Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	SAM88RC
Core Size	8-Bit
Speed	12MHz
Connectivity	SIO, UART/USART
Peripherals	LCD, LVR, PWM, WDT
Number of I/O	54
Program Memory Size	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-BQFP
Supplier Device Package	80-QFP
Purchase URL	https://www.e-xfl.com/product-detail/zilog/s3f8s6bxzz-qt8b

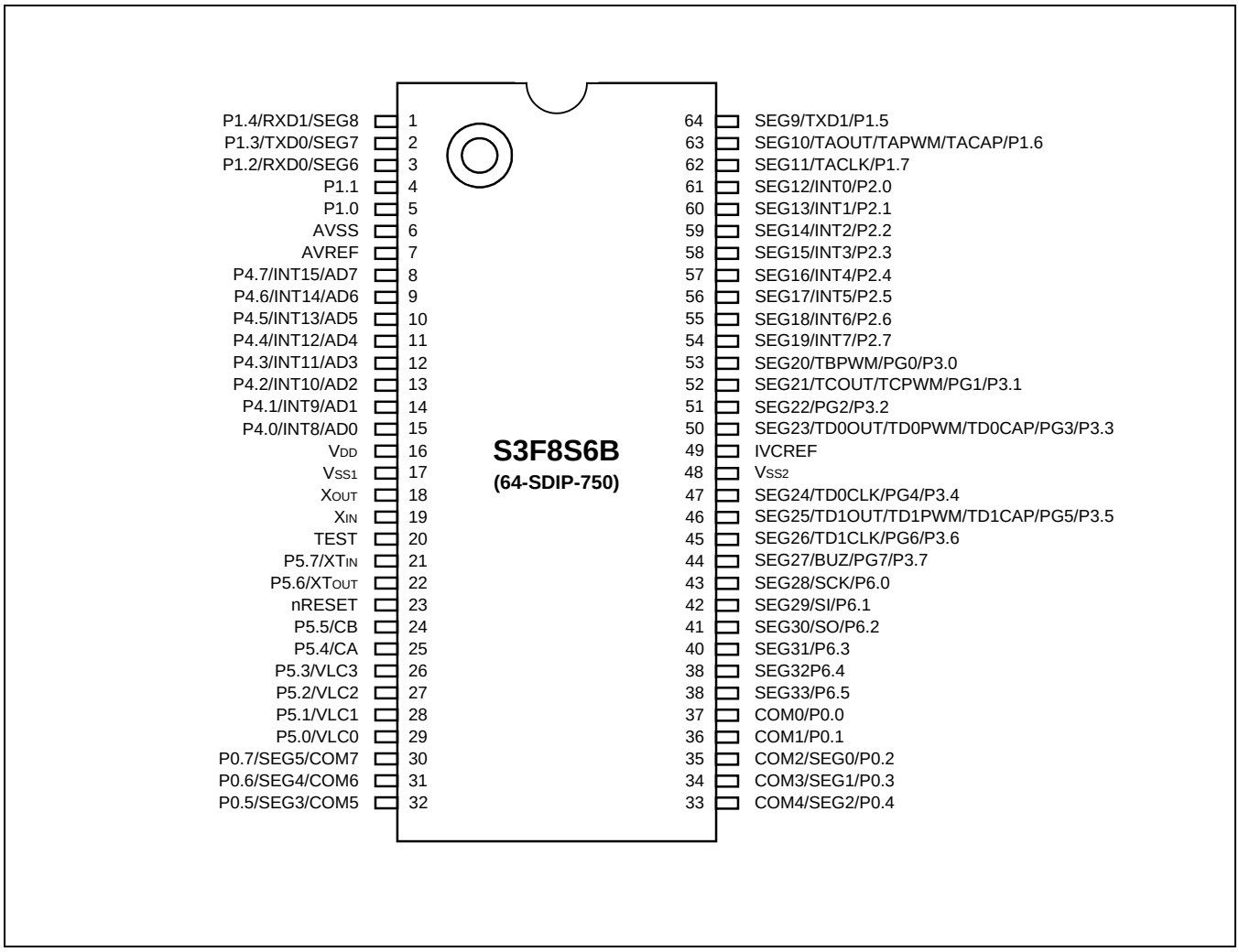


Figure 1-4 S3F8S6B Pin Assignments (64-SDIP-750)

