

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 "<u>Embedded -</u> <u>Microcontrollers</u>"

### Details

E-XF

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	4MHz
Connectivity	UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	3.5KB (2K x 14)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	224 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	18-SOIC (0.295", 7.50mm Width)
Supplier Device Package	18-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f628t-04i-so

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong

#### Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION. QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

### Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, rfPIC and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Octopus, Omniscient Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICkit, PICtail, PIC<sup>32</sup> logo, REAL ICE, rfLAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2009, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.



## QUALITY MANAGEMENT SYSTEM CERTIFIED BY DNV ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and mulfacture of development systems is ISO 9001:2000 certified.

### **Pin Diagrams**



NOTES:

NOTES:

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset <sup>(1)</sup>	Details on Page
Bank 2											
100h	INDF	Addressing	g this location	cal register)	XXXX XXXX	30					
101h	TMR0	Timer0 Mo	dule's Registe	er						XXXX XXXX	47
102h	PCL	Program C	Counter's (PC)	Least Sign	ificant Byte					0000 0000	30
103h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	С	0001 1xxx	24
104h	FSR	Indirect Da	ata Memory A	ddress Poin	ter	•		•	•	xxxx xxxx	30
105h	_	Unimplem	ented							_	—
106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	38
107h	—	Unimplem	ented							—	—
108h	—	Unimplem	ented							—	
109h	—	Unimplem	ented							—	
10Ah	PCLATH	—	—	—	Write	Buffer for u	pper 5 bits o	f Program C	ounter	0 0000	30
10Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	26
10Ch	—	Unimplem	ented							—	
10Dh	—	Unimplem	ented	—							
10Eh	—	Unimplem	Jnimplemented								
10Fh	—	Unimplem	ented							—	
110h	—	Unimplem	ented							—	
111h	—	Unimplem	ented							—	
112h	—	Unimplem	ented							—	
113h	—	Unimplem	ented							—	
114h	—	Unimplem	ented							—	
115h	_	Unimplem	ented							_	
116h	_	Unimplem	ented							_	
117h	_	Unimplem	ented							_	
118h	_	Unimplem	ented							_	
119h	—	Unimplem	ented							—	—
11Ah	_	Unimplem	ented							_	
11Bh	—	Unimplem	ented							—	—
11Ch	—	Unimplem	ented							—	—
11Dh	—	Unimplem	ented							—	—
11Eh	—	Unimplem	nimplemented								—
11Fh	—	Unimplem	ented							—	—

### TABLE 4-5: SPECIAL FUNCTION REGISTERS SUMMARY BANK2

Legend:- = Unimplemented locations read as '0', u = unchanged, x = unknown, q = value depends on condition, shaded = unimplemented.Note1:For the initialization condition for registers tables, refer to Table 14-6 and Table 14-7.

## 7.0 TIMER1 MODULE

The Timer1 module is a 16-bit timer/counter consisting of two 8-bit registers (TMR1H and TMR1L) which are readable and writable. The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The Timer1 Interrupt, if enabled, is generated on overflow of the TMR1 register pair which latches the interrupt flag bit TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing the Timer1 interrupt enable bit TMR1IE (PIE1<0>).

Timer1 can operate in one of two modes:

- As a timer
- As a counter

The Operating mode is determined by the clock select bit, TMR1CS (T1CON<1>).

-n = Value at POR

In Timer mode, the TMR1 register pair value increments every instruction cycle. In Counter mode, it increments on every rising edge of the external clock input.

Timer1 can be enabled/disabled by setting/clearing control bit TMR1ON (T1CON<0>).

Timer1 also has an internal "Reset input". This Reset can be generated by the CCP module (Section 9.0 "Capture/Compare/PWM (CCP) Module"). Register 7-1 shows the Timer1 control register.

For the PIC16F627A/628A/648A, when the Timer1 oscillator is enabled (T1OSCEN is set), the RB7/T1OSI/PGD and RB6/T1OSO/T1CKI/PGC pins become inputs. That is, the TRISB<7:6> value is ignored.

