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	20MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, WDT
Number of I/O	18
Program Memory Size	3.5KB (2K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	20-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f687-e-ss

TABLE 1-2: PINOUT DESCRIPTION – PIC16F677 (CONTINUED)

Name	Function	Input Type	Output Type	Description
RB7	RB7	TTL	CMOS	General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up.
RC0/AN4/C2IN+	RC0	ST	CMOS	General purpose I/O.
	AN4	AN	—	A/D Channel 4 input.
	C2IN+	AN	—	Comparator C2 non-inverting input.
RC1/AN5/C12IN1-	RC1	ST	CMOS	General purpose I/O.
	AN5	AN	—	A/D Channel 5 input.
	C12IN1-	AN	—	Comparator C1 or C2 inverting input.
RC2/AN6/C12IN2-	RC2	ST	CMOS	General purpose I/O.
	AN6	AN	—	A/D Channel 6 input.
	C12IN2-	AN	—	Comparator C1 or C2 inverting input.
RC3/AN7/C12IN3-	RC3	ST	CMOS	General purpose I/O.
	AN7	AN	—	A/D Channel 7 input.
	C12IN3-	AN	—	Comparator C1 or C2 inverting input.
RC4/C2OUT	RC4	ST	CMOS	General purpose I/O.
	C2OUT	—	CMOS	Comparator C2 output.
RC5	RC5	ST	CMOS	General purpose I/O.
RC6/AN8/$\overline{SS}$	RC6	ST	CMOS	General purpose I/O.
	AN8	AN	—	A/D Channel 8 input.
	$\overline{SS}$	ST	—	Slave Select input.
RC7/AN9/SDO	RC7	ST	CMOS	General purpose I/O.
	AN9	AN	—	A/D Channel 9 input.
	SDO	—	CMOS	SPI data output.
VSS	VSS	Power	—	Ground reference.
VDD	VDD	Power	—	Positive supply.

Legend: AN = Analog input or output CMOS=CMOS compatible input or output
TTL = TTL compatible input ST= Schmitt Trigger input with CMOS levels
HV = High Voltage XTAL= Crystal

4.5.1 RC0/AN4/C2IN+

The RC0 is configurable to function as one of the following:

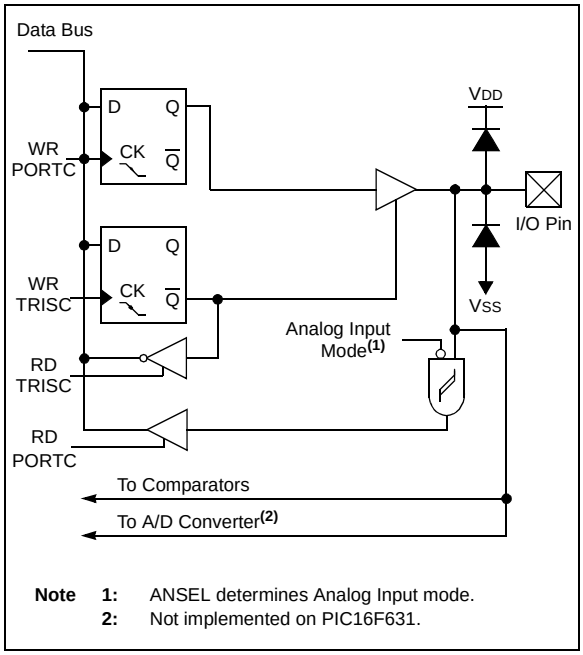
- a general purpose I/O
- an analog input for the ADC (except PIC16F631)
- an analog input to Comparator C2

4.5.2 RC1/AN5/C12IN1-

The RC1 is configurable to function as one of the following:

- a general purpose I/O
- an analog input for the ADC
- an analog input to Comparator C1 or C2

FIGURE 4-11: BLOCK DIAGRAM OF RC0 AND RC1



Note 1: ANSEL determines Analog Input mode.
2: Not implemented on PIC16F631.

4.5.3 RC2/AN6/C12IN2-/P1D

The RC2/AN6/P1D(1) is configurable to function as one of the following:

- a general purpose I/O
- an analog input for the ADC (except PIC16F631)
- a PWM output
- an analog input to Comparator C1 or C2

Note 1: P1D is available on PIC16F685/ PIC16F690 only.

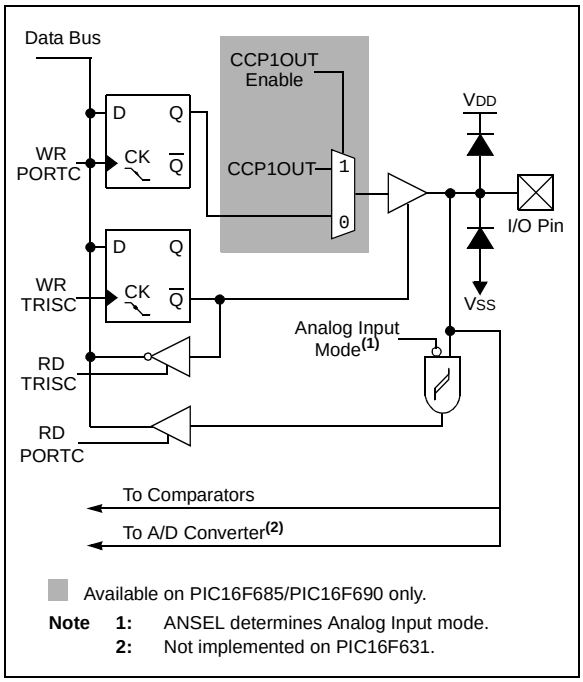
4.5.4 RC3/AN7/C12IN3-/P1C

The RC3/AN7/P1C(1) is configurable to function as one of the following:

- a general purpose I/O
- an analog input for the ADC (except PIC16F631)
- a PWM output
- a PWM output
- an analog input to Comparator C1 or C2

Note 1: P1C is available on PIC16F685/ PIC16F690 only.

FIGURE 4-12: BLOCK DIAGRAM OF RC2 AND RC3



Available on PIC16F685/PIC16F690 only.

Note 1: ANSEL determines Analog Input mode.
2: Not implemented on PIC16F631.

4.5.5 RC4/C2OUT/P1B

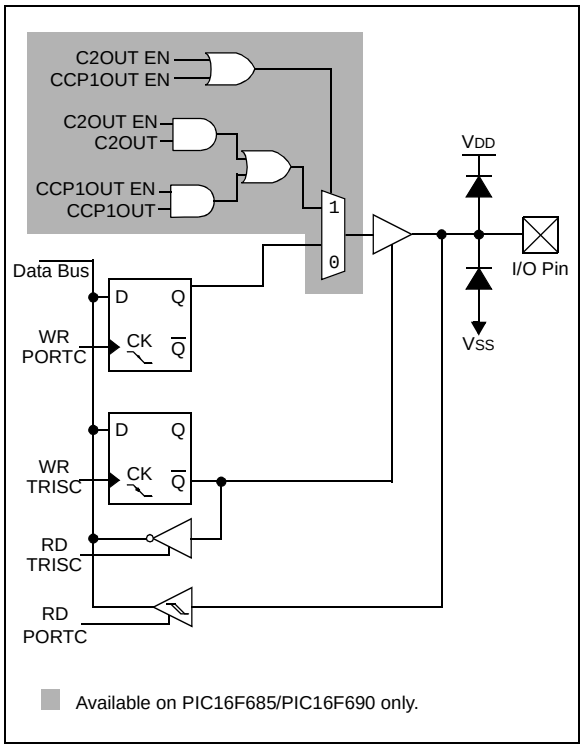
The RC4/C2OUT/P1B(1, 2) is configurable to function as one of the following:

- a general purpose I/O
- a digital output from Comparator C2
- a PWM output

Note 1: Enabling both C2OUT and P1B will cause a conflict on RC4 and create unpredictable results. Therefore, if C2OUT is enabled, the ECCP+ can not be used in Half-Bridge or Full-Bridge mode and vise-versa.

