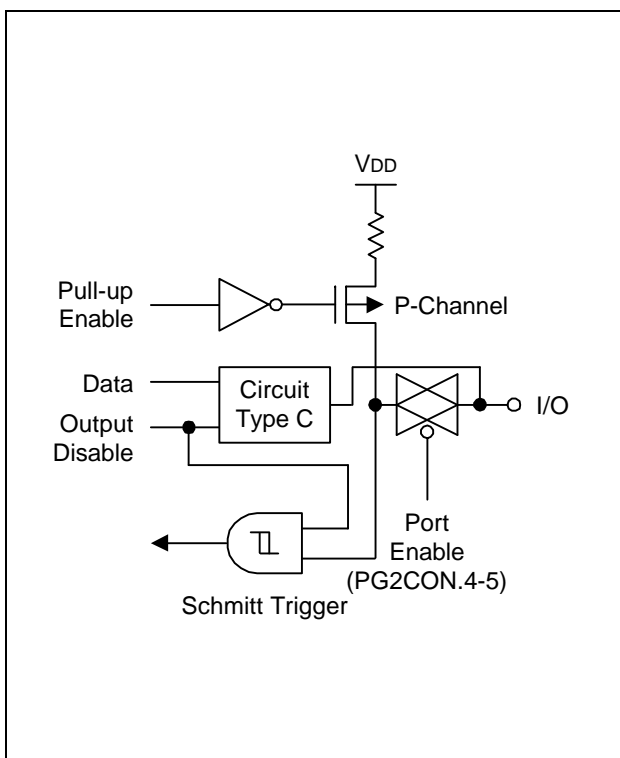


Figure 1-10. Pin Circuit Type D-7 (P1)

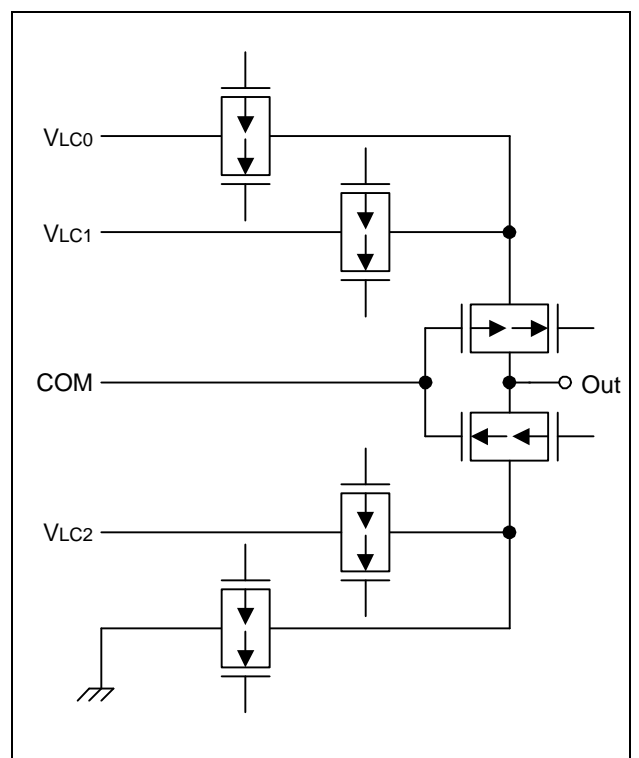


Figure 1-12. Pin Circuit Type H (COM0-COM3)

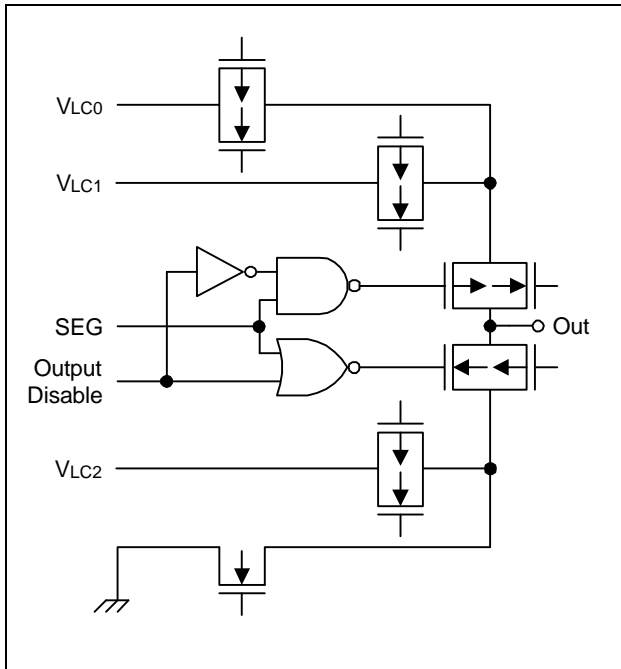


Figure 1-13. Pin Circuit Type H-39

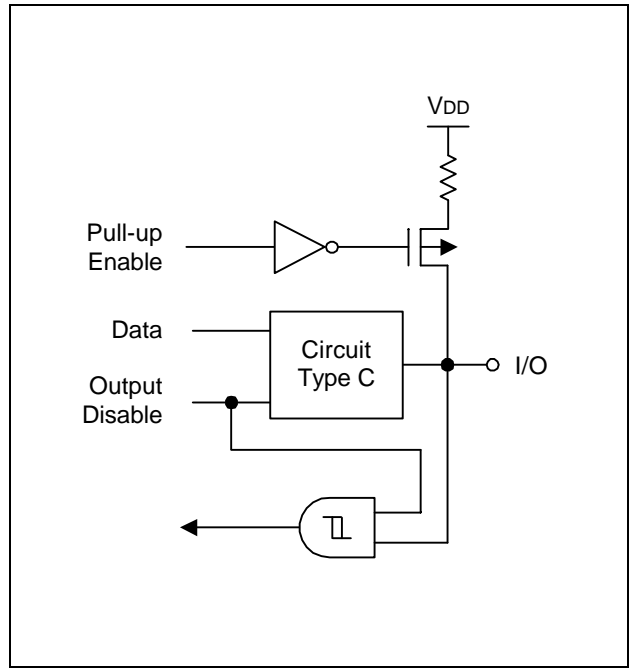


Figure 1-15. Pin Circuit Type D-4 (P2.4-P2.7)

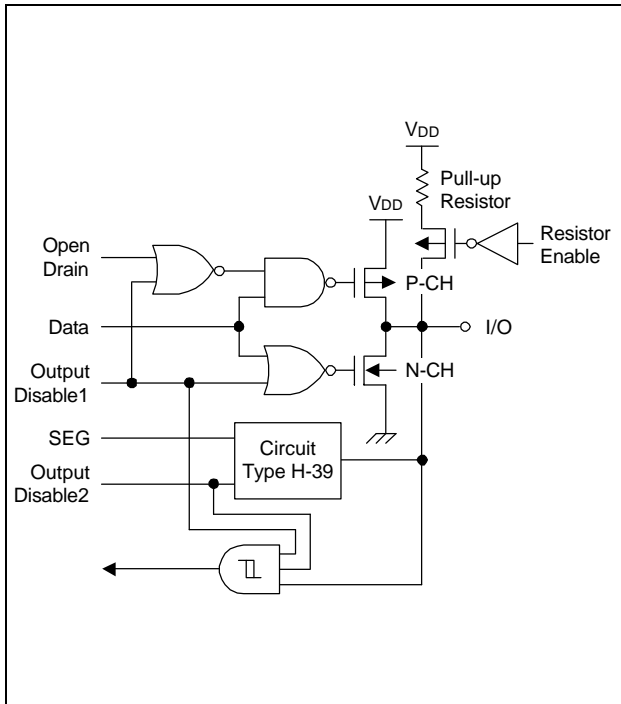


Figure 1-14. Pin Circuit Type H-41 (P4-P8)

# 2 ADDRESS SPACES

## OVERVIEW

The S3C830A microcontroller has two types of address space:

- Internal program memory (ROM)
- Internal register file

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 S3C830A has an internal 48-Kbyte mask-programmable ROM.

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

A 20-byte LCD display register file is implemented.

There are 2,134 mapped registers in the internal register file. Of these, 2,064 are for general-purpose. (This number includes a 16-byte working register common area used as a "scratch area" for data operations, eight 192-byte prime register areas, and eight 64-byte areas (Set 2)). Thirteen 8-bit registers are used for the CPU and the system control, and 57 registers are mapped for peripheral controls and data registers. Ten register locations are not mapped.

## PROGRAM MEMORY (ROM)

Program memory (ROM) stores program codes or table data. The S3C830A has 48K bytes internal mask-programmable program memory.

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.

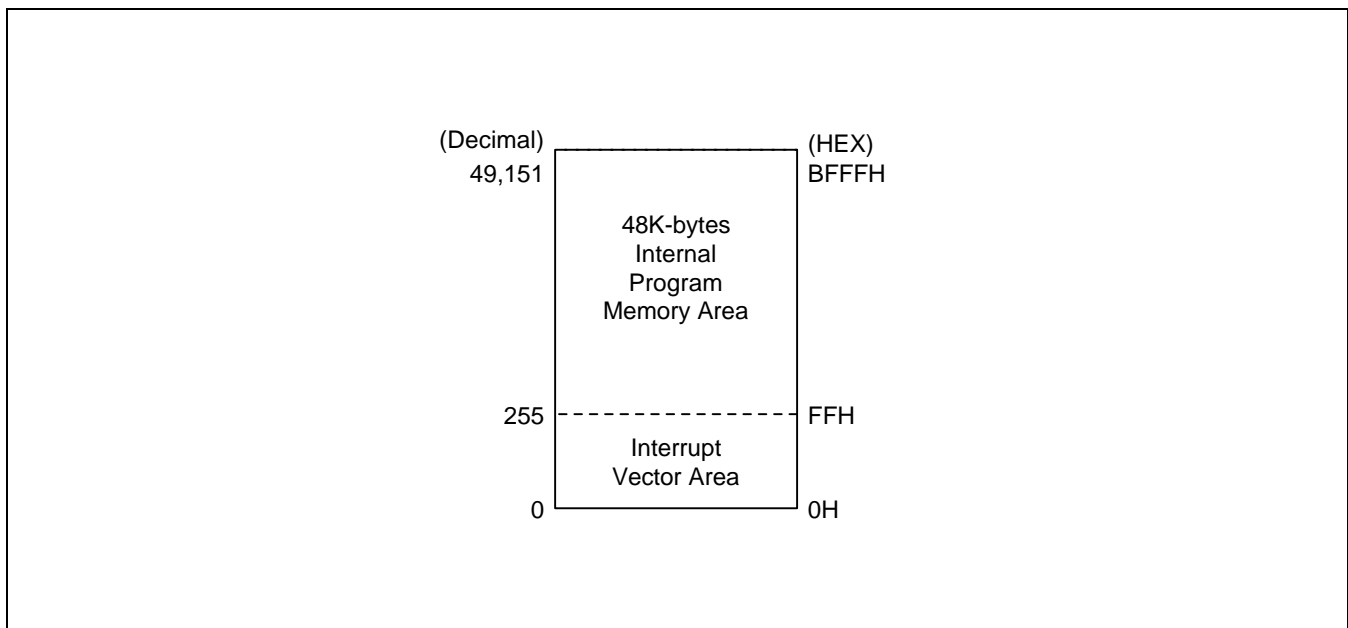


Figure 2-1. Program Memory Address Space

## REGISTER ARCHITECTURE

In the S3C830A 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 case of S3C830A the total number of addressable 8-bit registers is 2134. Of these 2134 registers, 13 bytes are for CPU and system control registers, 57 bytes are for peripheral control and data registers, 16 bytes are used as a shared working registers, and 2048 registers are for general-purpose use, page 0-page 7 (including 20 bytes for LCD display registers).

You can always address set 1 register locations, regardless of which of the eight register pages is currently selected. Set 1 locations, however, can only be addressed using register 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. S3C830A Register Type Summary**

Register Type	Number of Bytes
General-purpose registers (including the 16-byte common working register area, eight 192-byte prime register area (including LCD data registers), and eight 64-byte set 2 area).	2,064
CPU and system control registers	13
Mapped clock, peripheral, I/O control, and data registers	57
<b>Total Addressable Bytes</b>	<b>2,134</b>

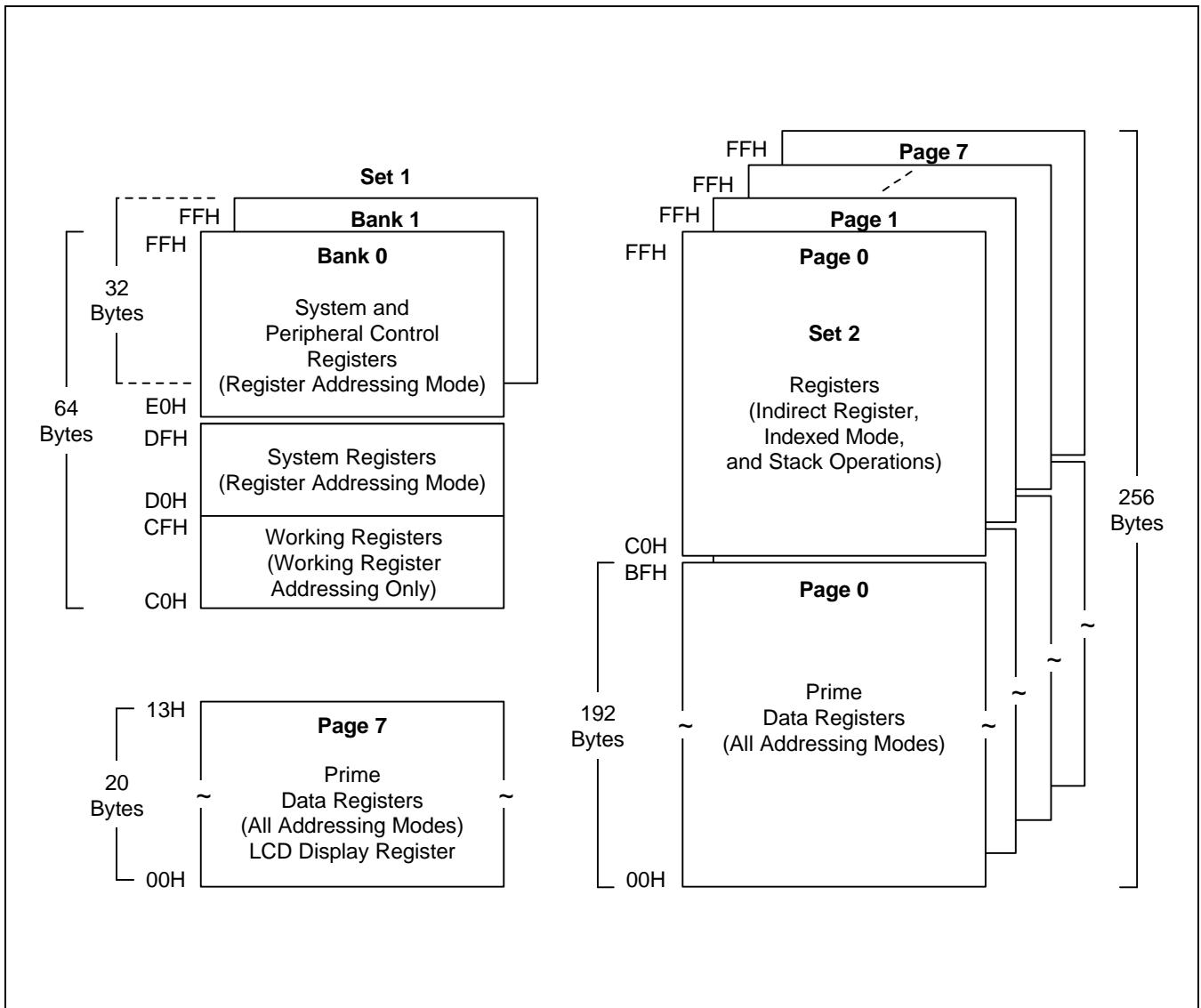
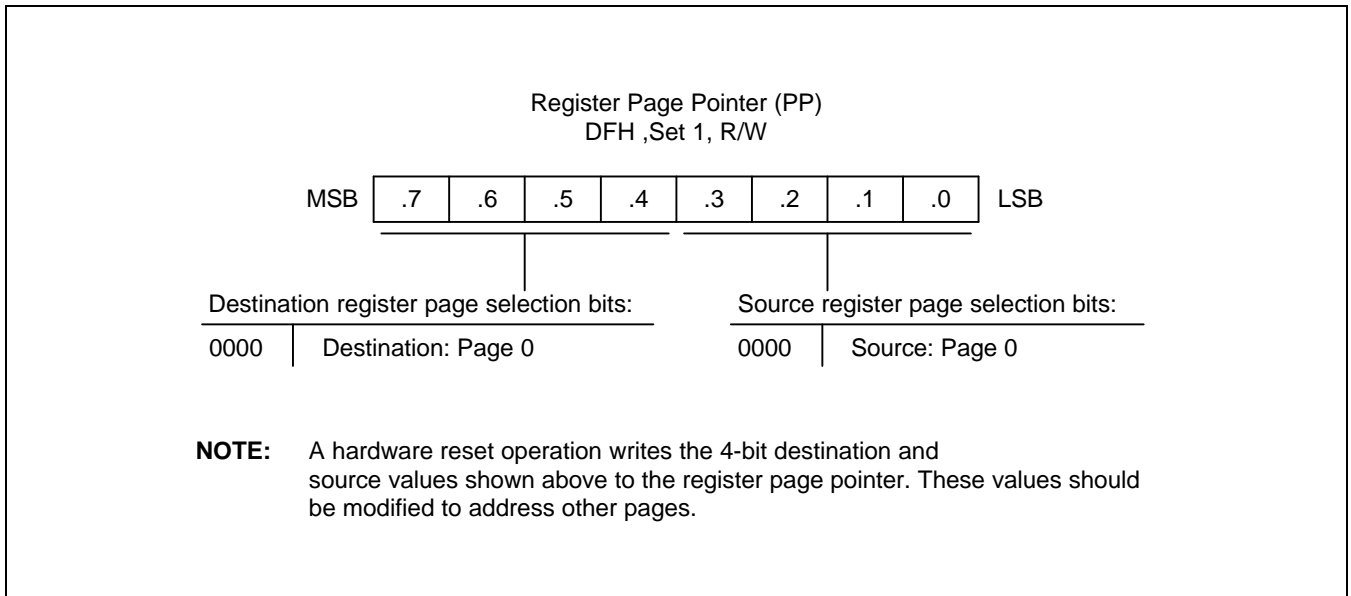


Figure 2-2. Internal Register File Organization

## 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 S3C830A 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 "0000", 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)

	LD	PP,#00H	; Destination ← 0, Source ← 0
	SRP	#0C0H	
RAMCL0	LD	R0,#0FFH	; Page 0 RAM clear starts
	CLR	@R0	
	DJNZ	R0,RAMCL0	
	CLR	@R0	; R0 = 00H
	LD	PP,#10H	; Destination ← 1, Source ← 0
	LD	R0,#0FFH	; Page 1 RAM clear starts
RAMCL1	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 two 32-byte areas (bank 0 and bank 1) of set 1 (E0H–FFH) contains 57 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 S3C830A, the set 2 address range (C0H–FFH) is accessible on pages 0-7.

