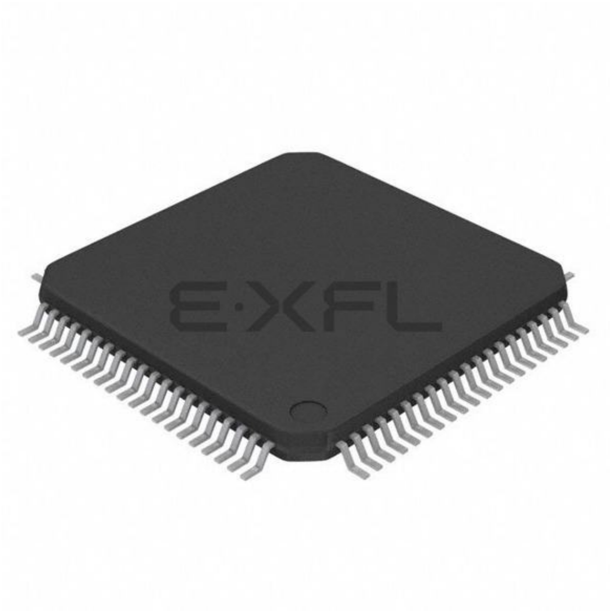
Microchip Technology - PIC18F84J90-I/PT Datasheet



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	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LCD, LVD, POR, PWM, WDT
Number of I/O	67
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f84j90-i-pt

1.0 DEVICE OVERVIEW

This document contains device-specific information for the following devices:

- PIC18F63J90
- PIC18F64J90
- PIC18F65J90
- PIC18F83J90
- PIC18F84J90
- PIC18F85J90

This family combines the traditional advantages of all PIC18 microcontrollers – namely, high computational performance and a rich feature set – with a versatile on-chip LCD driver, while maintaining an extremely competitive price point. These features make the PIC18F85J90 family a logical choice for many high-performance applications where price is a primary consideration.

1.1 Core Features

1.1.1 nanoWatt TECHNOLOGY

All of the devices in the PIC18F85J90 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal RC oscillator, power consumption during code execution can be reduced by as much as 90%.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled but the peripherals still active. In these states, power consumption can be reduced even further, to as little as 4% of normal operation requirements.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the user to incorporate power-saving ideas into their application's software design.

1.1.2 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F85J90 family offer six different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes, using crystals or ceramic resonators.
- Two External Clock modes, offering the option of a divide-by-4 clock output.
- A Phase Lock Loop (PLL) frequency multiplier, available to the External Oscillator modes which allows clock speeds of up to 40 MHz.
- An internal oscillator block which provides an 8 MHz clock (±2% accuracy) and an INTRC source (approximately 31 kHz, stable over temperature and V_{DD}), as well as a range of six user-selectable clock frequencies, between 125 kHz to 4 MHz, for a total of eight clock frequencies. This option frees the two oscillator pins for use as additional general purpose I/O.

The internal oscillator block provides a stable reference source that gives the family additional features for robust operation:

- **Fail-Safe Clock Monitor:** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up:** This option allows the internal oscillator to serve as the clock source from Power-on Reset, or wake-up from Sleep mode, until the primary clock source is available.

1.1.3 MEMORY OPTIONS

The PIC18F85J90 family provides a range of program memory options, from 8 Kbytes to 32 Kbytes of code space. The Flash cells for program memory are rated to last up to 1000 erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 20 years.

The PIC18F85J90 family also provides plenty of room for dynamic application data, with up to 2048 bytes of data RAM.

1.1.4 EXTENDED INSTRUCTION SET

The PIC18F85J90 family implements the optional extension to the PIC18 instruction set, adding 8 new instructions and an Indexed Addressing mode. Enabled as a device configuration option, the extension has been specifically designed to optimize re-entrant application code originally developed in high-level languages, such as 'C'.

1.1.5 EASY MIGRATION

Regardless of the memory size, all devices share the same rich set of peripherals, allowing for a smooth migration path as applications grow and evolve.

The consistent pinout scheme used throughout the entire family also aids in migrating to the next larger device. This is true when moving between the 64-pin members, between the 80-pin members, or even jumping from 64-pin to 80-pin devices.

The PIC18F85J90 family is also largely pin compatible with other PIC18 families, such as the PIC18F8720 and PIC18F8722, as well as the PIC18F8490 family of microcontrollers with LCD drivers. This allows a new dimension to the evolution of applications, allowing developers to select different price points within Microchip's PIC18 portfolio, while maintaining a similar feature set.

TABLE 1-4: PIC18F8XJ90 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number	Pin Type	Buffer Type	Description
	TQFP			
				PORTD is a bidirectional I/O port.
RD0/SEG0	72			
RD0		I/O	ST	Digital I/O.
SEG0		O	Analog	SEG0 output for LCD.
RD1/SEG1	69			
RD1		I/O	ST	Digital I/O.
SEG1		O	Analog	SEG1 output for LCD.
RD2/SEG2	68			
RD2		I/O	ST	Digital I/O.
SEG2		O	Analog	SEG2 output for LCD.
RD3/SEG3	67			
RD3		I/O	ST	Digital I/O.
SEG3		O	Analog	SEG3 output for LCD.
RD4/SEG4	66			
RD4		I/O	ST	Digital I/O.
SEG4		O	Analog	SEG4 output for LCD.
RD5/SEG5	65			
RD5		I/O	ST	Digital I/O.
SEG5		O	Analog	SEG5 output for LCD.
RD6/SEG6	64			
RD6		I/O	ST	Digital I/O.
SEG6		O	Analog	SEG6 output for LCD.
RD7/SEG7	63			
RD7		I/O	ST	Digital I/O.
SEG7		O	Analog	SEG7 output for LCD.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I²C™ = I²C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to VDD)

Note 1: Default assignment for CCP2 when the CCP2MX Configuration bit is set.

2: Alternate assignment for CCP2 when the CCP2MX Configuration bit is cleared.