2: P1B is available on PIC16F685/ PIC16F690 only.

FIGURE 4-13: BLOCK DIAGRAM OF RC4



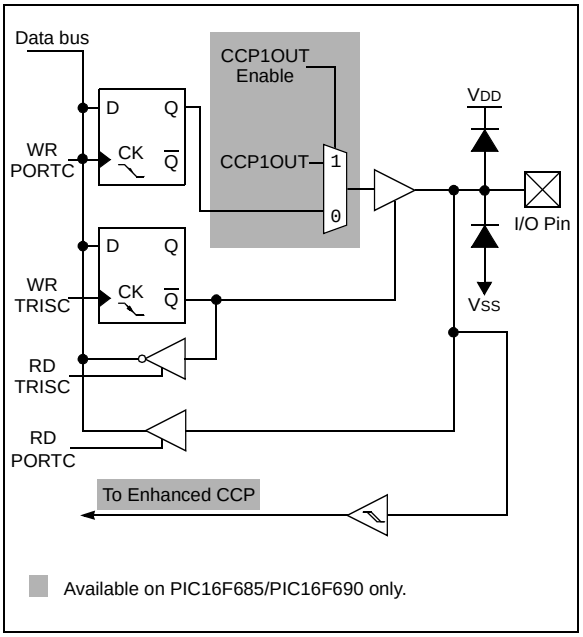
4.5.6 RC5/CCP1/P1A

The RC5/CCP1/P1A(1) is configurable to function as one of the following:

- a general purpose I/O
- a digital input/output for the Enhanced CCP
- a PWM output

Note 1: CCP1 and P1A are available on PIC16F685/PIC16F690 only.

FIGURE 4-14: BLOCK DIAGRAM OF RC5



REGISTER 7-1: T2CON: TIMER 2 CONTROL REGISTER(1)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
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 6-3 **TOUTPS<3:0>:** Timer2 Output Postscaler Select bits
0000 =1:1 Postscaler
0001 =1:2 Postscaler
0010 =1:3 Postscaler
0011 =1:4 Postscaler
0100 =1:5 Postscaler
0101 =1:6 Postscaler
0110 =1:7 Postscaler
0111 =1:8 Postscaler
1000 =1:9 Postscaler
1001 =1:10 Postscaler
1010 =1:11 Postscaler
1011 =1:12 Postscaler
1100 =1:13 Postscaler
1101 =1:14 Postscaler
1110 =1:15 Postscaler
1111 =1:16 Postscaler

bit 2 **TMR2ON:** Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 =Prescaler is 1
01 =Prescaler is 4
1x =Prescaler is 16

Note 1: PIC16F685/PIC16F690 only.

TABLE 7-1: SUMMARY OF ASSOCIATED TIMER2(1) REGISTERS

Name	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
INTCON	GIE	PEIE	T0IE	INTE	RABIE	T0IF	INTF	RABIF	0000 000x	0000 000x
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
PR2	Timer2 Module Period Register								1111 1111	1111 1111
TMR2	Holding Register for the 8-bit TMR2 Register								0000 0000	0000 0000
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000

Legend: x = unknown, u = unchanged, – = unimplemented read as '0'. Shaded cells are not used for Timer2 module.
Note 1: PIC16F685/PIC16F690 only.

9.1 ADC Configuration

When configuring and using the ADC the following functions must be considered:

- Port configuration
- Channel selection
- ADC voltage reference selection
- ADC conversion clock source
- Interrupt control
- Results formatting

9.1.1 PORT CONFIGURATION

The ADC can be used to convert both analog and digital signals. When converting analog signals, the I/O pin should be configured for analog by setting the associated TRIS and ANSEL bits. See the corresponding port section for more information.

Note:	Analog voltages on any pin that is defined as a digital input may cause the input buffer to conduct excess current.

9.1.2 CHANNEL SELECTION

The CHS bits of the ADCON0 register determine which channel is connected to the sample and hold circuit.

When changing channels, a delay is required before starting the next conversion. Refer to **Section 9.2 "ADC Operation"** for more information.

9.1.3 ADC VOLTAGE REFERENCE

The VCFG bit of the ADCON0 register provides control of the positive voltage reference. The positive voltage reference can be either VDD or an external voltage source. The negative voltage reference is always connected to the ground reference.

9.1.4 CONVERSION CLOCK

The source of the conversion clock is software selectable via the ADCS bits of the ADCON1 register. There are seven possible clock options:

- FOSC/2
- FOSC/4
- FOSC/8
- FOSC/16
- FOSC/32
- FOSC/64
- FRC (dedicated internal oscillator)

The time to complete one bit conversion is defined as TAD. One full 10-bit conversion requires 11 TAD periods as shown in Figure 9-2.

For correct conversion, the appropriate TAD specification must be met. See A/D conversion requirements in **Section 17.0 "Electrical Specifications"** for more information. Table 9-1 gives examples of appropriate ADC clock selections.

Note:	Unless using the FRC, any changes in the system clock frequency will change the ADC clock frequency, which may adversely affect the ADC result.

11.4.4 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the ECCPASx bits of the ECCPAS register. A shutdown event may be generated by:

- A logic '0' on the INT pin
- Comparator C1
- Comparator C2
- Setting the ECCPASE bit in firmware

A shutdown condition is indicated by the ECCPASE (Auto-Shutdown Event Status) bit of the ECCPAS register. If the bit is a '0', the PWM pins are operating normally. If the bit is a '1', the PWM outputs are in the shutdown state.

When a shutdown event occurs, two things happen:

The ECCPASE bit is set to '1'. The ECCPASE will remain set until cleared in firmware or an auto-restart occurs (see **Section 11.4.5 "Auto-Restart Mode"**).

The enabled PWM pins are asynchronously placed in their shutdown states. The PWM output pins are grouped into pairs [P1A/P1C] and [P1B/P1D]. The state of each pin pair is determined by the PSSAC and PSSBD bits of the ECCPAS register. Each pin pair may be placed into one of three states:

- Drive logic '1'
- Drive logic '0'
- Tri-state (high-impedance)

