Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

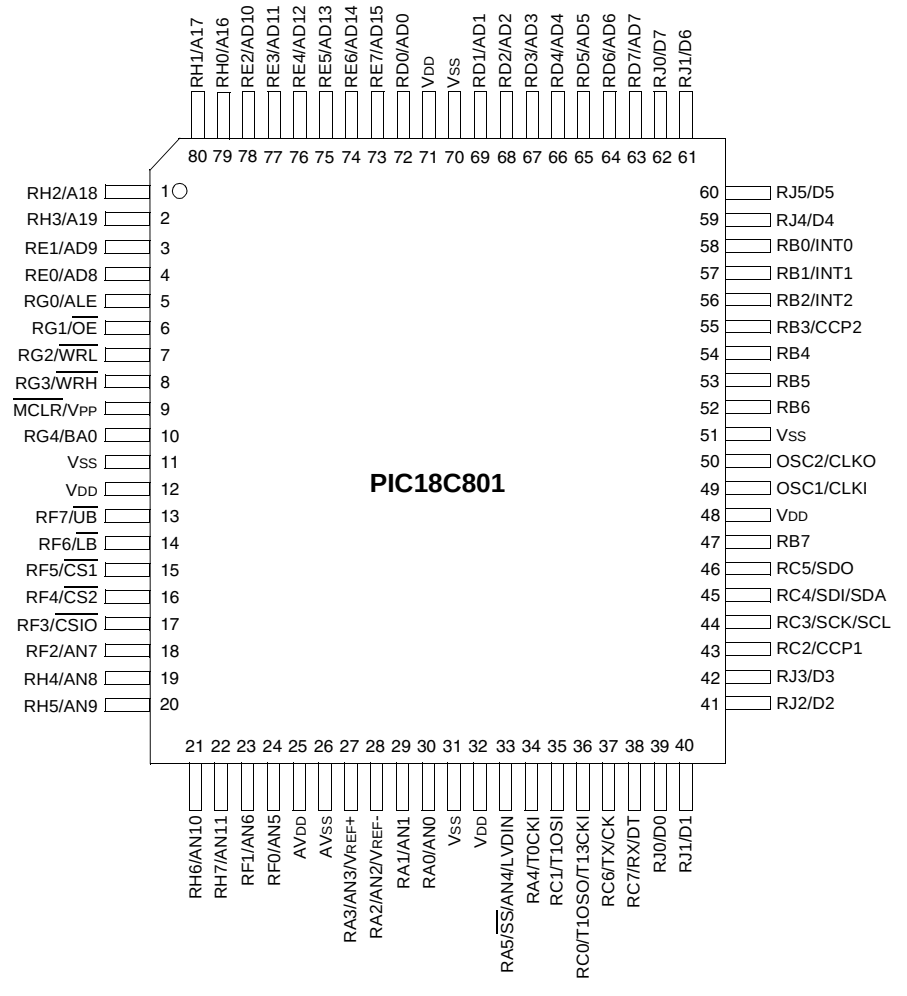
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	25MHz
Connectivity	EBI/EMI, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	37
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	External
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/pic18c801t-i-pt

Pin Diagrams (Cont.'d)

80-Pin TQFP



TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@mail.microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com/cn** to receive the most current information on all of our products.

NOTES:

TABLE 1-2: PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18C601		PIC18C801				
	TQFP	PLCC	TQFP	PLCC			
							PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT0	48	60	58	72			
RB0					I/O	TTL	Digital I/O.
INT0					I	ST	External interrupt 0.
RB1/INT1	47	59	57	71			
RB1					I/O	TTL	Digital I/O.
INT1					I	ST	External interrupt 1.
RB2/INT2	46	58	56	70			
RB2					I/O	TTL	Digital I/O.
INT2					I	ST	External interrupt 2.
RB3/CCP2	45	57	55	69			
RB3					I/O	TTL	Digital I/O.
CCP2					I/O	ST	Capture2 input, Compare2 output, PWM2 output.
RB4	44	56	54	68	I/O	TTL	Digital I/O, Interrupt-on-change pin.
RB5	43	55	53	67	I/O	TTL	Digital I/O, Interrupt-on-change pin.
RB6	42	54	52	66	I/O	TTL	Digital I/O, Interrupt-on-change pin.
					I	ST	ICSP programming clock.
RB7	37	48	47	60	I/O	TTL	Digital I/O, Interrupt-on-change pin.
					I/O	ST	ICSP programming data.