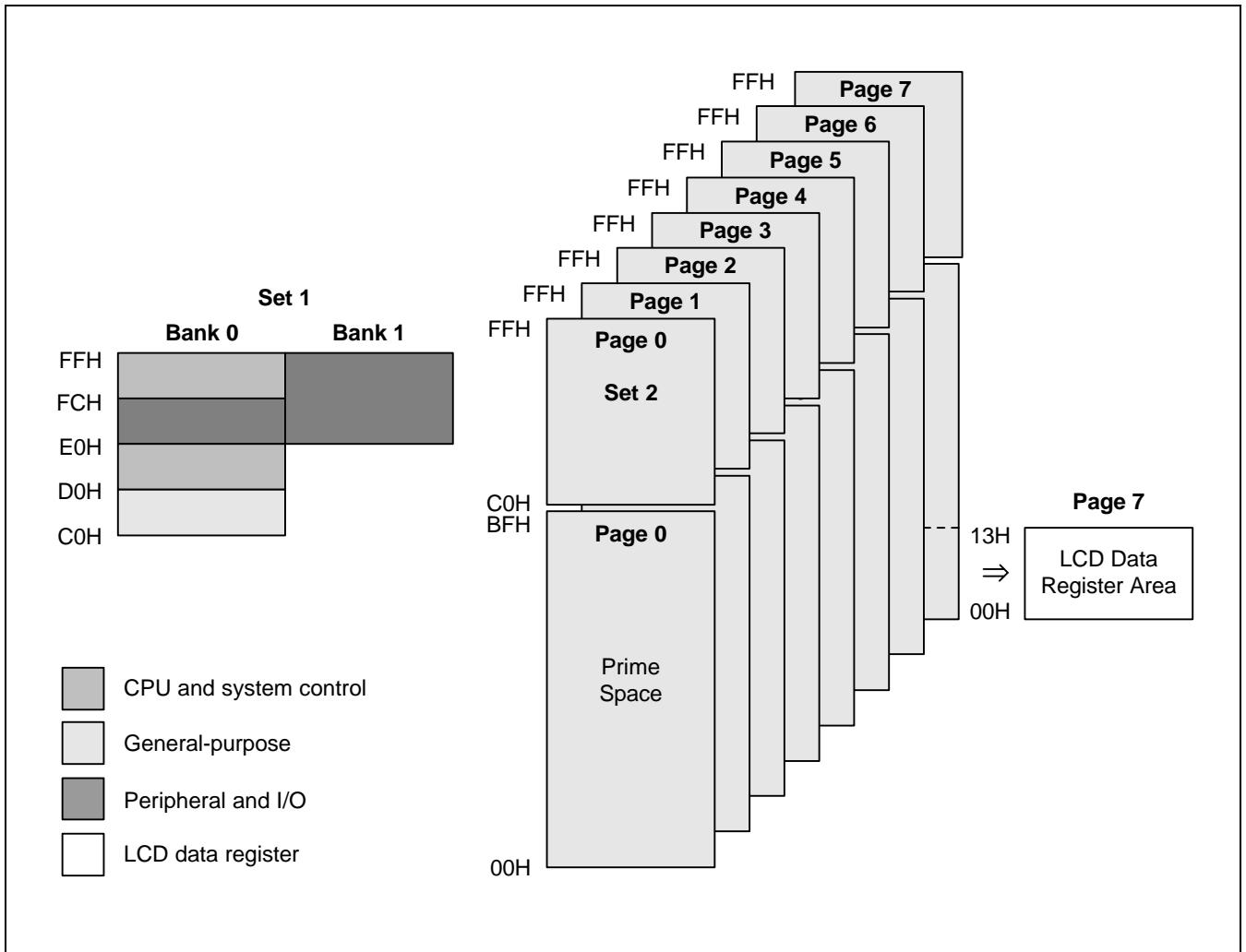
The logical division of set 1 and set 2 is maintained by means of addressing mode restrictions. You can use only Register addressing mode to access set 1 locations. In order to access registers in set 2, you must use Register Indirect addressing mode or Indexed addressing mode.

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

**PRIME REGISTER SPACE**

The lower 192 bytes (00H–BFH) of the S3C830A's eight 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, 3, 4, 5, 6, or 7 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 32 8-byte register groups or "slices." Each slice comprises of eight 8-bit registers.

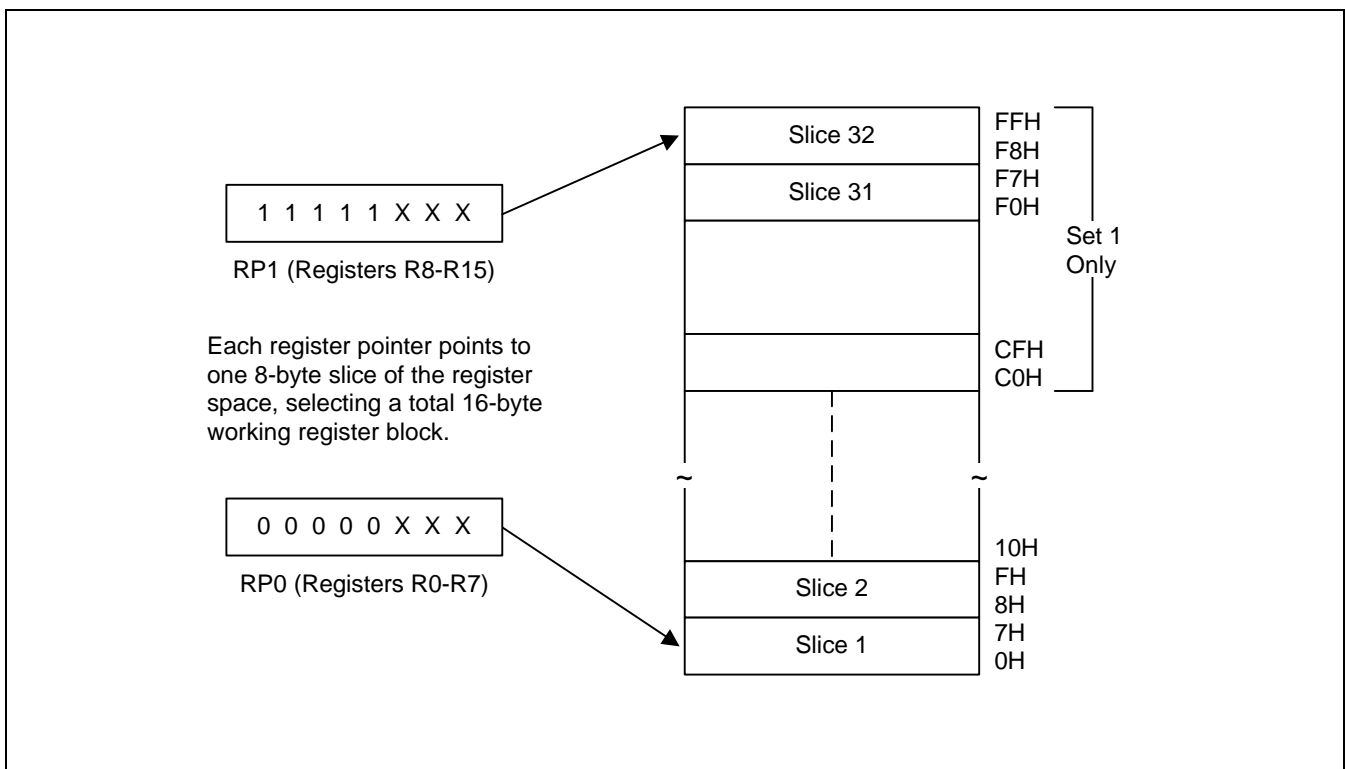
Using the two 8-bit register pointers, RP1 and RP0, two working register slices can be selected at any one time to form a 16-byte working register block. Using the register pointers, you can move this 16-byte register block anywhere in the addressable register file, except the set 2 area.

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 24 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.

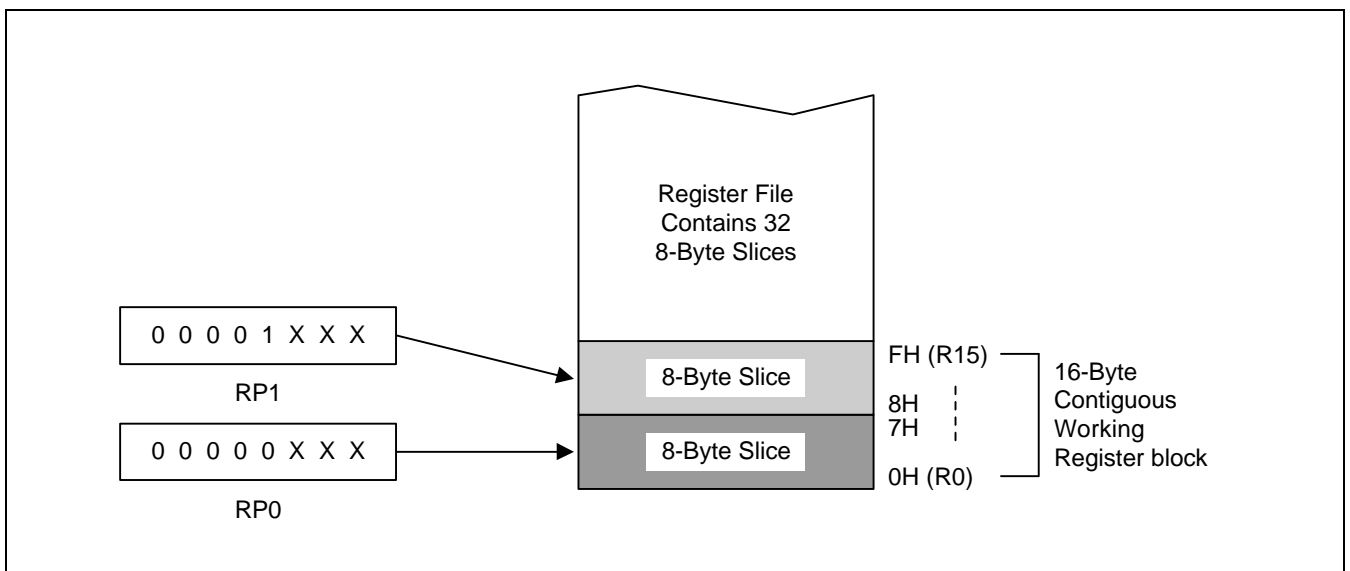
The selected 16-byte working register block usually consists of two contiguous 8-byte slices. As a general programming guideline, it is recommended that RP0 point to the "lower" slice and RP1 point to the "upper" slice (see Figure 2-6). In some cases, it may be necessary to define working register areas 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 ← 0F8H
    