EK / - I.	TICON-			<b>NEGISTER</b>	ADDRESS.			
	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	_	—	T1CKPS1	T1CKPS0	T10SCEN	T1SYNC	TMR1CS	TMR10N
	bit 7			· · ·		1		bit 0
bit 7-6	Unimplem	ented: Rea	ad as '0'					
bit 5-4	T1CKPS<1	<b>i :0&gt;</b> : Timer	1 Input Cloci	k Prescale S	elect bits			
	11 <b>= 1:8 P</b> i	rescale valı	ue					
	10 = 1:4 Pi	rescale valu	Je					
	01 = 1:2 Pi	rescale valu	Je					
hit 3	TIOSCEN	· Timer1 Or	scillator Enal	hle Control b	it			
DIEG	1 = Oscillat	tor is enabl		JC 0011101 2				
	0 = Oscillat	tor is shut c	off(1)					
bit 2	T1SYNC: 7	Fimer1 Exte	ernal Clock Ir	nput Synchro	onization Contro	ol bit		
	TMR1CS =	<u>= 1</u>						
	1 = Do not	synchroniz	e external cl	ock input				
	0 = Synchr	onize exter	nal clock inp	out				
	<u>- This hit is i</u>	<u>: 0</u> anored Tin	ner1 uses th	e internal clc	vek when TMR <sup>4</sup>	1CS = 0		
hit 1		Timer1 Clor		alact hit		$\mathbf{U}\mathbf{U}=\mathbf{U}.$		
	1 = Extern:	al clock from	m nin RB6/T		I/PGC (on the	risina edae	.)	
	0 = Interna	I clock (Fo:	sc/4)	1000/1101		Tonig cage	)	
bit 0	TMR1ON:	Timer1 On	bit					
	1 = Enable	s Timer1						
	0 = Stops 7	limer1						
	Note 1:	The oscilla	ator inverter a	and feedbacl	k resistor are tu	irned off to	eliminate p	ower drain.
	Legend:							
	R = Reada	able bit	VV = V	Nritable bit	U = Unimpl	emented b	it. read as '	0'

'1' = Bit is set

'0' = Bit is cleared

REGISTER 7-1: T1CON – TIMER1 CONTROL REGISTER (ADDRESS: 10h)

x = Bit is unknown

### 9.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available: the CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

*PWM duty cycle* =

(CCPR1L:CCP1CON<5:4>) · Tosc · TMR2 prescale value

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR1H until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

When the CCPR1H and 2-bit latch match TMR2 concatenated with an internal 2-bit Q clock or 2 bits of the TMR2 prescaler, the CCP1 pin is cleared.

Maximum PWM resolution (bits) for a given PWM frequency:



Note: If the PWM duty cycle value is longer than the PWM period the CCP1 pin will not be cleared.

For an example PWM period and duty cycle calculation, see the *PIC<sup>®</sup> Mid-Range Reference Manual* (DS33023).

### 9.3.3 SET-UP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

- 1. Set the PWM period by writing to the PR2 register.
- 2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
- Make the CCP1 pin an output by clearing the TRISB<3> bit.
- 4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.

### TABLE 9-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 20 MHz

PWM Frequency	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.5

### TABLE 9-4: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	TOIE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	EEIF	CMIF	RCIF	TXIF	_	CCP1IF	TMR2IF	TMR1IF	0000 -000	0000 -000
8Ch	PIE1	EEIE	CMIE	RCIE	TXIE	_	CCP1IE	TMR2IE	TMR1IE	0000 -000	0000 -000
86h, 186h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
11h	TMR2	Timer2 M	odule's Reg	ister						0000 0000	0000 0000
92h	PR2	Timer2 M	odule's Peri	od Register						1111 1111	1111 1111
12h	T2CON		TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	uuuu uuuu
15h	CCPR1L	Capture/0	Compare/PV	/M Register	1 (LSB)					xxxx xxxx	uuuu uuuu
16h	CCPR1H	Capture/0	Capture/Compare/PWM Register 1 (MSB)								uuuu uuuu
17h	CCP1CON	_	_	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	00 0000	00 0000

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by PWM and Timer2.

### 12.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit baud rate generator. The SPBRG register controls the period of a free running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 12-1 shows the formula for computation of the baud rate for different USART modes, which only apply in Master mode (internal clock).

Given the desired baud rate and Fosc, the nearest integer value for the SPBRG register can be calculated using the formula in Table 12-1. From this, the error in baud rate can be determined.