Legend: TTL = TTL compatible input
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
Analog = Analog input
I = Input
O = Output
P = Power
OD = Open Drain (no P diode to V_{DD})

TABLE 1-2: PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number				Pin Type	Buffer Type	Description
	PIC18C601		PIC18C801				
	TQFP	PLCC	TQFP	PLCC			
							PORTJ is a bi-directional I/O port.
RJ0/D0	—	—	39	52			
RJ0					I/O	ST	Digital I/O.
D0					I/O	TTL	System bus data bit 0.
RJ1/D1	—	—	40	53			
RJ1					I/O	ST	Digital I/O.
D1					I/O	TTL	System bus data bit 1.
RJ2/D2	—	—	41	54			
RJ2					I/O	ST	Digital I/O.
D2					I/O	TTL	System bus data bit 2.
RJ3/D3	—	—	42	55			
RJ3					I/O	ST	Digital I/O.
D3					I/O	TTL	System bus data bit 3.
RJ4/D4	—	—	59	73			
RJ4					I/O	ST	Digital I/O.
D4					I/O	TTL	System bus data bit 4.
RJ5/D5	—	—	60	74			
RJ5					I/O	ST	Digital I/O.
D5					I/O	TTL	System bus data bit 5.
RJ6/D6	—	—	61	75			
RJ6					I/O	ST	Digital I/O.
D6					I/O	TTL	System bus data bit 6.
RJ7/D7	—	—	62	76			
RJ7					I/O	ST	Digital I/O.
D7					I/O	TTL	System bus data bit 7.
VSS	9, 25, 41, 56	19, 36, 53, 68	11,31, 51, 70	23, 44, 65, 84	P	—	Ground reference for logic and I/O pins.
VDD	10,26, 38, 57	2, 20, 37, 49	12,32, 48, 71	2, 24, 45, 61	P	—	Positive supply for logic and I/O pins.
AVSS	20	30	26	38	P	—	Ground reference for analog modules.
AVDD	19	29	25	37	P	—	Positive supply for analog modules.

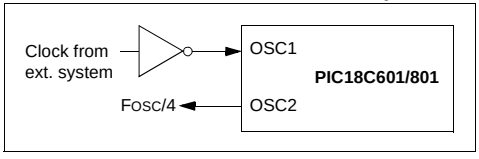
Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open Drain (no P diode to VDD)

2.4 External Clock Input

The EC oscillator mode requires an external clock source to be connected to the OSC1 pin. The feedback device between OSC1 and OSC2 is turned off in these modes to save current. There is no oscillator start-up time required after a Power-on Reset or after a recovery from SLEEP mode.

In the EC oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 2-3 shows the pin connections for the EC oscillator mode.

FIGURE 2-3: EXTERNAL CLOCK INPUT OPERATION (EC OSC CONFIGURATION)



2.5 HS4 (PLL)

A Phase Lock Loop (PLL) circuit is provided as a software programmable option for users that want to multiply the frequency of the incoming crystal oscillator signal by 4. For an input clock frequency of 6 MHz, the internal clock frequency will be multiplied to 24 MHz. This is useful for customers who are concerned with EMI due to high frequency crystals.

The PLL is enabled by configuring HS oscillator mode and setting the PLLEN bit in the OSCON register. If HS oscillator mode is not selected, or PLLEN bit in OSCCON register is clear, the PLL is not enabled and the system clock will come directly from OSC1. HS oscillator mode is the default for PIC18C601/801. In all other modes, the PLLEN bit and the SCS1 bit are forced to '0'.

A PLL lock timer is used to ensure that the PLL has locked before device execution starts. The PLL lock timer has a time-out, referred to as TPLL.