3.5 Internal Oscillator Block

The PIC18F85J90 family of devices includes an internal oscillator block which generates two different clock signals; either can be used as the microcontroller's clock source. This may eliminate the need for an external oscillator circuit on the OSC1 and/or OSC2 pins.

The main output is the Fast RC oscillator, or INTOSC, an 8 MHz clock source which can be used to directly drive the device clock. It also drives a postscaler, which can provide a range of clock frequencies from 31 kHz to 4 MHz. INTOSC is enabled when a clock frequency from 125 kHz to 8 MHz is selected. The INTOSC output can also be enabled when 31 kHz is selected, depending on the INTSRC bit (OSCTUNE<7>).

The other clock source is the Internal RC oscillator (INTRC), which provides a nominal 31 kHz output. INTRC is enabled if it is selected as the device clock source. It is also enabled automatically when any of the following are enabled:

- Power-up Timer
- Fail-Safe Clock Monitor
- Watchdog Timer
- Two-Speed Start-up

These features are discussed in greater detail in **Section 23.0 "Special Features of the CPU"**.

The clock source frequency (INTOSC direct, INTOSC with postscaler or INTRC direct) is selected by configuring the IRCF bits of the OSCCON register. The default frequency on device Resets is 1 MHz.

3.5.1 OSC1 AND OSC2 PIN CONFIGURATION

Whenever the internal oscillator is configured as the default clock source (FOSC2 = 0), the OSC1 and OSC2 pins are reconfigured automatically as port pins, RA6 and RA7. In this mode, they function as general digital I/O. All oscillator functions on the pins are disabled.

3.5.2 INTERNAL OSCILLATOR OUTPUT FREQUENCY AND TUNING

The internal oscillator block is calibrated at the factory to produce an INTOSC output frequency of 8 MHz. It can be adjusted in the user's application by writing to TUN<5:0> (OSCTUNE<5:0>) in the OSCTUNE register (Register 3-2).

When the OSCTUNE register is modified, the INTOSC frequency will begin shifting to the new frequency. The oscillator will stabilize within 1 ms. Code execution continues during this shift. There is no indication that the shift has occurred.

The INTRC oscillator operates independently of the INTOSC source. Any changes in INTOSC across voltage and temperature are not necessarily reflected by changes in INTRC or vice versa. The frequency of INTRC is not affected by OSCTUNE.

3.5.3 INTOSC FREQUENCY DRIFT

The INTOSC frequency may drift as V_{DD} or temperature changes, and can affect the controller operation in a variety of ways. It is possible to adjust the INTOSC frequency by modifying the value in the OSCTUNE register. This will have no effect on the INTRC clock source frequency.

Tuning INTOSC requires knowing when to make the adjustment, in which direction it should be made and in some cases, how large a change is needed. Three compensation techniques are shown here.

3.5.3.1 Compensating with the EUSART

An adjustment may be required when the EUSART begins to generate framing errors or receives data with errors while in Asynchronous mode. Framing errors indicate that the device clock frequency is too high. To adjust for this, decrement the value in OSCTUNE to reduce the clock frequency. On the other hand, errors in data may suggest that the clock speed is too low. To compensate, increment OSCTUNE to increase the clock frequency.

3.5.3.2 Compensating with the Timers

This technique compares device clock speed to some reference clock. Two timers may be used; one timer is clocked by the peripheral clock, while the other is clocked by a fixed reference source, such as the Timer1 oscillator.

Both timers are cleared, but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value is much greater than expected, then the internal oscillator block is running too fast. To adjust for this, decrement the OSCTUNE register.

3.5.3.3 Compensating with the CCP Module in Capture Mode

A CCP module can use free-running Timer1 (or Timer3), clocked by the internal oscillator block and an external event with a known period (i.e., AC power frequency). The time of the first event is captured in the CCPRxH:CCPRxL registers and is recorded for use later. When the second event causes a capture, the time of the first event is subtracted from the time of the second event. Since the period of the external event is known, the time difference between events can be calculated.

If the measured time is much greater than the calculated time, the internal oscillator block is running too fast. To compensate, decrement the OSCTUNE register. If the measured time is much less than the calculated time, the internal oscillator block is running too slow. To compensate, increment the OSCTUNE register.

7.5 Writing to Flash Program Memory

The minimum programming block is 32 words or 64 bytes. Word or byte programming is not supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 64 holding registers used by the table writes for programming.

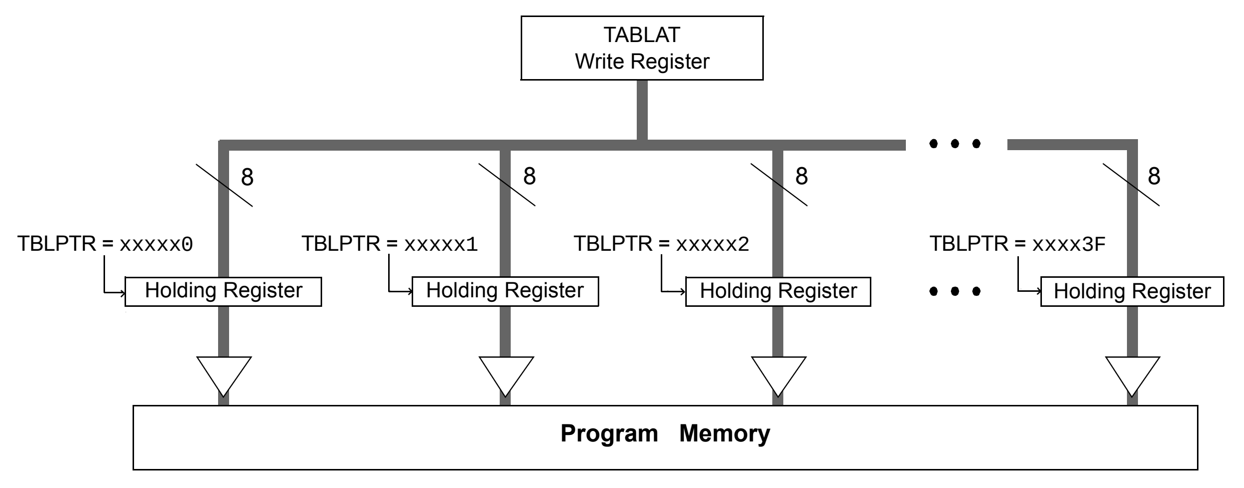
Since the Table Latch (TABLAT) is only a single byte, the `TBLWT` instruction may need to be executed 64 times for each programming operation. All of the table write operations will essentially be short writes because only the holding registers are written. At the end of updating the 64 holding registers, the EECON1 register must be written to in order to start the programming operation with a long write.