1.7 Pin Circuits

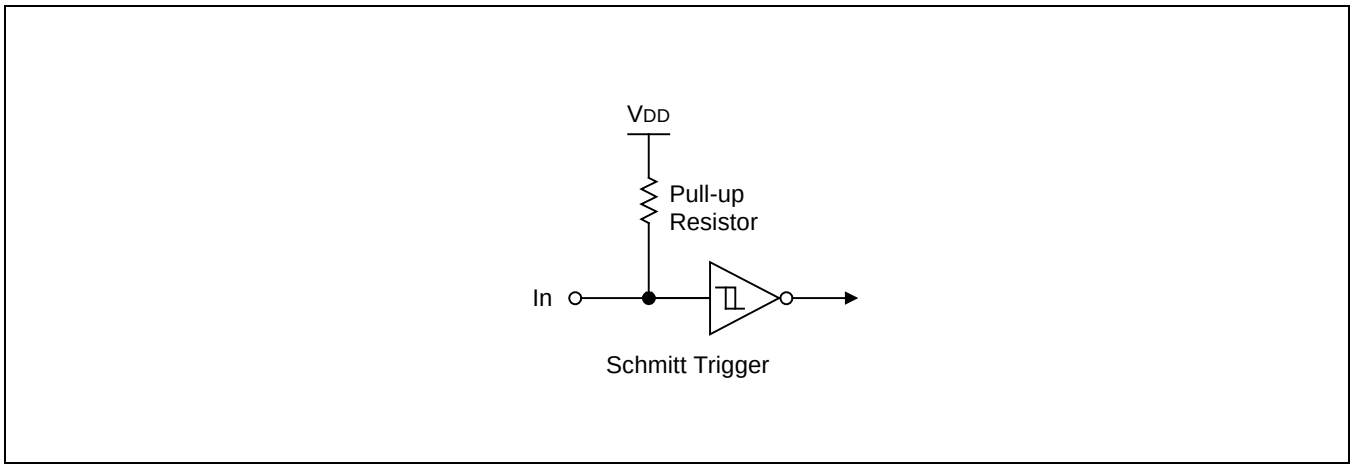


Figure 1-5 Pin Circuit Type B

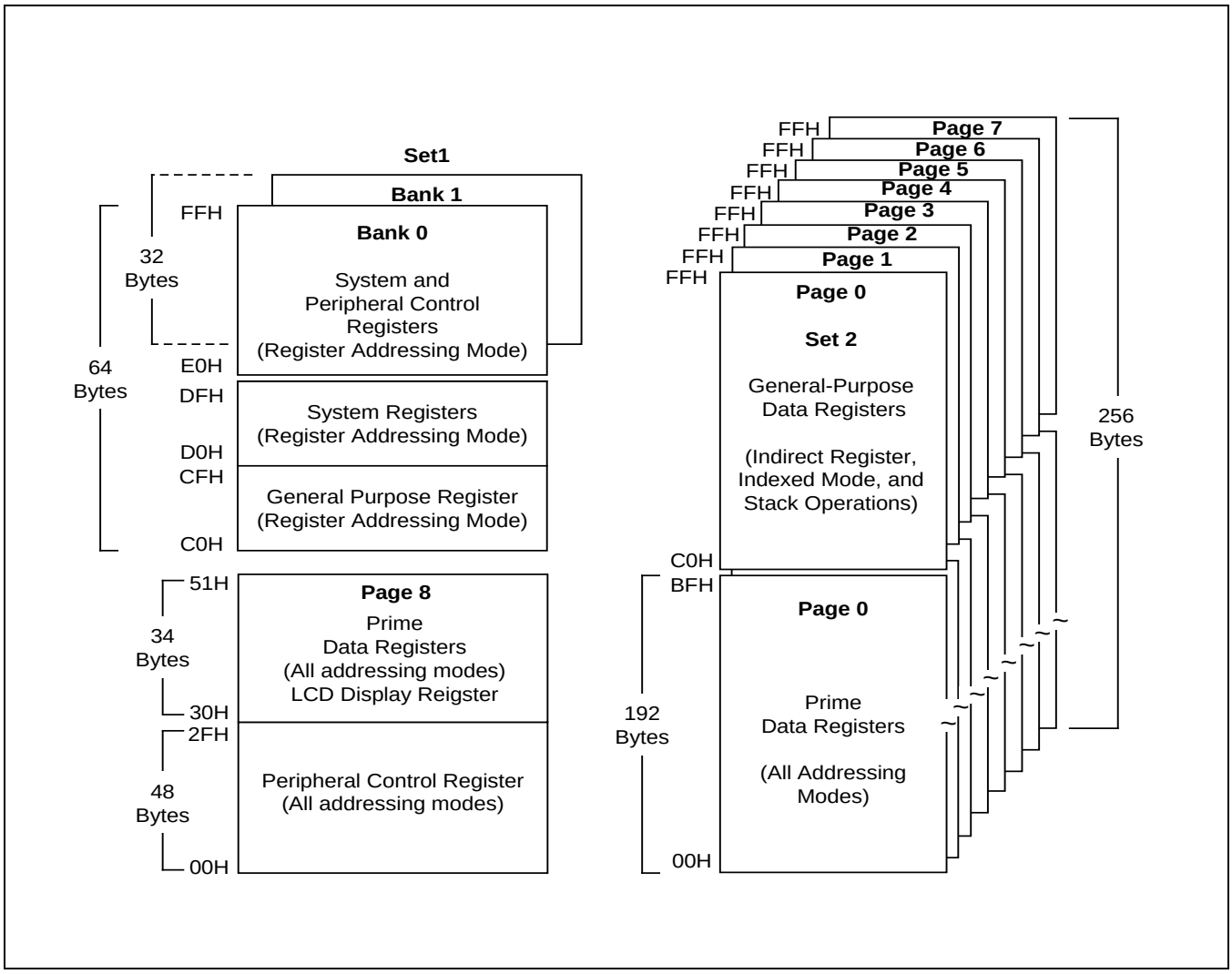


Figure 2-3 Internal Register File Organization (S3F8S6B)

4.1.19 P0CONL: Port 0 Control Register-Low Byte (E1H, 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	RW	RW	RW	RW	RW	RW	RW	RW
Addressing Mode	Register addressing mode only							

.7–.6 P0.3/SEG1/COM3 Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Alternative function (LCD signal)
1	1	Output mode, push-pull

.5–.4 P0.2/SEG0/COM2 Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Alternative function (LCD signal)
1	1	Output mode, push-pull

.3–.2 P0.1/COM1 Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Alternative function (LCD signal)
1	1	Output mode, push-pull

.1–.0 P0.0/COM0 Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Alternative function (LCD signal)
1	1	Output mode, push-pull

4.1.39 P5CONH: Port 5 Control Register-High Byte (F0H, 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	RW	RW	RW	RW	RW	RW	RW	RW
Addressing Mode	Register addressing mode only							

.7–.6 P5.7/XTIN Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Alternative function (XT_{IN})
1	1	Output mode, push-pull

.5–.4 P5.6/XTOUT Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Alternative function (XT_{OUT})
1	1	Output mode, push-pull

.3–.2 P5.5/CB (NOTE) Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Not available
1	1	Output mode, push-pull

.1–.0 P5.4/CA (NOTE) Configuration Bits

0	0	Input mode
0	1	Input mode, pull-up
1	0	Not available
1	1	Output mode, push-pull

NOTE: Refer to LCON register in Chapter 15

6.3.17 CP - Compare

CP dst,src

Operation: dst – src

The source operand is compared to (subtracted from) the destination operand, and the appropriate flags are set accordingly. The contents of both operands are unaffected by the comparison.

Flags:

C: Set if a "borrow" occurred (src > dst); cleared otherwise.

Z: Set if the result is "0"; cleared otherwise.

S: Set if the result is negative; cleared otherwise.

V: Set if arithmetic overflow occurred; cleared otherwise.

D: Unaffected.

H: Unaffected.

Format:

	Bytes	Cycles	Opcode (Hex)	Addr Mode dst	Addr Mode src
opc | dst | src	2	4	A2	r	r
		6	A3	r	lr
opc | src | dst	3	6	A4	R	R
		6	A5	R	IR
opc | dst | src	3	6	A6	R	IM

Examples: 1. Given: R1 = 02H and R2 = 03H:

```
CP      R1,R2   →       Set the C and S flags
```

Destination working register R1 contains the value 02H and source register R2 contains the value 03H. The statement "CP R1,R2" subtracts the R2 value (source/subtrahend) from the R1 value (destination/minuend). Because a "borrow" occurs and the difference is negative, C and S are "1".