FIGURE 2-4: PLL BLOCK DIAGRAM

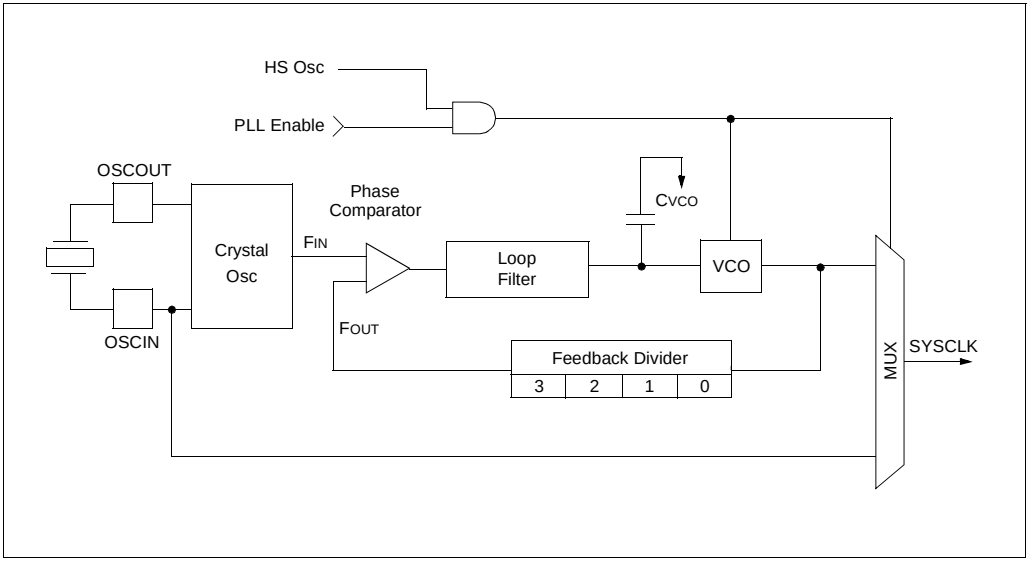


TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset	$\overline{\text{MCLR}}$ Reset WDT Reset Reset Instruction Stack Over/Underflow Reset	Wake-up via WDT or Interrupt
POSTINC2	601	801	**(Note 5)**	**(Note 5)**	**(Note 5)**
POSTDEC2	601	801	**(Note 5)**	**(Note 5)**	**(Note 5)**
PREINC2	601	801	**(Note 5)**	**(Note 5)**	**(Note 5)**
PLUSW2	601	801	**(Note 5)**	**(Note 5)**	**(Note 5)**
FSR2H	601	801	---- 0000	---- 0000	---- uuuu
FSR2L	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	601	801	---x xxxx	---u uuuu	---u uuuu
TMR0H	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR0L	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	601	801	1111 1111	1111 1111	uuuu uuuu
OSCCON	601	801	--00 0-00	--uu u-u0	--uu u-uu
LVDCON	601	801	--00 0101	--00 0101	--uu uuuu
WDTCON	601	801	---- 1111	---- uuuu	---- uuuu
RCON(4)	601	801	0r-1 11qr	0r-1 qqur	ur-u qqur
TMR1H	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	601	801	0-00 0000	u-uu uuuu	u-uu uuuu
TMR2	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
PR2	601	801	1111 1111	1111 1111	1111 1111
T2CON	601	801	-000 0000	-000 0000	-uuu uuuu
SSPBUF	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	601	801	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	601	801	0000 0000	0000 0000	uuuu uuuu
SSPCON1	601	801	0000 0000	0000 0000	uuuu uuuu
SSPCON2	601	801	0000 0000	0000 0000	uuuu uuuu
ADRESH	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	601	801	--00 0000	--00 0000	--uu uuuu
ADCON1	601	801	-000 0000	-000 0000	-uuu uuuu
ADCON2	601	801	0--- -000	0--- -000	u--- -uuu
CCPR1H	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	601	801	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	601	801	--00 0000	--00 0000	--uu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition, r = reserved, maintain '0'

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (00008h or 00018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH, and TOSL are updated with the current value of the PC. The SKPTR is modified to point to the next location in the hardware stack.