The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

The on-chip timer controls the write time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device.

> **Note 1:** Unlike previous PIC® devices, members of the PIC18F85J90 family do not reset the holding registers after a write occurs. The holding registers must be cleared or overwritten before a programming sequence.
>
> **2:** To maintain the endurance of the program memory cells, each Flash byte should not be programmed more than one time between erase operations. Before attempting to modify the contents of the target cell a second time, a block erase of the target block, or a bulk erase of the entire memory, must be performed.

FIGURE 7-5: TABLE WRITES TO FLASH PROGRAM MEMORY



7.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

The sequence of events for programming an internal program memory location should be:

1. Read 1024 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load the Table Pointer register with the address being erased.
4. Execute the block erase procedure.
5. Load the Table Pointer register with the address of the first byte being written, minus 1.
6. Write the 64 bytes into the holding registers with auto-increment.
7. Set the WREN bit (EECON1<2>) to enable byte writes.
8. Disable interrupts.
9. Write 55h to EECON2.
10. Write 0AAh to EECON2.
11. Set the WR bit; this will begin the write cycle.
12. The CPU will stall for the duration of the write for TIW (see parameter D133A).
13. Re-enable interrupts.
14. Repeat steps 6 through 13 until all 1024 bytes are written to program memory.
15. Verify the memory (table read).

An example of the required code is shown in Example 7-3 on the following page.

> **Note:** Before setting the WR bit, the Table Pointer address needs to be within the intended address range of the 64 bytes in the holding register.

9.2 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Request (Flag) registers (PIR1, PIR2, PIR3).

> **Note 1:** Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE (INTCON<7>).
>
> **2:** User software should ensure the appropriate interrupt flag bits are cleared prior to enabling an interrupt and after servicing that interrupt.

REGISTER 9-4: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1

U-0	R/W-0	R-0	R-0	R/W-0	U-0	R/W-0	R/W-0
—	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **ADIF:** A/D Converter Interrupt Flag bit
1 = An A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

bit 5 **RC1IF:** EUSART Receive Interrupt Flag bit
1 = The EUSART receive buffer, RCREG1, is full (cleared when RCREG1 is read)
0 = The EUSART receive buffer is empty

bit 4 **TX1IF:** EUSART Transmit Interrupt Flag bit
1 = The EUSART transmit buffer, TXREG1, is empty (cleared when TXREG1 is written)
0 = The EUSART transmit buffer is full

bit 3 **SSPIF:** Master Synchronous Serial Port Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **Unimplemented:** Read as ‘0’

bit 1 **TMR2IF:** TMR2 to PR2 Match Interrupt Flag bit
1 = TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = TMR1 register overflowed (must be cleared in software)
0 = TMR1 register did not overflow

TABLE 10-2: OUTPUT DRIVE LEVELS FOR VARIOUS PORTS

Low	Medium	High
PORTA<5:0>	PORTD	PORTA<7:6>
PORTF	PORTE	PORTB
PORTG	PORTJ[(1)]	PORTC
PORTH[(1)]		

Note 1: Not available on 64-pin devices.

10.1.3 PULL-UP CONFIGURATION

Four of the I/O ports (PORTB, PORTD, PORTE and PORTJ) implement configurable weak pull-ups on all pins. These are internal pull-ups that allow floating digital input signals to be pulled to a consistent level without the use of external resistors.

The pull-ups are enabled with a single bit for each of the ports: $\overline{\text{RBPU}}$ (INTCON2<7>) for PORTB, and RDPU, REPU and PJPU (PORTG<7:5>) for the other ports.

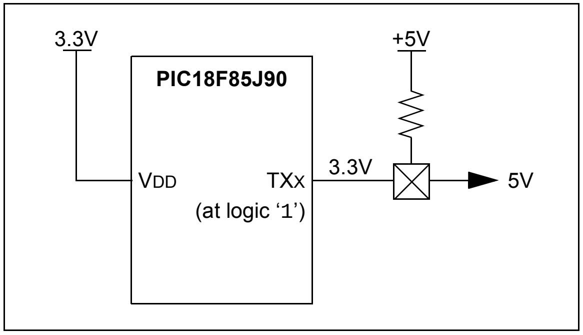
10.1.4 OPEN-DRAIN OUTPUTS

The output pins for several peripherals are also equipped with a configurable, open-drain output option. This allows the peripherals to communicate with external digital logic, operating at a higher voltage level, without the use of level translators.

The open-drain option is implemented on port pins specifically associated with the data and clock outputs of the USARTs, the MSSP module (in SPI mode) and the CCP modules. This option is selectively enabled by setting the open-drain control bit for the corresponding module in TRISG and LATG. Their configuration is discussed in more detail in **Section 10.4 "PORTC, TRISC and LATC Registers"**, **Section 10.6 "PORTE, TRISE and LATE Registers"** and **Section 10.8 "PORTG, TRISG and LATG Registers"**.

When the open-drain option is required, the output pin must also be tied through an external pull-up resistor provided by the user to a higher voltage level, up to 5V (Figure 10-2). When a digital logic high signal is output, it is pulled up to the higher voltage level.