FIGURE 11-14: AUTO-SHUTDOWN BLOCK DIAGRAM

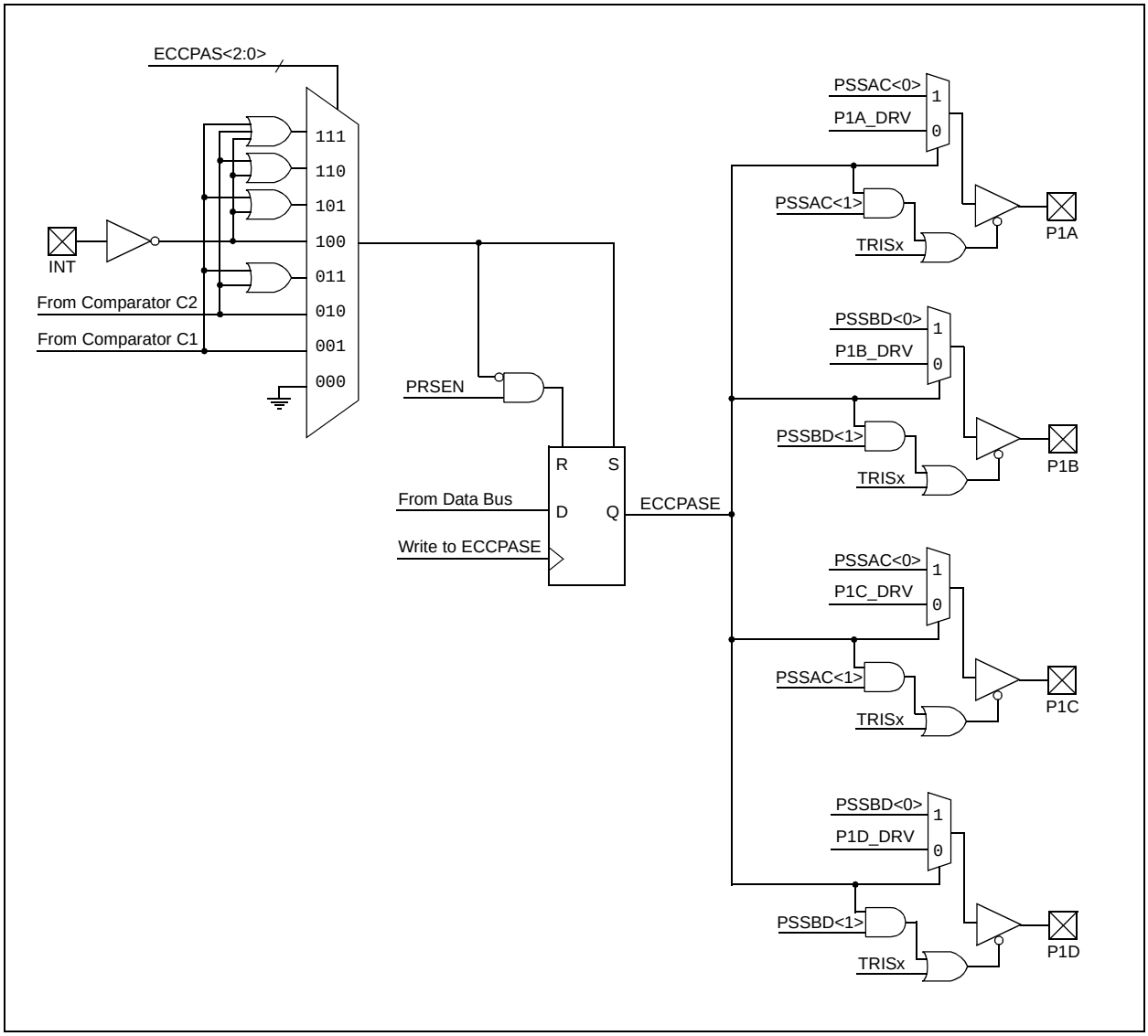


TABLE 12-5: BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 20.000 MHz			FOSC = 18.432 MHz			FOSC = 11.0592 MHz			FOSC = 8.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	—	—	—	—	—	—	—	—	—
1200	1221	1.73	255	1200	0.00	239	1200	0.00	143	1202	0.16	103
2400	2404	0.16	129	2400	0.00	119	2400	0.00	71	2404	0.16	51
9600	9470	-1.36	32	9600	0.00	29	9600	0.00	17	9615	0.16	12
10417	10417	0.00	29	10286	-1.26	27	10165	-2.42	16	10417	0.00	11
19.2k	19.53k	1.73	15	19.20k	0.00	14	19.20k	0.00	8	—	—	—
57.6k	—	—	—	57.60k	0.00	7	57.60k	0.00	2	—	—	—
115.2k	—	—	—	—	—	—	—	—	—	—	—	—

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 4.000 MHz			FOSC = 3.6864 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300	0.16	207	300	0.00	191	300	0.16	103	300	0.16	51
1200	1202	0.16	51	1200	0.00	47	1202	0.16	25	1202	0.16	12
2400	2404	0.16	25	2400	0.00	23	2404	0.16	12	—	—	—
9600	—	—	—	9600	0.00	5	—	—	—	—	—	—
10417	10417	0.00	5	—	—	—	10417	0.00	2	—	—	—
19.2k	—	—	—	19.20k	0.00	2	—	—	—	—	—	—
57.6k	—	—	—	57.60k	0.00	0	—	—	—	—	—	—
115.2k	—	—	—	—	—	—	—	—	—	—	—	—

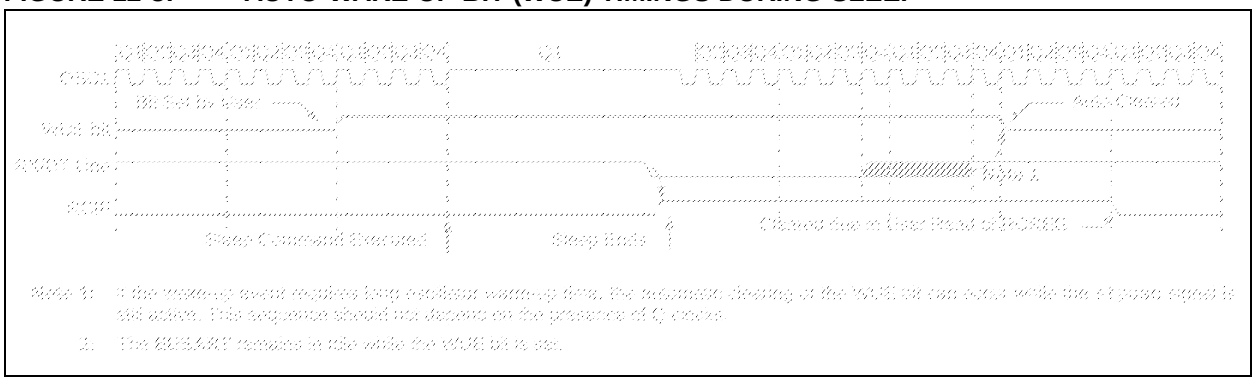
BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 0											
	FOSC = 20.000 MHz			FOSC = 18.432 MHz			FOSC = 11.0592 MHz			FOSC = 8.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	—	—	—	—	—	—	—	—	—
1200	—	—	—	—	—	—	—	—	—	—	—	—
2400	—	—	—	—	—	—	—	—	—	2404	0.16	207
9600	9615	0.16	129	9600	0.00	119	9600	0.00	71	9615	0.16	51
10417	10417	0.00	119	10378	-0.37	110	10473	0.53	65	10417	0.00	47
19.2k	19.23k	0.16	64	19.20k	0.00	59	19.20k	0.00	35	19231	0.16	25
57.6k	56.82k	-1.36	21	57.60k	0.00	19	57.60k	0.00	11	55556	-3.55	8
115.2k	113.64k	-1.36	10	115.2k	0.00	9	115.2k	0.00	5	—	—	—