Example 12-1 shows the calculation of the baud rate error for the following conditions:

Fosc = 16 MHz Desired Baud Rate = 9600 BRGH = 0 SYNC = 0

### EQUATION 12-1: CALCULATING BAUD RATE ERROR

$$Desired Baud Rate = \frac{Fosc}{64(x+1)}$$

$$9600 = \frac{16000000}{64(x+1)}$$

$$x = 25.042$$

$$Calculated Baud Rate = \frac{16000000}{64(25+1)} = 9615$$

$$Error = \frac{(Calculated Baud Rate - Desired Baud Rate)}{Desired Baud Rate}$$

$$= \frac{9615 - 9600}{9600} = 0.16\%$$

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the FOSC/(16(X + 1)) equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register causes the BRG timer to be reset (or cleared) and ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

The data on the RB1/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

### TABLE 12-1: BAUD RATE FORMULA

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = Fosc/(64(X+1))	Baud Rate = Fosc/(16(X+1))
1	(Synchronous) Baud Rate = Fosc/(4(X+1))	NA

**Legend:** X = value in SPBRG (0 to 255)

### TABLE 12-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
98h	TXSTA	CSRC	TX9	TXEN	SYNC		BRGH	TRMT	TX9D	0000 -010	0000 -010
18h	RCSTA	SPEN	RX9	SREN	CREN	ADEN	FERR	OERR	RX9D	0000 000x	0000 000x
99h	SPBRG	Baud Ra	3aud Rate Generator Register								0000 0000

**Legend:** x = unknown, - = unimplemented read as '0'. Shaded cells are not used for the BRG.

BAUD	Fosc = 20 MHz		SPBRG	16 MHz		SPBRG	10 MHz		SPBRG
RATE (K)	KBAUD	ERROR	(decimal)	KBAUD	ERROR	(decimal)	KBAUD	ERROR	(decimal)
0.3	NA	_	_	NA	_	_	NA	_	_
1.2	NA	—	_	NA	—	_	NA	_	_
2.4	NA	—	_	NA	—	_	NA	_	_
9.6	NA	—	_	NA	—	_	9.766	+1.73%	255
19.2	19.53	+1.73%	255	19.23	+0.16%	207	19.23	+0.16%	129
76.8	76.92	+0.16%	64	76.92	+0.16%	51	75.76	-1.36%	32
96	96.15	+0.16%	51	95.24	-0.79%	41	96.15	+0.16%	25
300	294.1	-1.96	16	307.69	+2.56%	12	312.5	+4.17%	7
500	500	0	9	500	0	7	500	0	4
HIGH	5000	_	0	4000	_	0	2500	_	0
LOW	19.53	_	255	15.625	_	255	9.766	_	255

### TABLE 12-3: BAUD RATES FOR SYNCHRONOUS MODE

BAUD	Fosc = 7.15	909 MHz	SPBRG	5.0688 MHz		SPBRG	4 MHz		SPBRG
RATE (K)	KBAUD	ERROR	(decimal)	KBAUD	ERROR	(decimal)	KBAUD	ERROR	(decimal)
0.3	NA		_	NA		_	NA	_	_
1.2	NA	_	_	NA	_	_	NA	_	_
2.4	NA	_	_	NA	_	_	NA	_	_
9.6	9.622	+0.23%	185	9.6	0	131	9.615	+0.16%	103
19.2	19.24	+0.23%	92	19.2	0	65	19.231	+0.16%	51
76.8	77.82	+1.32	22	79.2	+3.13%	15	75.923	+0.16%	12
96	94.20	-1.88	18	97.48	+1.54%	12	1000	+4.17%	9
300	298.3	-0.57	5	316.8	5.60%	3	NA	_	_
500	NA	_	_	NA	_	_	NA	_	_
HIGH	1789.8	_	0	1267	_	0	100	_	0
LOW	6.991	_	255	4.950	_	255	3.906	_	255