4: See Table 3-2 for RESET value for specific condition.

5: This is not a physical register. It is an indirect pointer that addresses another register. The contents returned is the value contained in the addressed register.

TABLE 9-9: PORTE FUNCTIONS

Name	Bit#	Buffer Type	Function
RE0/AD8/A8(2)	bit0	ST/TTL(1)	Input/output port pin or Address/Data bit 8
RE1/AD9/A9(2)	bit1	ST/TTL(1)	Input/output port pin or Address/Data bit 9
RE2/AD10/A10(2)	bit2	ST/TTL(1)	Input/output port pin or Address/Data bit 10
RE3/AD11/A11(2)	bit3	ST/TTL(1)	Input/output port pin or Address/Data bit 11
RE4/AD12/A12(2)	bit4	ST/TTL(1)	Input/output port pin or Address/Data bit 12
RE5/AD13/A13(2)	bit5	ST/TTL(1)	Input/output port pin or Address/Data bit 13
RE6/AD14/A14(2)	bit6	ST/TTL(1)	Input/output port pin or Address/Data bit 14
RE7/AD15/A15(2)	bit7	ST/TTL(1)	Input/output port pin or Address/Data bit 15

Legend: ST = Schmitt Trigger input, TTL = TTL input

Note 1: Input buffers are Schmitt Triggers when in I/O mode and TTL buffers when in System Bus mode.

2: REx is used as a multiplexed address/data bus for PIC18C601 and PIC18C801 in 16-bit mode, and as an address only for PIC18C801 in 8-bit mode.

TABLE 9-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

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
TRISE	PORTE Data Direction Control Register								1111 1111	1111 1111
PORTE	Read PORTE pin/Write PORTE Data Latch								xxxx xxxx	uuuu uuuu
LATE	Read PORTE Data Latch/Write PORTE Data Latch								xxxx xxxx	uuuu uuuu
MEMCON	EBDIS	PGRM	WAIT1	WAIT0	—	—	WM1	WM0	0000 --00	0000 --00

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTE.

NOTES:

NOTES:

15.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received, simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

- Serial Data Out (SDO) - RC5/SDO
- Serial Data In (SDI) - RC4/SDI/SDA
- Serial Clock (SCK) - RC3/SCK/SCL/LVOIN

Additionally, a fourth pin may be used when in any Slave mode of operation:

- Slave Select ($\overline{SS}$) - RA5/$\overline{SS}$/AN4

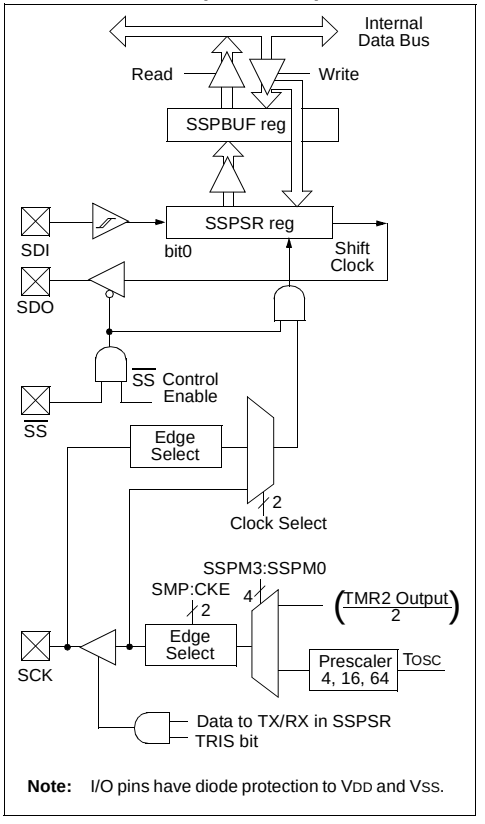
15.3.1 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits SSPCON1<5:0> and SSPSTAT<7:6>. These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock polarity (Idle state of SCK)
- Data input sample phase (middle or end of data output time)
- Clock edge (output data on rising/falling edge of SCK)
- Clock rate (Master mode only)
- Slave Select mode (Slave mode only)

Figure 15-1 shows the block diagram of the MSSP module, when in SPI mode.