TABLE 12-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 20.000 MHz			FOSC = 18.432 MHz			FOSC = 11.0592 MHz			FOSC = 8.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.00	16665	300.0	0.00	15359	300.0	0.00	9215	300.0	0.00	6666
1200	1200	-0.01	4166	1200	0.00	3839	1200	0.00	2303	1200	-0.02	1666
2400	2400	0.02	2082	2400	0.00	1919	2400	0.00	1151	2401	0.04	832
9600	9597	-0.03	520	9600	0.00	479	9600	0.00	287	9615	0.16	207
10417	10417	0.00	479	10425	0.08	441	10433	0.16	264	10417	0	191
19.2k	19.23k	0.16	259	19.20k	0.00	239	19.20k	0.00	143	19.23k	0.16	103
57.6k	57.47k	-0.22	86	57.60k	0.00	79	57.60k	0.00	47	57.14k	-0.79	34
115.2k	116.3k	0.94	42	115.2k	0.00	39	115.2k	0.00	23	117.6k	2.12	16

BAUD RATE	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 4.000 MHz			FOSC = 3.6864 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	300.0	0.01	3332	300.0	0.00	3071	299.9	-0.02	1666	300.1	0.04	832
1200	1200	0.04	832	1200	0.00	767	1199	-0.08	416	1202	0.16	207
2400	2398	0.08	416	2400	0.00	383	2404	0.16	207	2404	0.16	103
9600	9615	0.16	103	9600	0.00	95	9615	0.16	51	9615	0.16	25
10417	10417	0.00	95	10473	0.53	87	10417	0.00	47	10417	0.00	23
19.2k	19.23k	0.16	51	19.20k	0.00	47	19.23k	0.16	25	19.23k	0.16	12
57.6k	58.82k	2.12	16	57.60k	0.00	15	55.56k	-3.55	8	—	—	—
115.2k	111.1k	-3.55	8	115.2k	0.00	7	—	—	—	—	—	—

FIGURE 12-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



12.3.3 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 ‘0’ bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TXSTA register. The Break character transmission is then initiated by a write to the TXREG. The value of data written to TXREG will be ignored and all ‘0’s will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TXSTA register indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 12-9 for the timing of the Break character sequence.

12.3.3.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXREG with a dummy character to initiate transmission (the value is ignored).
4. Write ‘55h’ to TXREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXREG becomes empty, as indicated by the TXIF, the next data byte can be written to TXREG.

12.3.4 RECEIVING A BREAK CHARACTER

The Enhanced EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RCSTA register and the Received data as indicated by RCREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

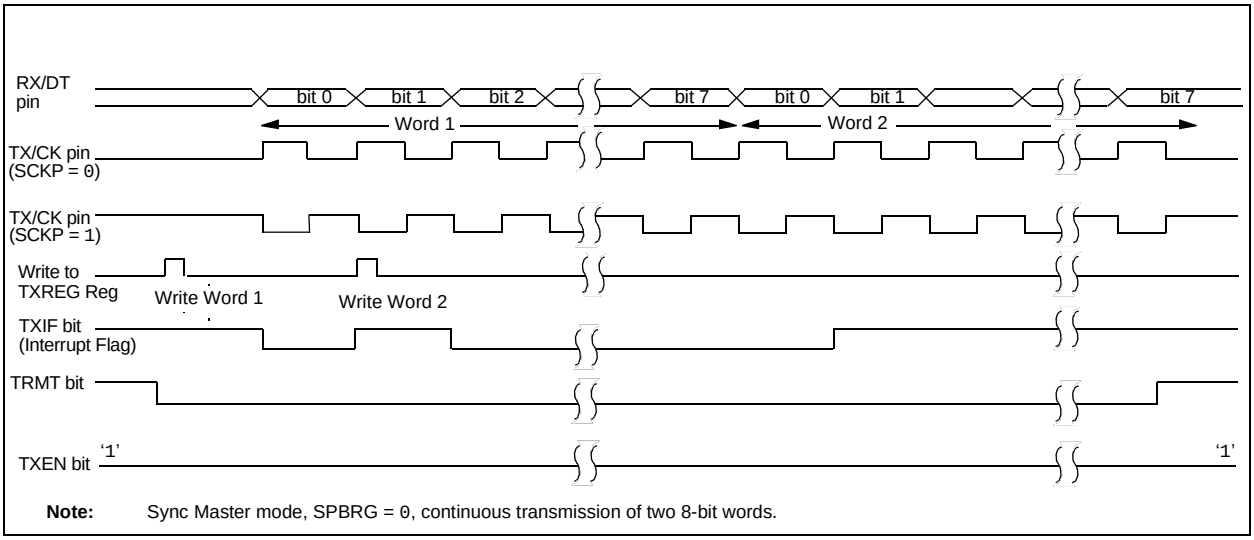
A Break character has been received when;

- RCIF bit is set
- FERR bit is set
- RCREG = 00h

The second method uses the Auto-Wake-up feature described in **Section 12.3.2 “Auto-Wake-up on Break”**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUDCTL register before placing the EUSART in Sleep mode.

FIGURE 12-10: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRG = 0, continuous transmission of two 8-bit words.

FIGURE 12-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

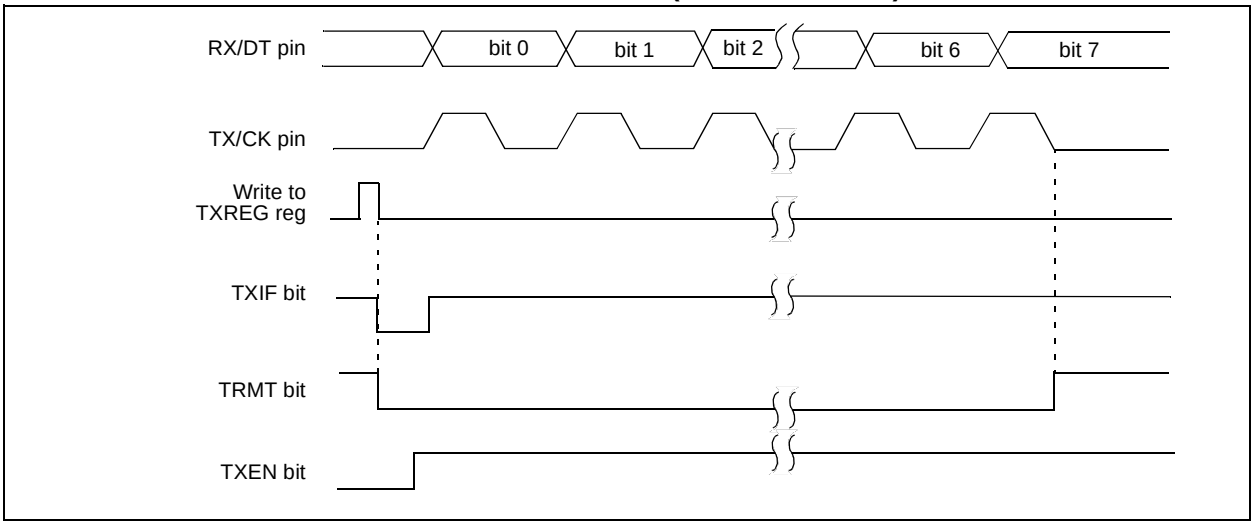


TABLE 12-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

Name	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
BAUDCTL	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	01-0 0-00
INTCON	GIE	PEIE	T0IE	INTE	RABIE	T0IF	INTF	RABIF	0000 000x	0000 000x
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
RCREG	EUSART Receive Data Register								0000 0000	0000 0000
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
SPBRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0	0000 0000	0000 0000
SPBRGH	BRG15	BRG14	BRG13	BRG12	BRG11	BRG10	BRG9	BRG8	0000 0000	0000 0000
TRISB	TRISB7	TRISB6	TRISB5	TRISB4					1111 ----	1111 ----
TXREG	EUSART Transmit Data Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010

Legend: x = unknown, – = unimplemented read as '0'. Shaded cells are not used for Synchronous Master Transmission.