FIGURE 10-2: USING THE OPEN-DRAIN OUTPUT (USART SHOWN AS EXAMPLE)



10.2 PORTA, TRISA and LATA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding Data Direction and Data Latch registers are TRISA and LATA.

RA4/T0CKI is a Schmitt Trigger input. All other PORTA pins have TTL input levels and full CMOS output drivers.

The RA4 pin is multiplexed with the Timer0 clock input and one of the LCD segment drives. RA5 and RA<3:0> are multiplexed with analog inputs for the A/D Converter.

The operation of the analog inputs as A/D Converter inputs is selected by clearing or setting the PCFG<3:0> control bits in the ADCON1 register. The corresponding TRISA bits control the direction of these pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

Note: RA5 and RA<3:0> are configured as analog inputs on any Reset and are read as '0'. RA4 is configured as a digital input.

OSC2/CLKO/RA6 and OSC1/CLKI/RA7 normally serve as the external circuit connections for the external (primary) oscillator circuit (HS Oscillator modes), or the external clock input and output (EC Oscillator modes). In these cases, RA6 and RA7 are not available as digital I/O and their corresponding TRIS and LAT bits are read as '0'. When the device is configured to use INTOSC or INTRC as the default oscillator mode (FOSC2 Configuration bit is '0'), RA6 and RA7 are automatically configured as digital I/O; the oscillator and clock in/clock out functions are disabled.

RA1, RA4 and RA5 are multiplexed with LCD segment drives, controlled by bits in the LCDSE1 and LCDSE2 registers. I/O port functionality is only available when the LCD segments are disabled.