```



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

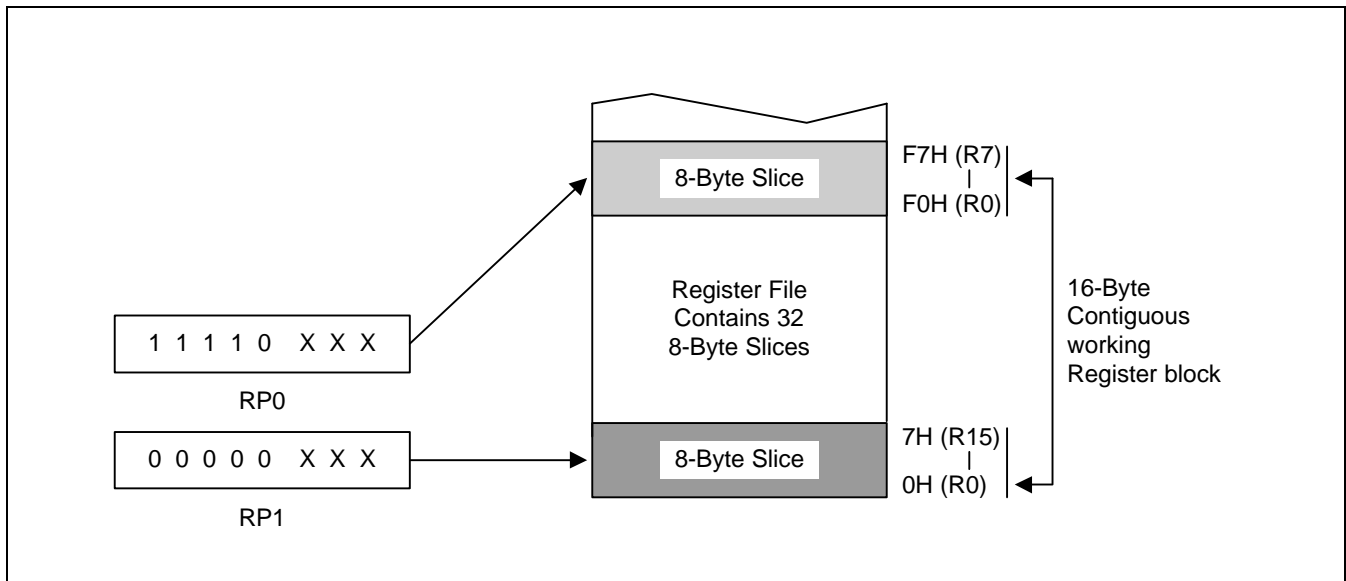


Figure 2-7. Non-Contiguous 16-Byte Working Register Block

**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.

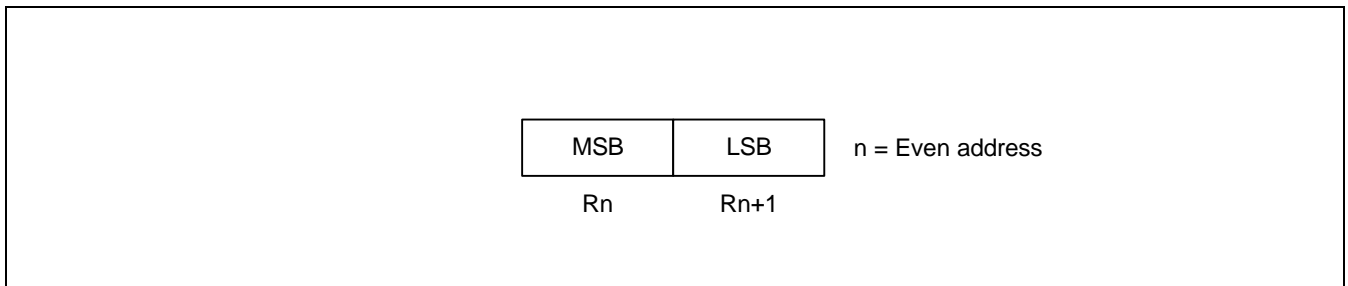
## REGISTER ADDRESSING

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**

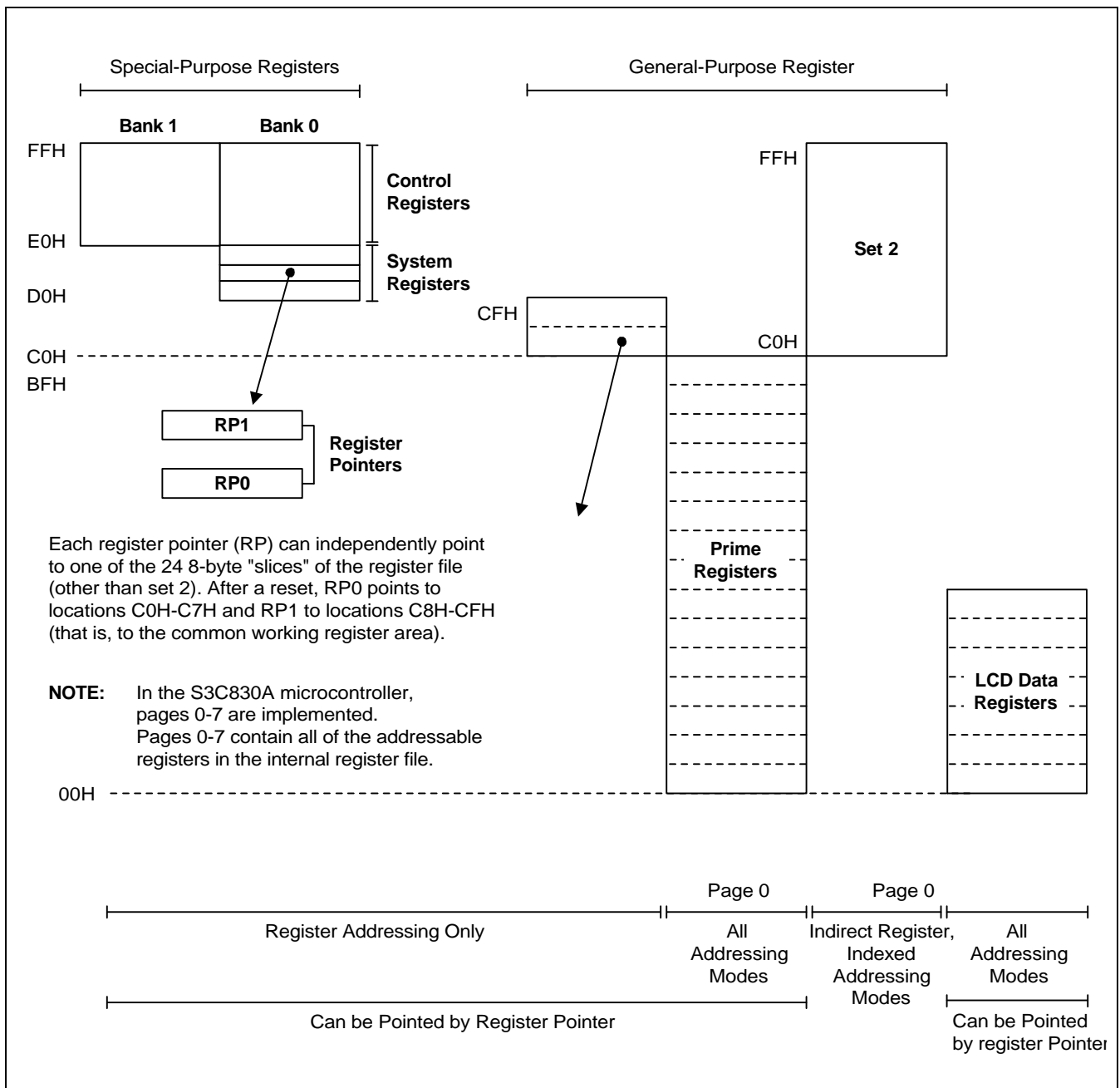


Figure 2-9. Register File Addressing

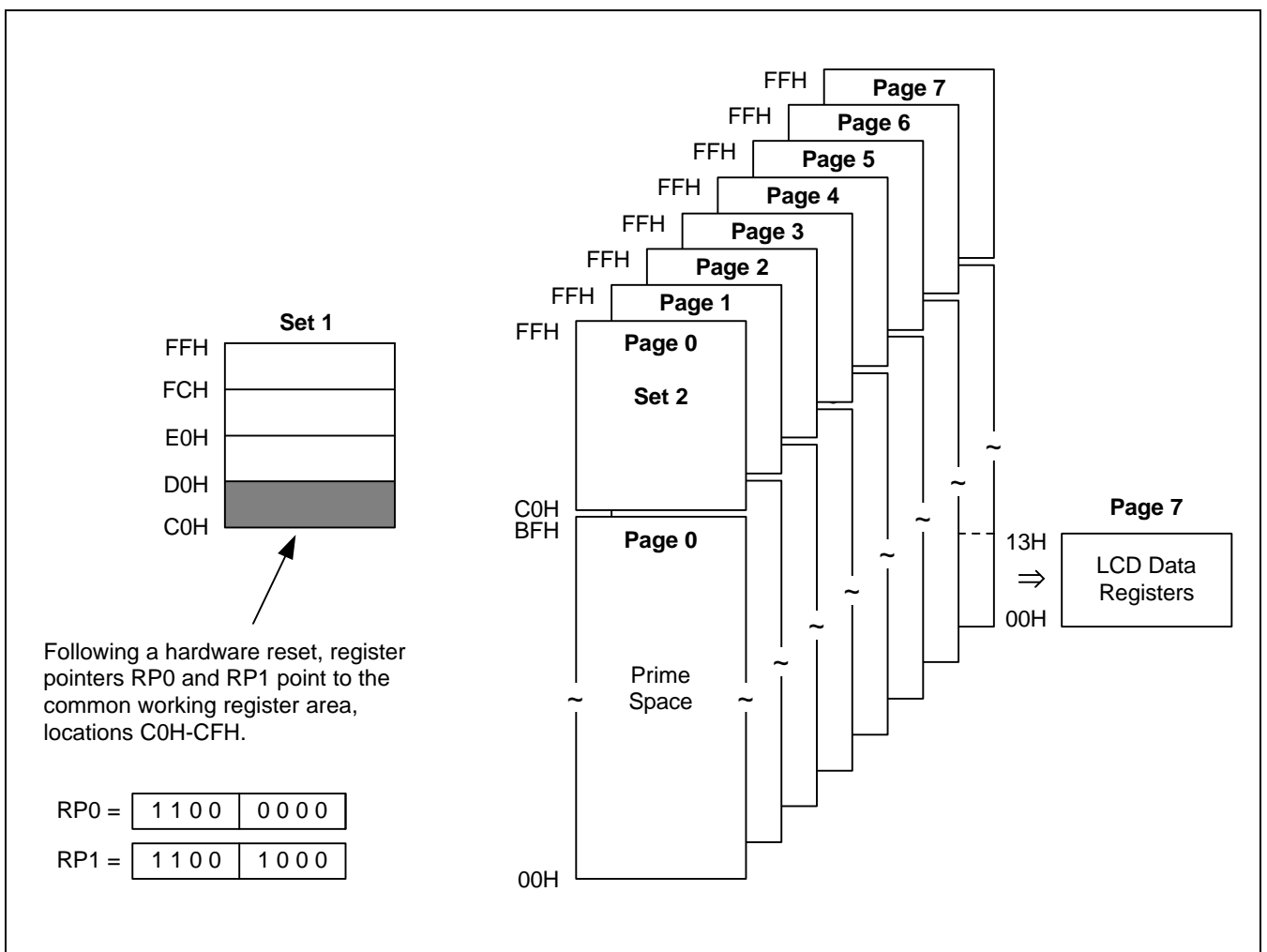
**COMMON WORKING REGISTER AREA (C0H–CFH)**

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**
- ```
LD    0C2H,40H                ; Invalid addressing mode!
```

Use working register addressing instead:

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

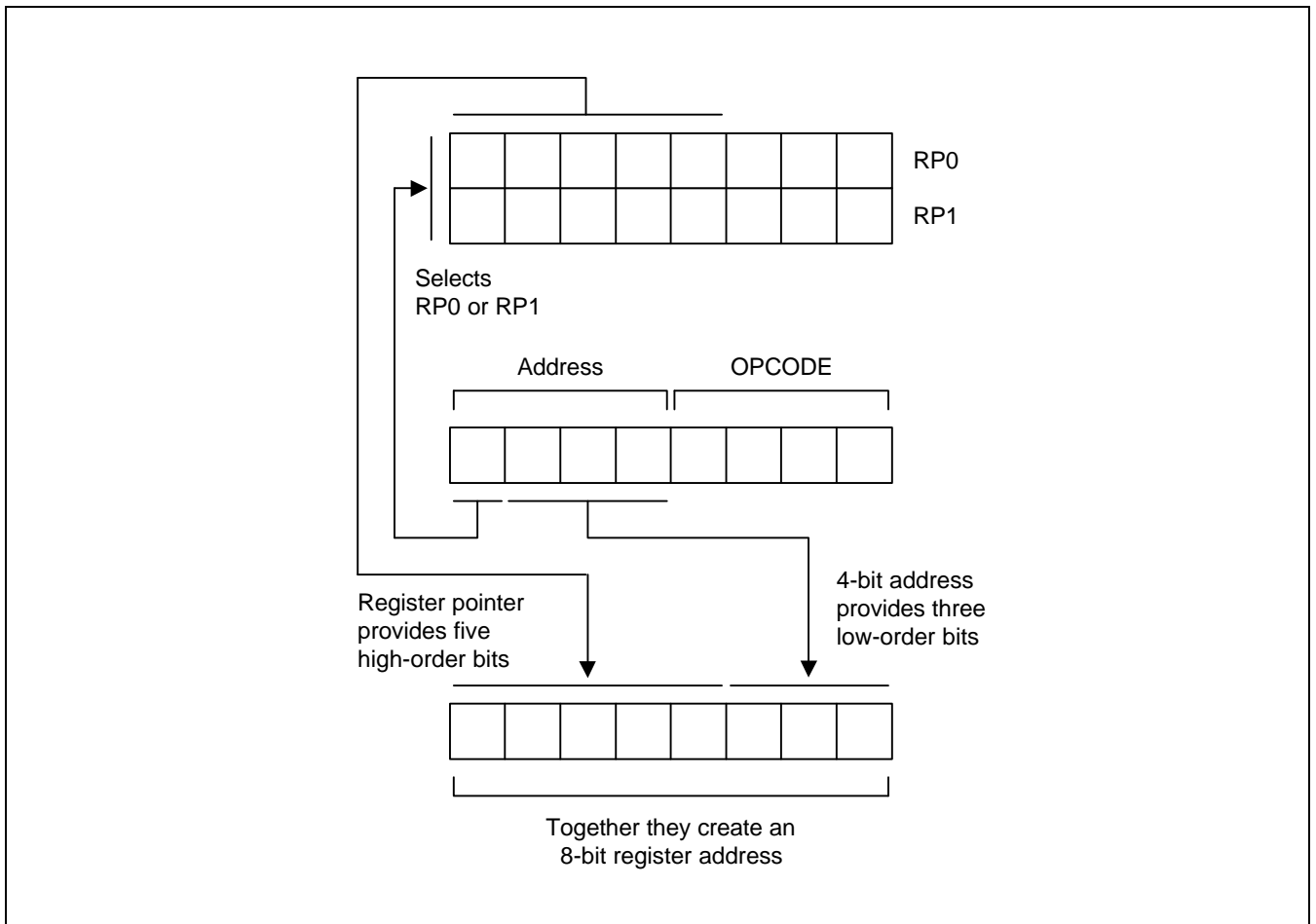


Figure 2-11. 4-Bit Working Register Addressing

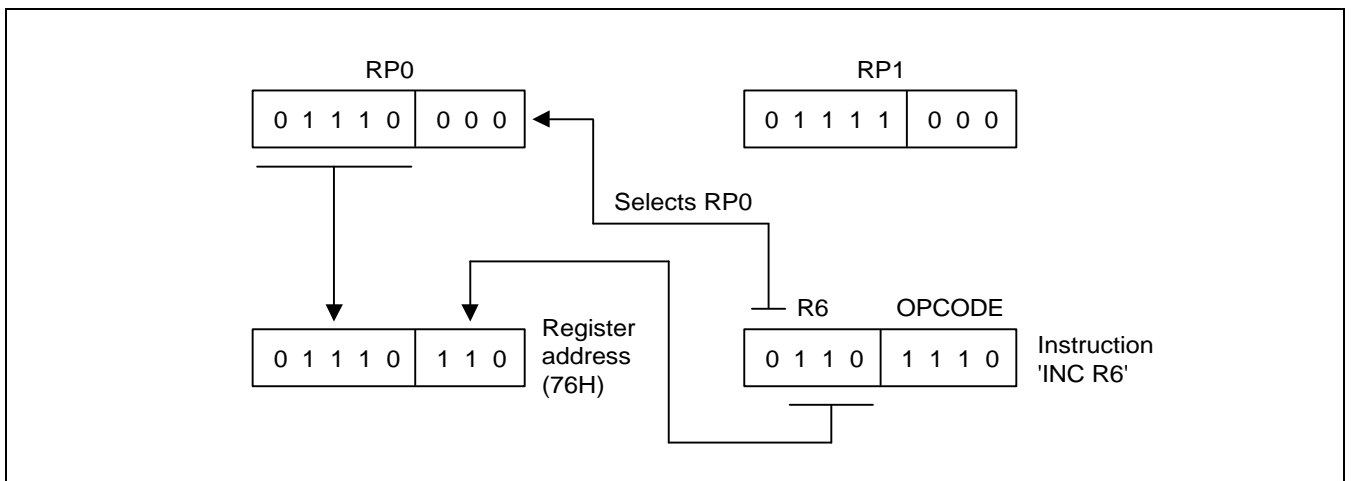


Figure 2-12. 4-Bit Working Register Addressing Example

## 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).

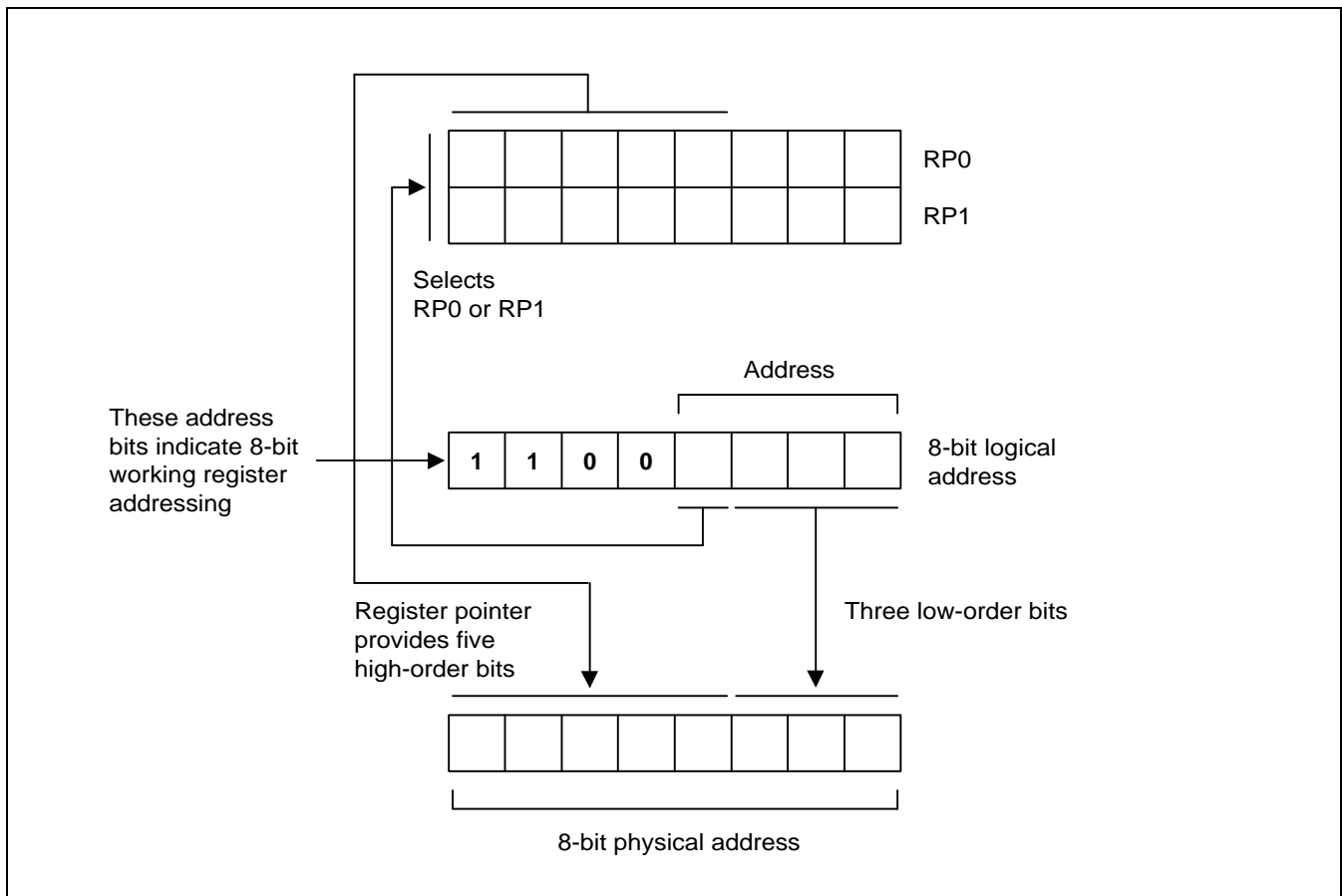


Figure 2-13. 8-Bit Working Register Addressing

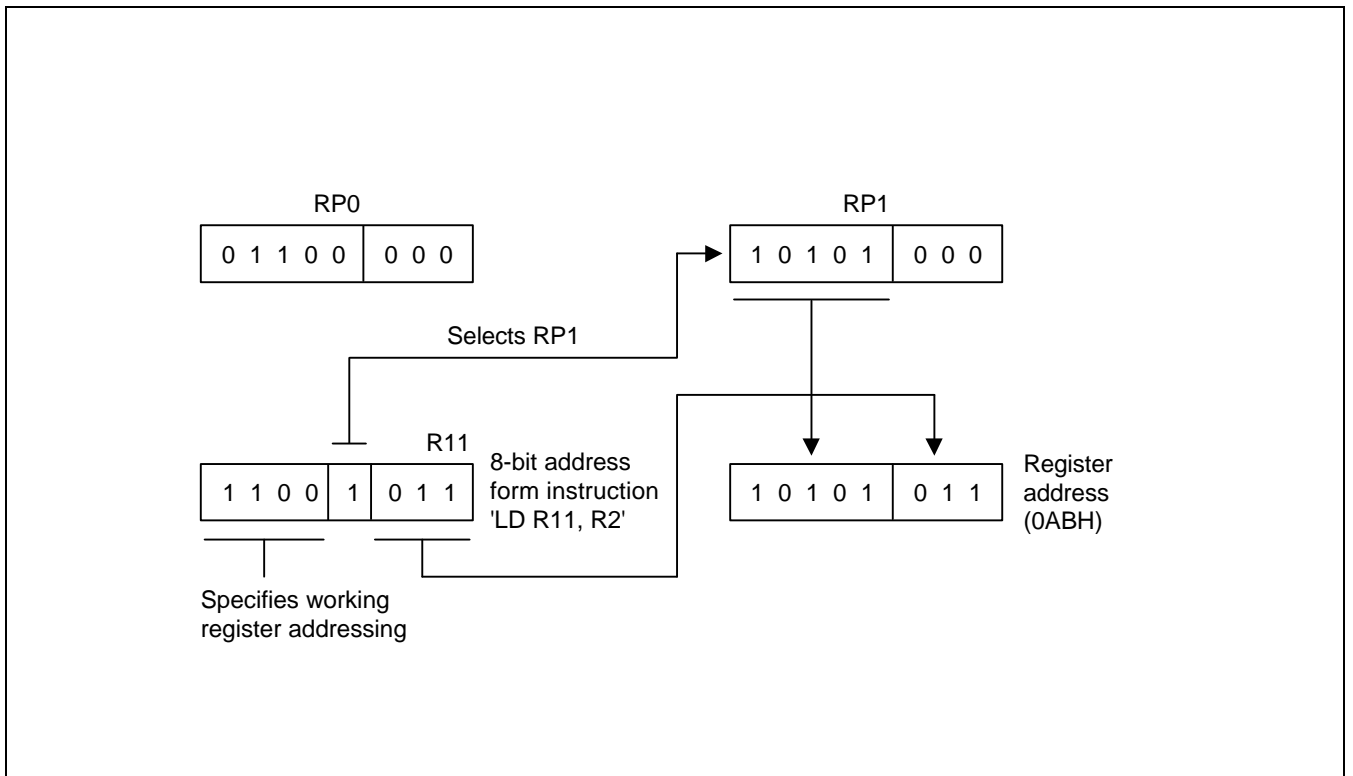


Figure 2-14. 8-Bit Working Register Addressing Example

## SYSTEM AND USER STACK

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 S3C830A 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.

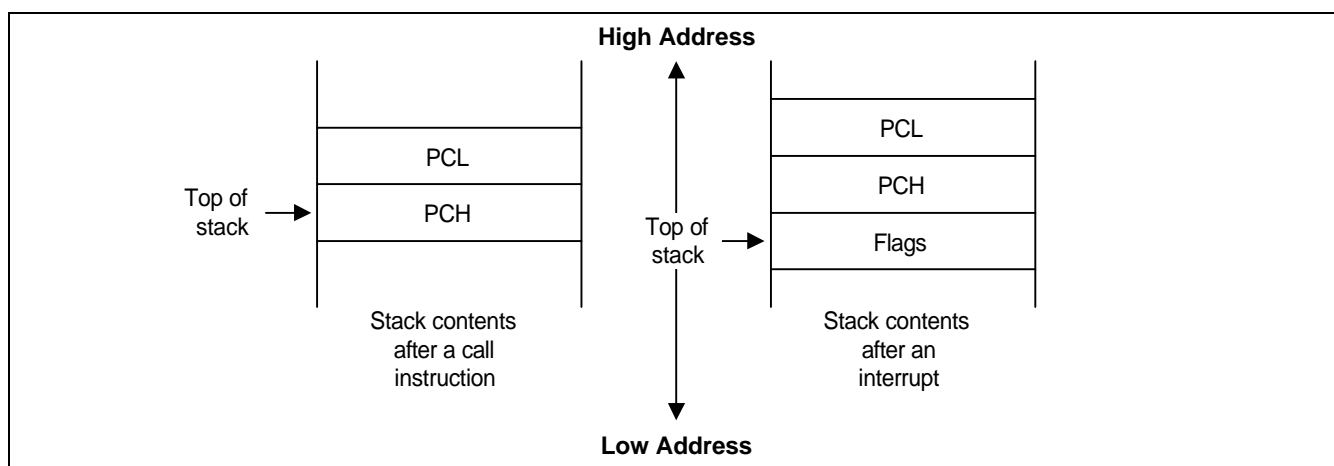


Figure 2-15. Stack Operations

### 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.

Because only internal memory space is implemented in the S3C830A, 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.

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

```

NOTES

# 3 ADDRESSING MODES

## OVERVIEW

Instructions that are stored in program memory are fetched for execution using the program counter. Instructions indicate the operation to be performed and the data to be operated on. Addressing mode is the method used to determine the location of the data operand. The operands specified in SAM88RC 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. The seven addressing modes and their symbols are:

- 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 (R), the operand value is the content of a specified register or register pair (see Figure 3-1).

