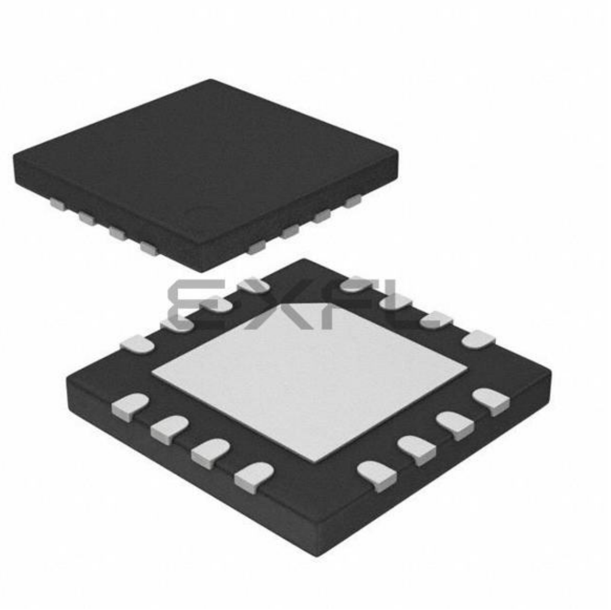
Microchip Technology - PIC16F18325-I/JQ Datasheet



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	32MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	12
Program Memory Size	14KB (8K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 11x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	16-UQFN Exposed Pad
Supplier Device Package	16-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f18325-i-jq

2.3 Master Clear ($\overline{\text{MCLR}}$) Pin

The $\overline{\text{MCLR}}$ pin provides three specific device functions:

- Device Reset (when MCLRE = 1)
- Digital input pin (when MCLRE = 0)
- Device Programming and Debugging

If programming and debugging are not required in the end application then either set the MCLRE configuration bit to ‘1’ and use the pin as a digital input or clear the MCLRE Configuration bit and leave the pin open to use the internal weak pull-up. The addition of other components, to help increase the application’s resistance to spurious Resets from voltage sags, may be beneficial. A typical configuration is shown in Figure 2-1. Other circuit designs may be implemented, depending on the application’s requirements.

During programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the $\overline{\text{MCLR}}$ pin. Consequently, specific voltage levels (V_{IH} and V_{IL}) and fast signal transitions must not be adversely affected. Therefore, the programmer $\overline{\text{MCLR}}/V_{PP}$ output should be connected directly to the pin so that R1 isolates the capacitor, C1 from the $\overline{\text{MCLR}}$ pin during programming and debugging operations.

Any components associated with the $\overline{\text{MCLR}}$ pin should be placed within 0.25 inch (6 mm) of the pin.

2.4 ICSP™ Pins

The ICSPCLK and ICSPDAT pins are used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of ohms, not to exceed 100Ω.

Pull-up resistors, series diodes and capacitors on the ICSPCLK and ICSPDAT pins are not recommended as they will interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they should be isolated from the programmer by resistors between the application and the device pins or removed from the circuit during programming. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits, and pin input voltage high (VIH) and input low (VIL) requirements.

For device emulation, ensure that the “Communication Channel Select” (i.e., ICSPCLK/ICSPDAT pins), programmed into the device, matches the physical connections for the ICSP to the Microchip debugger/emulator tool.

For more information on available Microchip development tools connection requirements, refer to **Section 37.0 “Development Support”**.

TABLE 4-4: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-31 (CONTINUED)

Address	Name	PIC16(L)F18325	PIC16(L)F18345	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
Bank 5													
CPU CORE REGISTERS; see Table 4-2 for specifics													
28Ch	ODCONA			—	—	ODCA5	ODCA4	—	ODCA2	ODCA1	ODCA0	`--00 -000`	`--00 -000`
28Dh	ODCONB	X	—	Unimplemented								—	—
		—	X	ODCB7	ODCB6	ODCB5	ODCB4	—	—	—	—	`0000 ----`	`0000 ----`
28Eh	ODCONC	X	—	—	—	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	`--00 0000`	`--00 0000`
		—	X	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	`0000 0000`	`0000 0000`
28Fh	—	—		Unimplemented								—	—
290h	—	—		Unimplemented								—	—
291h	CCPR1L			CCPR1<7:0>								`xxxx xxxx`	`xxxx xxxx`
292h	CCPR1H			CCPR1<15:8>								`xxxx xxxx`	`xxxx xxxx`
293h	CCP1CON			CCP1EN	—	CCP1OUT	CCP1FMT	CCP1MODE<3:0>				`0-x0 0000`	`0-x0 0000`
294h	CCP1CAP			—		—	—	CCP1CTS<3:0>				`---- 0000`	`---- xxxx`
295h	CCPR2L			CCPR2<7:0>								`xxxx xxxx`	`xxxx xxxx`
296h	CCPR2H			CCPR2<15:8>								`xxxx xxxx`	`xxxx xxxx`
297h	CCP2CON			CCP2EN	—	CCP2OUT	CCP2FMT	CCP2MODE<3:0>				`0-x0 0000`	`0-x0 0000`
298h	CCP2CAP			—	—	—	—	CCP2CTS<3:0>				`---- 0000`	`---- xxxx`
299h	—	—		Unimplemented								—	—
29Ah	—	—		Unimplemented								—	—
29Bh	—	—		Unimplemented								—	—
29Ch	—	—		Unimplemented								—	—
29Dh	—	—		Unimplemented								—	—
29Eh	—	—		Unimplemented								—	—
29Fh	CCPTMRS			C4TSEL<1:0>		C3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<1:0>		`0101 0101`	`0101 0101`

Legend: x = unknown, u = unchanged, q =depends on condition, - = unimplemented, read as ‘`0`’, `r` = reserved. Shaded locations unimplemented, read as ‘`0`’.

Note 1: Only on PIC16F18325/18345.

2: Register accessible from both User and ICD Debugger.