FIGURE 15-1: MSSP BLOCK DIAGRAM (SPI MODE)



Note: I/O pins have diode protection to VDD and VSS.

15.4.16.2 Bus Collision During a Repeated START Condition

During a Repeated START condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from low level to high level.

b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user de-asserts SDA and the pin is allowed to float high, the BRG is loaded with SSPADD<6:0> and counts down to 0. The SCL pin is then de-asserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', see Figure 15-24). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high to low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high to low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated START condition (Figure 15-25).

If, at the end of the BRG time-out both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated START condition is complete.

FIGURE 15-24: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

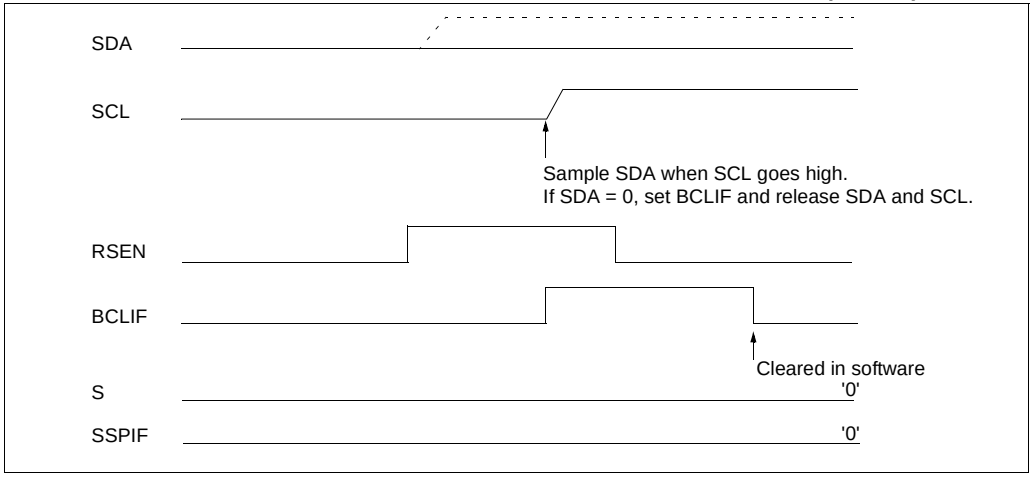
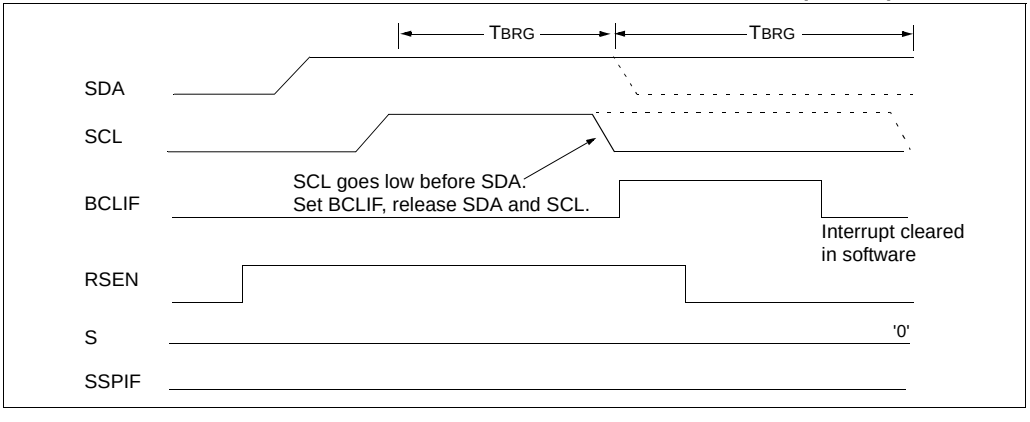