BAUD	Fosc = 3.57	'9545 MHz	SPBRG	1 MHz		SPBRG	32.768 kHz		SPBRG
RATE (K)	KBAUD	ERROR	value (decimal)	KBAUD	ERROR	value (decimal)	KBAUD	ERROR	value (decimal)
0.3	NA	_	_	NA		_	0.303	+1.14%	26
1.2	NA	_	_	1.202	+0.16%	207	1.170	-2.48%	6
2.4	NA	—	—	2.404	+0.16%	103	NA	_	_
9.6	9.622	+0.23%	92	9.615	+0.16%	25	NA	_	_
19.2	19.04	-0.83%	46	19.24	+0.16%	12	NA	—	_
76.8	74.57	-2.90%	11	83.34	+8.51%	2	NA	—	_
96	99.43	+3.57%	8	NA	_	_	NA	_	_
300	298.3	0.57%	2	NA	_	_	NA	_	_
500	NA	_	—	NA	—	—	NA	—	—
HIGH	894.9	_	0	250	_	0	8.192	_	0
LOW	3.496	—	255	0.9766	—	255	0.032	—	255

## 12.2 USART Asynchronous Mode

In this mode, the USART uses standard non-return-tozero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8-bit. A dedicated 8-bit baud rate generator is used to derive baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The USART's transmitter and receiver are functionally independent, but use the same data format and baud rate. The baud rate generator produces a clock either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during Sleep.

Asynchronous mode is selected by clearing bit SYNC (TXSTA<4>).

The USART Asynchronous module consists of the following important elements:

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

### 12.2.1 USART ASYNCHRONOUS TRANSMITTER

The USART transmitter block diagram is shown in Figure 12-1. The heart of the transmitter is the Transmit (serial) Shift Register (TSR). The shift register obtains its data from the read/write transmit buffer, 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 (PIR1<4>) is set. This interrupt can be enabled/ disabled by setting/clearing enable bit TXIE (PIE1<4>). 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<1>) 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.

Transmission is enabled by setting enable bit TXEN (TXSTA<5>). The actual transmission will not occur until the TXREG register has been loaded with data and the Baud Rate Generator (BRG) has produced a shift clock (Figure 12-1). The transmission can also be started by first loading the TXREG register and then setting enable bit TXEN. Normally when transmission is first started, the TSR register is empty, so a transfer to the TXREG register will result in an immediate transfer to TSR resulting in an empty TXREG. A back-to-back transfer is thus possible (Figure 12-3). Clearing enable bit TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. As a result the RB2/TX/CK pin will revert to high-impedance.

In order to select 9-bit transmission, transmit bit TX9 (TXSTA<6>) should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to the TXREG register. This is because a data write to the TXREG register can result in an immediate transfer of the data to the TSR register (if the TSR is empty). In such a case, an incorrect ninth data bit may be loaded in the TSR register.



## 12.5 USART Synchronous Slave Mode

Synchronous Slave mode differs from the Master mode in the fact that the shift clock is supplied externally at the RB2/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<7>).

### 12.5.1 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 now be set.
- e) If enable bit TXIE is set, the interrupt will wake the chip from Sleep and if the global interrupt is enabled, the program will branch to the interrupt vector (0004h).

Follow these steps when setting up a Synchronous Slave Transmission:

- 1. TRISB<1> and TRISB<2> should both be set to '1' to configure the RB1/RX/DT and RB2/TX/CK pins as inputs. Output drive, when required, is controlled by the peripheral circuitry.
- Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
- 3. Clear bits CREN and SREN.
- 4. If interrupts are desired, then set enable bit TXIE.
- 5. If 9-bit transmission is desired, then set bit TX9.
- 6. Enable the transmission by setting enable bit TXEN.
- 7. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
- 8. Start transmission by loading data to the TXREG register.

## 13.0 DATA EEPROM MEMORY

The EEPROM data memory is readable and writable during normal operation (full VDD range). This memory is not directly mapped in the register file space. Instead it is indirectly addressed through the Special Function Registers (SFRs). There are four SFRs used to read and write this memory. These registers are:

- EECON1
- EECON2 (Not a physically implemented register)
- EEDATA
- EEADR

EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. PIC16F627A/628A devices have 128 bytes of data EEPROM with an address range from 0h to 7Fh. The PIC16F648A device has 256 bytes of data EEPROM with an address range from 0h to FFh. The EEPROM data memory allows byte read and write. A byte write automatically erases the location and writes the new data (erase before write). The EEPROM data memory is rated for high erase/write cycles. The write time is controlled by an on-chip timer. The write time will vary with voltage and temperature, as well as from chip-to-chip. Please refer to AC specifications for exact limits.