REGISTER 5-2: CONFIGURATION WORD 2: SUPERVISORS

R/P-1	R/P-1	R/P-1	U-1	R/P-1	U-1
$\overline{\text{DEBUG}}$	STVREN	PPS1WAY	—	BORV	—
bit 13					bit 8

R/P-1	R/P-1	R/P-1	U-1	R/P-1	R/P-1	R/P-1	R/P-1
BOREN1	BOREN0	$\overline{\text{LPBOREN}}$	—	WDTE1	WDTE0	$\overline{\text{PWRTE}}$	MCLRE
bit 7							bit 0

Legend:

R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘`1`’
‘`0`’ = Bit is cleared	‘`1`’ = Bit is set	n = Value when blank

bit 13 **$\overline{\text{DEBUG}}$:** Debugger Enable bit[1]
`1 = OFF` Background debugger disabled; ICSPCLK and ICSPDAT are general purpose I/O pins
`0 = ON` Background debugger enabled; ICSPCLK and ICSPDAT are dedicated to the debugger

bit 12 **STVREN:** Stack Overflow/Underflow Reset Enable bit
`1 = ON` Stack Overflow or Underflow will cause a Reset
`0 = OFF` Stack Overflow or Underflow will not cause a Reset

bit 11 **PPS1WAY:** PPSLOCK One-Way Set Enable bit
`1 = ON` The PPSLOCK bit can be cleared and set only once; PPS registers remain locked after one clear/set cycle
`0 = OFF` The PPSLOCK bit can be set and cleared repeatedly (subject to the unlock sequence)

bit 10 **Unimplemented:** Read as ‘`1`’

bit 9 **BORV:** Brown-out Reset Voltage Selection bit[2]
`1 = LOW` Brown-out Reset voltage (V_{BOR}) set to 1.9V on LF, and 2.45V on F devices
`0 = HIGH` Brown-out Reset voltage (V_{BOR}) set to 2.7V
The higher voltage setting is recommended for operation at or above 16 MHz.

bit 8 **Unimplemented:** Read as ‘`1`’

bit 7-6 **BOREN<1:0>:** Brown-out Reset Enable bits
When enabled, Brown-out Reset Voltage (V_{BOR}) is set by the BORV bit
`11 = ON` Brown-out Reset is enabled; SBOREN bit is ignored
`10 = SLEEP` Brown-out Reset is enabled while running, disabled in Sleep; SBOREN bit is ignored
`01 = SBOREN` Brown-out Reset is enabled according to SBOREN
`00 = OFF` Brown-out Reset is disabled

bit 5 **$\overline{\text{LPBOREN}}$:** Low-Power BOR Enable bit
`1 = OFF` ULPBOR is disabled
`0 = ON` ULPBOR is enabled

bit 4 **Unimplemented:** Read as ‘`1`’

bit 3-2 **WDTE<1:0>:** Watchdog Timer Enable bit
`11 = ON` WDT is enabled; SWDTEN is ignored
`10 = SLEEP` WDT is enabled while running and disabled in Sleep/Idle; SWDTEN is ignored
`01 = SWDTEN` WDT is controlled by the SWDTEN bit in the WDTCON register
`00 = OFF` WDT is disabled; SWDTEN is ignored

bit 1 **$\overline{\text{PWRTE}}$:** Power-up Timer Enable bit
`1 = OFF` PWRT is disabled
`0 = ON` PWRT is enabled

bit 0 **MCLRE:** Master Clear ($\overline{\text{MCLR}}$) Enable bit
<u>If LVP = `1`</u>:
RA3 pin function is $\overline{\text{MCLR}}$.
<u>If LVP = `0`</u>:
`1 = ON` $\overline{\text{MCLR}}$ pin is $\overline{\text{MCLR}}$.
`0 = OFF` $\overline{\text{MCLR}}$ pin function is port-defined function.

Note 1: The DEBUG bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a ‘`1`’.

2: See V_{BOR} parameter for specific trip point voltages.

6.2.1 BOR IS ALWAYS ON

When the BOREN bits of Configuration Words are programmed to '11', the BOR is always on. The device start-up will be delayed until the BOR is ready and VDD is higher than the BOR threshold.

BOR protection is active during Sleep. The BOR does not delay wake-up from Sleep.

6.2.2 BOR IS OFF IN SLEEP

When the BOREN bits of Configuration Words are programmed to '10', the BOR is on, except in Sleep. The device start-up will be delayed until the BOR is ready and VDD is higher than the BOR threshold.

BOR protection is not active during Sleep, but device wake-up will be delayed until the BOR can determine that VDD is higher than the BOR threshold. The device wake-up will be delayed until the BOR is ready.

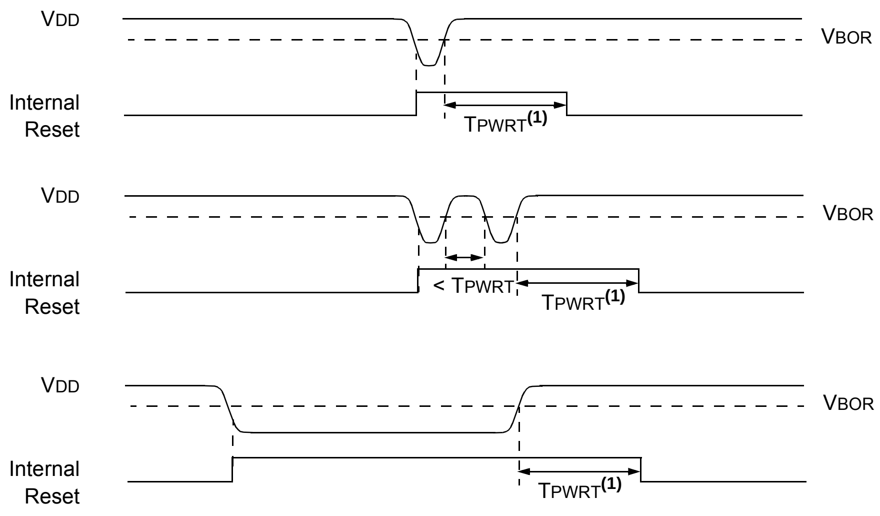
6.2.3 BOR CONTROLLED BY SOFTWARE

When the BOREN bits of Configuration Words are programmed to '01', the BOR is controlled by the SBOREN bit of the BORCON register. The device wake from Sleep is not delayed by the BOR Ready condition or the VDD level only when the SBOREN bit is cleared in software and the device is starting up from a non POR/BOR Reset event.

BOR protection begins as soon as the BOR circuit is ready. The status of the BOR circuit is reflected in the BORRDY bit of the BORCON register.

BOR protection is unchanged by Sleep.