EXAMPLE 10-1: INITIALIZING PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output latches
CLRF    LATA    ; Alternate method to
                ; clear output data latches
MOVLW   07h     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   0BFh    ; Value used to initialize
                ; data direction
MOVWF   TRISA   ; Set RA<7, 5:0> as inputs,
                ; RA<6> as output
```

TABLE 10-15: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	60
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	60
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	59
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	59
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	59
LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16	59
LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24	59

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used by PORTF.

NOTES:

NOTES:

15.3 Compare Mode

In Compare mode, the 16-bit CCPR2 register value is constantly compared against either the TMR1 or TMR3 register pair value. When a match occurs, the CCP2 pin can be:

- driven high
- driven low
- toggled (high-to-low or low-to-high)
- remain unchanged (that is, reflects the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCP2M<3:0>). At the same time, the interrupt flag bit, CCP2IF, is set.

15.3.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the appropriate TRIS bit.

Note:	Clearing the CCP2CON register will force the RC1 or RE7 compare output latch (depending on device configuration) to the default low level. This is not the PORTC or PORTE I/O data latch.

15.3.2 TIMER1/TIMER3 MODE SELECTION

Timer1 and/or Timer3 must be running in Timer mode, or Synchronized Counter mode, if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

15.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCP2M<3:0> = `1010`), the CCP2 pin is not affected. Only a CCP interrupt is generated, if enabled, and the CCP2IE bit is set.

15.3.4 SPECIAL EVENT TRIGGER

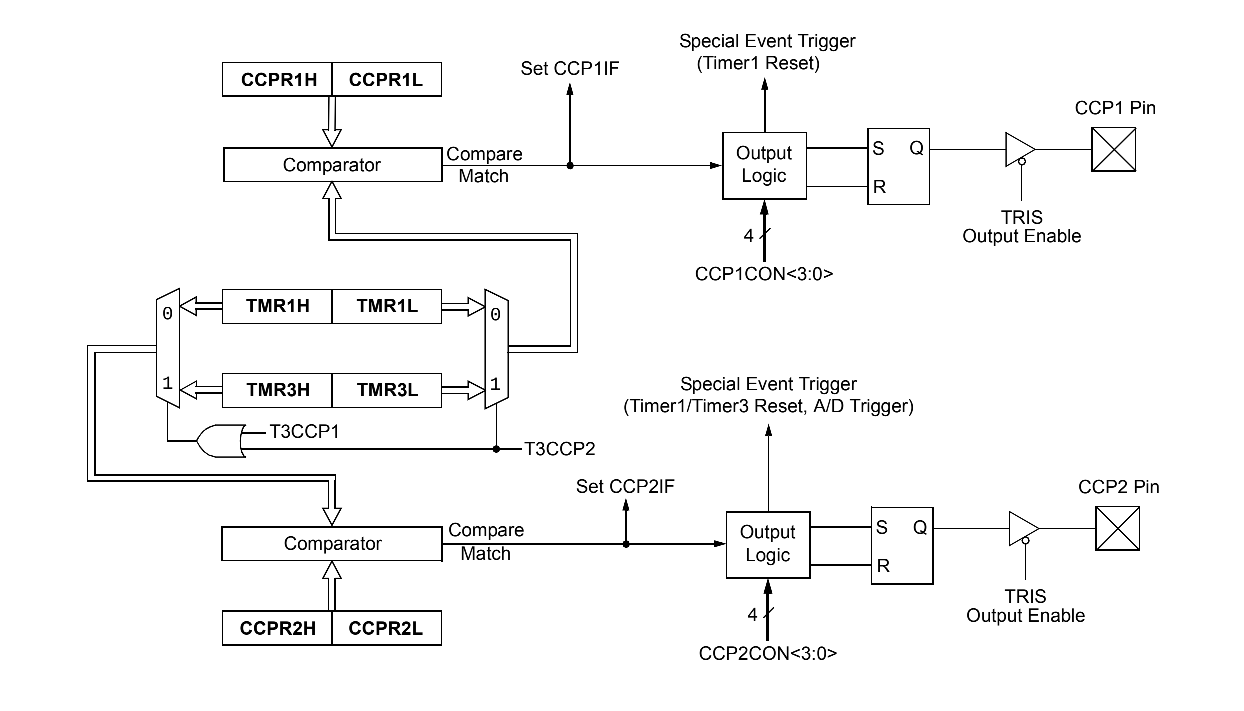
Both CCP modules are equipped with a Special Event Trigger. This is an internal hardware signal generated in Compare mode to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCP2M<3:0> = `1011`).

For either CCP module, the Special Event Trigger resets the timer register pair for whichever timer resource is currently assigned as the module's time base. This allows the CCPRx registers to serve as a programmable period register for either timer.

The Special Event Trigger for CCP2 can also start an A/D conversion. In order to do this, the A/D Converter must already be enabled.

Note:	The Special Event Trigger of CCP1 only resets Timer1/Timer3 and cannot start an A/D conversion even when the A/D Converter is enabled.

FIGURE 15-3: COMPARE MODE OPERATION BLOCK DIAGRAM



17.3.9 OPERATION IN POWER-MANAGED MODES

In SPI Master mode, module clocks may be operating at a different speed than when in Full-Power mode; in the case of Sleep mode, all clocks are halted.

In Idle modes, a clock is provided to the peripherals. That clock should be from the primary clock source, the secondary clock (Timer1 oscillator at 32.768 kHz) or the INTRC source. See **Section 3.3 "Clock Sources and Oscillator Switching"** for additional information.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSP interrupts are enabled, they can wake the controller from Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle mode is not desired, MSSP interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the devices wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in any power-managed mode and data to be shifted into the SPI Transmit/Receive Shift register. When all 8 bits have been received, the MSSP interrupt flag bit will be set and if enabled, will wake the device.