Working register addressing differs from Register addressing in that 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).

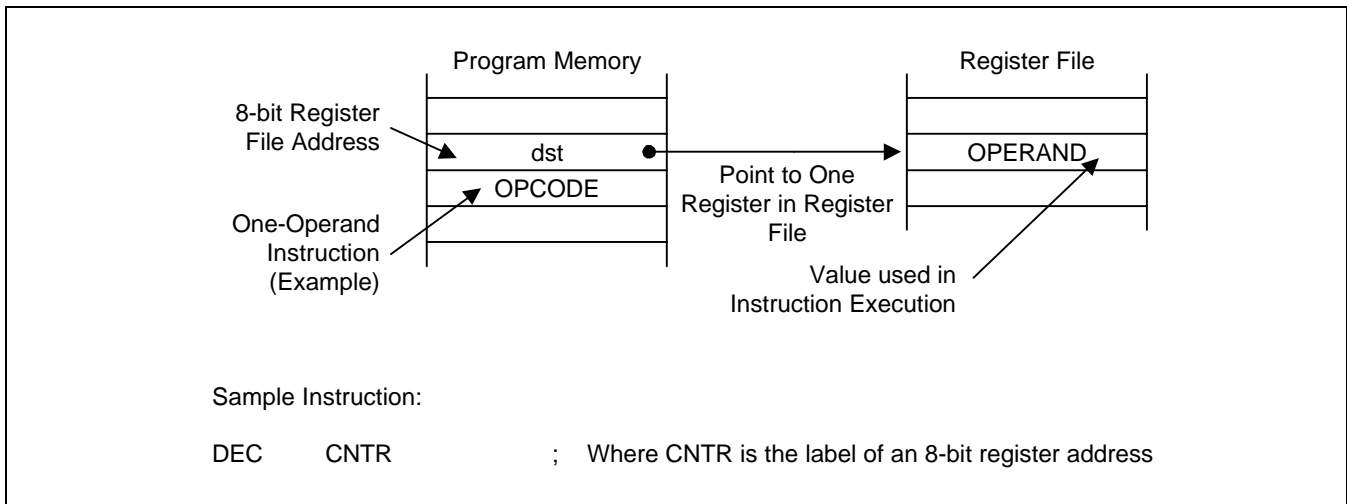


Figure 3-1. Register Addressing

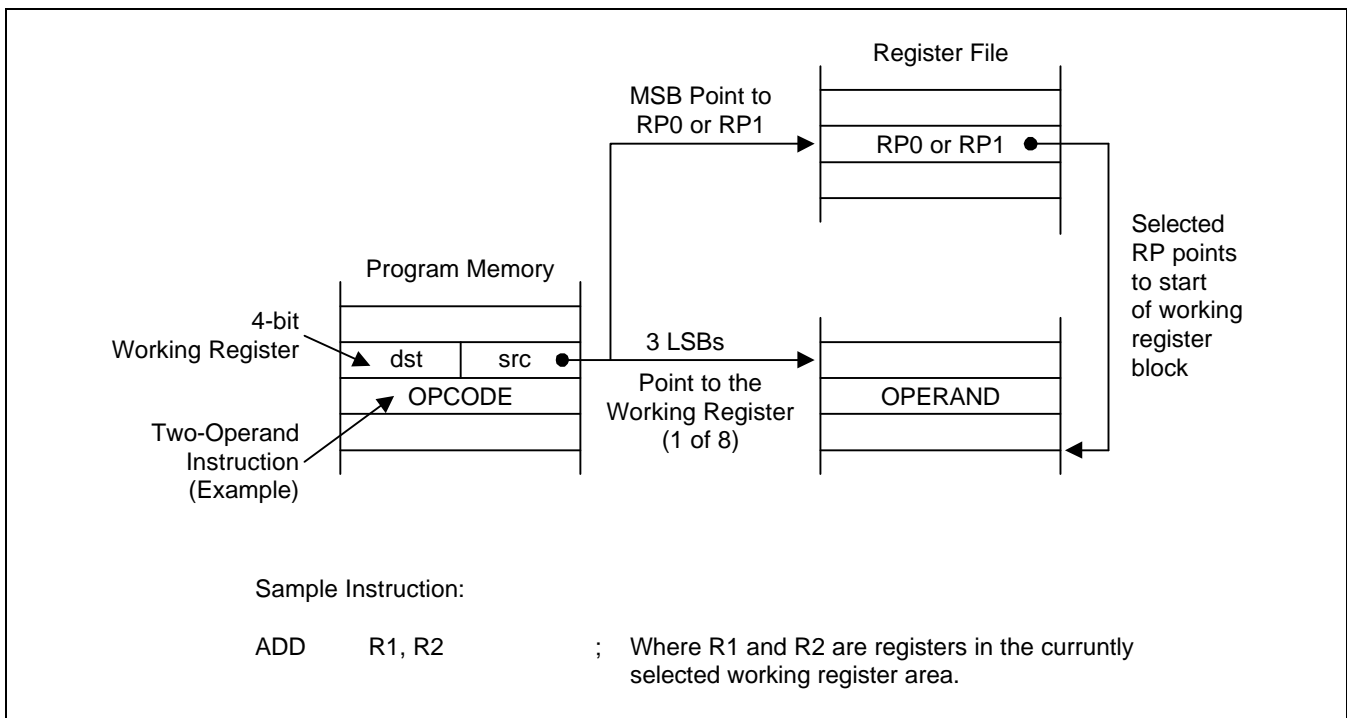


Figure 3-2. Working Register Addressing

### 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 (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. Please note, however, that you cannot access locations C0H–FFH in set 1 using the Indirect Register addressing mode.

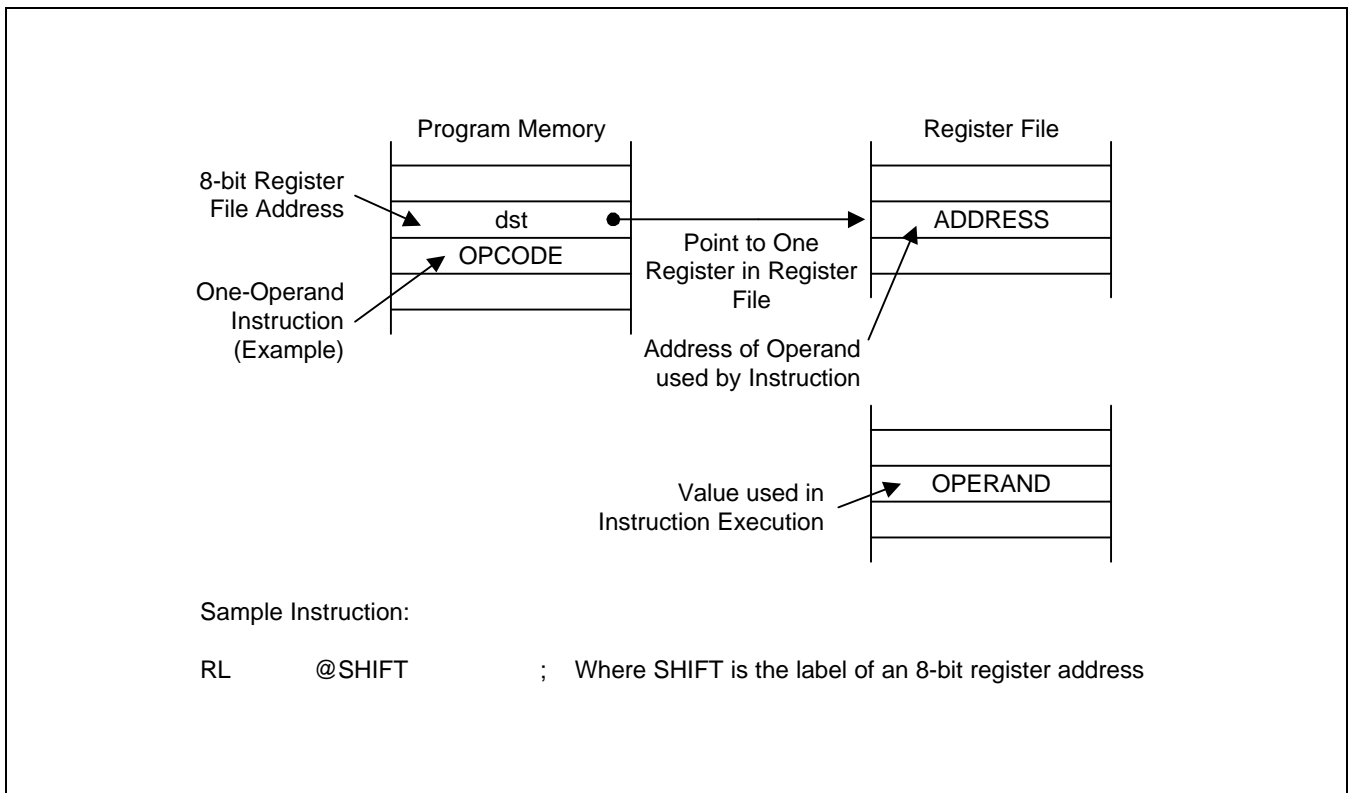


Figure 3-3. Indirect Register Addressing to Register File

INDIRECT REGISTER ADDRESSING MODE (Continued)

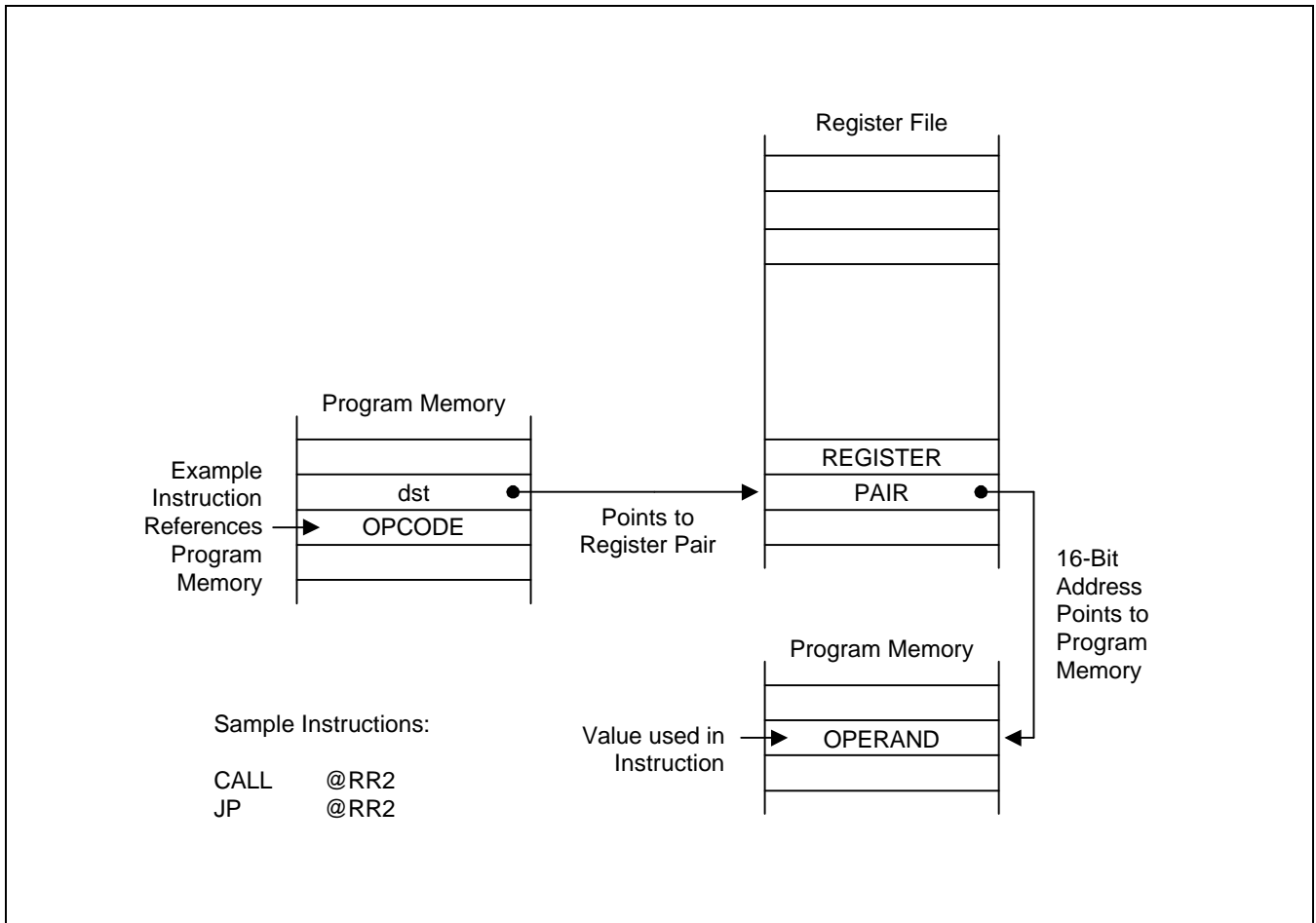


Figure 3-4. Indirect Register Addressing to Program Memory

### INDIRECT REGISTER ADDRESSING MODE (Continued)

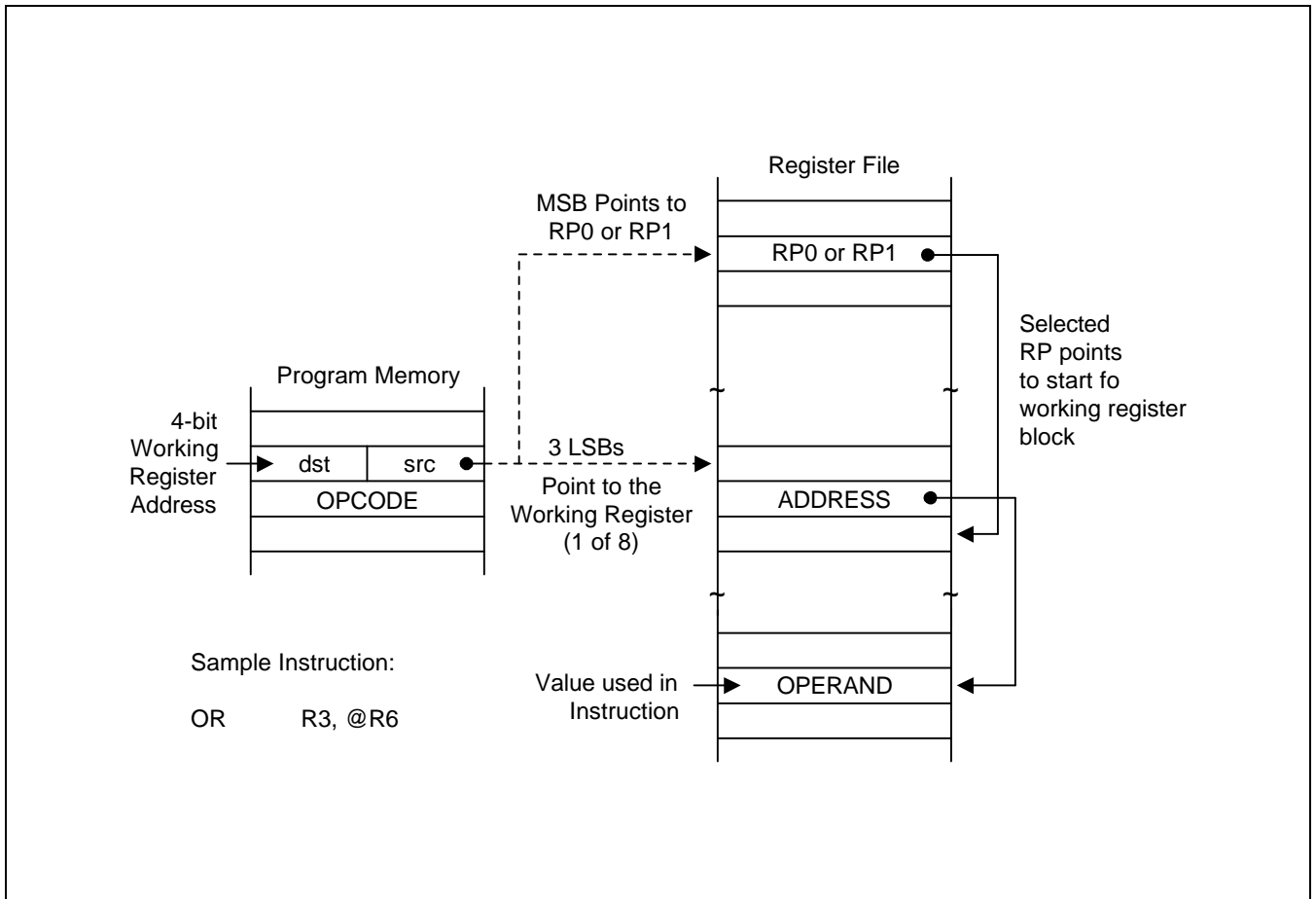
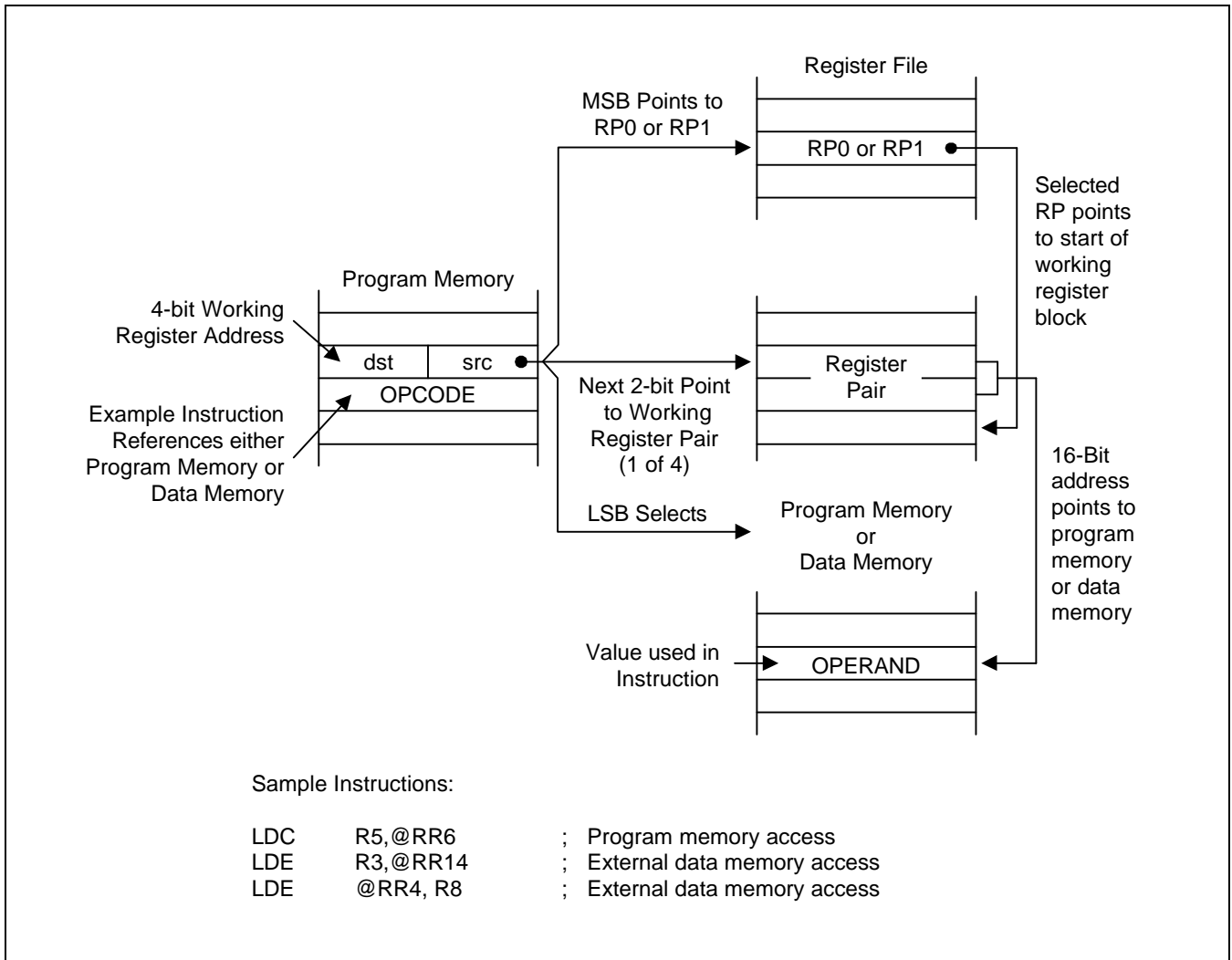


Figure 3-5. Indirect Working Register Addressing to Register File

**INDIRECT REGISTER ADDRESSING MODE (Concluded)**



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

### INDEXED ADDRESSING MODE (X)

Indexed (X) addressing mode adds an offset value to a base address during instruction execution in order to calculate the effective operand address (see Figure 3-7). You can use Indexed addressing mode to access locations in the internal register file or in external memory. Please note, however, that you cannot access locations C0H–FFH in set 1 using Indexed addressing mode.