FIGURE 15-25: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



16.2 USART Asynchronous Mode

In this mode, data is transmitted in non-return-to-zero (NRZ) format. Data consists of one START bit, eight or nine data bits and one STOP bit. Data is transmitted in serial fashion with LSb first. An on-chip 8-bit baud rate generator can be programmed to generate the desired baud rate. The baud rate generator produces a clock, either x16 or x64 of the bit shift rate, depending on the BRGH bit (TXSTA register). USART does not automatically calculate the parity bit for the given data byte. If parity is to be transmitted, USART must be programmed to transmit nine bits and software must set/clear ninth data bit as parity bit. Asynchronous mode is stopped during SLEEP.

Asynchronous mode is selected by clearing the SYNC bit (TXSTA register).

The USART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

16.2.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 16-1. The heart of the transmitter is the Transmit (serial) Shift Register (TSR). The TSR register obtains its data from the Read/Write Transmit Buffer register (TXREG). The TXREG register is loaded with data in software. The TSR register is not loaded until the STOP bit has been transmitted from the previous load. As soon as the STOP bit is transmitted, the TSR is loaded with new data from the TXREG register (if available). Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and flag bit TXIF (PIR registers) is set. This interrupt can be enabled/disabled by setting/clearing enable bit TXIE (PIE registers). Flag bit TXIF will be set, regardless of the state of enable bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register. While flag bit TXIF indicated the status of the TXREG register, another bit TRMT (TXSTA register) shows the status of the TSR register. Status bit TRMT is a read only bit, which is set when the TSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty.

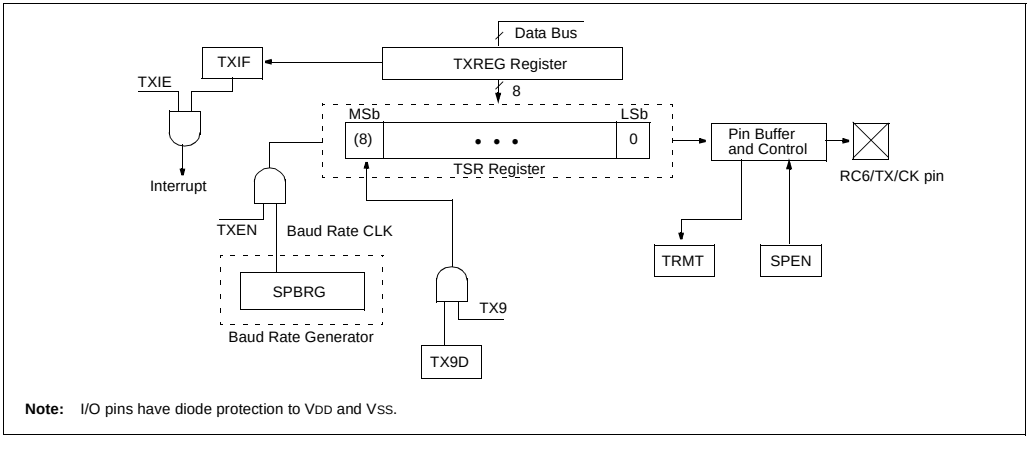
Note 1: The TSR register is not mapped in data memory, so it is not available to the user.

2: Flag bit TXIF is set when enable bit TXEN is set.

Steps to follow when setting up an Asynchronous Transmission:

1. Initialize the SPBRG register for the appropriate baud rate. If a high speed baud rate is desired, set bit BRGH (Section 16.1).
2. Enable the asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set transmit bit TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit TXEN, which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).

FIGURE 16-1: USART TRANSMIT BLOCK DIAGRAM



Note: I/O pins have diode protection to VDD and VSS.

16.4 USART Synchronous Slave Mode

Synchronous Slave mode differs from the Master mode, in that the shift clock is supplied externally at the RC6/TX/CK pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in SLEEP mode. Slave mode is entered by clearing bit CSRC (TXSTA register).