2. Given: R1 = 05H and R2 = 0AH:

```
CP      R1,R2
JP      UGE,SKIP
INC     R1
SKIP    LD      R3,R1
```

In this example, destination working register R1 contains the value 05H which is less than the contents of the source working register R2 (0AH). The statement "CP R1,R2" generates C = "1" and the JP instruction does not jump to the SKIP location. After the statement "LD R3,R1" executes, the value 06H remains in working register R3.

6.3.32 IRET - Interrupt Return

IRET IRET (Normal) IRET (Fast)

Operation:

FLAGS ← @SP
SP ← SP + 1
PC ← @SP
SP ← SP + 2
SYM(0) ← 1

PC ↔ IP
FLAGS ← FLAGS'
FIS ← 0

This instruction is used at the end of an interrupt service routine. It restores the flag register and the program counter. It also re-enables global interrupts. A "normal IRET" is executed only if the fast interrupt status bit (FIS, bit one of the FLAGS register, 0D5H) is cleared (= "0"). If a fast interrupt occurred, IRET clears the FIS bit that was set at the beginning of the service routine.

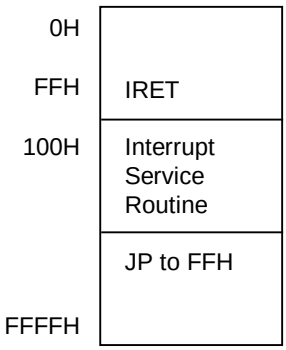
Flags: All flags are restored to their original settings (that is, the settings before the interrupt occurred).

Format:

IRET (Normal)	Bytes	Cycles	Opcode (Hex)
opc	1	10 (internal stack) 12 (internal stack)	BF

IRET (Fast)	Bytes	Cycles	Opcode (Hex)
opc	1	6	BF

Example: In the figure below, the instruction pointer is initially loaded with 100H in the main program before interrupts are enabled. When an interrupt occurs, the program counter and instruction pointer are swapped. This causes the PC to jump to address 100H and the IP to keep the return address. The last instruction in the service routine normally is a jump to IRET at address FFH. This causes the instruction pointer to be loaded with 100H "again" and the program counter to jump back to the main program. Now, the next interrupt can occur and the IP is still correct at 100H.



NOTE: In the fast interrupt example above, if the last instruction is not a jump to IRET, you must pay attention to the order of the last two instructions. The IRET cannot be immediately proceeded by a clearing of the interrupt status (as with a reset of the IPR register).

6.3.70 WFI - Wait for Interrupt

WFI

Operation:

The CPU is effectively halted until an interrupt occurs, except that DMA transfers can still take place during this wait state. The WFI status can be released by an internal interrupt, including a fast interrupt .

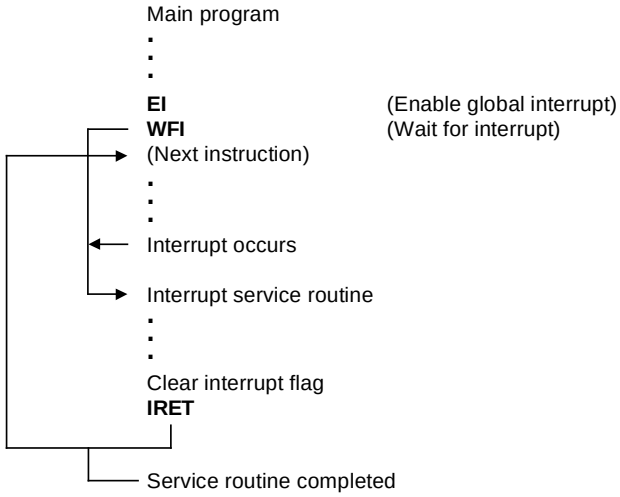
Flags: No flags are affected.

Format:

	Bytes	Cycles	Opcode (Hex)
opc	1	4n	3F

NOTE: (n = 1, 2, 3, …)

Example: The following sample program structure shows the sequence of operations that follow a "WFI" statement:



9.2.1 Port 0

Port 0 is an 8-bit I/O port with individually configurable pins. Port 0 pins are accessed directly by writing or reading the port 0 data register, P0 at location 00H in page 8. P0.0–P0.7 can serve as inputs (with or without pull-ups), and push-pull outputs. And, they can serve as segment pins for LCD also.

9.2.1.1 Port 0 Control Register (P0CONH, P0CONL)

Port 0 has two 8-bit control registers: P0CONH for P0.4-P0.7 and P0CONL for P0.0-P0.3. A reset clears the P0CONH and P0CONL registers to "00H", configuring all pins to input mode. You use control registers settings to select input (with or without pull-ups) or push-pull output mode and enable the alternative functions.