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 that base address (see Figure 3-9).

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

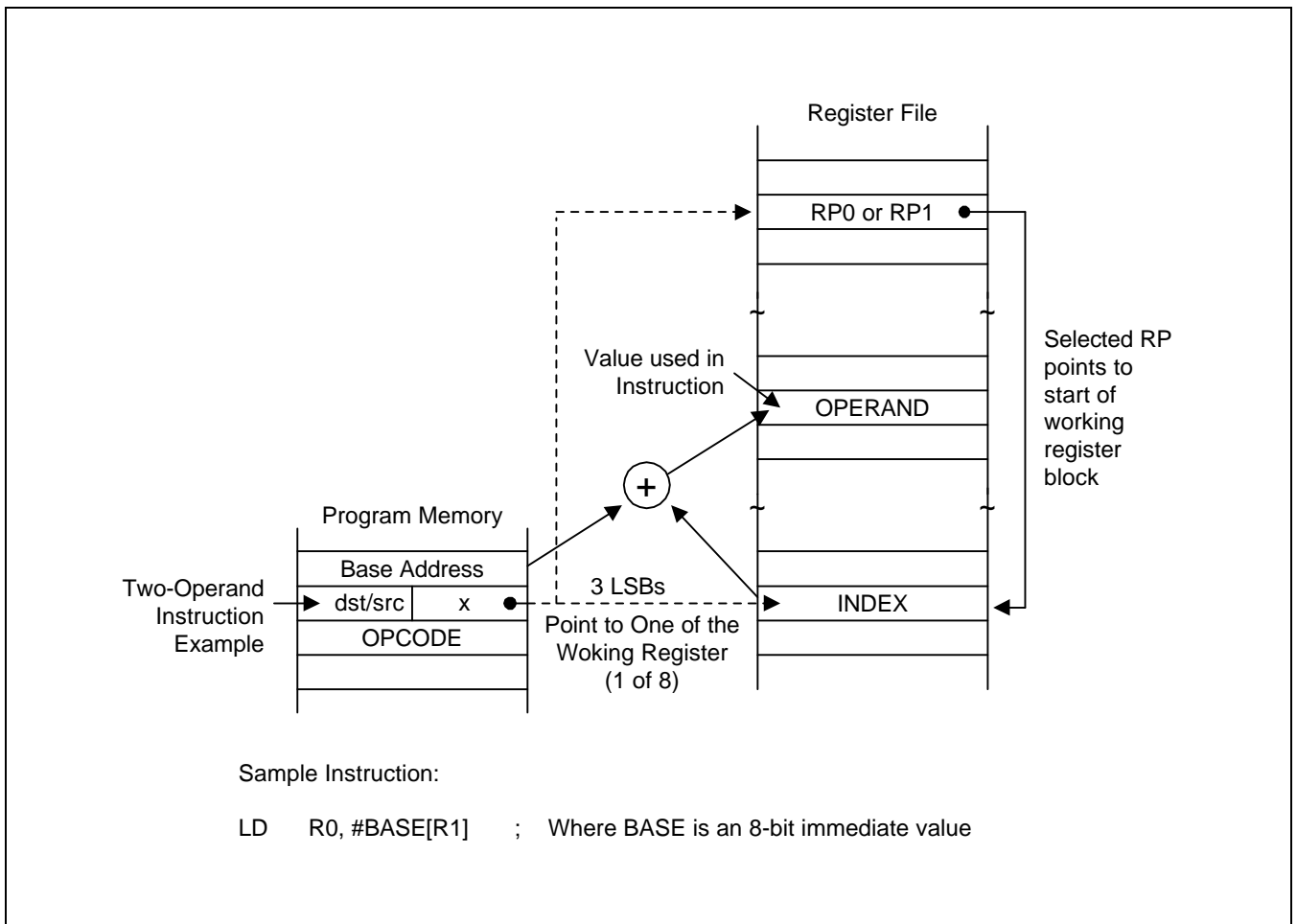


Figure 3-7. Indexed Addressing to Register File

INDEXED ADDRESSING MODE (Continued)

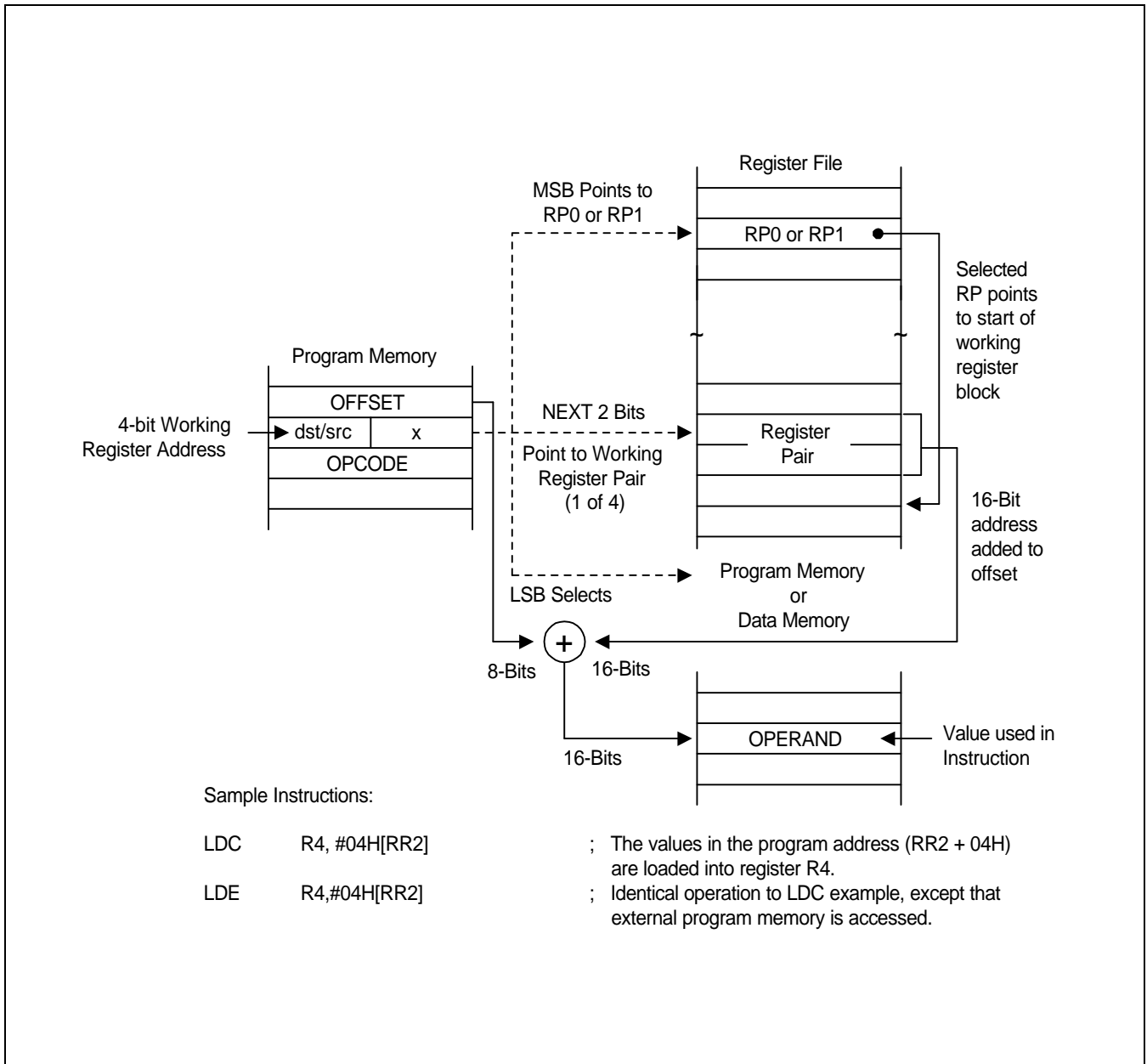
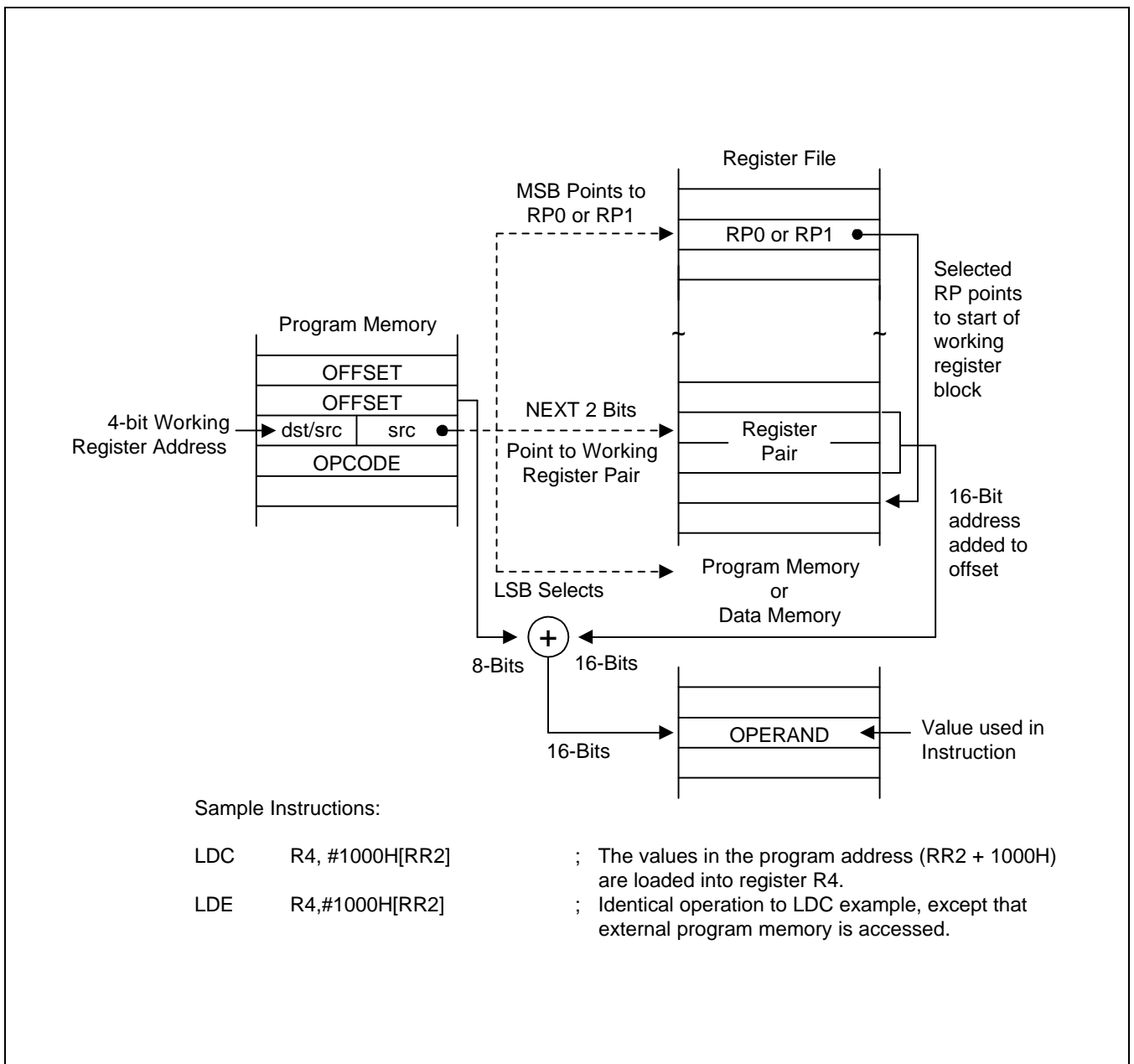


Figure 3-8. Indexed Addressing to Program or Data Memory with Short Offset

**INDEXED ADDRESSING MODE (Concluded)**



**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.

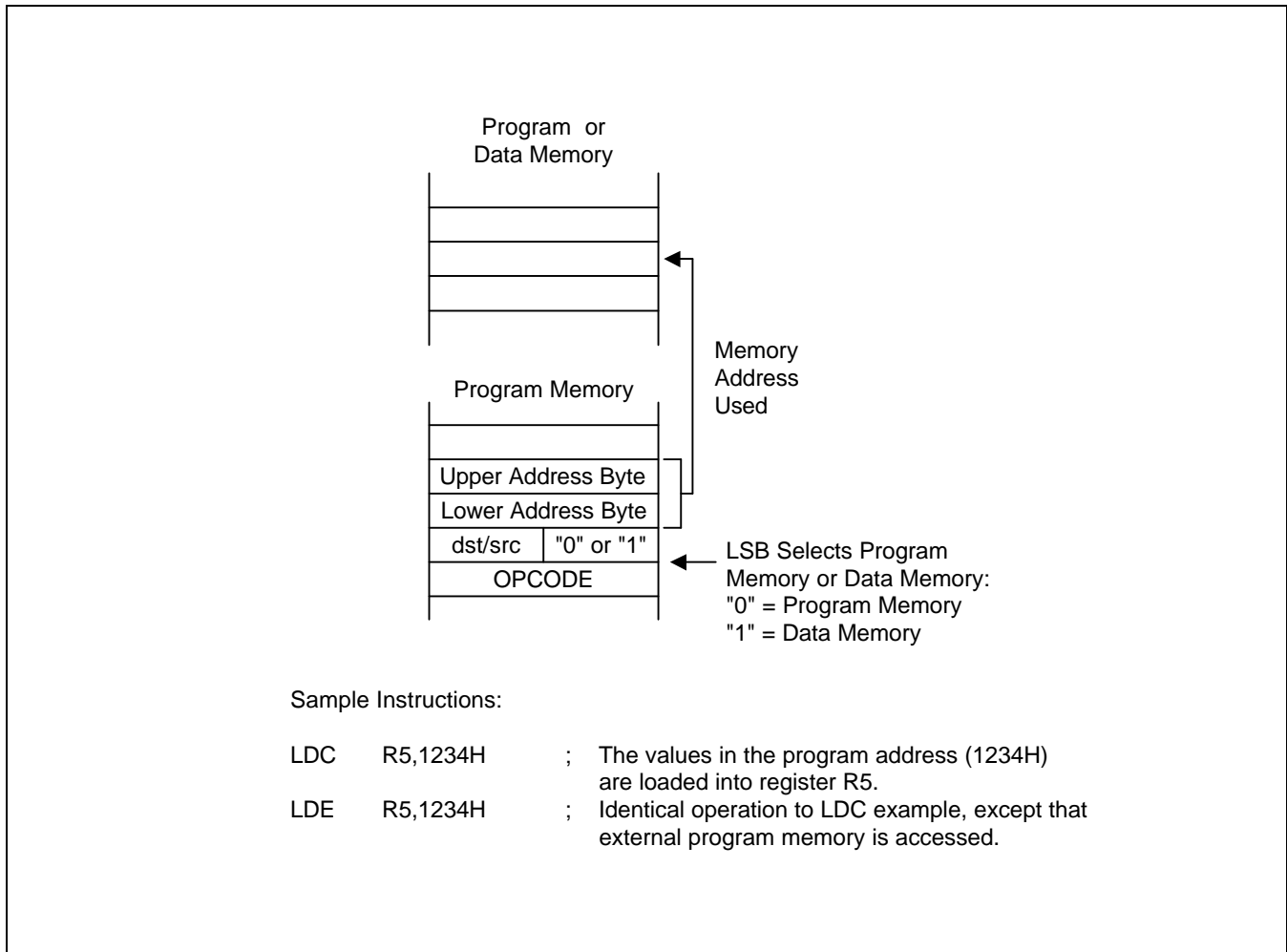
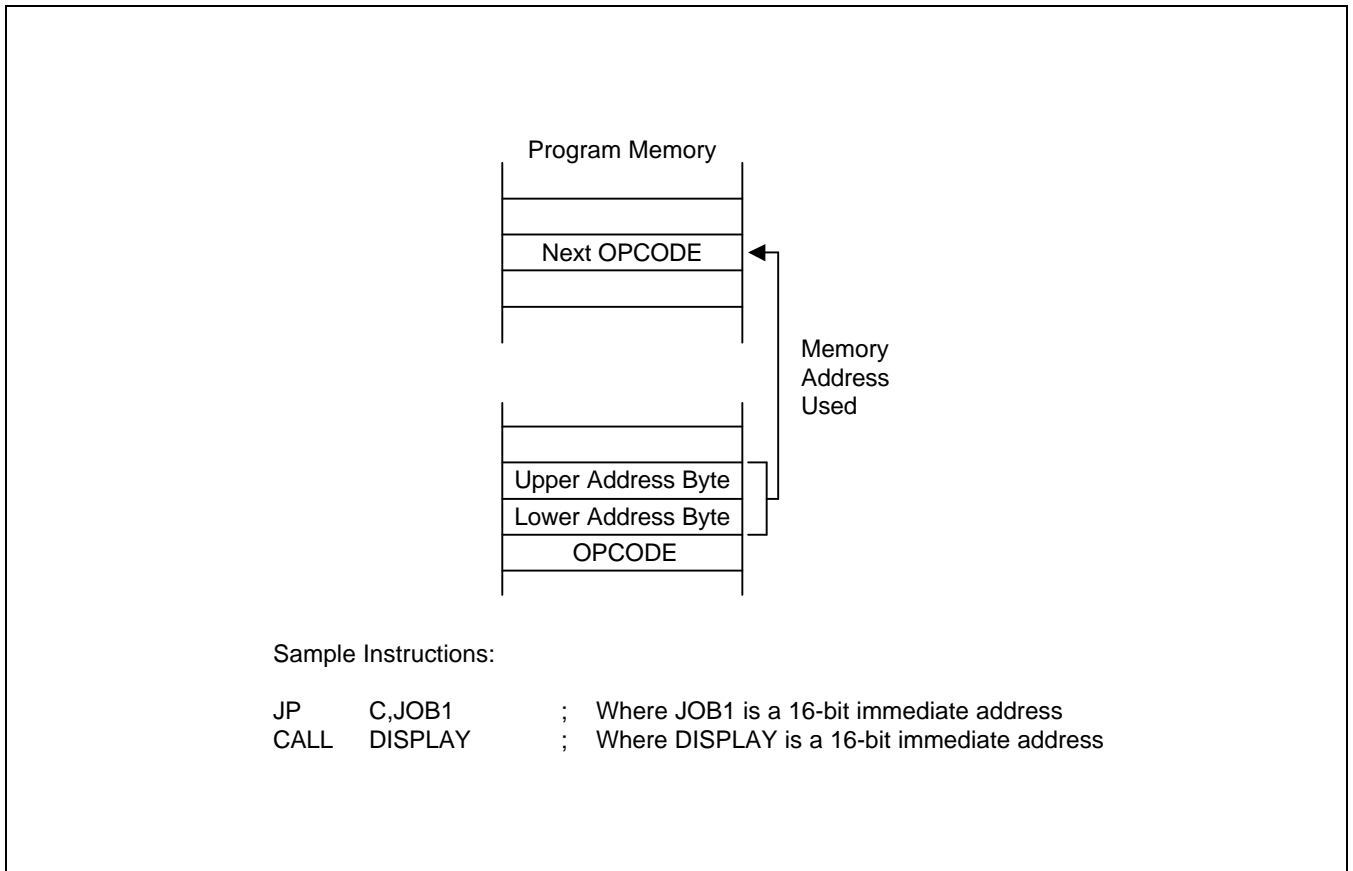