16.4.1 USART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical, 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:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in TXREG register.
c) Flag bit TXIF will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit TXIF will be set.
e) If enable bit TXIE is set, the interrupt will wake the chip from SLEEP. If the global interrupt is enabled, the program will branch to the interrupt vector.

When setting up a Synchronous Slave Transmission, follow these steps:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.

16.4.2 USART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of the SLEEP mode and bit SREN, which is a "don't care" in Slave mode.

If receive is enabled by setting bit CREN prior to the SLEEP instruction, then a word may be received during SLEEP. On completely receiving the word, the RSR register will transfer the data to the RCREG register, and if enable bit RCIE bit is set, the interrupt generated will wake the chip from SLEEP. If the global interrupt is enabled, the program will branch to the interrupt vector.

When setting up a Synchronous Slave Reception, follow these steps:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit RCIF will be set when reception is complete. An interrupt will be generated if enable bit RCIE was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.

TABLE 16-10: 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
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000	-000 0000
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000	-000 0000
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	-000 0000	-000 0000
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREG	USART Transmit Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	ADDEN	BRGH	TRMT	TX9D	0000 0010	0000 0010
SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

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

17.1 A/D Acquisition Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 17-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5kΩ**. After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

Note:	When the conversion is started, the holding capacitor is disconnected from the input pin.

To calculate the minimum acquisition time, Equation 17-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 17-1 shows the calculation of the minimum required acquisition time TACQ. This calculation is based on the following application system assumptions:

CHOLD	=	120 pF
Rs	=	2.5 kΩ
Conversion Error	≤	1/2 LSb
VDD	=	5V → Rss = 7 kΩ
Temperature	=	50°C (system max.)
VHOLD	=	0V @ time = 0

EQUATION 17-1: ACQUISITION TIME

$$\begin{aligned} T_{ACQ} &= \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= T_{AMP} + T_C + T_{COFF} \end{aligned}$$

EQUATION 17-2: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

or

$$T_C = -(120\ pF)(1\ k\Omega + R_{SS} + R_S) \ln(1/2047)$$

EXAMPLE 17-1: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$

Temperature coefficient is only required for temperatures > 25°C.

$T_{ACQ} = 2\ \mu s + T_C + [(Temp - 25°C)(0.05\ \mu s/°C)]$

$T_C = -C_{HOLD}(R_{IC} + R_{SS} + R_S) \ln(1/2047)$
$-120\ pF\ (1\ k\Omega + 7\ k\Omega + 2.5\ k\Omega) \ln(0.0004885)$
$-120\ pF\ (10.5\ k\Omega) \ln(0.0004885)$
$-1.26\ \mu s\ (-7.6241)$
$9.61\ \mu s$

$T_{ACQ} = 2\ \mu s + 9.61\ \mu s + [(50°C - 25°C)(0.05\ \mu s/°C)]$
$11.61\ \mu s + 1.25\ \mu s$
$12.86\ \mu s$

17.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 12 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable. There are seven possible options for TAD:

- 2TOSC
- 4TOSC
- 8TOSC
- 16TOSC
- 32TOSC
- 64TOSC
- Internal RC oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time of 1.6 μs.

Table 17-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

17.3 Configuring Analog Port Pins

The ADCON1, TRISA, TRISF and TRISH registers control the operation of the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CHS3:CHS0 bits and the TRIS bits.

Note 1: When reading the port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.

2: Analog levels on any pin defined as a digital input may cause the input buffer to consume current out of the device's specification limits.

TABLE 17-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency	
Operation	**ADCS2:ADCS0**	**PIC18C601/801**	**PIC18LC601/801(5)**
2TOSC	000	1.25 MHz	666 kHz
4TOSC	100	2.50 MHz	1.33 MHz
8TOSC	001	5.00 MHz	2.67 MHz
16TOSC	101	10.0 MHz	5.33 MHz
32TOSC	010	20.0 MHz	10.67 MHz
64TOSC	110	—	—
RC	x11	—	—

Note 1: The RC source has a typical TAD time of 4 μs.
2: These values violate the minimum required TAD time.
3: For faster conversion times, the selection of another clock source is recommended.
4: For device frequencies above 1 MHz, the device must be in SLEEP for the entire conversion or the A/D accuracy may be out of specification.
5: This column is for the **LC** devices only.