When the device is code-protected, the CPU can continue to read and write the data EEPROM memory. A device programmer can no longer access this memory.

Additional information on the data EEPROM is available in the *PIC<sup>®</sup> Mid-Range Reference Manual* (DS33023).

### REGISTER 13-1: EEDATA – EEPROM DATA REGISTER (ADDRESS: 9Ah)

| R/W-x  |
|--------|--------|--------|--------|--------|--------|--------|--------|
| EEDAT7 | EEDAT6 | EEDAT5 | EEDAT4 | EEDAT3 | EEDAT2 | EEDAT1 | EEDAT0 |
| bit 7  |        |        |        |        |        |        | bit 0  |

bit 7-0 **EEDATn**: Byte value to Write to or Read from data EEPROM memory location.

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

### REGISTER 13-2: EEADR – EEPROM ADDRESS REGISTER (ADDRESS: 9Bh)

| R/W-x |
|-------|-------|-------|-------|-------|-------|-------|-------|
| EADR7 | EADR6 | EADR5 | EADR4 | EADR3 | EADR2 | EADR1 | EADR0 |
| bit 7 |       |       |       |       |       |       | bit 0 |

bit 7 PIC16F627A/628A

Unimplemented Address: Must be set to '0'

### PIC16F648A

**EEADR**: Set to '1' specifies top 128 locations (128-255) of EEPROM Read/Write Operation **EEADR**: Specifies one of 128 locations of EEPROM Read/Write Operation

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 6-0

### 13.1 EEADR

The PIC16F648A EEADR register addresses 256 bytes of data EEPROM. All eight bits in the register (EEADR<7:0>) are required.

The PIC16F627A/628A EEADR register addresses only the first 128 bytes of data EEPROM so only seven of the eight bits in the register (EEADR<6:0>) are required. The upper bit is address decoded. This means that this bit should always be '0' to ensure that the address is in the 128 byte memory space.

### 13.2 EECON1 and EECON2 Registers

EECON1 is the control register with four low order bits physically implemented. The upper-four bits are non-existent and read as '0's.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set, in software. They are cleared in hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a  $\overline{\text{MCLR}}$  Reset or a WDT Time-out Reset during normal operation. In these situations, following Reset, the user can check the WRERR bit and rewrite the location. The data and address will be unchanged in the EEDATA and EEADR registers.

Interrupt flag bit EEIF in the PIR1 register is set when write is complete. This bit must be cleared in software.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the data EEPROM write sequence.

### REGISTER 13-3: EECON1 – EEPROM CONTROL REGISTER 1 (ADDRESS: 9Ch)

				•		,	
U-0	U-0	U-0	U-0	R/W-x	R/W-0	R/S-0	R/S-0
_	_	—	—	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7-4	Unimplemented: Read as '0'
bit 3	WRERR: EEPROM Error Flag bit
	<ul> <li>1 = A write operation is prematurely terminated (any MCLR Reset, any WDT Reset during normal operation or BOR Reset)</li> <li>a = The write operation completed</li> </ul>
bit 2	WREN: EEPROM Write Enable bit
	1 = Allows write cycles
	0 = Inhibits write to the data EEPROM
bit 1	WR: Write Control bit
	<ul> <li>1 = initiates a write cycle. (The bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.</li> <li>0 = Write cycle to the data EEPROM is complete</li> </ul>
bit 0	RD: Read Control bit
	<ul> <li>1 = Initiates an EEPROM read (read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software).</li> </ul>
	0 = Does not initiate an EEPROM read
	Legend:

W = Writable bit

'1' = Bit is set

R = Readable bit

-n = Value at POR

x = Bit is unknown

U = Unimplemented bit, read as '0'

'0' = Bit is cleared

NOTES:

### 14.8.1 WAKE-UP FROM SLEEP

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

- 1. External Reset input on MCLR pin
- 2. Watchdog Timer wake-up (if WDT was enabled)
- 3. Interrupt from RB0/INT pin, RB port change, or any peripheral interrupt, which is active in Sleep.