Figure 3-10. Direct Addressing for Load Instructions

## 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.

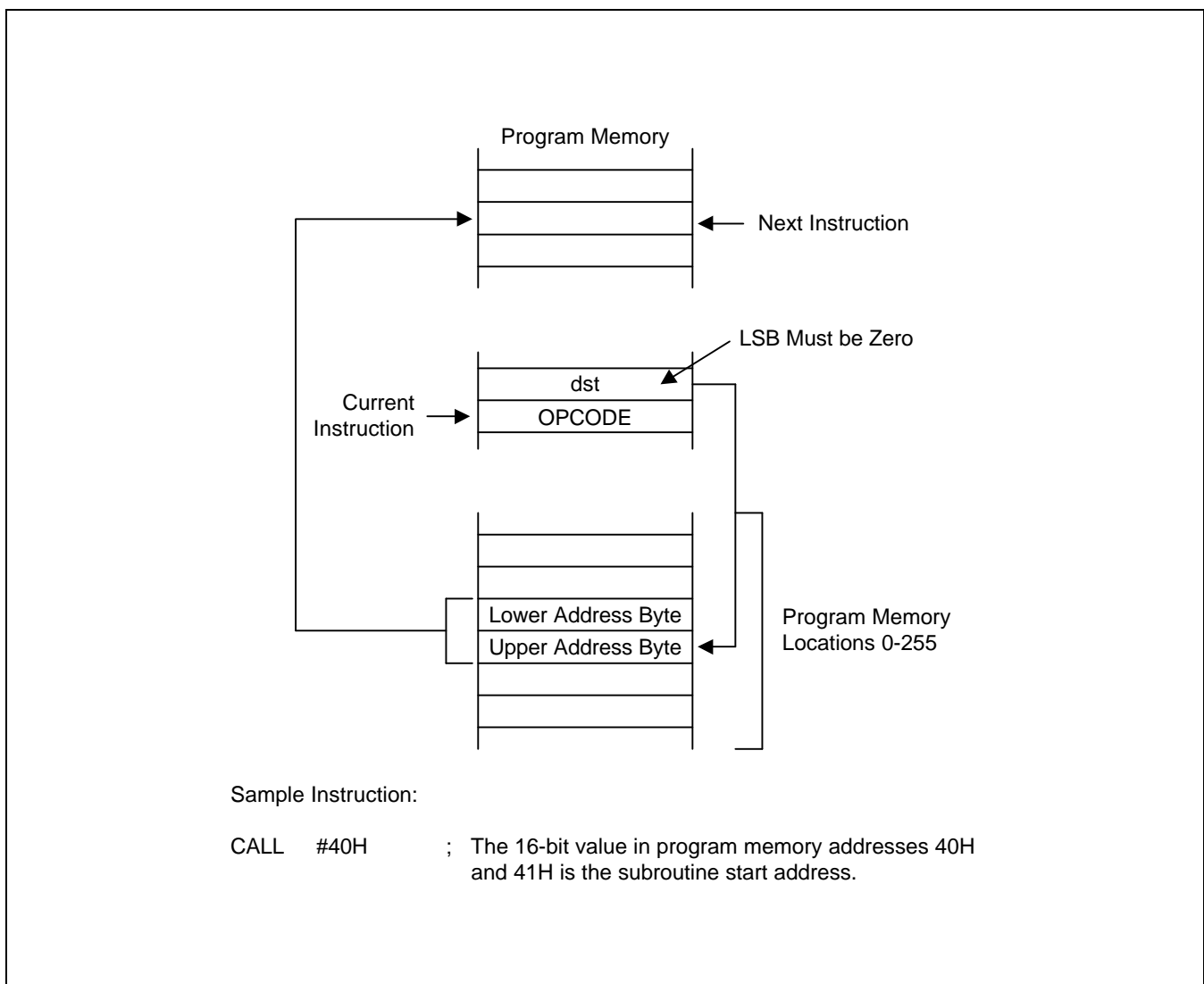


Figure 3-12. Indirect Addressing

## 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.

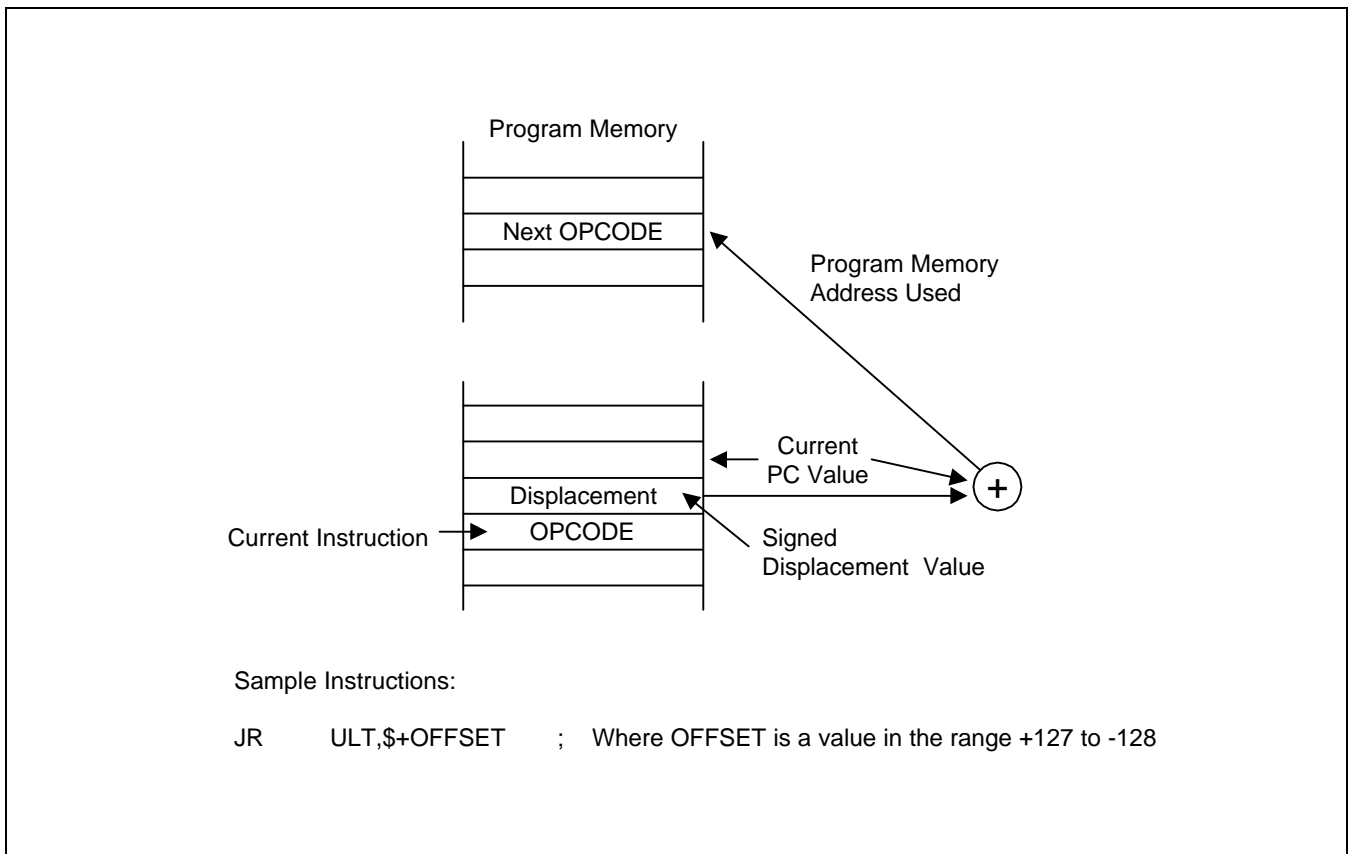
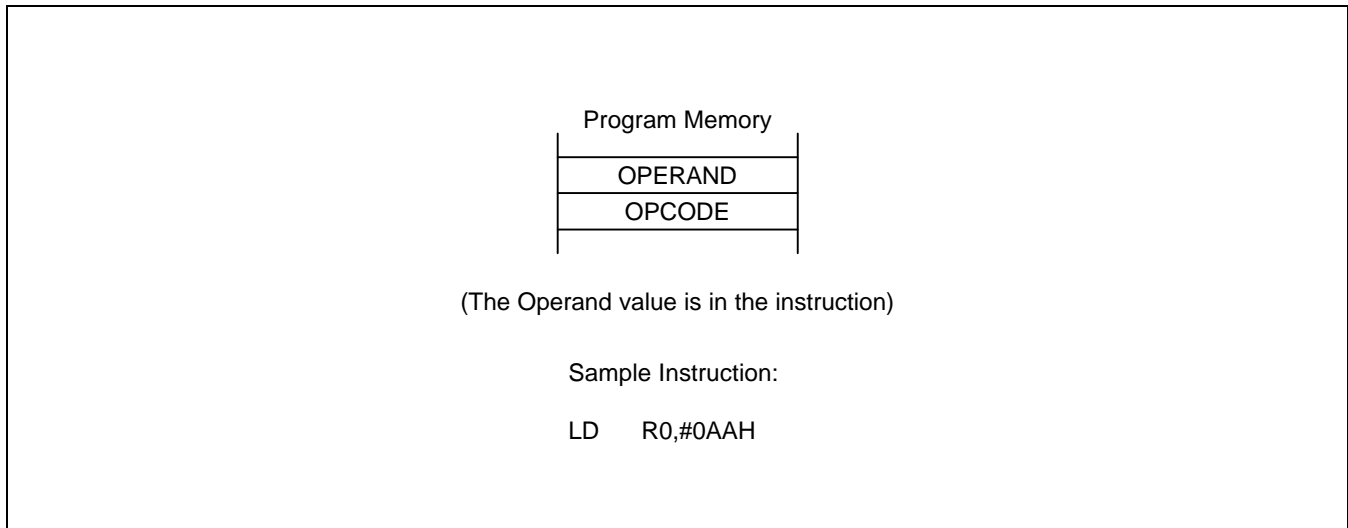


Figure 3-13. Relative Addressing

## IMMEDIATE MODE (IM)

In Immediate (IM) addressing 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**

# 20 ELECTRICAL DATA

## OVERVIEW

In this chapter, S3C830A electrical characteristics are presented in tables and graphs. The information is arranged in the following order:

- Absolute maximum ratings
- D.C. electrical characteristics
- A.C. electrical characteristics
- Input/output capacitance
- Data retention supply voltage in stop mode
- A/D converter electrical characteristics
- PLL electrical characteristics
- Low voltage reset electrical characteristics
- Serial I/O timing characteristics
- Oscillation characteristics
- Oscillation stabilization time

Table 20-1. 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	$V_I$	Ports 0–8	– 0.3 to $V_{DD} + 0.3$	
Output voltage	$V_O$	–	– 0.3 to $V_{DD} + 0.3$	
Output current high	$I_{OH}$	One I/O pin active	– 15	mA
		All I/O pins active	– 60	
Output current low	$I_{OL}$	One I/O pin active	+ 30	
		Total pin current for port	+ 100	
Operating temperature	$T_A$		– 25 to + 85	$^\circ\text{C}$
Storage temperature	$T_{STG}$		– 65 to + 150	

Table 20-2. D.C. Electrical Characteristics

 $(T_A = -25\text{ }^\circ\text{C to } +85\text{ }^\circ\text{C}, V_{DD} = 3.0\text{ V to } 5.5\text{ V})$ 

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Operating voltage	$V_{DD}$	fx = 0.4–4.5 MHz (except PLL/IFC)	3.0	–	5.5	V
		fx = 4.5 MHz (PLL/IFC)	4.5	–	5.5	
Input high voltage	$V_{IH1}$	Ports 0–8	$0.8 V_{DD}$	–	$V_{DD}$	
	$V_{IH2}$	RESET, CE	$0.8 V_{DD}$		$V_{DD}$	
	$V_{IH3}$	$X_{IN}$ , $X_{OUT}$	$V_{DD}-0.1$		$V_{DD}$	
Input low voltage	$V_{IL1}$	Ports 0–8	–	–	$0.2 V_{DD}$	
	$V_{IL2}$	RESET, CE			$0.2 V_{DD}$	
	$V_{IL3}$	$X_{IN}$ , $X_{OUT}$			0.1	
Output high voltage	$V_{OH1}$	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ EO0, EO1; $I_{OH} = -1\text{ mA}$	$V_{DD} - 2.0$	–	$V_{DD}$	
	$V_{OH2}$	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ Other output ports; $I_{OH} = -1\text{ mA}$	$V_{DD} - 1.0$	–	$V_{DD}$	
Output low voltage	$V_{OL1}$	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ EO0, EO1; $I_{OL} = 1\text{ mA}$	–	–	2.0	
	$V_{OL2}$	$V_{DD} = 4.5\text{ V to } 5.5\text{ V}$ Other output ports; $I_{OL} = 10\text{ mA}$	–	–	2.0	

Table 20-2. D.C. Electrical Characteristics (Continued)

(T<sub>A</sub> = -25 °C to + 85 °C, V<sub>DD</sub> = 3.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Input high leakage current	I <sub>LIH1</sub>	V <sub>IN</sub> = V <sub>DD</sub> All input pins except X <sub>IN</sub> , X <sub>OUT</sub>	–	–	3	uA
	I <sub>LIH2</sub>	V <sub>IN</sub> = V <sub>DD</sub> , X <sub>IN</sub> , X <sub>OUT</sub>			20	
Input low leakage current	I <sub>LIL1</sub>	V <sub>IN</sub> = 0 V All input pins except RESET, X <sub>IN</sub> , X <sub>OUT</sub>	–	–	-3	
	I <sub>LIL2</sub>	V <sub>IN</sub> = 0 V, X <sub>IN</sub> , X <sub>OUT</sub>			-20	
Output high leakage current	I <sub>LOH</sub>	V <sub>OUT</sub> = V <sub>DD</sub> All output pins	–	–	3	
Output low leakage current	I <sub>LOL</sub>	V <sub>OUT</sub> = 0 V All output pins	–	–	-3	
Pull-up resistor	R <sub>L1</sub>	V <sub>IN</sub> = 0 V; V <sub>DD</sub> = 5 V Port 0–8	25	50	100	kΩ
	R <sub>L2</sub>	V <sub>IN</sub> = 0 V; V <sub>DD</sub> = 5 V; RESET	150	250	400	
Pull-down resistor	R <sub>D</sub>	V <sub>IN</sub> = V <sub>DD</sub> , V <sub>DD</sub> = 5 V VCOFM, VCOAM, AMIF and FMIF	15	35	45	kΩ
Oscillator feed back resistors	R <sub>OSC</sub>	V <sub>DD</sub> = 5 V, T <sub>A</sub> = 25 °C X <sub>IN</sub> = V <sub>DD</sub> , X <sub>OUT</sub> = 0 V	300	750	1500	kΩ
LCD voltage dividing resistor	R <sub>LCD</sub>	T <sub>A</sub> = 25 °C	70	110	150	kΩ
V <sub>LCD</sub> – COM <sub>i</sub>   voltage drop (i = 0–3)	V <sub>DC</sub>	–15 μA per common pin	–	45	120	mV
V <sub>LCD</sub> – SEG <sub>x</sub>   voltage drop (x = 0–39)	V <sub>DS</sub>	–15 μA per common pin	–	45	120	mV
Middle output voltage	V <sub>LC0</sub>	V <sub>DD</sub> = 3.0 V to 5.5 V	0.6V <sub>DD</sub> – 0.2	0.6V <sub>DD</sub>	0.6V <sub>DD</sub> + 0.2	V
	V <sub>LC1</sub>		0.4V <sub>DD</sub> – 0.2	0.4V <sub>DD</sub>	0.4V <sub>DD</sub> + 0.2	
	V <sub>LC2</sub>		0.2V <sub>DD</sub> – 0.2	0.2V <sub>DD</sub>	0.2V <sub>DD</sub> + 0.2	



Table 20-2. D.C. Electrical Characteristics (Concluded)

(T<sub>A</sub> = -25 °C to + 85 °C, V<sub>DD</sub> = 3.0 V to 5.5 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Supply current <sup>(1)</sup>	I <sub>DD1</sub>	Run mode: 4.5 MHz crystal oscillator CE = V <sub>DD</sub> V <sub>DD</sub> = 5 V ± 10 % C1 = C2 = 22pF	–	5.0	15	mA
	I <sub>DD2</sub>	Run mode: 4.5 MHz crystal oscillator CE = 0 V V <sub>DD</sub> = 5 V ± 10 % C1 = C2 = 22pF	–	2.6	5.5	
	I <sub>DD3</sub>	Idle mode: 4.5 MHz crystal oscillator V <sub>DD</sub> = 5 V ± 10 %	–	0.6	2.0	
	I <sub>DD4</sub> <sup>(2)</sup>	Stop mode (in LVR disable): CE = 0 V, T <sub>A</sub> = 25 °C V <sub>DD</sub> = 5 V ± 10 %	–	0.5	3	μA

**NOTES:**

1. Supply current does not include current drawn through internal pull-up resistors, PWM, or external output current loads.
2. I<sub>DD4</sub> is current when the main clock oscillation stops.
3. Every values in this table is measured when bits 4–3 of the system clock control register (CLKCON.4–3) is set to 11B.