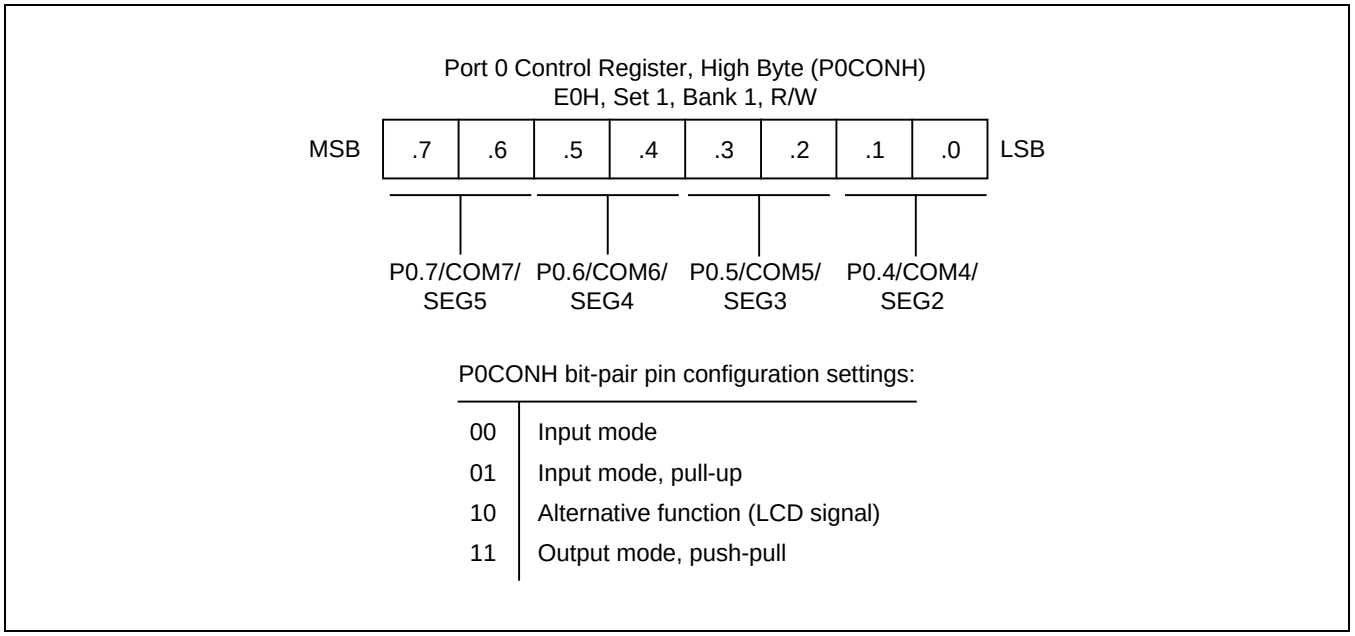
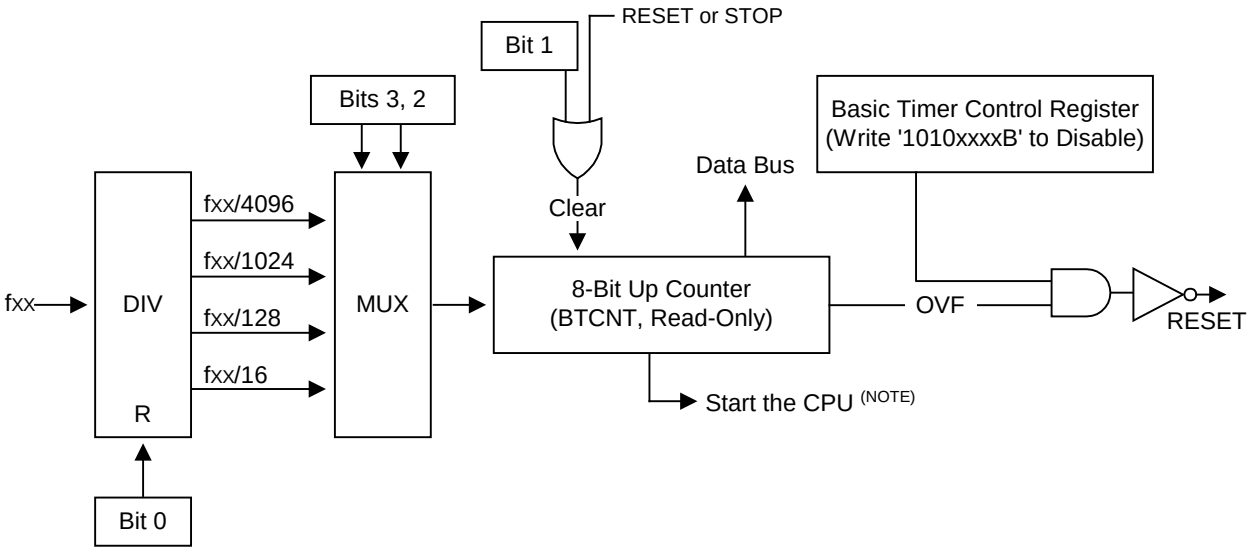


Figure 9-2 Port 0 High-Byte Control Register (P0CONH)



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

Figure 10-2 Basic Timer Block Diagram

11.1.3.4 Capture Mode

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

Timer A also gives you capture input source: the signal edge at the TACAP pin. You select the capture input by setting the values of the timer A capture input selection bits in the port 1 control register, P1CONH.5–.4, (set 1, bank 1, E2H). When P1CONH.5–.4 is "00" the TACAP input is selected.

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

By reading the captured data value in TADATA, and assuming a specific value for the timer A clock frequency, you can calculate the pulse width (duration) of the signal that is being input at the TACAP pin (see *Figure 11-4*).

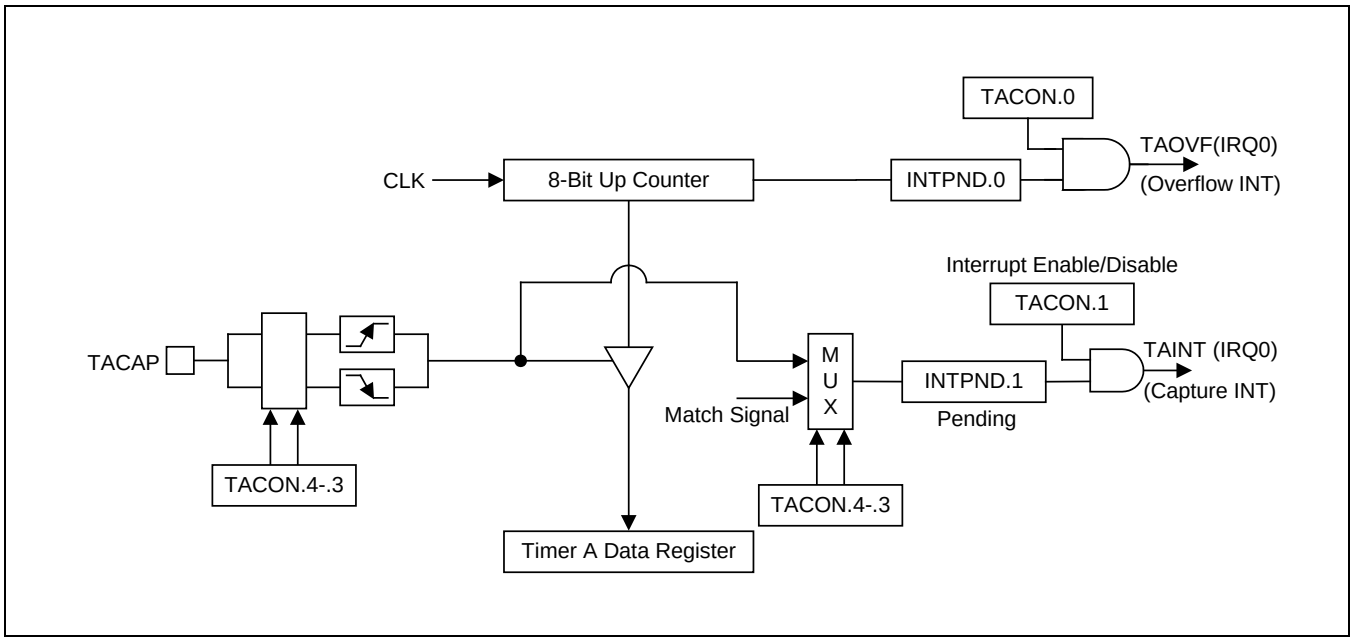


Figure 11-4 Simplified Timer A Function Diagram: Capture Mode

12 8-Bit Timer C

12.1 8-Bit Timer C

12.1.1 Overview

The 8-bit timer C is an 8-bit general-purpose timer/counter.