FIGURE 6-2: BROWN-OUT SITUATIONS



Note 1: TPWRT delay only if $\overline{\text{PWRTE}}$ bit is programmed to '0'.

6.2.4 BOR ALWAYS OFF

When the BOREN bits of Configuration Word 2 are programmed to '00', the BOR is always disable. In the configuration, setting the SWBOREN bit will have no affect on BOR operation.

6.10 Start-up Sequence

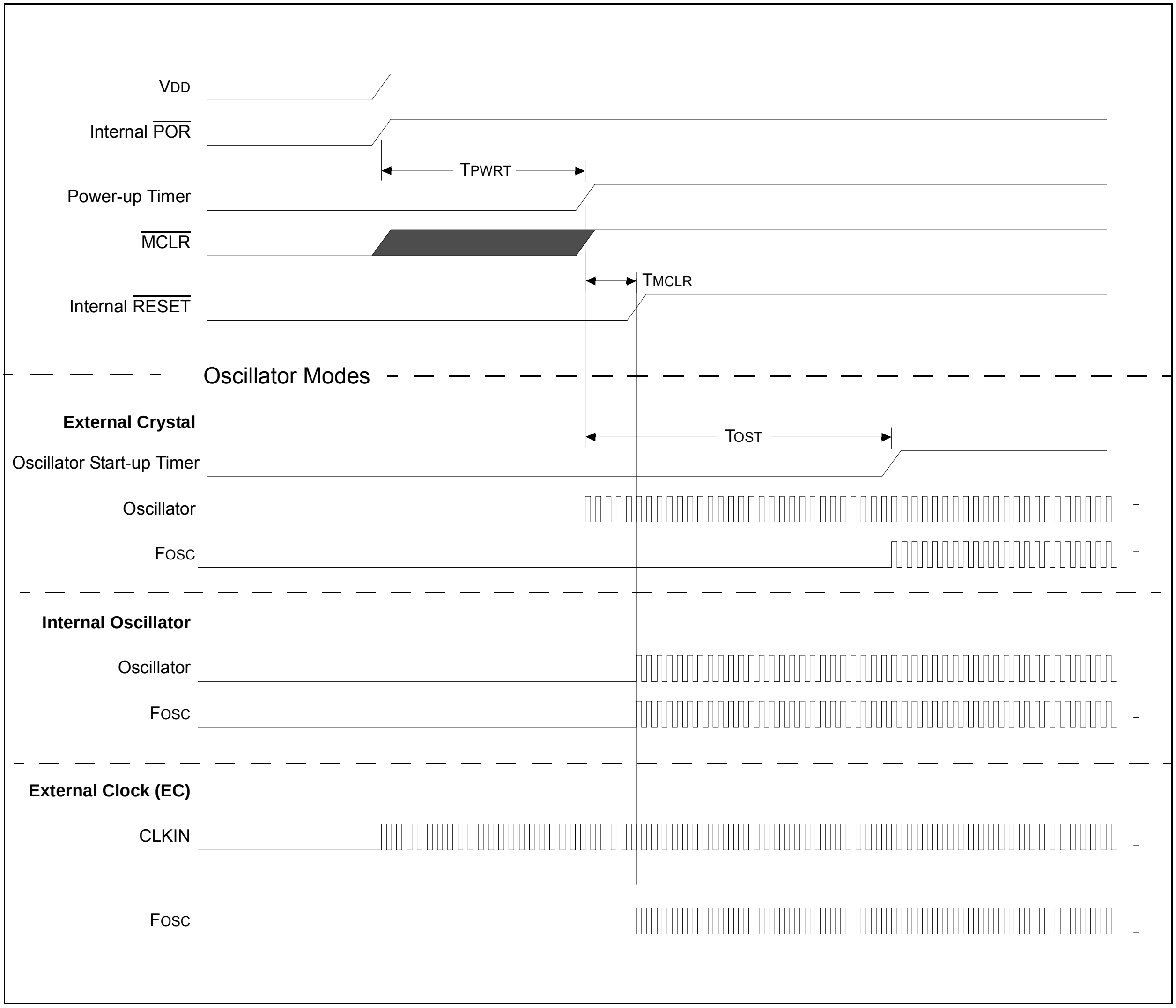
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. $\overline{\text{MCLR}}$ must be released (if enabled).
3. Oscillator start-up timer runs to completion (if required for oscillator source).

The total time-out will vary based on oscillator configuration and Power-up Timer Configuration. See **Section 7.0 "Oscillator Module"** for more information.

The Power-up Timer and oscillator start-up timer run independently of $\overline{\text{MCLR}}$ Reset. If $\overline{\text{MCLR}}$ is kept low long enough, the Power-up Timer will expire. Upon bringing $\overline{\text{MCLR}}$ high, the device will begin execution after 10 FOSC cycles (see Figure 6-3). This is useful for testing purposes or to synchronize more than one device operating in parallel.

FIGURE 6-3: RESET START-UP SEQUENCE



7.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (ECH, ECM, ECL mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes).

There is also a secondary oscillator block which is optimized for a 32.768 kHz external clock source, which can be used as an alternate clock source.

There are two internal oscillator blocks:

- HFINTOSC
- LFINTOSC

The HFINTOSC can produce clock frequencies from 1-16 MHz. The LFINTOSC generates a 31 kHz clock frequency.

There is a PLL that can be used by the external oscillator. See **7.2.1.4 “4x PLL”** for more details. Additionally, there is a PLL that can be used by the HFINTOSC at certain frequencies. See **Section 7.2.2.2 “2x PLL”** for more details.

7.2.1 EXTERNAL CLOCK SOURCES

An external clock source can be used as the device system clock by performing one of the following actions:

- Program the RSTOSC<2:0> bits in the Configuration Words to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the NOSC<2:0> and NDIV<3:0> bits in the OSCCON1 register to switch the system clock source.

See **Section 7.3 “Clock Switching”** for more information.

7.2.1.1 EC Mode

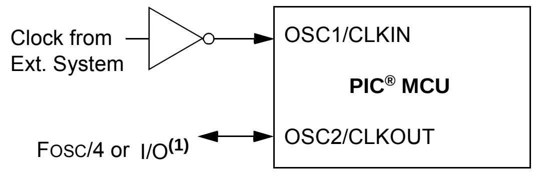
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the CLKIN input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. Figure 7-2 shows the pin connections for EC mode.