POP **Pop Top of Return Stack**

Syntax: [*label*] POP

Operands: None

Operation: (TOS) → bit bucket

Status Affected: None

Encoding:

0000	0000	0000	0110

Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack.

This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Pop TOS value	No operation

Example:

```
POP
GOTO    NEW
```

Before Instruction
- TOS = 0031A2h
- Stack (1 level down) = 014332h

After Instruction
- TOS = 014332h
- PC = NEW

PUSH **Push Top of Return Stack**

Syntax: [*label*] PUSH

Operands: None

Operation: (PC+2) → TOS

Status Affected: None

Encoding:

0000	0000	0000	0101

Description: The PC+2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack.

This instruction allows implementing a software stack by modifying TOS, and then push it onto the return stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Push PC+2 onto return stack	No operation	No operation

Example:

```
PUSH
```

Before Instruction
- TOS = 00345Ah
- PC = 000124h

After Instruction
- PC = 000126h
- TOS = 000126h
- Stack (1 level down) = 00345Ah

RRNCF **Rotate Right f (no carry)**

Syntax: [*label*] RRNCF f [,d [,a]]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

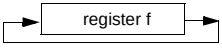
Operation: (f<n>) → dest<n-1>,
(f<0>) → dest<7>

Status Affected: N,Z

Encoding:

0100	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is 0, the result is placed in WREG. If 'd' is 1, the result is placed back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, the Bank will be selected as per the BSR value.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `RRNCF REG`

Before Instruction
REG = 1101 0111
N = ?
Z = ?

After Instruction
REG = 1110 1011
N = 1
Z = 0

Example 2: `RRNCF REG, 0, 0`

Before Instruction
WREG = ?
REG = 1101 0111
N = ?
Z = ?

After Instruction
WREG = 1110 1011
REG = 1101 0111
N = 1
Z = 0

SETF **Set f**

Syntax: [*label*] SETF f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: FFh → f

Status Affected: None

Encoding:

0110	100a	ffff	ffff

Description: The contents of the specified register are set to FFh. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, the Bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `SETF REG`

Before Instruction
REG = 5Ah

After Instruction
REG = 0FFh

TBLWT **Table Write**

Syntax: [*label*] TBLWT (*; *+; *-; +*)

Operands: None

Operation: if TBLWT*,
(TABLAT) → Prog Mem (TBLPTR) or Holding Register;
TBLPTR - No Change;
if TBLWT*+,
(TABLAT) → Prog Mem (TBLPTR) or Holding Register;
(TBLPTR) +1 → TBLPTR;
if TBLWT*-,
(TABLAT) → Prog Mem (TBLPTR) or Holding Register;
(TBLPTR) -1 → TBLPTR;
if TBLWT+*,
(TBLPTR) +1 → TBLPTR;
(TABLAT) → Prog Mem (TBLPTR) or Holding Register;

Status Affected: None

Encoding:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction is used to program the contents of Program Memory (P.M.).

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2 MByte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLWT instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2 (many if long write is to on-chip EPROM program memory)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register or Memory)

TBLWT (Cont.)

Example 1: `TBLWT *+;`

Before Instruction
TABLAT = 55h
TBLPTR = 00A356h
MEMORY(00A356h) = 0FFh

After Instructions (table write completion)
TABLAT = 55h
TBLPTR = 00A357h
MEMORY(00A356h) = 55h

Example 2: `TBLWT +*;`

Before Instruction
TABLAT = 34h
TBLPTR = 01389Ah
MEMORY(01389Ah) = 0FFh
MEMORY(01389Bh) = 0FFh

After Instruction (table write completion)
TABLAT = 34h
TBLPTR = 01389Bh
MEMORY(01389Ah) = 0FFh
MEMORY(01389Bh) = 34h

U

W

X