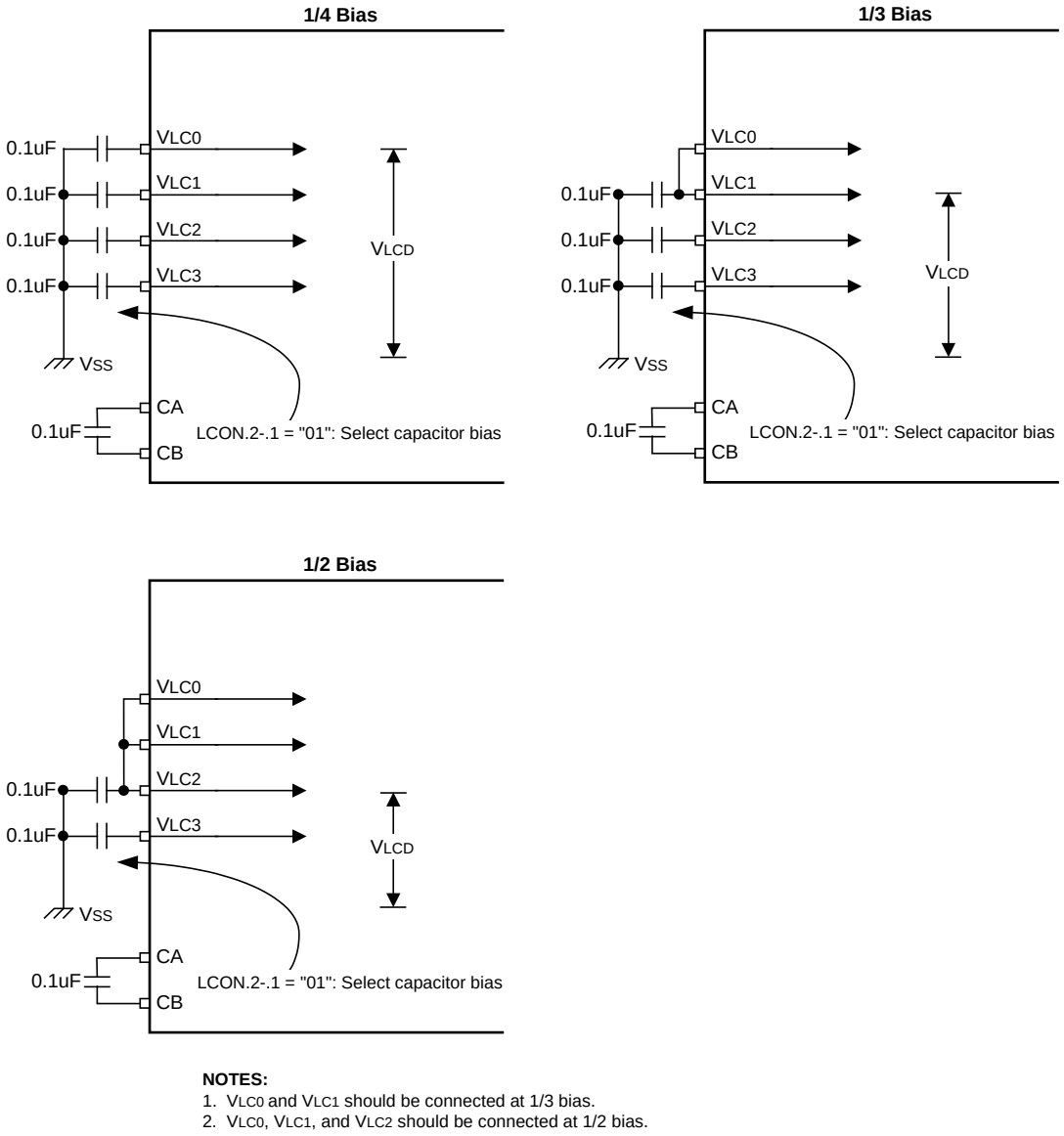
Timer C has two operating mode, you can select one of them using the appropriate TCCON setting:

- Interval timer mode (Toggle output at TCOUT pin), only match interrupt occurs
- PWM mode (TCPWM pin), match and overflow interrupt can occur

Timer C has the following functional components:

- Clock frequency divider with multiplexer
- 8-bit counter, 8-bit comparator, and 8-bit reference data register (TCDATA)
- PWM or match output (TCOUT/TCPWM)
- Timer C match/overflow interrupt (IRQ2, vector D4H) generation
- Timer C control register, TCCON (set 1, bank0, ECH, read/write)

15.8 Capacitor Bias Pin Connection



NOTES:
1. V_{LC0} and V_{LC1} should be connected at 1/3 bias.
2. V_{LC0}, V_{LC1}, and V_{LC2} should be connected at 1/2 bias.

Figure 15-8 Capacitor Bias Pin Connection

15.10 Segment (SEG) Signals

The 34 LCD segment signal pins are connected to corresponding display RAM locations at page 8. Bits of the display RAM are synchronized with the common signal output pins.

When the bit value of a display RAM location is "1", a select signal is sent to the corresponding segment pin. When the display bit is "0", a 'no-select' signal to the corresponding segment pin.