The first event will cause a device Reset. The two latter events are considered a continuation of program execution. The TO and PD bits in the Status register can be used to determine the cause of device Reset. PD bit, which is set on power-up, is cleared when Sleep is invoked. TO bit is cleared if WDT wake-up occurred. When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is 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 and then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have an 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 not enter Sleep. The SLEEP instruction is executed as a NOP instruction.

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

### FIGURE 14-17: WAKE-UP FROM SLEEP THROUGH INTERRUPT

OSC1	; Q1 Q2 Q3 Q4; '/~_/~_/	Q1 Q2 Q3 Q4	Q1	Q1	Q2  Q3  Q4 ~_/~_/~_	Q1  Q2  Q3  Q4  /	; q1  q2  q3  q4; /~/	Q1 Q2 Q3  Q4 ~
CLKOUT <sup>(4)</sup>	1		Tos	T(1,2)		\/	<u>ب</u> ۲	<u>_</u>
INT pin	1 1 1 1 1 1	1 1 1		1 1 1		1 1 1	1 1 1 1 1 1	1 1 1
INTCON<1>	ı ı )⊢────┼		<u></u>	Inte	errupt Latend	у	·	
	'i i i i			I I	(Note 2)	I I		1
(INTCON<7>	)	1	Processor in Sleep	1		·	<u>   </u>         	
Instruction F	low	, ,		1		1	, , , , , , , , , , , , , , , , , , ,	I I
PC	X PC X	PC + 1	PC + 2	<u> </u>	PC + 2	X PC + 2	<u>χ 0004h<sup>(3)</sup> χ</u>	0005h
Instruction { Fetched {	Inst(PC) = Sleep	Inst(PC + 1)		In	st(PC + 2)	1 1 1	Inst(0004h)	Inst(0005h)
Instruction { Executed {	Inst(PC - 1)	Sleep		In	st(PC + 1)	Dummy cycle	Dummy cycle	Inst(0004h)
Note 1: > 2: 1	(T, HS or LP Oscilla ost = 1024 Tosc (	ator mode assur drawing not to s	ned. cale). Approxima	itely 1 μs	delay will b	e there for RC O	scillator mode.	

**3:** GIE = 1 assumed. In this case, after wake-up the processor jumps to the interrupt routine. If GIE = 0, execution will continue in-line.

4: CLKOUT is not available in these Oscillator modes, but shown here for timing reference.

### 14.9 Code Protection

With the Code-Protect bit is cleared (Code-Protect enabled), the contents of the program memory locations are read out as '0'. See "*PIC16F627A/628A/648A EEPROM Memory Programming Specification*" (DS41196) for details.

Note:	Only a Bulk Erase function can set the CP					
	and CPD bits by turning off the code					
	protection. The entire data EEPROM and					
	Flash program memory will be erased to					
	turn the code protection off.					

### 14.10 User ID Locations

Four memory locations (2000h-2003h) are designated as user ID locations where the user can store checksum or other code-identification numbers. These locations are not accessible during normal execution but are readable and writable during Program/Verify. Only the Least Significant 4 bits of the user ID locations are used for checksum calculations although each location has 14 bits.

AND Literal with W

ANDLW

ADDLW	Add Literal and W				
Syntax:	[ <i>label</i> ] ADDLW k				
Operands:	$0 \le k \le 255$				
Operation:	(W) + k –	→ (W)			
Status Affected:	C, DC, Z				
Encoding:	11	111x	kkkk	kkkk	
Description:	The contents of the W register are added to the eight bit literal 'k' and the result is placed in the W register.			gister literal I in the	
Words:	1				
Cycles:	1				
Example	ADDLW	0x15			
	Before In: W After Instr W	structio = 0x1 ruction = 0x2	n 10 25		