Table 20-3. A.C. Electrical Characteristics

( $T_A = -25\text{ }^\circ\text{C}$  to  $+85\text{ }^\circ\text{C}$ ,  $V_{DD} = 3.0\text{ V}$  to  $5.5\text{ V}$ )

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Interrupt input high, low width (P1.0–P1.7)	tINTH, tINTL	P1.0–P1.7, $V_{DD} = 5\text{ V}$	200	–		ns
RESET input low width	tRSL	$V_{DD} = 5\text{ V}$	10	–	–	us

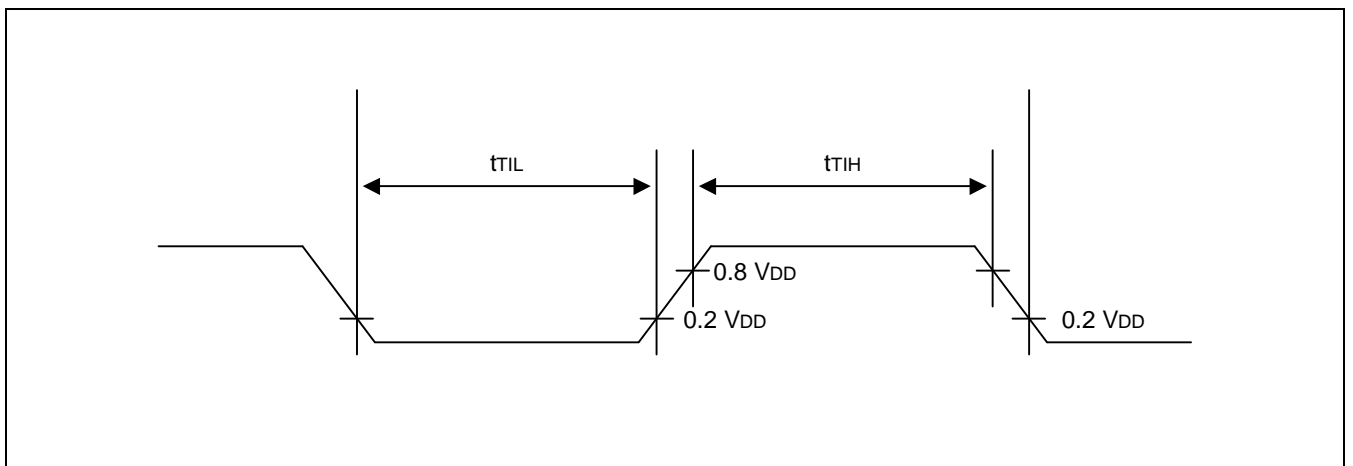


Figure 20-1. Input Timing for External Interrupts (Ports 1)

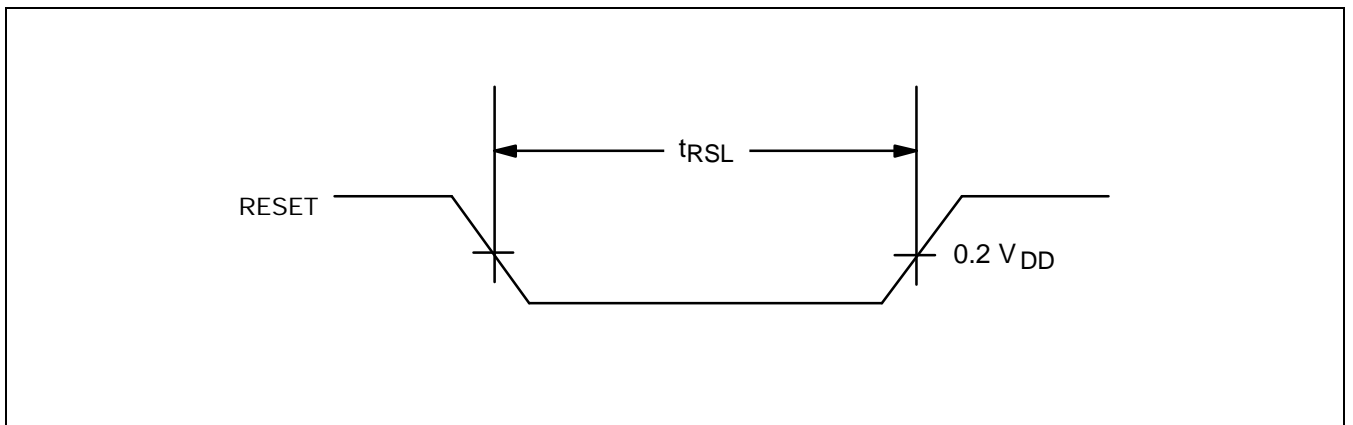


Figure 20-2. Input Timing for RESET

Table 20-4. Input/Output Capacitance

(T<sub>A</sub> = -25 °C to +85 °C, V<sub>DD</sub> = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Input capacitance	C <sub>IN</sub>	f = 1 MHz; unmeasured pins are returned to V <sub>SS</sub>	-	-	10	pF
Output capacitance	C <sub>OUT</sub>					
I/O capacitance	C <sub>IO</sub>					

Table 20-5. Data Retention Supply Voltage in Stop Mode

(T<sub>A</sub> = -25 °C to +85 °C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Data retention supply voltage	V <sub>DDDR</sub>		3.0	-	5.5	V
Data retention supply current	I <sub>DDDR</sub>	V <sub>DDDR</sub> = 2 V (T <sub>A</sub> = 25 °C) Stop mode (in LVR disable)	-	-	1	uA

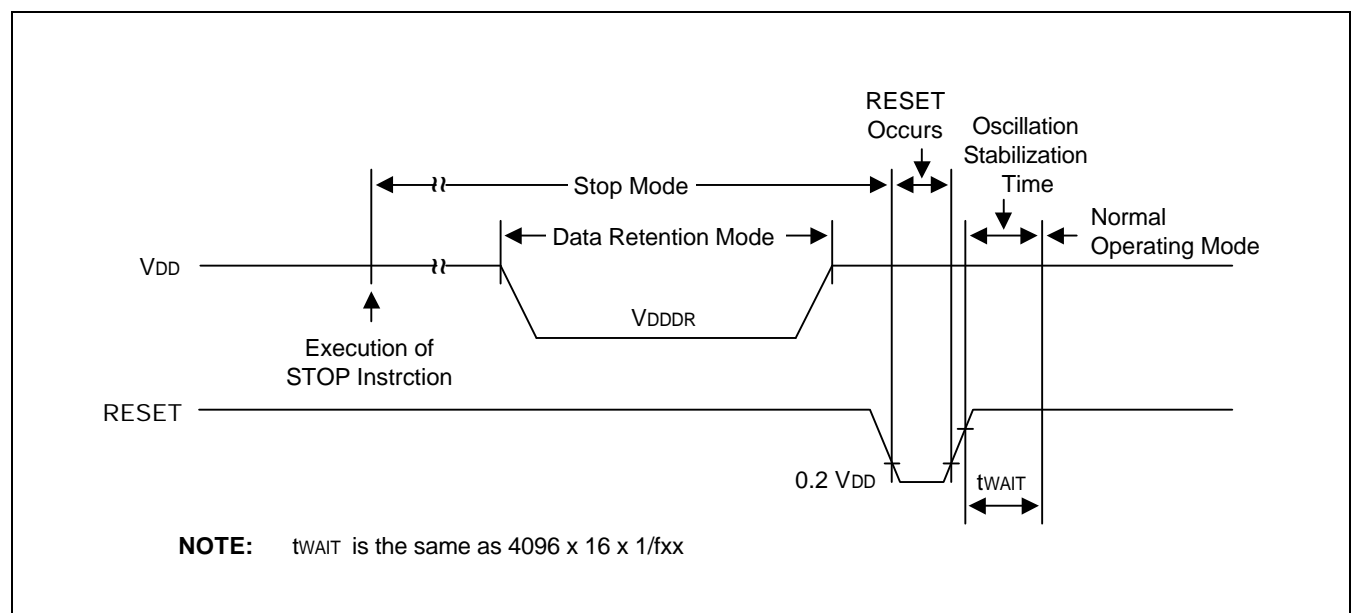


Figure 20-3. Stop Mode Release Timing Initiated by RESET

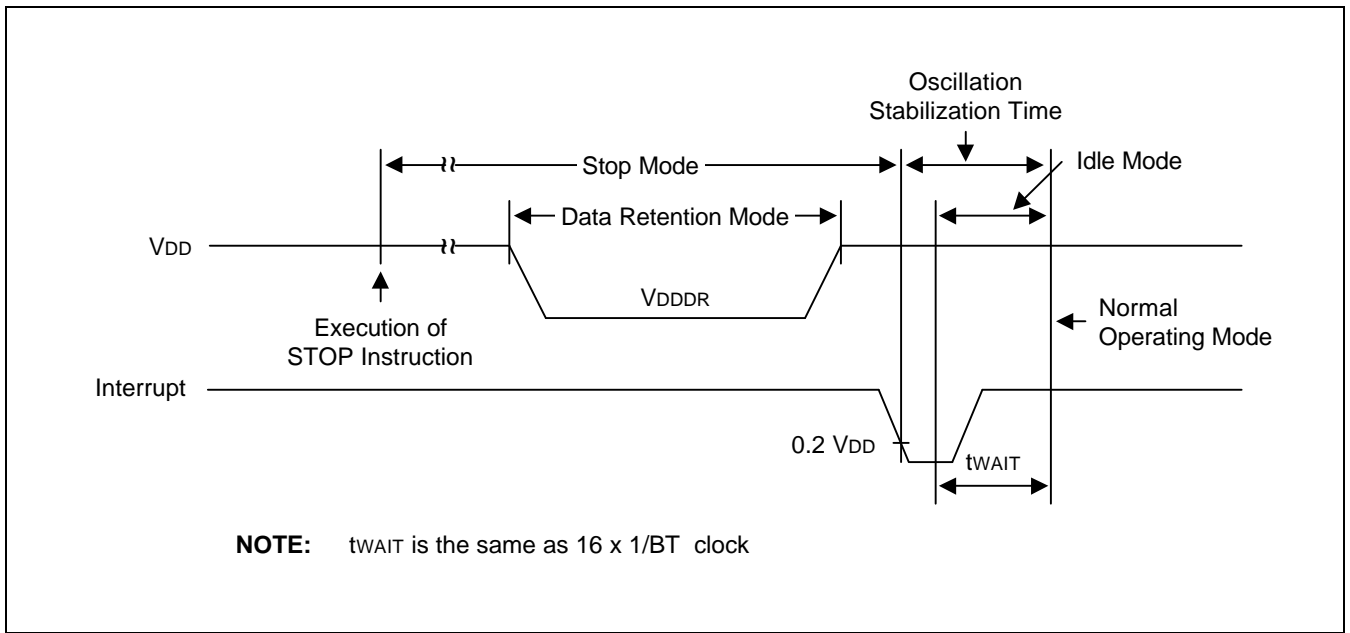


Figure 20-4. Stop Mode Release Timing Initiated by Interrupts

Table 20-6. A/D Converter Electrical Characteristics

(T<sub>A</sub> = -25 °C to +85 °C, V<sub>DD</sub> = 3.5 V to 5.5 V, V<sub>SS</sub> = 0 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
A/D converting resolution	–	–	–	8	–	bits
Absolute accuracy	–	–	–	–	±2	LSB
A/D conversion time	t <sub>CON</sub>	Conversion clock = f <sub>xx</sub>	50/f <sub>xx</sub>	–	–	μs
Analog input voltage	V <sub>IAN</sub>	–	V <sub>SS</sub>	–	V <sub>DD</sub>	V
Analog input impedance	R <sub>AN</sub>	V <sub>DD</sub> = 5 V	2	1000	–	MΩ

Table 20-7. PLL Electrical Characteristics

(T<sub>A</sub> = -25 °C to +85 °C, V<sub>DD</sub> = 4.5 V to 5.5 V)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
VCOFM, VCOAM, FMIF and AMIF input voltage (peak to peak)	V <sub>IN</sub>	Sine wave input	0.3	–	V <sub>DD</sub>	V
Frequency	fV <sub>COAM</sub>	VCOAM mode, sine wave input; V <sub>IN</sub> = 0.3V <sub>P-P</sub>	0.5	–	30	MHz
	fV <sub>COFM</sub>	VCOFM mode, sine wave input; V <sub>IN</sub> = 0.3V <sub>P-P</sub>	30		150	
	f <sub>AMIF</sub>	AMIF mode, sine wave input; V <sub>IN</sub> = 0.3V <sub>P-P</sub>	0.1		1.0	
	f <sub>FMIF</sub>	FMIF mode, sine wave input; V <sub>IN</sub> = 0.3V <sub>P-P</sub>	5		15	

Table 20-8. Low Voltage Reset Electrical Characteristics

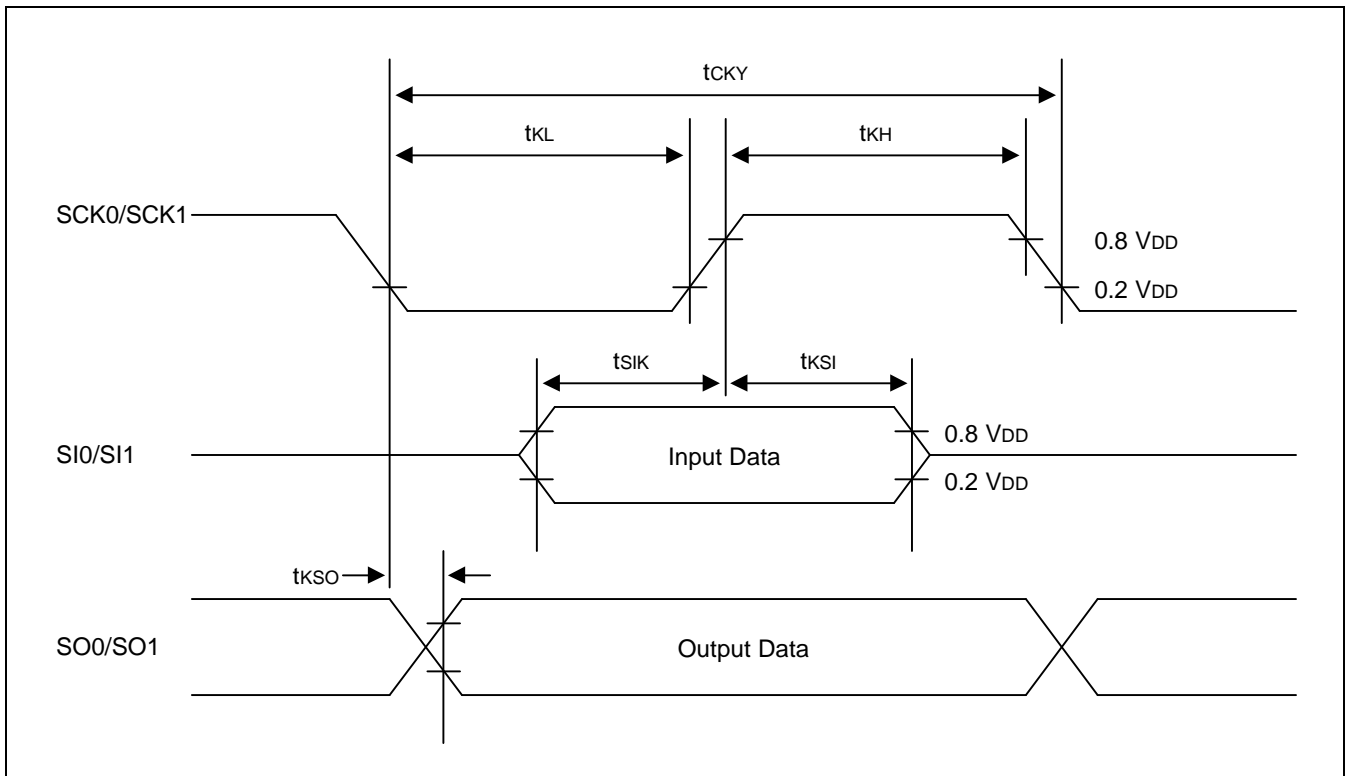
(T<sub>A</sub> = -25 °C to +85 °C)

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
Detect voltage range	V <sub>DET</sub>		3.0	3.5	4.0	V
LVR operating current	I <sub>BL</sub>	–	–	10	25	μA

**Table 20-9. Synchronous SIO Electrical Characteristics**

( $T_A = -25\text{ }^\circ\text{C}$  to  $+85\text{ }^\circ\text{C}$ ,  $V_{DD} = 3.0\text{ V}$  to  $5.5\text{ V}$ )

Parameter	Symbol	Conditions	Min	Typ	Max	Unit
SCK0/SCK1 cycle time	$t_{CKY}$	External SCK0/SCK1 source	1000	-	-	ns
		Internal SCK0/SCK1 source	1000			
SCK0/SCK1 high, low width	$t_{KH}, t_{KL}$	External SCK0/SCK1 source	500	-	-	
		Internal SCK0/SCK1 source	$t_{CKY}/2 - 50$			
SI setup time to SCK0/SCK1 high	$t_{SIK}$	External SCK0/SCK1 source	250	-	-	
		Internal SCK0/SCK1 source	250			
SI hold time to SCK0/SCK1 high	$t_{KSI}$	External SCK0/SCK1 source	400	-	-	
		Internal SCK0/SCK1 source	400			
Output delay for SCK0/SCK1 to SO	$t_{KSO}$	External SCK0/SCK1 source	-	-	300	
		Internal SCK0/SCK1 source	-		250	



**Figure 20-5. Serial Data Transfer Timing**

Table 20-10. Main Oscillator Characteristics (fx)

(T<sub>A</sub> = -25 °C to +85 °C, V<sub>DD</sub> = 3.0 V to 5.5 V)

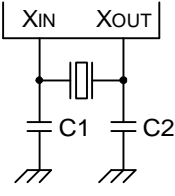
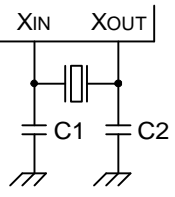
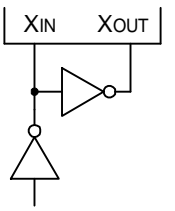
Oscillator	Clock Circuit	Test Condition	Min	Typ	Max	Unit
Crystal		Crystal oscillation frequency	0.4	–	4.5	MHz
Ceramic		Ceramic oscillation frequency	0.4	–	4.5	MHz
External clock		X <sub>IN</sub> input frequency	0.4	–	4.5	MHz

Table 20-11. Main Oscillator Clock Stabilization Time (t<sub>ST1</sub>)(T<sub>A</sub> = -25 °C to +85 °C, V<sub>DD</sub> = 3.0 V to 5.5 V)