12.4.2 SYNCHRONOUS SLAVE MODE

The following bits are used to configure the EUSART for Synchronous slave operation:

- SYNC = 1
- CSRC = 0
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXSTA register configures the device for synchronous operation. Clearing the CSRC bit of the TXSTA register configures the device as a slave. Clearing the SREN and CREN bits of the RCSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCSTA register enables the EUSART. If the RX/DT or TX/CK pins are shared with an analog peripheral the analog I/O functions must be disabled by clearing the corresponding ANSEL bits.

12.4.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see **Section 12.4.1.3 "Synchronous Master Transmission")**, except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

1. The first character will immediately transfer to the TSR register and transmit.
2. The second word will remain in TXREG register.
3. The TXIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXREG register will transfer the second character to the TSR and the TXIF bit will now be set.
5. If the PEIE and TXIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

12.4.2.2 Synchronous Slave Transmission Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the CREN and SREN bits.
3. If interrupts are desired, set the TXIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
4. If 9-bit transmission is desired, set the TX9 bit.
5. Enable transmission by setting the TXEN bit.
6. If 9-bit transmission is selected, insert the Most Significant bit into the TX9D bit.
7. Start transmission by writing the Least Significant eight bits to the TXREG register.

TABLE 12-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	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
BAUDCTL	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	01-0 0-00
INTCON	GIE	PEIE	T0IE	INTE	RABIE	T0IF	INTF	RABIF	0000 000x	0000 000x
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
RCREG	EUSART Receive Data Register								0000 0000	0000 0000
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
SPBRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0	0000 0000	0000 0000
SPBRGH	BRG15	BRG14	BRG13	BRG12	BRG11	BRG10	BRG9	BRG8	0000 0000	0000 0000
TRISB	TRISB7	TRISB6	TRISB5	TRISB4					1111 ----	1111 ----
TXREG	EUSART Transmit Data Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010

Legend: x = unknown, – = unimplemented read as '0'. Shaded cells are not used for Synchronous Slave Transmission.

13.6 Slave Mode

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCK. When the last bit is latched, the SSPIF interrupt flag bit is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCK pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device will wake-up from Sleep.

13.7 Slave Select Synchronization

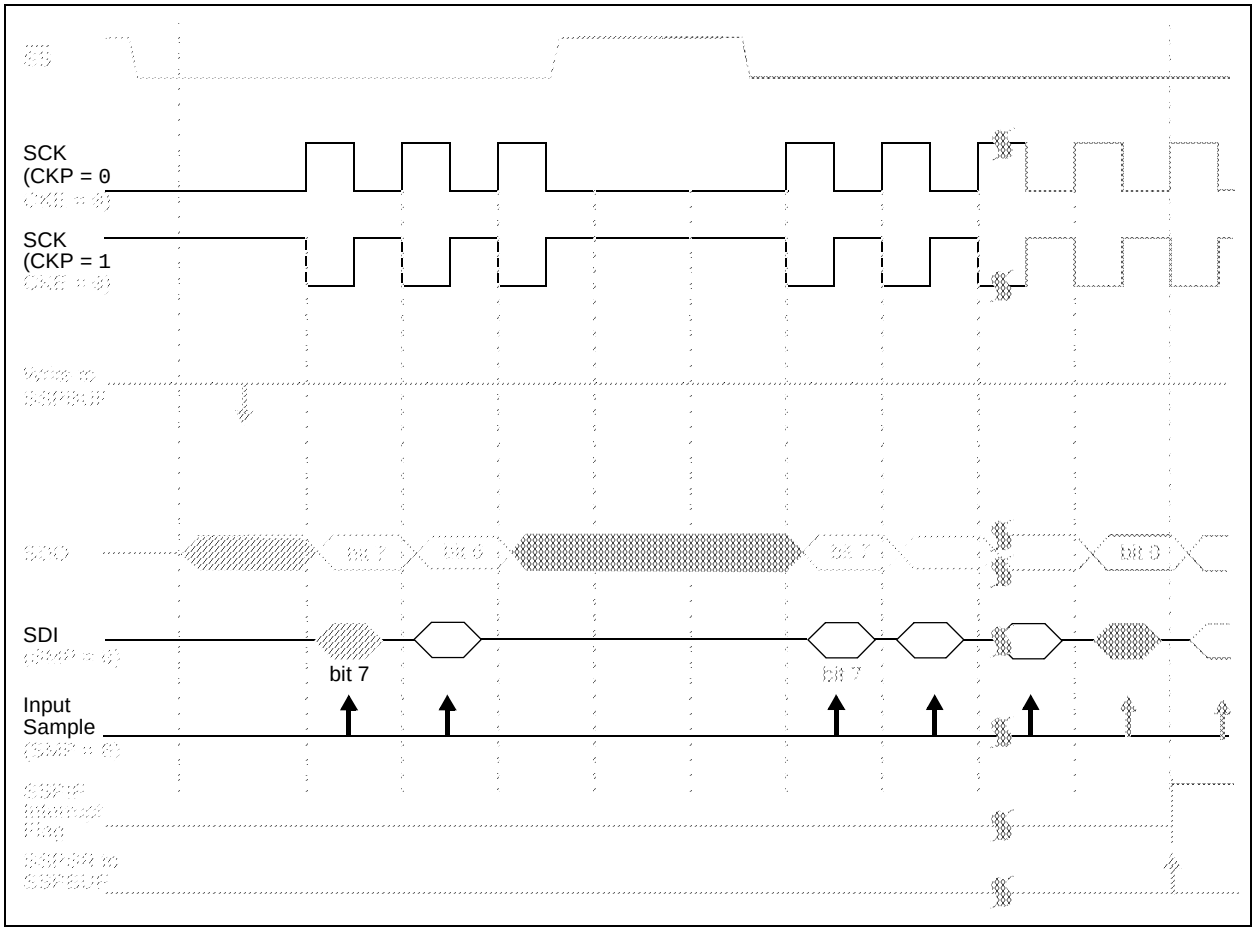
The $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = 04h). The pin must not be driven low for the $\overline{SS}$ pin to function as an input. The data latch must be high. When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven. When the $\overline{SS}$ pin goes high, the SDO pin is no longer driven, even if in the middle of a transmitted byte, and becomes a floating output. External pull-up/pull-down resistors may be desirable, depending on the application.

Note 1:	When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to V_{DD}.
2:	If the SPI is used in Slave Mode with CKE set, then the $\overline{SS}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SS}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDO pin can be connected to the SDI pin. When the SPI needs to operate as a receiver, the SDO pin can be configured as an input. This disables transmissions from the SDO. The SDI can always be left as an input (SDI function) since it cannot create a bus conflict.

FIGURE 13-4: SLAVE SYNCHRONIZATION WAVEFORM



13.12.4 TRANSMISSION

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit, and pin RB6/SCK/SCL is held low. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then, pin RB6/SCK/SCL should be enabled by setting bit CKP (SSPCON<4>). The master must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master by stretching the clock. The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 13-10).

An SSP interrupt is generated for each data transfer byte. Flag bit SSPIF must be cleared in software, and the SSPSTAT register is used to determine the status of the byte. Flag bit SSPIF is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the $\overline{ACK}$ pulse from the master receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not $\overline{ACK}$), then the data transfer is complete. When the $\overline{ACK}$ is latched by the slave, the slave logic is reset (resets SSPSTAT register) and the slave then monitors for another occurrence of the Start bit. If the SDA line was low ($\overline{ACK}$), the transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then pin RB6/SCK/SCL should be enabled by setting bit CKP.