17.3.10 EFFECTS OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

17.3.11 BUS MODE COMPATIBILITY

Table 17-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 17-1: SPI BUS MODES

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
`0, 0`	`0`	`1`
`0, 1`	`0`	`0`
`1, 0`	`1`	`1`
`1, 1`	`1`	`0`

There is also an SMP bit which controls when the data is sampled.

TABLE 17-2: REGISTERS ASSOCIATED WITH SPI OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	60
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	60
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	60
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	60
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	60
TRISG	SPIOD	CCP2OD	CCP1OD	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	60
SSPBUF	MSSP Receive Buffer/Transmit Register								58
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	58
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	58

Legend: Shaded cells are not used by the MSSP module in SPI mode.

17.4 I^{2}C Mode

The MSSP module in I^{2}C mode fully implements all master and slave functions (including general call support) and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications as well as 7-bit and 10-bit addressing.

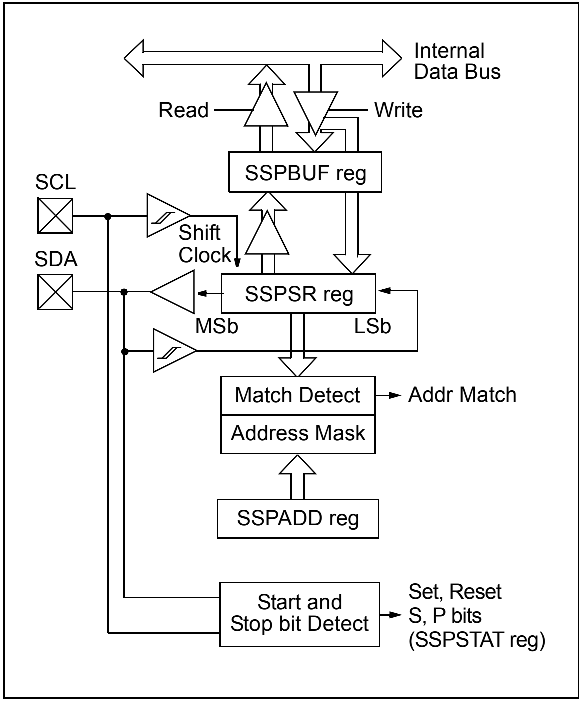
Note:	Disabling the MSSP module by clearing the SSPEN (SSPCON1<5>) bit may not reset the module. It is recommended to clear the SSPSTAT, SSPCON1 and SSPCON2 registers and select the mode prior to setting the SSPEN bit to enable the MSSP module.

Two pins are used for data transfer:

- Serial clock (SCL) – RC3/SCK/SCL
- Serial data (SDA) – RC4/SDI/SDA

The user must configure these pins as inputs by setting the TRISC<4:3> bits.

FIGURE 17-7: MSSP BLOCK DIAGRAM (I^{2}C™ MODE)



17.4.1 REGISTERS

The MSSP module has six registers for I^{2}C operation. These are:

- MSSP Control Register 1 (SSPCON1)
- MSSP Control Register 2 (SSPCON2)
- MSSP Status Register (SSPSTAT)
- MSSP Serial Receive/Transmit Buffer Register (SSPBUF)
- MSSP Shift Register (SSPSR) – Not directly accessible
- MSSP Address Register (SSPADD)

SSPCON1, SSPCON2 and SSPSTAT are the control and status registers in I^{2}C mode operation. The SSPCON1 and SSPCON2 registers are readable and writable. The lower 6 bits of the SSPSTAT are read-only. The upper two bits of the SSPSTAT are read/write.

Many of the bits in SSPCON2 assume different functions, depending on whether the module is operating in Master or Slave mode; bits<5:2> also assume different names in Slave mode. The different aspects of SSPCON2 are shown in Register 17-5 (for Master mode) and Register 17-6 (Slave mode).

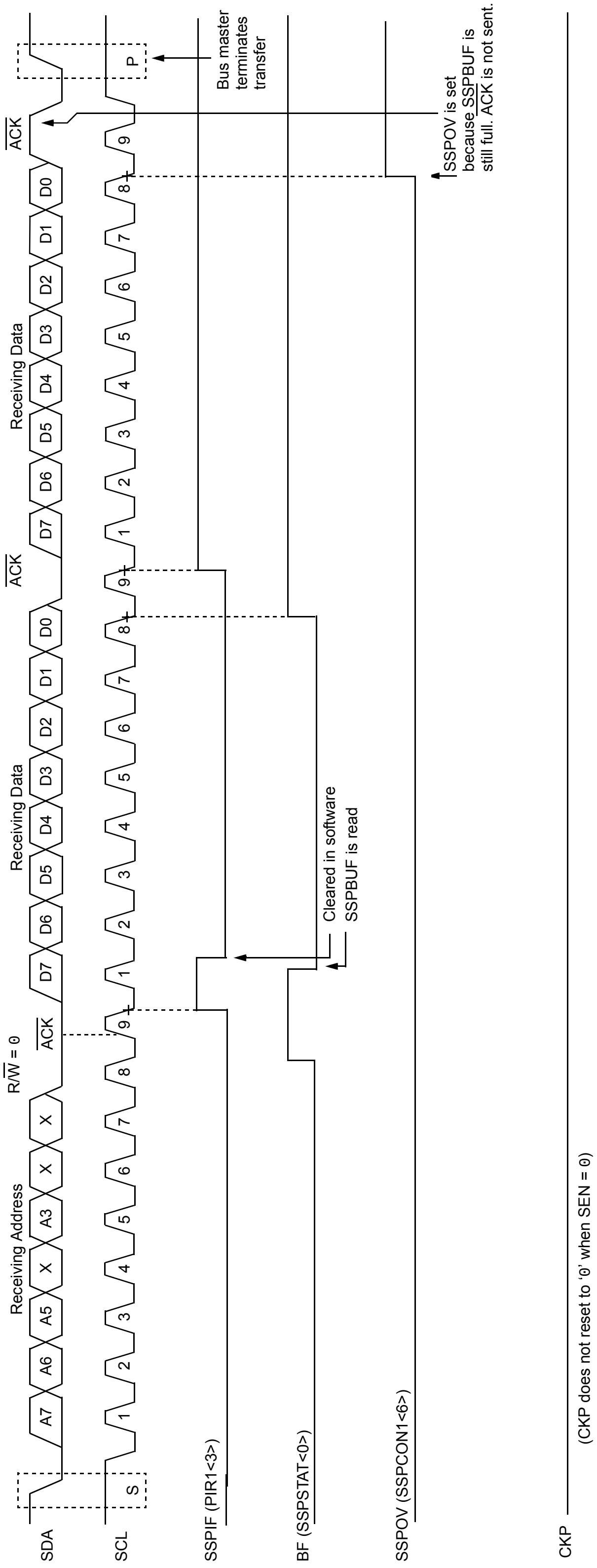
SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

SSPADD register holds the slave device address when the MSSP is configured in I^{2}C Slave mode. When the MSSP is configured in Master mode, the lower seven bits of SSPADD act as the Baud Rate Generator reload value.

In receive operations, SSPSR and SSPBUF together create a double-buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double-buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.

FIGURE 17-9: I²C™ SLAVE MODE TIMING WITH SEN = 0 AND ADMSK<5:1> = 01011 (RECEPTION, 7-BIT ADDRESS)



Note 1: x = Don't care (i.e., address bit can be either a '1' or a '0').

2: In this example, an address equal to A7.A6.A5.X.A3.X.X will be Acknowledged and cause an interrupt.