EC mode has three power modes to select from through Configuration Words:

- ECH – High power, <= 32 MHz
- ECM – Medium power, <= 8 MHz
- ECL – Low power, <= 0.1 MHz

The Oscillator Start-up Timer (OST) is disabled when EC mode is selected. Therefore, there is no delay in operation after a Power-on Reset (POR) or wake-up from Sleep. Because the PIC® MCU design is fully static, stopping the external clock input will have the effect of halting the device while leaving all data intact. Upon restarting the external clock, the device will resume operation as if no time had elapsed.

FIGURE 7-2: EXTERNAL CLOCK (EC) MODE OPERATION



Note 1: Output depends upon $\overline{\text{CLKOUTEN}}$ bit of the Configuration Words.

7.2.1.3 Oscillator Start-up Timer (OST)

If the oscillator module is configured for LP, XT or HS modes, the Oscillator Start-up Timer (OST) counts 1024 oscillations from OSC1. This occurs following a Power-on Reset (POR), Brown-out Reset (BOR), or a wake-up from Sleep. The OST ensures that the oscillator circuit, using a quartz crystal resonator or ceramic resonator, has started and is providing a stable system clock to the oscillator module.

7.2.1.4 4x PLL

The oscillator module contains a PLL that can be used with external clock sources to provide a system clock source. The input frequency for the PLL must fall within specifications. See the PLL Clock Timing Specifications in Table 35-9.

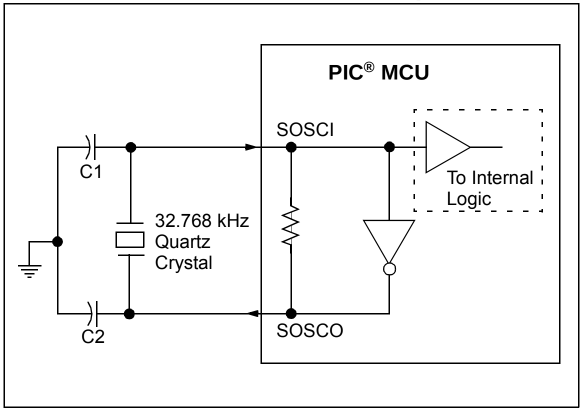
The PLL may be enabled for use by one of two methods:

1. Program the RSTOSC bits in the Configuration Word 1 to enable the EXTOSC with 4x PLL.
2. Write the NOSC<2:0> bits in the OSCCON1 register to enable the EXTOSC with 4x PLL.

7.2.1.5 Secondary Oscillator

The secondary oscillator is a separate oscillator block that can be used as an alternate system clock source. The secondary oscillator is optimized for 32.768 kHz, and can be used with an external crystal oscillator connected to the SOSCI and SOSCO device pins, or an external clock source connected to the SOSCIN pin. The secondary oscillator can be selected during run-time using clock switching. Refer to **Section 7.3 "Clock Switching"** for more information.

FIGURE 7-5: QUARTZ CRYSTAL OPERATION (SECONDARY OSCILLATOR)



Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

2: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

3: For oscillator design assistance, reference the following Microchip Application Notes:

- AN826, "*Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices*" (DS00826)
- AN849, "*Basic PICmicro® Oscillator Design*" (DS00849)
- AN943, "*Practical PICmicro® Oscillator Analysis and Design*" (DS00943)
- AN949, "*Making Your Oscillator Work*" (DS00949)
- TB097, "*Interfacing a Micro Crystal MS1V-T1K 32.768 kHz Tuning Fork Crystal to a PIC16F690/SS*" (DS91097)
- AN1288, "*Design Practices for Low-Power External Oscillators*" (DS01288)

REGISTER 8-4: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TMR6IE	C2IE	C1IE	NVMIE	SSP2IE	BCL2IE	TMR4IE	NCO1IE
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **TMR6IE:** TMR6 to PR6 Match Interrupt Enable bit
1 = TMR6 to PR6 match interrupt is enabled
0 = TMR6 to PR6 match is not enabled

bit 6 **C2IE:** Comparator C2 Interrupt Enable bit
1 = Enables the Comparator C2 interrupt
0 = Disables the Comparator C2 interrupt

bit 5 **C1IE:** Comparator C1 Interrupt Enable bit
1 = Enables the Comparator C1 interrupt
0 = Disables the Comparator C1 interrupt

bit 4 **NVMIE:** NVM Interrupt Enable Bit
1 = ENVM task complete interrupt enabled
0 = NVM interrupt not enabled

bit 3 **SSP2IE:** Master Synchronous Serial Port (MSSP2) Interrupt Enable bit
1 = Enables the MSSP2 interrupt
0 = Disables the MSSP2 interrupt

bit 2 **BCL2IE:** MSSP2 Bus Collision Interrupt Enable bit
1 = MSSP bus collision interrupt enabled
0 = MSSP bus collision interrupt not enabled

bit 1 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
1 = TMR4 to PR4 match interrupt is enabled
0 = TMR4 to PR4 match is not enabled

bit 0 **NCO1IE:** NCO Interrupt Enable bit
1 = NCO rollover interrupt enabled
0 = NCO rollover interrupt not enabled

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

REGISTER 8-7: PIR0: PERIPHERAL INTERRUPT REQUEST REGISTER 0

U-0	U-0	R/W/HS-0/0	R-0	U-0	U-0	U-0	R/W/HS-0/0
—	—	TMR0IF	IOCIF[1]	—	—	—	INTF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HS= Hardware Set

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 4 **IOCIF:** Interrupt-on-Change Interrupt Flag bit (read-only)
1 = An enabled edge was detected by the IOC module. One of the IOCF bits is set.
0 = No enabled edge is was detected by the IOC module. None of the IOCF bits is set.
Pins are individually masked via IOCxP and IOCxN.

bit 3-1 **Unimplemented:** Read as ‘0’

bit 0 **INTF:** INT External Interrupt Flag bit[1]
1 = The INT external interrupt occurred (must be cleared in software)
0 = The INT external interrupt did not occur

Note 1: The IOCIF bit is the logical OR of all the IOCAF-IOCCF flags. Therefore, to clear the IOCIF flag, application firmware must clear all of the IOCAF-IOCCF register bits.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

19.2 Register Definitions: PWM Control

REGISTER 19-1: PWMxCON: PWM CONTROL REGISTER

R/W-0/0	U-0	R-0	R/W-0/0	U-0	U-0	U-0	U-0
PWMxEN	—	PWMxOUT	PWMxPOL	—	—	—	—
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **PWMxEN:** PWM Module Enable bit
1 = PWM module is enabled
0 = PWM module is disabled

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **PWMxOUT:** PWM Module Output Level when bit is read.

bit 4 **PWMxPOL:** PWMx Output Polarity Select bit
1 = PWM output is active-low.
0 = PWM output is active-high.

bit 3-0 **Unimplemented:** Read as ‘0’

REGISTER 19-2: PWMxDCH: PWM DUTY CYCLE HIGH BITS

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
PWMxDC<9:2>							
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **PWMxDC<9:2>:** PWM Duty Cycle Most Significant bits
These bits are the MSbs of the PWM duty cycle. The two LSbs are found in PWMxDCL Register.