FIGURE 13-10: I²C™ WAVEFORMS FOR TRANSMISSION (7-BIT ADDRESS)

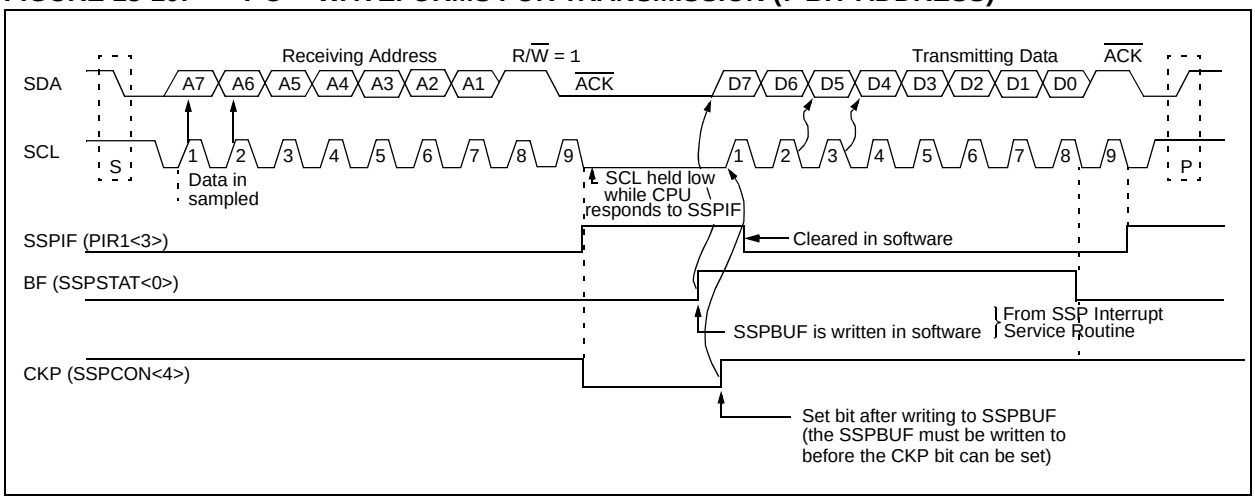


TABLE 14-4: INITIALIZATION CONDITION FOR REGISTER (CONTINUED)

Register	Address	Power-on Reset	$\overline{\text{MCLR}}$ Reset WDT Reset (Continued) Brown-out Reset(1)	Wake-up from Sleep through Interrupt Wake-up from Sleep through WDT Time-out
TRISB	86h/186h	1111 ----	1111 ----	uuuu ----
TRISC	87h/187h	1111 1111	1111 1111	uuuu uuuu
PIE1	8Ch	-000 0000	-000 0000	-uuu uuuu
PIE2	8Dh	0000 ----	0000 ----	uuuu uuuu
PCON	8Eh	--01 --0x	--0u --uq[1, 5]	--uu --uu
OSCCON	8Fh	-110 q000	-110 q000	-uuu uuuu
OSCTUNE	90h	---0 0000	---u uuuu	---u uuuu
PR2	92h	1111 1111	1111 1111	uuuu uuuu
SSPADD	93h	0000 0000	1111 1111	uuuu uuuu
SSPMSK(6)	93h	---- ----	1111 1111	uuuu uuuu
SSPSTAT	94h	0000 0000	1111 1111	uuuu uuuu
WPUA	95h	--11 -111	--11 -111	uuuu uuuu
IOCA	96h	--00 0000	--00 0000	--uu uuuu
WDTCON	97h	---0 1000	---0 1000	---u uuuu
TXSTA	98h	0000 0010	0000 0010	uuuu uuuu
SPBRG	99h	0000 0000	0000 0000	uuuu uuuu
SPBRGH	9Ah	0000 0000	0000 0000	uuuu uuuu
BAUDCTL	9Bh	01-0 0-00	01-0 0-00	uu-u u-uu
ADRESL	9Eh	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON1	9Fh	-000 ----	-000 ----	-uuu ----
EEDAT	10Ch	0000 0000	0000 0000	uuuu uuuu
EEADR	10Dh	0000 0000	0000 0000	uuuu uuuu
EEDATH	10Eh	--00 0000	--00 0000	--uu uuuu
EEADRH	10Fh	---- 0000	---- 0000	---- uuuu
WPUB	115h	1111 ----	1111 ----	uuuu ----
IOCB	116h	0000 ----	0000 ----	uuuu ----
VRCON	118h	0000 0000	0000 0000	uuuu uuuu
CM1CON0	119h	0000 -000	0000 -000	uuuu -uuu
CM2CON0	11Ah	0000 -000	0000 -000	uuuu -uuu
CM2CON1	11Bh	00-- --00	00-- --10	uu-- --uu
ANSEL	11Eh	1111 1111	1111 1111	uuuu uuuu
ANSELH	11Fh	---- 1111	---- 1111	---- uuuu
EECON1	18Ch	x--- x000	u--- q000	---- uuuu
EECON2	18Dh	---- ----	---- ----	---- ----
PSTRCON	19Dh	---0 0001	---0 0001	---u uuuu
SRCON	19EH	0000 00--	0000 00--	uuuu uu--

Legend: u = unchanged, x = unknown, – = unimplemented bit, reads as '0', q = value depends on condition.

Note 1: If VDD goes too low, Power-on Reset will be activated and registers will be affected differently.

2: One or more bits in INTCON and/or PIR1 will be affected (to cause wake-up).

3: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

4: See Table 14-5 for Reset value for specific condition.

5: If Reset was due to brown-out, then bit 0 = 0. All other Resets will cause bit 0 = u.

6: Accessible only when SSPM<3:0> = 1001.

TABLE 14-5: INITIALIZATION CONDITION FOR SPECIAL REGISTERS

14.4 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users may wish to save key registers during an interrupt (e.g., W and STATUS registers). This must be implemented in software.

Since the upper 16 bytes of all GPR banks are common in the PIC16F631/677/685/687/689/690 (see Figures 2-2 and 2-3), temporary holding registers, W_TEMP and STATUS_TEMP, should be placed in here. These 16 locations do not require banking and therefore, make it easier to context save and restore. The same code shown in Example 14-1 can be used to:

- Store the W register
- Store the STATUS register
- Execute the ISR code
- Restore the Status (and Bank Select Bit register)
- Restore the W register

Note: The PIC16F631/677/685/687/689/690 normally does not require saving the PCLATH. However, if computed GOTO's are used in the ISR and the main code, the PCLATH must be saved and restored in the ISR.

EXAMPLE 14-1: SAVING STATUS AND W REGISTERS IN RAM

```
MOVWF  W_TEMP           ;Copy W to TEMP register
SWAPF  STATUS,W         ;Swap status to be saved into W
CLRF   STATUS           ;bank 0, regardless of current bank, Clears IRP,RP1,RP0
MOVWF  STATUS_TEMP      ;Save status to bank zero STATUS_TEMP register
:
:(ISR)                  ;Insert user code here
:
SWAPF  STATUS_TEMP,W    ;Swap STATUS_TEMP register into W
                        ;(sets bank to original state)
MOVWF  STATUS           ;Move W into STATUS register
SWAPF  W_TEMP,F         ;Swap W_TEMP
SWAPF  W_TEMP,W         ;Swap W_TEMP into W
```

14.6 Power-Down Mode (Sleep)

The Power-Down mode is entered by executing a SLEEP instruction.