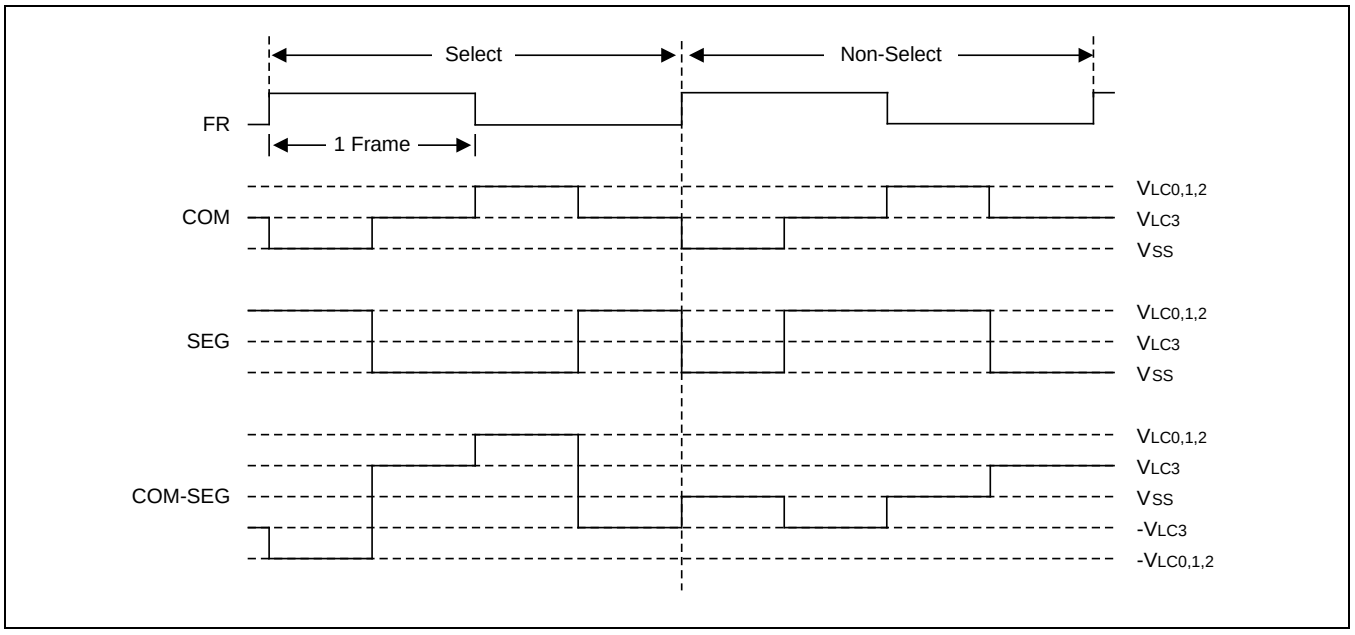


Figure 15-9 Select/No-Select Signal in 1/2 Duty, 1/2 Bias Display Mode

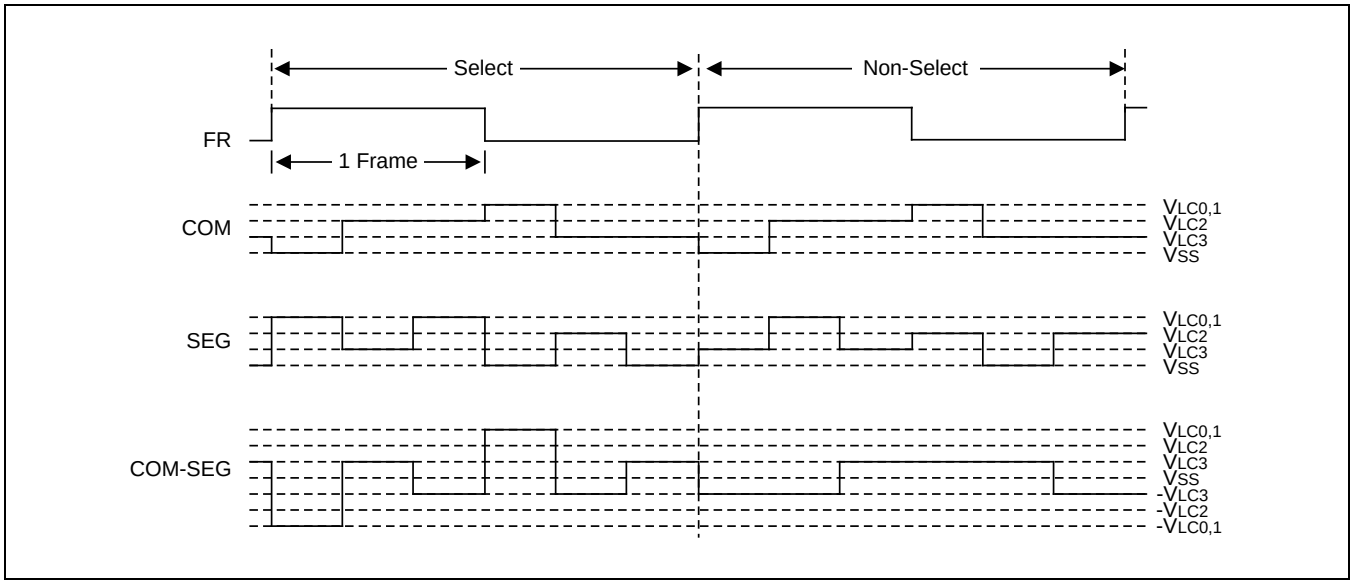


Figure 15-10 Select/No-Select Signal in 1/3 Duty, 1/3 Bias Display Mode

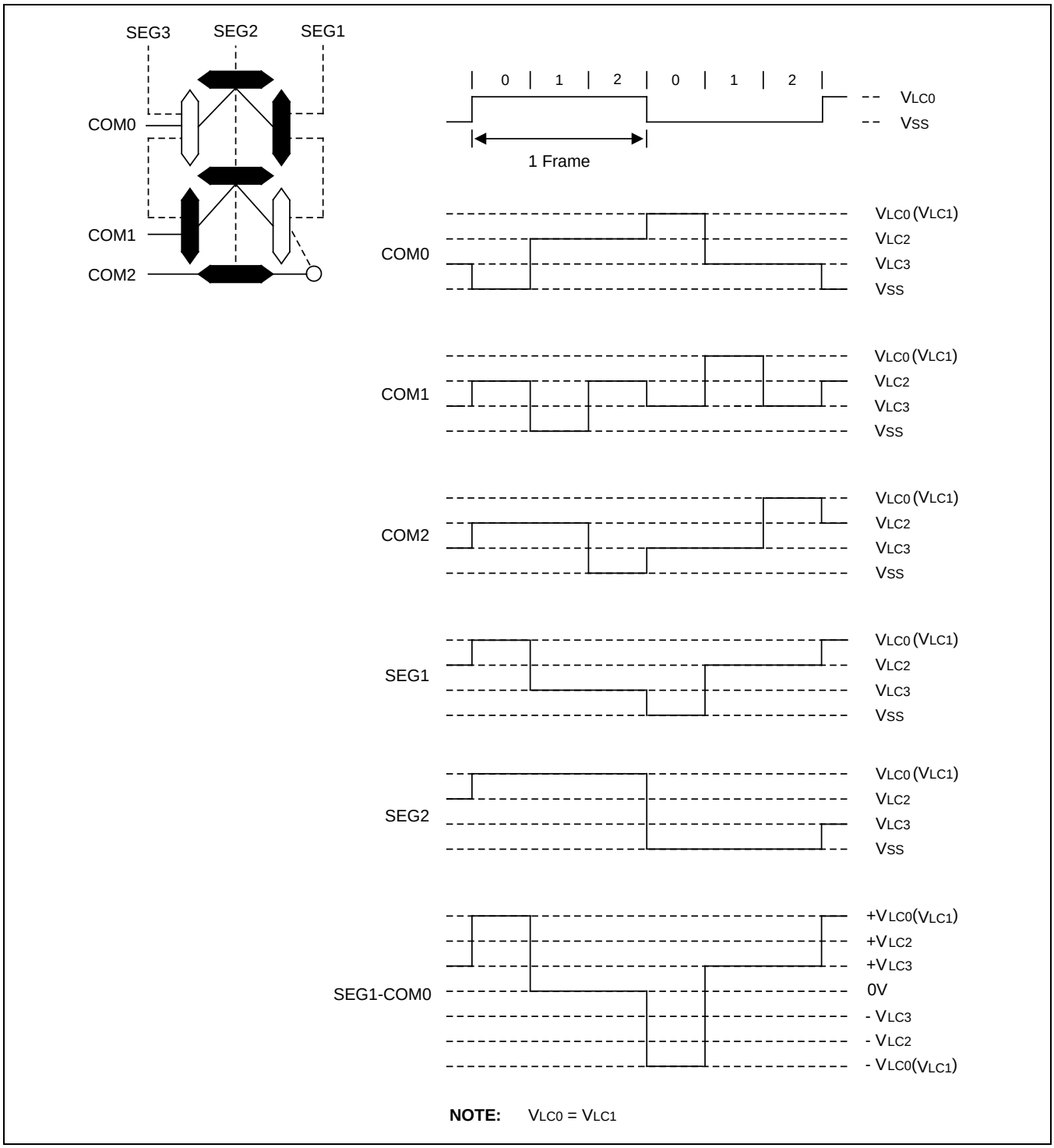


Figure 15-12 LCD Signal Waveforms (1/3 Duty, 1/3 Bias)

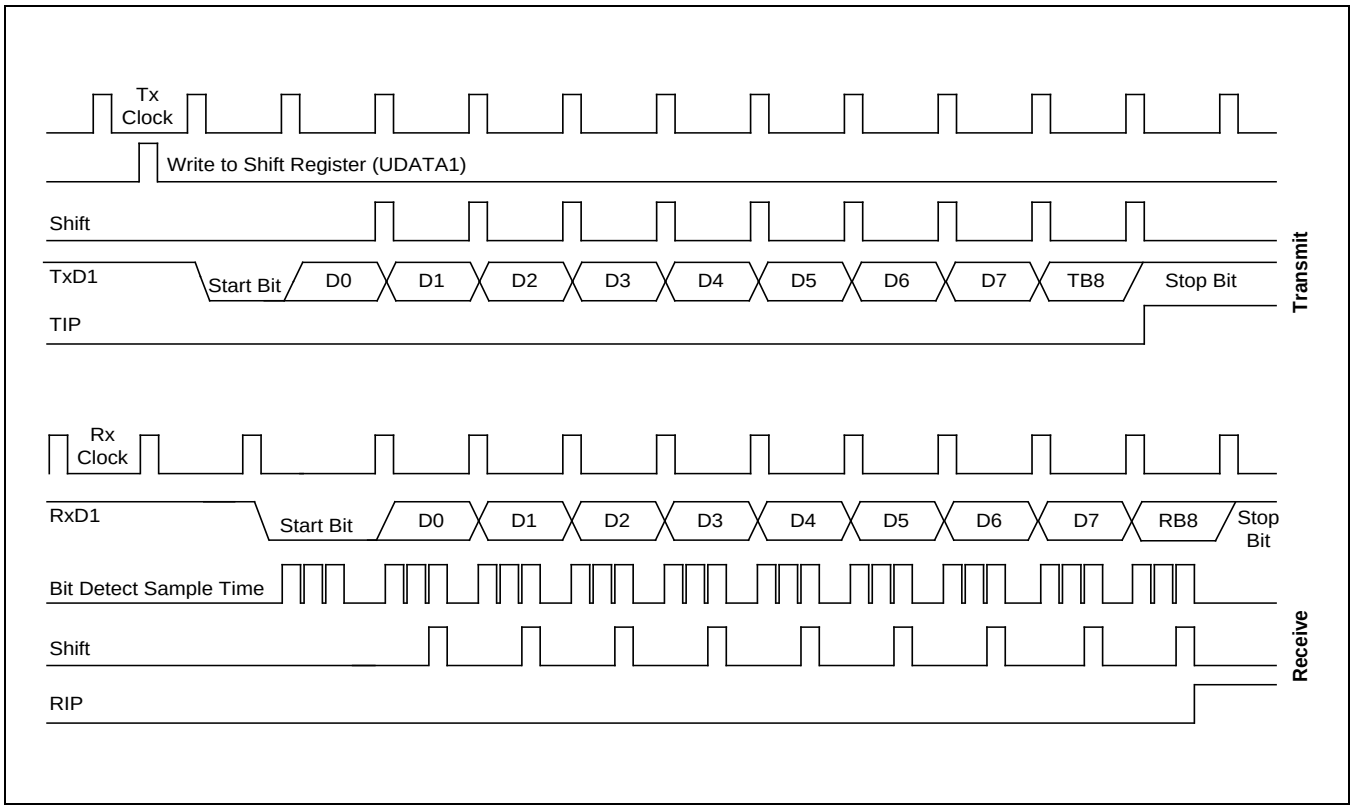


Figure 19-9 Timing Diagram for Serial Port Mode 3 Operation

20 Pattern Generation Module

20.1 Overview

20.1.1 PAttern Gneration Flow

You can output up to 8-bit through P3.0-P3.7 by tracing the following sequence. First of all, you have to change the PGDATA into what you want to output. And then you have to set the PGCON to enable the pattern generation module and select the triggering signal. From now, bits of PGDATA are on the P3.0-P3.7 whenever the selected triggering signal happens.