17.4.4 CLOCK STRETCHING

Both 7-Bit and 10-Bit Slave modes implement automatic clock stretching during a transmit sequence.

The SEN bit (SSPCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCL pin to be held low at the end of each data receive sequence.

17.4.4.1 Clock Stretching for 7-Bit Slave Receive Mode (SEN = 1)

In 7-Bit Slave Receive mode, on the falling edge of the ninth clock at the end of the $\overline{\text{ACK}}$ sequence, if the BF bit is set, the CKP bit in the SSPCON1 register is automatically cleared, forcing the SCL output to be held low. The CKP being cleared to '0' will assert the SCL line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCL line low, the user has time to service the ISR and read the contents of the SSPBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 17-15).

> **Note 1:** If the user reads the contents of the SSPBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.
>
> **2:** The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

17.4.4.2 Clock Stretching for 10-Bit Slave Receive Mode (SEN = 1)

In 10-Bit Slave Receive mode, during the address sequence, clock stretching automatically takes place but CKP is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address and following the receive of the second byte of the 10-bit address with the R/$\overline{\text{W}}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

> **Note:** If the user polls the UA bit and clears it by updating the SSPADD register before the falling edge of the ninth clock occurs and if the user hasn't cleared the BF bit by reading the SSPBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching on the basis of the state of the BF bit only occurs during a data sequence, not an address sequence.

17.4.4.3 Clock Stretching for 7-Bit Slave Transmit Mode

The 7-Bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCL line low, the user has time to service the ISR and load the contents of the SSPBUF before the master device can initiate another transmit sequence (see Figure 17-10).

> **Note 1:** If the user loads the contents of SSPBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.
>
> **2:** The CKP bit can be set in software regardless of the state of the BF bit.

17.4.4.4 Clock Stretching for 10-Bit Slave Transmit Mode

In 10-Bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-Bit Slave Receive mode. The first two addresses are followed by a third address sequence which contains the high-order bits of the 10-bit address and the R/$\overline{\text{W}}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled by the BF flag as in 7-Bit Slave Transmit mode (see Figure 17-13).

17.4.4.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCL output is forced to '0'. However, clearing the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external I^{2}C master device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the I^{2}C bus have deasserted SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see Figure 17-14).

FIGURE 17-14: CLOCK SYNCHRONIZATION TIMING

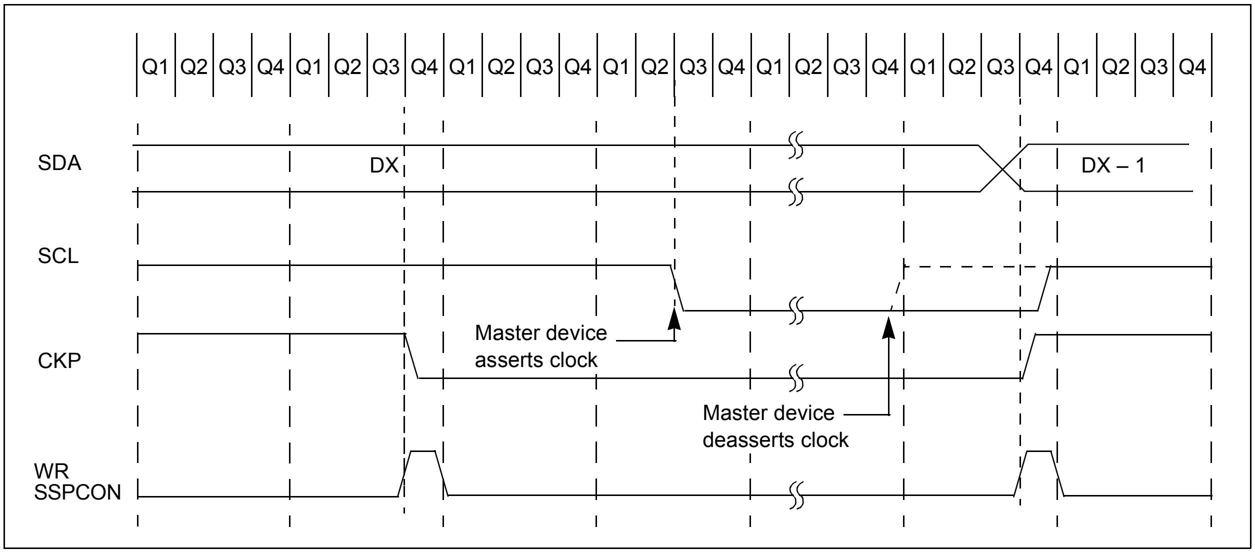

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4
SDA
DX
DX – 1
SCL
CKP
Master device asserts clock
Master device deasserts clock
WR SSPCON

FIGURE 17-16: I²C™ SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)

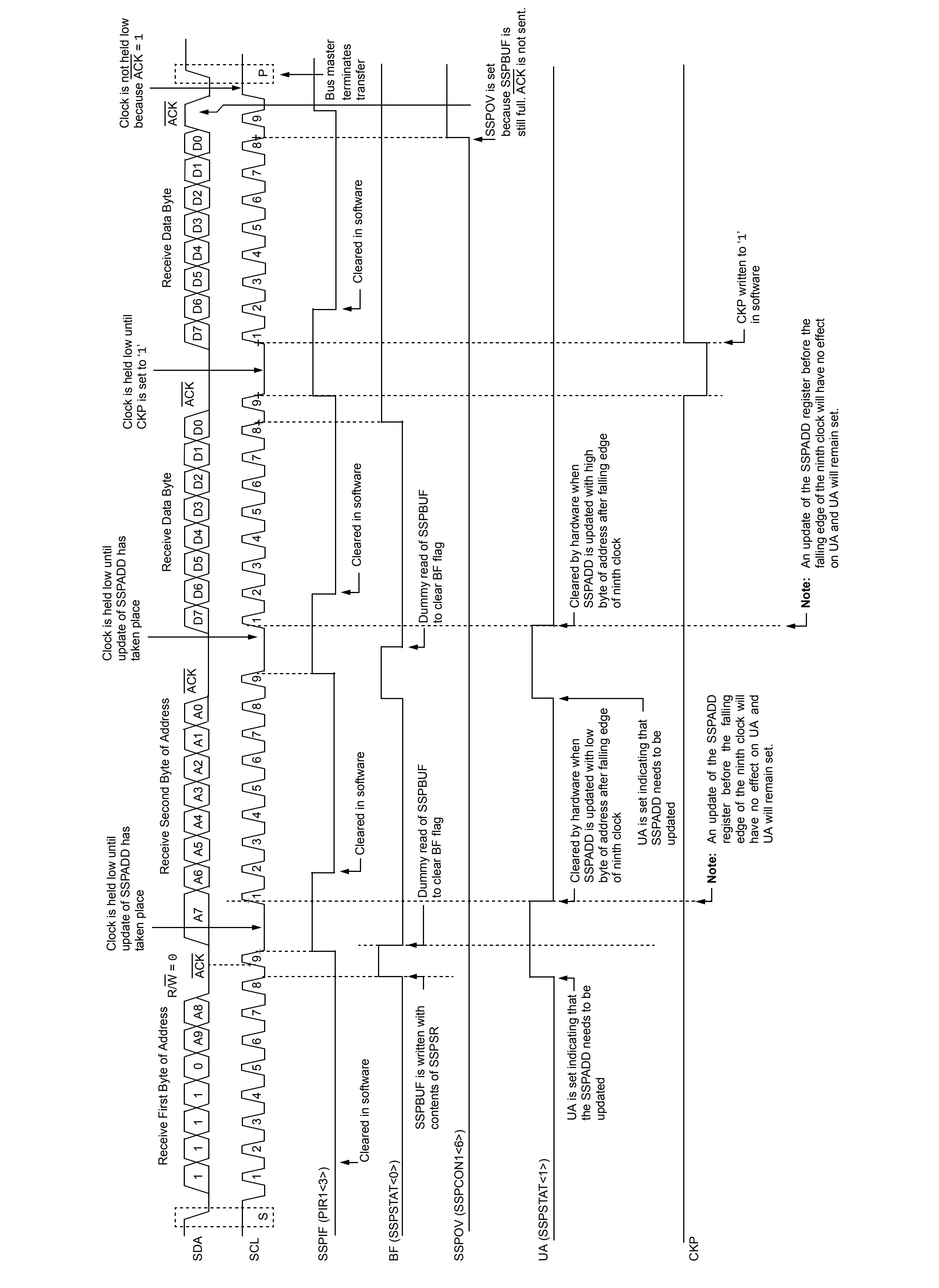

SDA
SCL
SSPIF (PIR1<3>)
BF (SSPSTAT<0>)
SSPOV (SSPCON1<6>)
UA (SSPSTAT<1>)
CKP
S
P
Receive First Byte of Address
1 1 1 1 0 A9 A8
R/W = 0
ACK
Receive Second Byte of Address
A7 A6 A5 A4 A3 A2 A1 A0
ACK
Receive Data Byte
D7 D6 D5 D4 D3 D2 D1 D0