If the Watchdog Timer is enabled:

- WDT will be cleared but keeps running.
- $\overline{PD}$ bit in the STATUS register is cleared.
- $\overline{TO}$ bit is set.
- Oscillator driver is turned off.
- I/O ports maintain the status they had before SLEEP was executed (driving high, low or high-impedance).

For lowest current consumption in this mode, all I/O pins should be either at V_{DD} or V_{SS}, with no external circuitry drawing current from the I/O pin and the comparators and CV_{REF} should be disabled. I/O pins that are high-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should also be at V_{DD} or V_{SS} for lowest current consumption. The contribution from on-chip pull-ups on PORTA should be considered.

The $\overline{MCLR}$ pin must be at a logic high level.

Note:	It should be noted that a Reset generated by a WDT time-out does not drive $\overline{MCLR}$ pin low.

14.6.1 WAKE-UP FROM SLEEP

The device can wake-up from Sleep through one of the following events:

1. External Reset input on $\overline{MCLR}$ pin.
2. Watchdog Timer Wake-up (if WDT was enabled).
3. Interrupt from RA2/INT pin, PORTA change or a peripheral interrupt.

The first event will cause a device Reset. The two latter events are considered a continuation of program execution. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of device Reset. The $\overline{PD}$ bit, which is set on power-up, is cleared when Sleep is invoked. $\overline{TO}$ bit is cleared if WDT Wake-up occurred.

The following peripheral interrupts can wake the device from Sleep:

1. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
2. ECCP Capture mode interrupt.
3. A/D conversion (when A/D clock source is FRC).
4. EEPROM write operation completion.
5. Comparator output changes state.
6. Interrupt-on-change.
7. External Interrupt from INT pin.
8. EUSART Break detect, I^2C slave.

Other peripherals cannot generate interrupts since during Sleep, no on-chip clocks are present.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up occurs regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction, then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

Note:	If the global interrupts are disabled (GIE is cleared), but any interrupt source has both its interrupt enable bit and the corresponding interrupt flag bits set, the device will immediately wake-up from Sleep. The SLEEP instruction is completely executed.

The WDT is cleared when the device wakes up from Sleep, regardless of the source of wake-up.

14.6.2 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a SLEEP instruction, the SLEEP instruction will complete as a NOP. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will not be cleared, the $\overline{TO}$ bit will not be set and the $\overline{PD}$ bit will not be cleared.
- If the interrupt occurs **during or after** the execution of a SLEEP instruction, the device will immediately wake-up from Sleep. The SLEEP instruction will be completely executed before the wake-up. Therefore, the WDT and WDT prescaler and postscaler (if enabled) will be cleared, the $\overline{TO}$ bit will be set and the $\overline{PD}$ bit will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

15.0 INSTRUCTION SET SUMMARY

The PIC16F690 instruction set is highly orthogonal and is comprised of three basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal and control** operations

Each PIC16 instruction is a 14-bit word divided into an **opcode,** which specifies the instruction type and one or more **operands,** which further specify the operation of the instruction. The formats for each of the categories is presented in Figure 15-1, while the various opcode fields are summarized in Table 15-1.

Table 15-2 lists the instructions recognized by the MPASM™ assembler.

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the W register. If 'd' is one, the result is placed in the file register specified in the instruction.

For **bit-oriented** instructions, 'b' represents a bit field designator, which selects the bit affected by the operation, while 'f' represents the address of the file in which the bit is located.

For **literal and control** operations, 'k' represents an 8-bit or 11-bit constant, or literal value.

One instruction cycle consists of four oscillator periods; for an oscillator frequency of 4 MHz, this gives a normal instruction execution time of 1 μs. All instructions are executed within a single instruction cycle, unless a conditional test is true, or the program counter is changed as a result of an instruction. When this occurs, the execution takes two instruction cycles, with the second cycle executed as a NOP.

All instruction examples use the format '0xhh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

15.1 Read-Modify-Write Operations

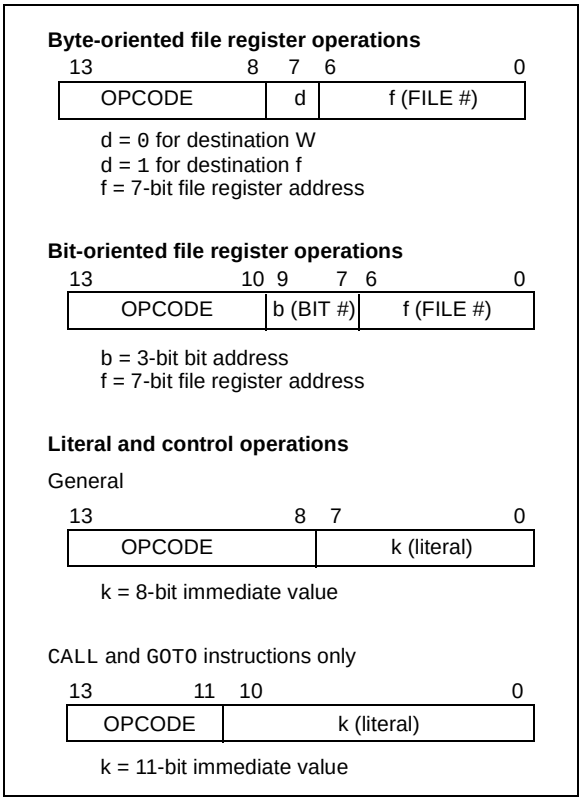
Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (RMW) operation. The register is read, the data is modified, and the result is stored according to either the instruction, or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

For example, a CLRF PORTA instruction will read PORTA, clear all the data bits, then write the result back to PORTA. This example would have the unintended consequence of clearing the condition that set the RAIF flag.

TABLE 15-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1.
PC	Program Counter
$\overline{TO}$	Time-out bit
C	Carry bit
DC	Digit carry bit
Z	Zero bit
$\overline{PD}$	Power-down bit

FIGURE 15-1: GENERAL FORMAT FOR INSTRUCTIONS



16.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® X IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICkit™ 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits and Starter Kits
- Third-party development tools

16.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

FIGURE 17-12: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)