REGISTER 19-3: PWMxDCL: PWM DUTY CYCLE LOW BITS

R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0	U-0	U-0
PWMxDC<1:0>		—	—	—	—	—	—
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **PWMxDC<1:0>:** PWM Duty Cycle Least Significant bits
These bits are the LSbs of the PWM duty cycle. The MSbs are found in PWMxDCH Register.

bit 5-0 **Unimplemented:** Read as ‘0’

REGISTER 22-3: ADACT: A/D AUTO-CONVERSION TRIGGER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ADACT<4:0>				
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **ADACT<4:0>:** Auto-Conversion Trigger Selection bits[(1)]

`10001`	=	Timer5 overflow[(2)]
`10000`	=	Timer3 overflow[(2)]
`1111`	=	CCP4
`1110`	=	CCP3
`1101`	=	CCP2
`1100`	=	CCP1
`1011`	=	CLC4
`1010`	=	CLC3
`1001`	=	CLC2
`1000`	=	CLC1
`0111`	=	Comparator C2
`0110`	=	Comparator C1
`0101`	=	Timer2-PR2 match
`0100`	=	Timer1 overflow[(2)]
`0011`	=	Timer0 overflow[(2)]
`0010`	=	Timer6-PR6 match
`0001`	=	Timer4-PR4 match
`0000`	=	No auto-conversion trigger selected

Note 1: This is a rising edge sensitive input for all sources.

2: Trigger corresponds to when the peripheral’s interrupt flag is set.

26.1.6 SYNCHRONOUS MODE

When the T0ASYNC bit of the T0CON1 register is clear (T0ASYNC = 0), the counter clock is synchronized to the system oscillator (FOSC/4). When operating in Synchronous mode, the counter clock frequency cannot exceed FOSC/4.

26.2 Clock Source Selection

The T0CS<2:0> bits of the T0CON1 register are used to select the clock source for Timer0. Register 26-4 displays the clock source selections.

26.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, Timer0 operates as a timer and will increment on multiples of the clock source, as determined by the Timer0 prescaler.

26.2.2 EXTERNAL CLOCK SOURCE

When an external clock source is selected, Timer0 can operate as either a timer or a counter. Timer0 will increment on multiples of the rising edge of the external clock source, as determined by the Timer0 prescaler.

26.3 Programmable Prescaler

A software programmable prescaler is available for exclusive use with Timer0. There are 16 prescaler options for Timer0 ranging in powers of two from 1:1 to 1:32768. The prescaler values are selected using the T0CKPS<3:0> bits of the T0CON1 register.

The prescaler is not directly readable or writable. Clearing the prescaler register can be done by writing to the TMR0L register, the T0CON0 register, or the T0CON1 register.

26.4 Programmable Postscaler

A software programmable postscaler (output divider) is available for exclusive use with Timer0. There are 16 postscaler options for Timer0 ranging from 1:1 to 1:16. The postscaler values are selected using the T0OUTPS<3:0> bits of the T0CON0 register.

The postscaler is not directly readable or writable. Clearing the postscaler register can be done by writing to the TMR0L register, the T0CON0 register, or the T0CON1 register.

26.5 Operation During Sleep

When operating synchronously, Timer0 will halt. When operating asynchronously, Timer0 will continue to increment and wake the device from Sleep (if Timer0 interrupts are enabled) provided that the input clock source is active.

26.6 Timer0 Interrupts

The Timer0 Interrupt Flag bit (TMR0IF) is set when either of the following conditions occur:

- 8-bit TMR0L matches the TMR0H value
- 16-bit TMR0 rolls over from FFFFh

When the postscaler bits (T0OUTPS<3:0>) are set to 1:1 operation (no division), the T0IF Flag bit will be set with every TMR0 match or rollover. In general, the TMR0IF Flag bit will be set every T0OUTPS +1 matches or rollovers.

If Timer0 interrupts are enabled (TMR0IE bit of the PIE0 register = 1), the CPU will be interrupted and the device may wake from Sleep (see **Section 26.5 "Operation During Sleep"** for more details).

26.7 Timer0 Output

The Timer0 output can be routed to any I/O pin via the RxyPPS output selection register (see **Section 13.0 "Peripheral Pin Select (PPS) Module"** for additional information). The Timer0 output can also be used by other peripherals, such as the auto-conversion trigger of the Analog-to-Digital Converter. Finally, the Timer0 output can be monitored through software via the Timer0 Output bit (T0OUT) of the T0CON0 register (Register 26-3).

TMR0_out will be one postscaled clock period when a match occurs between TMR0L and TMR0H in 8-bit mode, or when TMR0 rolls over in 16-bit mode. When a match condition occurs, the Timer0 output will toggle every T0OUTPS + 1 match. The total Timer0 period takes two match events to occur, and creates a 50% duty cycle output.