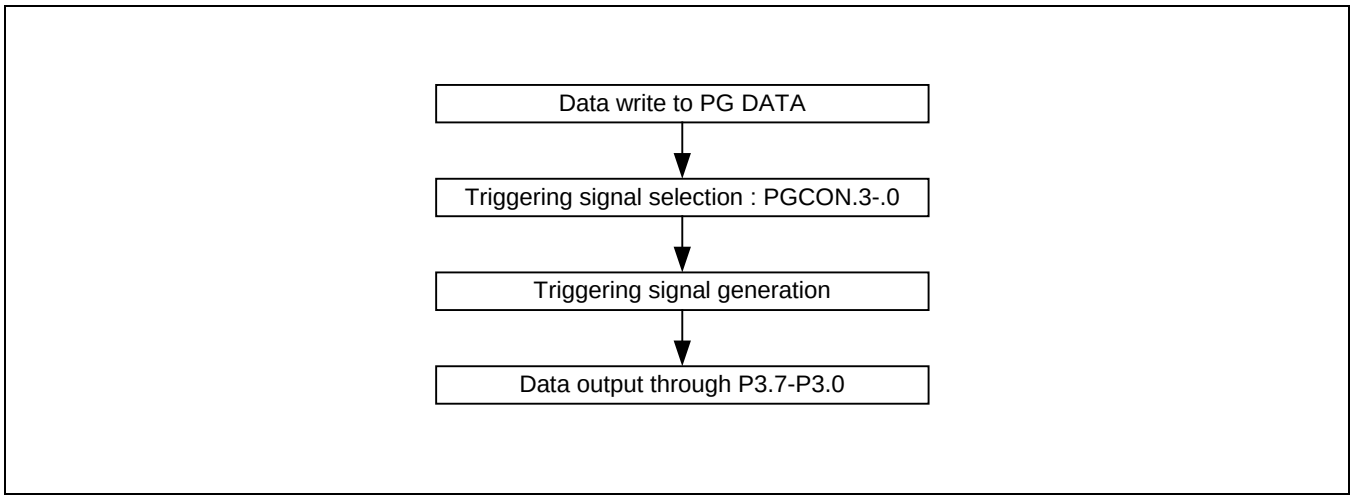


Figure 20-1 Pattern Generation Flow

21.2 User Program Mode

This mode supports sector erase, byte programming, byte read and one protection mode (Hard lock protection). The read protection mode is available only in Tool Program mode. So in order to make a chip into read protection, you need to select a read protection option when you program a initial your code to a chip by using Tool Program mode by using a programming tool.

The S3F8S6B has the pumping circuit internally; therefore, 12.5V into V_{PP} (Test) pin is not needed. To program a Flash memory in this mode several control registers will be used. There are four kind functions – programming, reading, sector erase and hard lock protection

NOTE:

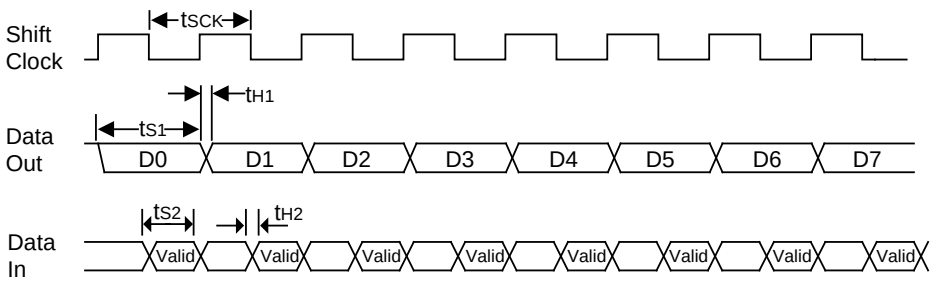
1. The user program mode cannot be used when the CPU operates with the subsystem clock.
2. Be sure to execute the DI instruction before starting user program mode. The user program mode checks the interrupt request register (IRQ). If an interrupt request is generated, user program mode is stopped.
3. User program mode is also stopped by an interrupt request that is masked even in the DI status. To prevent this, Be disable the interrupt by using the each peripheral interrupt enable bit.

Example 21-1 Sector Erase

```
                •
                •

                SB0
reErase:        LD      FMUSR,Temp0             ; User Program mode enable
                                                ; Temp0 = #0A5H
                                                ; Temp0 variable is must be setting another routine
                LD      FMSECH,#10H
                LD      FMSECL,#00H             ; Set sector address (1000H–107FH)
                CP      UserID_Code,#User_value ; Check user's ID code (written by user)
                                                ; User_value is any value by user
                JR      NE,Not_ID_Code          ; If not equal, jump to Not_ID_Code
                LD      FMCON,Temp1             ; Start sector erase
                                                ; Temp1 = #0A1H
                                                ; Temp1 variable is must be setting another routine
                NOP                             ; Dummy Instruction, This instruction must be
                                                  needed
                NOP                             ; Dummy Instruction, This instruction must be
                                                  needed
                LD      FMUSR,#0                ; User Program mode disable
                TM      FMCON,#00001000B        ; Check "Sector erase status bit"
                JR      NZ,reErase              ; Jump to reErase if fail
                •
                •
                •
Not_ID_Code:
                SB0
                LD      FMUSR,#0                ; User Program mode disable
                LD      FMCON,#0                ; Sector erase mode disable
                •
                •
                •
```

NOTE: In case of Flash User Mode, the Tmep0 to Temp1's data values are must be setting another routine. Temp0 to Temp (n) variables are should be defined by user.



NOTE: The symbols shown in this diagram are defined as follows:

f_{SCK}	Serial port clock cycle time
t_{S1}	Output data setup to clock rising edge
t_{S2}	Clock rising edge to input data valid
t_{H1}	Output data hold after clock rising edge
t_{H2}	Input data hold after clock rising edge

Figure 22-8 Timing Waveform for the UART Module