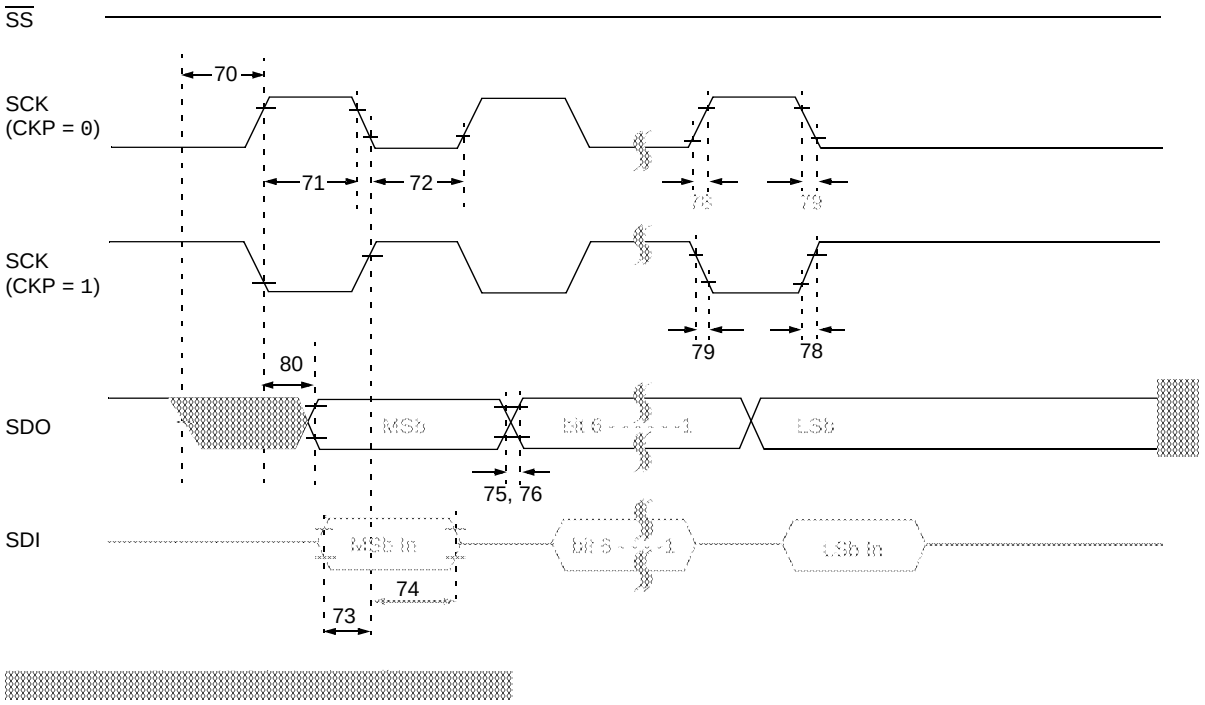
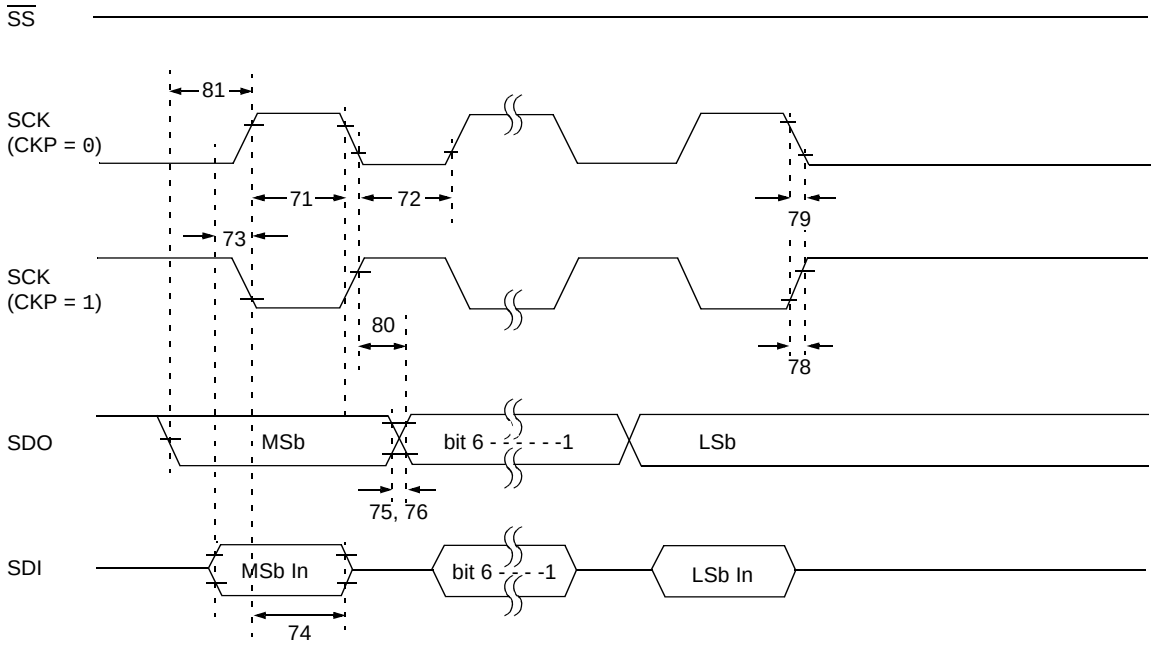


FIGURE 17-13: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)



Note: Refer to Figure 17-3 for load conditions.

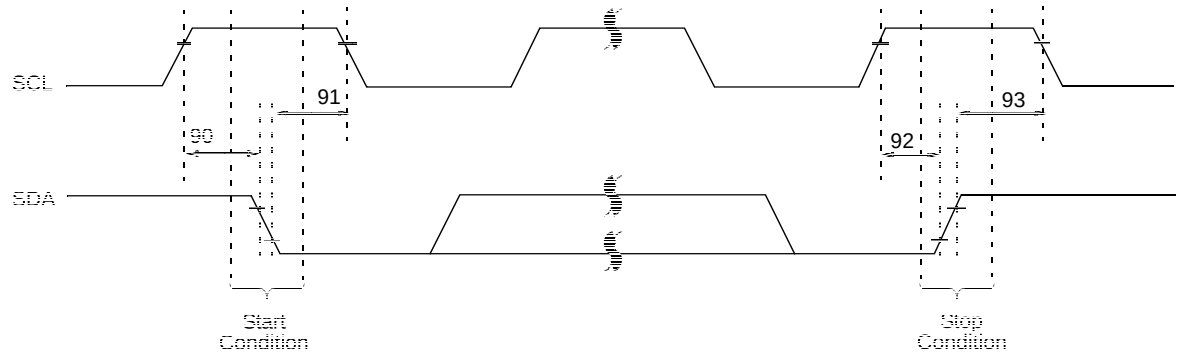
TABLE 17-12: SPI MODE REQUIREMENTS

Param No.	Symbol	Characteristic		Min.	Typ†	Max.	Units	Conditions
70*	TssL2scH, TssL2scL	$\overline{SS}\downarrow$ to SCK$\downarrow$ or SCK$\uparrow$ input		Tcy	—	—	ns	
71*	TscH	SCK input high time (Slave mode)		Tcy + 20	—	—	ns	
72*	TscL	SCK input low time (Slave mode)		Tcy + 20	—	—	ns	
73*	TdiV2scH, TdiV2scL	Setup time of SDI data input to SCK edge		100	—	—	ns	
74*	TscH2diL, TscL2diL	Hold time of SDI data input to SCK edge		100	—	—	ns	
75*	TdoR	SDO data output rise time	3.0-5.5V	—	10	25	ns	
			2.0-5.5V	—	25	50	ns	
76*	TdoF	SDO data output fall time		—	10	25	ns	
77*	TssH2doZ	$\overline{SS}\uparrow$ to SDO output high-impedance		10	—	50	ns	
78*	TscR	SCK output rise time (Master mode)	3.0-5.5V	—	10	25	ns	
			2.0-5.5V	—	25	50	ns	
79*	TscF	SCK output fall time (Master mode)		—	10	25	ns	
80*	TscH2doV, TscL2doV	SDO data output valid after SCK edge	3.0-5.5V	—	—	50	ns	
			2.0-5.5V	—	—	145	ns	
81*	TdoV2scH, TdoV2scL	SDO data output setup to SCK edge		Tcy	—	—	ns	
82*	TssL2doV	SDO data output valid after $\overline{SS}\downarrow$ edge		—	—	50	ns	
83*	TscH2ssH, TscL2ssH	$\overline{SS}\uparrow$ after SCK edge		1.5Tcy + 40	—	—	ns	

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 17-16: I^2C™ BUS START/STOP BITS TIMING



Note: Refer to Figure 17-3 for load conditions.