27.10 Register Definitions: Timer1/3/5 Control

REGISTER 27-1: TxCON[1]: TIMERx CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	U-0	R/W-0/u
TMRxCS<1:0>		TxCKPS<1:0>		TxSOSC	$\overline{\text{TxSYNC}}$	—	TMRxON
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-6 **TMRxCS<1:0>:** Timerx Clock Source Select bits

11 = Timerx clock Source is LFINTOSC
10 = Timerx clock source is pin or oscillator:
If TxSOSC = 0:
External clock from TxCKIPPS pin (on the rising edge)
If TxSOSC = 1:
Clock from SOSC, either crystal oscillator on TxSOSCI/TxSOSCO pins, or SOSCIN input
01 = Timerx clock source is system clock (FOSC)
00 = Timerx clock source is instruction clock (FOSC/4)

bit 5-4 **TxCKPS<1:0>:** Timerx Input Clock Prescale Select bits

11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **TxSOSC:** LP Oscillator Enable Control bit

1 = SOSC requested as the clock source
0 = TxCKI enabled as the clock source

bit 2 **$\overline{\text{TxSYNC}}$:** Timer1 Synchronization Control bit

TMRxCS<1:0> = 1x
1 = Do not synchronize external clock input
0 = Synchronize external clock input with system clock
TMRxCS<1:0> = 0x
This bit is ignored. Timer1 uses the internal clock and no additional synchronization is performed.

bit 1 **Unimplemented:** Read as '0'

bit 0 **TMRxON:** Timer1 On bit

1 = Enables Timerx
0 = Stops Timerx and clears Timerx gate flip-flop

Note 1: 'x' refers to either '1', '3' or '5' for the respective Timer1/3/5 registers.

30.4.5 START CONDITION

The I²C specification defines a Start condition as a transition of SDA from a high to a low state while SCL line is high. A Start condition is always generated by the master and signifies the transition of the bus from an Idle to an Active state. Figure 30-12 shows wave forms for Start and Stop conditions.

30.4.6 STOP CONDITION

A Stop condition is a transition of the SDA line from low-to-high state while the SCL line is high.

Note:	At least one SCL low time must appear before a Stop is valid, therefore, if the SDA line goes low then high again while the SCL line stays high, only the Start condition is detected.

30.4.7 RESTART CONDITION

A Restart is valid any time that a Stop would be valid. A master can issue a Restart if it wishes to hold the bus after terminating the current transfer. A Restart has the same effect on the slave that a Start would, resetting all slave logic and preparing it to clock in an address. The master may want to address the same or another slave. Figure 30-13 shows the wave form for a Restart condition.

In 10-bit Addressing Slave mode a Restart is required for the master to clock data out of the addressed slave. Once a slave has been fully addressed, matching both high and low address bytes, the master can issue a Restart and the high address byte with the $R/\overline{W}$ bit set. The slave logic will then hold the clock and prepare to clock out data.

30.4.8 START/STOP CONDITION INTERRUPT MASKING

The SCIE and PCIE bits of the SSPxCON3 register can enable the generation of an interrupt in Slave modes that do not typically support this function. Slave modes where interrupt on Start and Stop detect are already enabled, these bits will have no effect.

FIGURE 30-12: I²C START AND STOP CONDITIONS

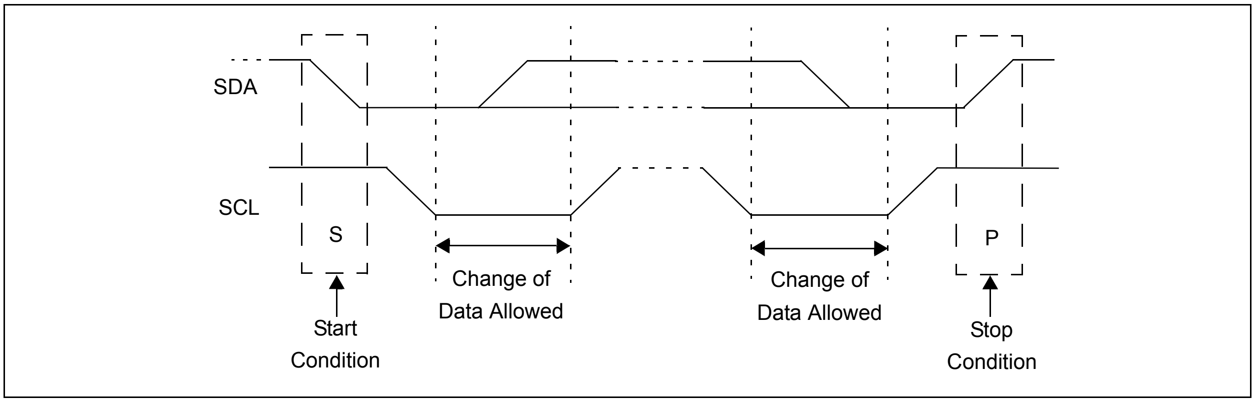
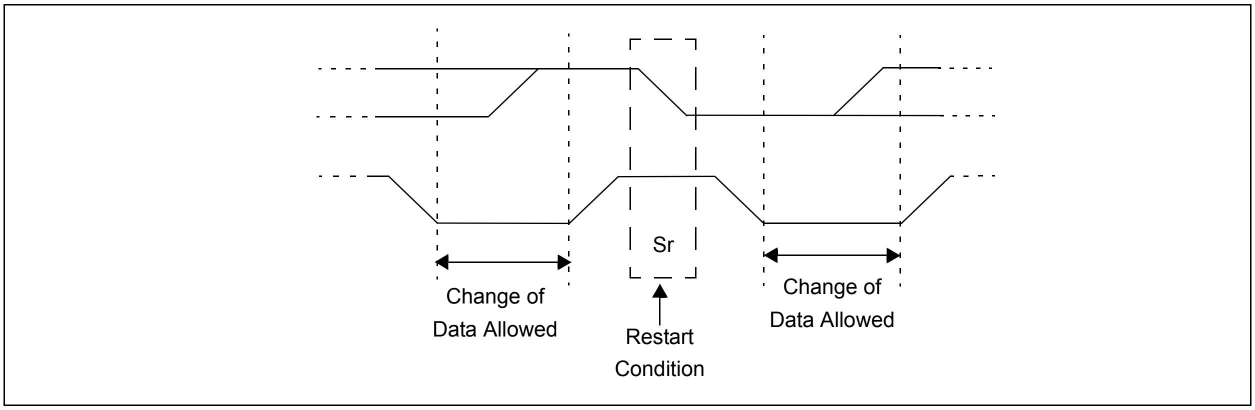


FIGURE 30-13: I²C RESTART CONDITION



30.5.2 SLAVE RECEPTION

When the R/$\overline{W}$ bit of a matching received address byte is clear, the R/$\overline{W}$ bit of the SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and acknowledged.