ACK
Receive Data Byte
D7 D6 D5 D4 D3 D2 D1 D0
ACK
1 2 3 4 5 6 7 8 9
Clock is held low until update of SSPADD has taken place
Clock is held low until update of SSPADD has taken place
Clock is held low until CKP is set to '1'
Clock is not held low because ACK = 1
Bus master terminates transfer
Cleared in software
Cleared in software
Cleared in software
Cleared in software
SSPBUF is written with contents of SSPSR
Dummy read of SSPBUF to clear BF flag
Dummy read of SSPBUF to clear BF flag
SSPOV is set because SSPBUF is still full. ACK is not sent.
UA is set indicating that the SSPADD needs to be updated
Cleared by hardware when SSPADD is updated with low byte of address after falling edge of ninth clock
UA is set indicating that SSPADD needs to be updated
Cleared by hardware when SSPADD is updated with high byte of address after falling edge of ninth clock
Note: An update of the SSPADD register before the falling edge of the ninth clock will have no effect on UA and UA will remain set.
Note: An update of the SSPADD register before the falling edge of the ninth clock will have no effect on UA and UA will remain set.
CKP written to '1' in software

FIGURE 18-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

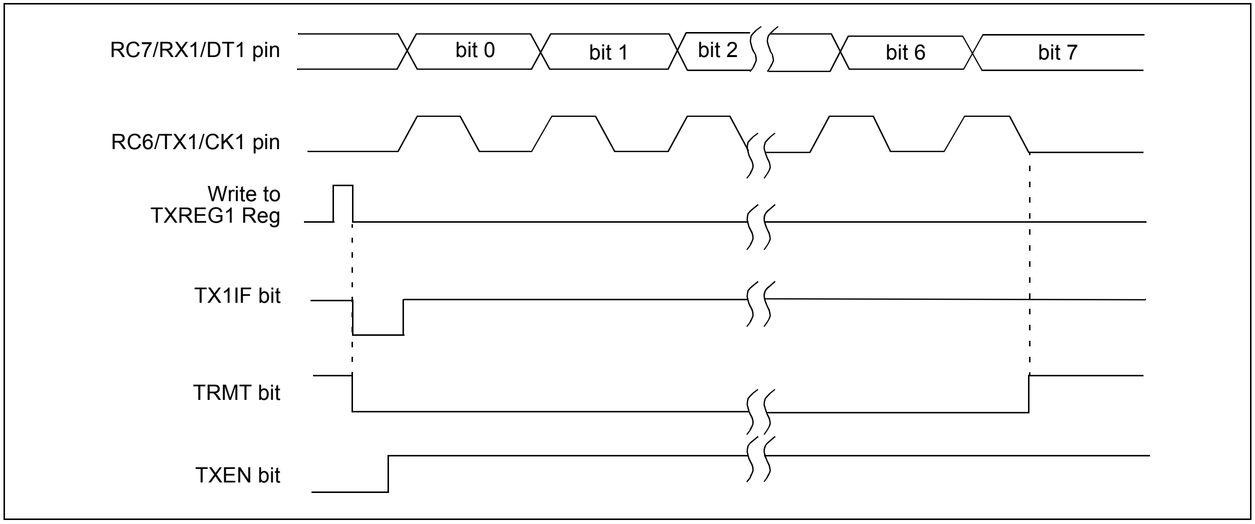


TABLE 18-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	57
PIR1	—	ADIF	RC1IF	TX1IF	SSPIF	—	TMR2IF	TMR1IF	60
PIE1	—	ADIE	RC1IE	TX1IE	SSPIE	—	TMR2IE	TMR1IE	60
IPR1	—	ADIP	RC1IP	TX1IP	SSPIP	—	TMR2IP	TMR1IP	60
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	59
TXREG1	EUSART Transmit Register								59
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	59
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	61
SPBRGH1	EUSART Baud Rate Generator Register High Byte								61
SPBRG1	EUSART Baud Rate Generator Register Low Byte								59
LATG	U2OD	U1OD	—	LATG4	LATG3	LATG2	LATG1	LATG0	60

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

BRA Unconditional Branch

Syntax:	BRA n
Operands:	-1024 ≤ n ≤ 1023
Operation:	(PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1101` | `0nnn` | `nnnn` | `nnnn`
Description:	Add the 2's complement number, '2n', to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

<u>Example:</u> `HERE BRA Jump`

Before Instruction
PC = address (HERE)
After Instruction
PC = address (Jump)

BSF Bit Set f

Syntax:	BSF f, b {,a}
Operands:	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]
Operation:	1 → f<b>
Status Affected:	None
Encoding:	`1000` | `bbba` | `ffff` | `ffff`
Description:	Bit 'b' in register 'f' is set. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank. If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 24.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

<u>Example:</u> `BSF FLAG_REG, 7, 1`

Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah

RCALL Relative Call

Syntax: RCALL n

Operands: $-1024 \leq n \leq 1023$

Operation: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

`1101`	`1nnn`	`nnnn`	`nnnn`

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' PUSH PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE     RCALL Jump`

Before Instruction
    PC = Address (`HERE`)
After Instruction
    PC = Address (`Jump`)
    TOS = Address (`HERE + 2`)

RESET Reset

Syntax: RESET

Operands: None

Operation: Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.

Status Affected: All

Encoding:

`0000`	`0000`	`1111`	`1111`

Description: This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start reset	No operation	No operation

Example: `RESET`

After Instruction
    Registers = Reset Value
    Flags* = Reset Value