Oscillator	Test Condition	Min	Typ	Max	Unit
Crystal	V <sub>DD</sub> = 4.5 V to 5.5 V	–	–	10	ms
Ceramic	Stabilization occurs when V <sub>DD</sub> is equal to the minimum oscillator voltage range.	–	–	4	ms
External clock	X <sub>IN</sub> input high and low level width (t <sub>XH</sub> , t <sub>XL</sub> )	111	–	1250	ns

**NOTE:** Oscillation stabilization time (t<sub>ST1</sub>) is the time required for the CPU clock to return to its normal oscillation frequency after a power-on occurs, or when Stop mode is ended by a RESET signal. The RESET should therefore be held at low level until the t<sub>ST1</sub> time has elapsed

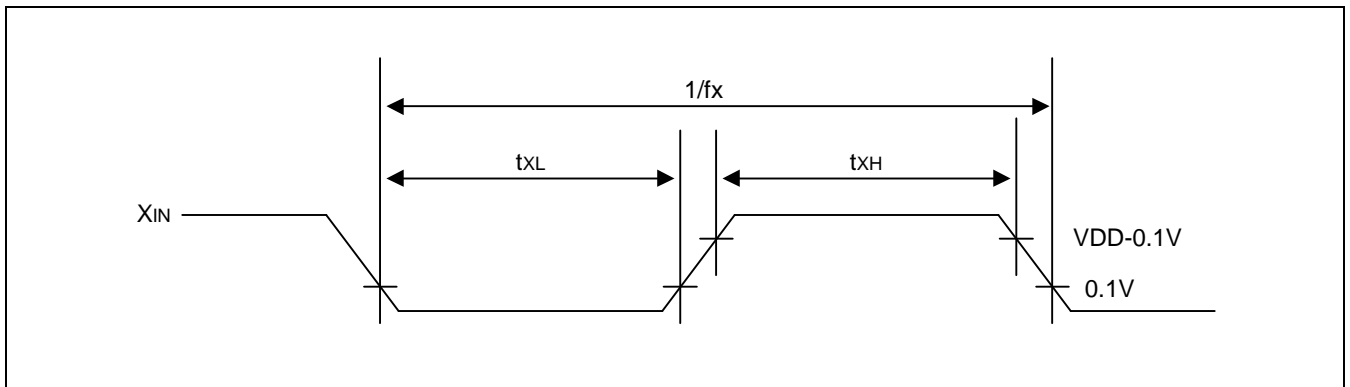


Figure 20-6. Clock Timing Measurement at X<sub>IN</sub>

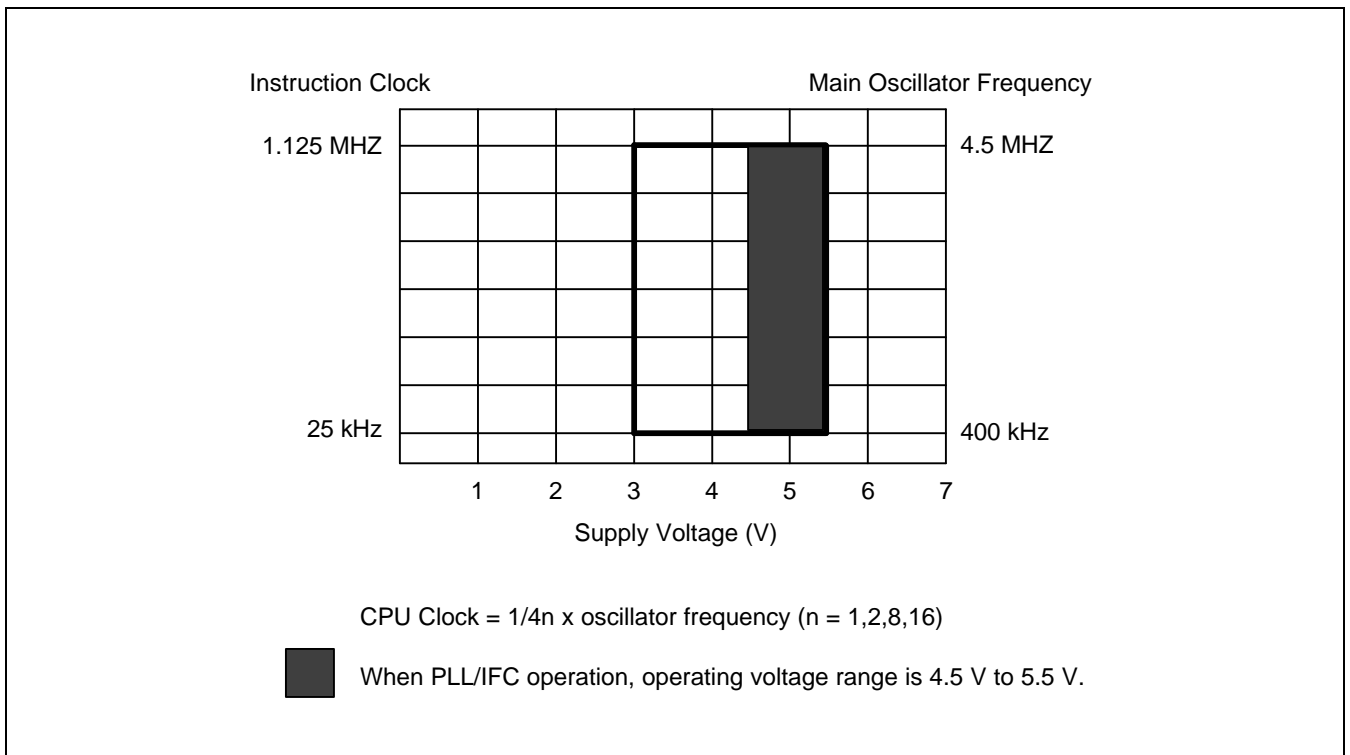


Figure 20-7. Operating Voltage Range



**NOTES**

# 21 MECHANICAL DATA

## OVERVIEW

The S3C830A microcontroller is currently available in 100-pin-QFP package.

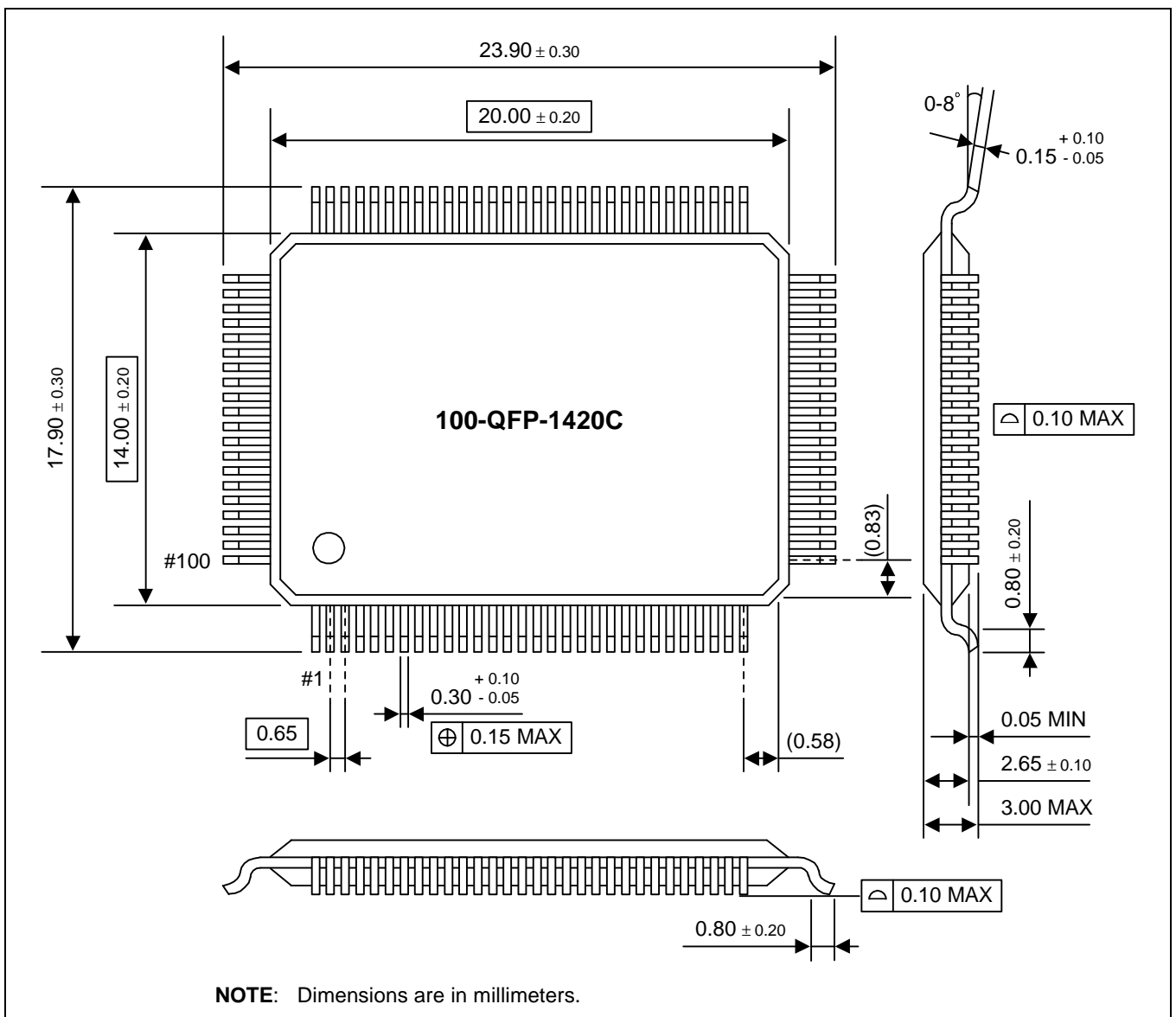


Figure 21-1. Package Dimensions (100-QFP-1420C)

NOTES

# 22

## S3P830A OTP

### OVERVIEW

The S3P830A single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the S3C830A microcontroller. It has an on-chip OTP ROM instead of a masked ROM. The EPROM is accessed by serial data format.

The S3P830A is fully compatible with the S3C830A, both in function in D.C. electrical characteristics and in pin configuration. Because of its simple programming requirements, the S3P830A is ideal as an evaluation chip for the S3C830A.

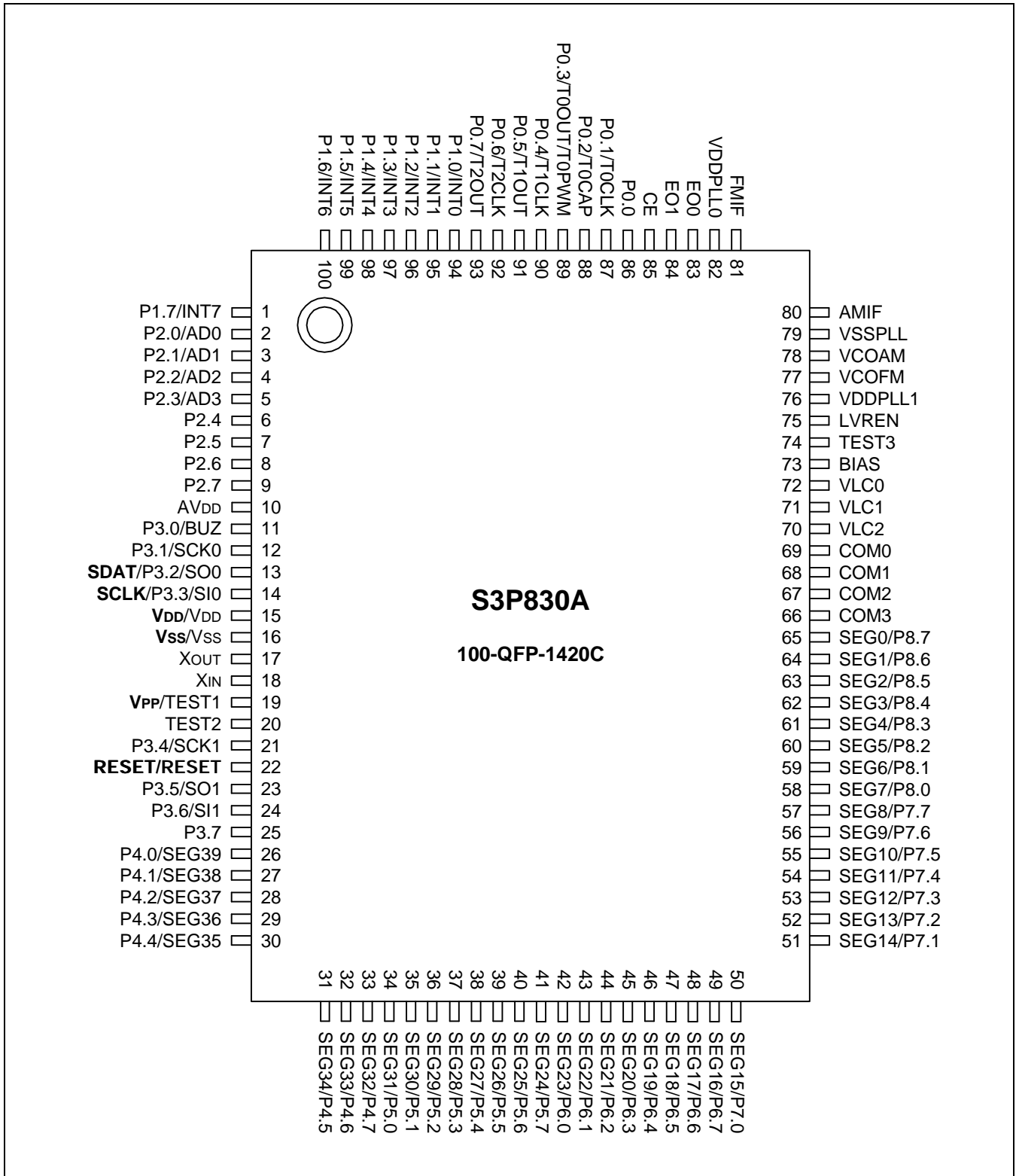


Figure 22-1. S3P830A Pin Assignments (100-Pin QFP Package)

Table 22-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip Pin Name	During Programming			
	Pin Name	Pin No.	I/O	Function
P3.2/SO0	SDAT	13	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input/push-pull output port.
P3.3/SI0	SCLK	14	I	Serial clock pin. Input only pin.
TEST1	V <sub>PP</sub>	19	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	22	I	Chip Initialization
V <sub>DD</sub> /V <sub>SS</sub>	V <sub>DD</sub> /V <sub>SS</sub>	15/16	–	Logic power supply pin. VDD should be tied to +5 V during programming.

Table 22-2. Comparison of S3P830A and S3C830A Features

Characteristic	S3P830A	S3C830A
Program Memory	48-Kbyte EPROM	48-Kbyte mask ROM
Operating Voltage (V <sub>DD</sub> )	3.0 V to 5.5 V	3.0 V to 5.5 V
OTP Programming Mode	V <sub>DD</sub> = 5 V, V <sub>PP</sub> (TEST1) = 12.5 V	
Pin Configuration	100 QFP	100 QFP
EPROM Programmability	User Program 1 time	Programmed at the factory

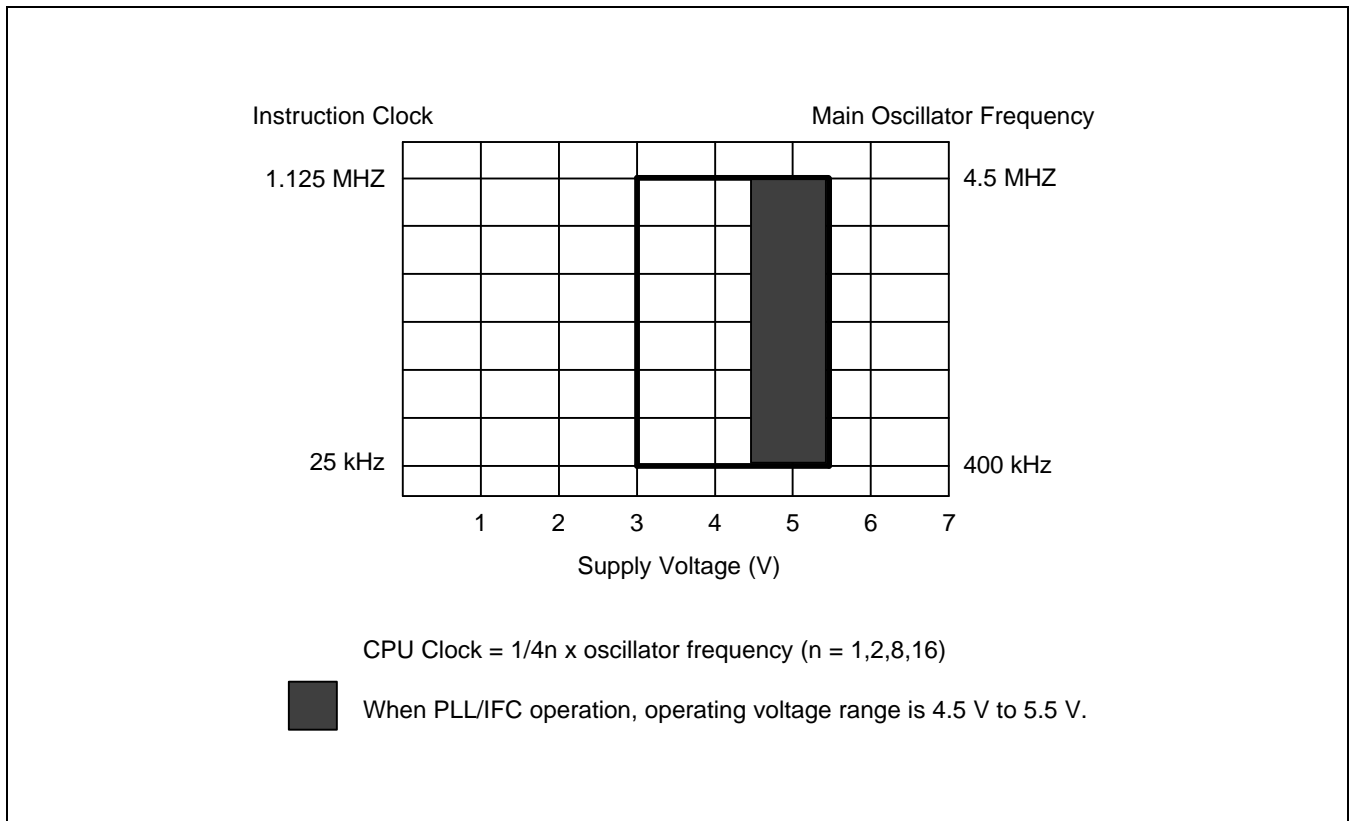
## OPERATING MODE CHARACTERISTICS

When 12.5 V is supplied to the V<sub>PP</sub> (TEST1) pin of the S3P830A, 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 21-3 below.

Table 22-3. Operating Mode Selection Criteria

VDD	VPP (TEST1)	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.

**Figure 22-2. Operating Voltage Range**