When the Overflow condition exists for a received address, then not Acknowledge is given. An Overflow condition is defined as either bit BF of the SSPxSTAT register is set, or bit SSPOV of the SSPxCON1 register is set. The BOEN bit of the SSPxCON3 register modifies this operation. For more information see Register 30-4.

An MSSP interrupt is generated for each transferred data byte. Flag bit, SSPxIF, must be cleared by software.

When the SEN bit of the SSPxCON2 register is set, SCL will be held low (clock stretch) following each received byte. The clock must be released by setting the CKP bit of the SSPxCON1 register.

30.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSPx module configured as an I^{2}C slave in 7-bit Addressing mode. Figure 30-14 and Figure 30-15 is used as a visual reference for this description.

This is a step-by-step process of what typically must be done to accomplish I^{2}C communication.

1. Start bit detected.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with R/$\overline{W}$ bit clear is received.
4. The slave pulls SDA low sending an $\overline{ACK}$ to the master, and sets SSPxIF bit.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF clearing the BF flag.
7. If SEN = 1; Slave software sets CKP bit to release the SCL line.
8. The master clocks out a data byte.
9. Slave drives SDA low sending an $\overline{ACK}$ to the master, and sets SSPxIF bit.
10. Software clears SSPxIF.
11. Software reads the received byte from SSPxBUF clearing BF.
12. Steps 8-12 are repeated for all received bytes from the master.
13. Master sends Stop condition, setting P bit of SSPxSTAT, and the bus goes idle.

30.5.2.2 7-bit Reception with AHEN and DHEN

Slave device reception with AHEN and DHEN set operate the same as without these options with extra interrupts and clock stretching added after the eighth falling edge of SCL. These additional interrupts allow time for the slave software to decide whether it wants to $\overline{ACK}$ the receive address or data byte.

This list describes the steps that need to be taken by slave software to use these options for I^{2}C communication. Figure 30-16 displays a module using both address and data holding. Figure 30-17 includes the operation with the SEN bit of the SSPxCON2 register set.

1. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Matching address with R/$\overline{W}$ bit clear is clocked in. SSPxIF is set and CKP cleared after the eighth falling edge of SCL.
3. Slave clears the SSPxIF.
4. Slave can look at the ACKTIM bit of the SSPxCON3 register to determine if the SSPxIF was after or before the $\overline{ACK}$.
5. Slave reads the address value from SSPxBUF, clearing the BF flag.
6. Slave sets $\overline{ACK}$ value clocked out to the master by setting ACKDT.
7. Slave releases the clock by setting CKP.
8. SSPxIF is set after an $\overline{ACK}$, not after a NACK.
9. If SEN = 1 the slave hardware will stretch the clock after the $\overline{ACK}$.
10. Slave clears SSPxIF.

Note: SSPxIF is still set after the ninth falling edge of SCL even if there is no clock stretching and BF has been cleared. Only if NACK is sent to master is SSPIF not set

11. SSPxIF set and CKP cleared after eighth falling edge of SCL for a received data byte.
12. Slave looks at ACKTIM bit of SSPxCON3 to determine the source of the interrupt.
13. Slave reads the received data from SSPxBUF clearing BF.
14. Steps 7-14 are the same for each received data byte.
15. Communication is ended by either the slave sending an $\overline{ACK}$ = 1, or the master sending a Stop condition. If a Stop is sent and Interrupt on Stop Detect is disabled, the slave will only know by polling the P bit of the SSPxSTAT register.

TABLE 30-2: MSSP CLOCK RATE W/BRG

FOSC	FCY	BRG Value	FCLOCK (2 Rollovers of BRG)
32 MHz	8 MHz	13h	400 kHz
32 MHz	8 MHz	19h	308 kHz
32 MHz	8 MHz	4Fh	100 kHz
16 MHz	4 MHz	09h	400 kHz
16 MHz	4 MHz	0Ch	308 kHz
16 MHz	4 MHz	27h	100 kHz
4 MHz	1 MHz	09h	100 kHz

Note: Refer to the I/O port electrical specifications in Table 35-4 to ensure the system is designed to support IOL requirements.

REGISTER 30-2: SSPxCON1: SSP CONTROL REGISTER 1

R/C/HS-0/0	R/C/HS-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
WCOL	SSPOV[1]	SSPEN	CKP	SSPM<3:0>			
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HS = Bit is set by hardware C = User cleared

bit 7 **WCOL:** Write Collision Detect bit (Transmit mode only)
1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit[1]
In SPI mode:
1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSPBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register (must be cleared in software).
0 = No overflow
In I^{2}C mode:
1 = A byte is received while the SSPBUF register is still holding the previous byte. SSPOV is a “don’t care” in Transmit mode (must be cleared in software).
0 = No overflow

bit 5 **SSPEN:** Synchronous Serial Port Enable bit
In both modes, when enabled, the following pins must be properly configured as input or output
In SPI mode:
1 = Enables serial port and configures SCK, SDO, SDI and $\overline{\text{SS}}$ as the source of the serial port pins[2]
0 = Disables serial port and configures these pins as I/O port pins
In I^{2}C mode:
1 = Enables the serial port and configures the SDA and SCL pins as the source of the serial port pins[3]
0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit
In SPI mode:
1 = Idle state for clock is a high level
0 = Idle state for clock is a low level
In I^{2}C Slave mode:
SCL release control
1 = Enable clock
0 = Holds clock low (clock stretch). (Used to ensure data setup time.)
In I^{2}C Master mode:
Unused in this mode

REGISTER 30-3: SSPxCON2: SSPx CONTROL REGISTER 2 (I²C MODE ONLY)[1]

R/W-0/0	R/HS/HC-0	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0
GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HC = Cleared by hardware S = User set

bit 7 **GCEN:** General Call Enable bit (in I²C Slave mode only)
1 = Enable interrupt when a general call address (0x00 or 00h) is received in the SSPSR
0 = General call address disabled

bit 6 **ACKSTAT:** Acknowledge Status bit (in I²C mode only)
1 = Acknowledge was not received
0 = Acknowledge was received

bit 5 **ACKDT:** Acknowledge Data bit (in I²C mode only)
In Receive mode:
Value transmitted when the user initiates an Acknowledge sequence at the end of a receive
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit (in I²C Master mode only)
In Master Receive mode:
1 = Initiate Acknowledge sequence on SDA and SCL pins, and transmit ACKDT data bit. Automatically cleared by hardware.
0 = Acknowledge sequence idle

bit 3 **RCEN:** Receive Enable bit (in I²C Master mode only)
1 = Enables Receive mode for I²C
0 = Receive idle

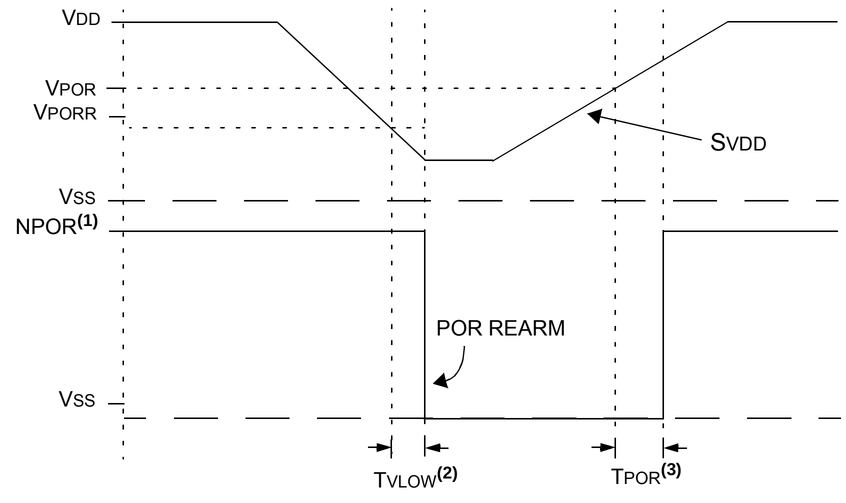
bit 2 **PEN:** Stop Condition Enable bit (in I²C Master mode only)
1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit (in I²C Master mode only)
1 = Initiate Repeated Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 **SEN:** Start Condition Enable/Stretch Enable bit
In Master mode:
1 = Initiate Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Start condition Idle
In Slave mode:
1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)
0 = Clock stretching is disabled

Note 1: For bits ACKEN, RCEN, PEN, RSEN, SEN: If the I²C module is not in the idle state, these bits may not be set (no spooling) and the SSPBUF may not be written.

FIGURE 35-3: POR AND POR REARM WITH SLOW RISING VDD

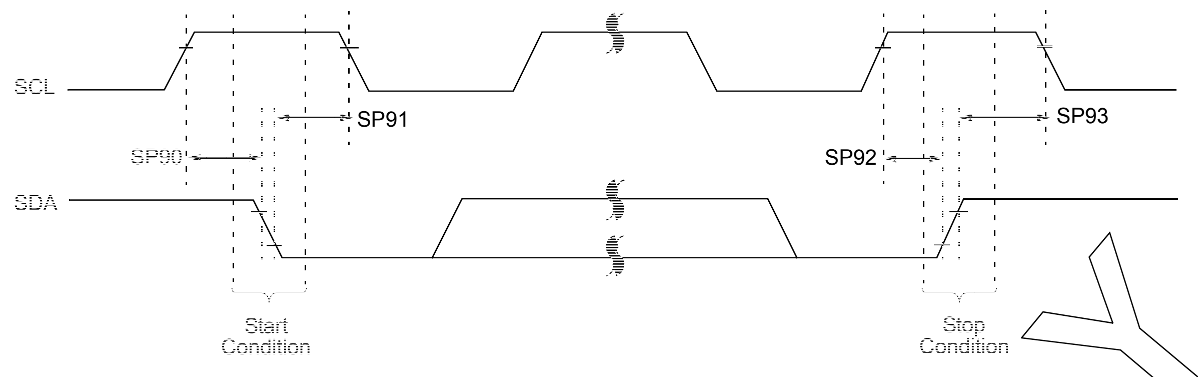


Note 1: When NPOR is low, the device is held in Reset.

2: TPOR 1 μs typical.

3: TVLOW 2.7 μs typical.

FIGURE 35-21: I²C BUS START/STOP BITS TIMING



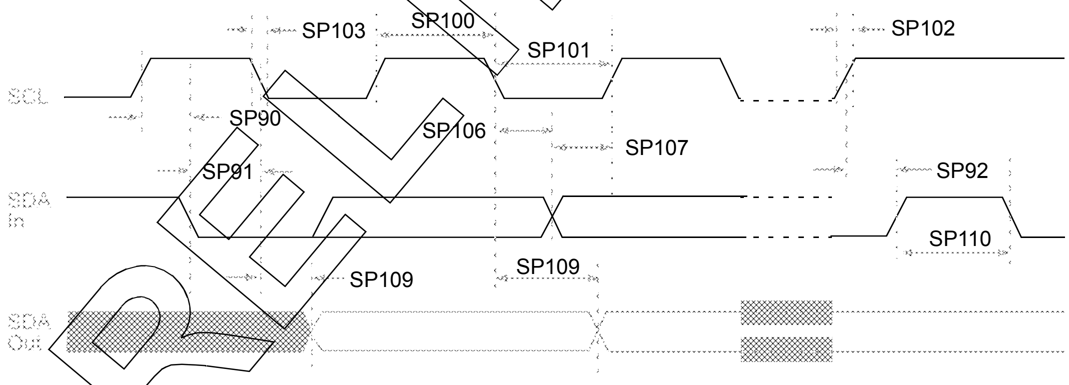
Note: Refer to Figure 35-4 for Load conditions.

TABLE 35-23: I²C BUS START/STOP BITS CHARACTERISTICS

Standard Operating Conditions (unless otherwise stated)								
Param. No.	**Symbol**	**Characteristic**		**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
SP90*	TSU:STA	Start condition	100 kHz mode	4700	—	—	ns	Only relevant for Repeated Start condition
		Setup time	400 kHz mode	600	—	—		
SP91*	THD:STA	Start condition	100 kHz mode	4000	—	—	ns	After this period, the first clock pulse is generated
		Hold time	400 kHz mode	600	—	—		
SP92*	TSU:STO	Stop condition	100 kHz mode	4700	—	—	ns	
		Setup time	400 kHz mode	600	—	—		
SP93	THD:STO	Stop condition	100 kHz mode	4000	—	—	ns	
		Hold time	400 kHz mode	600	—	—		

* These parameters are characterized but not tested.

FIGURE 35-22: I²C BUS DATA TIMING



Note: Refer to Figure 35-4 for Load conditions.