15.2 Instruction	Descriptions
------------------	--------------

Oymax.	
Operands:	$0 \le k \le 255$
Operation:	(W) .AND. (k) $\rightarrow$ (W)
Status Affected:	Z
Encoding:	11 1001 kkkk kkkk
Description:	The contents of W register are AND'ed with the eight bit literal 'k'. The result is placed in the W register.
Words:	1
Cycles:	1
Example	ANDLW 0x5F
	Before Instruction
	W = 0xA3
	W = 0x03
ANDWF	AND W with f
Syntax:	[ <i>label</i> ] ANDWF f.d
e yman.	
Operands:	0 ≤ f ≤ 127
Operands:	$0 \le f \le 127$ $d \in [0,1]$ (A) AND (f) (dept)
Operands: Operation:	$0 \le f \le 127$ d \equiv [0,1] (W) .AND. (f) \rightarrow (dest)
Operation: Status Affected:	$0 \le f \le 127$ $d \in [0,1]$ (W) .AND. (f) $\rightarrow$ (dest) Z
Operands: Operation: Status Affected: Encoding:	$0 \le f \le 127$ $d \in [0,1]$ (W) .AND. (f) $\rightarrow$ (dest) Z 00 0101 dfff ffff
Operands: Operation: Status Affected: Encoding: Description:	$\begin{array}{c} 0 \leq f \leq 127 \\ d \in [0,1] \\ (W) .AND. (f) \rightarrow (dest) \\ \hline Z \\ \hline 00 \\ Olo1 \\ dfff \\ ffff \\ \hline AND the W register with register \\ f'. If 'd' is '0', the result is stored \\ in the W register. If 'd' is '1', the \\ result is stored back in register \\ f'. \end{array}$
Operands: Operation: Status Affected: Encoding: Description: Words:	$0 \le f \le 127$ $d \in [0,1]$ (W) .AND. (f) $\rightarrow$ (dest) Z 00 0101 dfff ffff AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. 1
Operands: Operation: Status Affected: Encoding: Description: Words: Cycles:	$0 \le f \le 127$ $d \in [0,1]$ (W) .AND. (f) $\rightarrow$ (dest) Z 00 0101 dfff ffff AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. 1 1 1
Operands: Operation: Status Affected: Encoding: Description: Words: Cycles: Example	$0 \le f \le 127$ $d \in [0,1]$ (W) .AND. (f) $\rightarrow$ (dest) Z 00 0101 dfff ffff AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. 1 1 ANDWF REG1, 1

ADDWF	Add W and f				
Syntax:	[label] ADDWF f,d				
Operands:	$0 \le f \le 127$ $d \in [0,1]$				
Operation:	$(W) + (f) \to (dest)$				
Status Affected:	C, DC, Z				
Encoding:	00 0111 dfff ffff				
Description:	Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.				
Words:	1				
Cycles:	1				
Example	ADDWF REG1, 0				
	Before Instruction W = 0x17 REG1 = 0xC2 After Instruction W = 0xD9 REG1 = 0xC2 Z = 0 C = 0 DC = 0				

1

CLRF

REG1 **Before Instruction** 

After Instruction

Ζ

REG1 = 0x5A

REG1 = 0x00= 1

Cycles:

Example

BTFSS	Bit Test f, Skip if Set	CALL	Call Subroutine
Syntax:	[ label ] BTFSS f,b	Syntax:	[ <i>label</i> ] CALL k
Operands:	$0 \leq f \leq 127$	Operands:	$0 \leq k \leq 2047$
	0 ≤ b < 7	Operation:	(PC)+ 1 $\rightarrow$ TOS,
Operation:	skip if (f <b>) = <math>1</math></b>		$k \rightarrow PC<10:0>,$
Status Affected:	None	Status Affected	$(FCLATH^4.3^{-}) \to FC^12.11^{-}$
Encoding:	01 11bb bfff fff	Status Allected.	
Description:	If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution, is discarded and a NOP is executed instead, making this a two-cycle instruction.	Description:	Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The eleven bit imme- diate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two-cycle
Words:	1		instruction.
Cycles:	1(2)	Words:	1
Example	HERE BTFSS REG1	Cycles:	2
	FALSE GOTO PROCESS_CODE	Example	HERE CALL THERE
	Before Instruction PC = address HERE After Instruction if FLAG<1> = 0, PC = address FALSE		Before Instruction PC = Address HERE After Instruction PC = Address THERE TOS = Address HERE+1
	if $FLAG<1> = 1$ ,	CLRF	Clear f
	PC = address TRUE	Syntax:	[ <i>label</i> ] CLRF f
		Operands:	$0 \leq f \leq 127$
		Operation:	$\begin{array}{l} 00h \rightarrow (f) \\ 1 \rightarrow Z \end{array}$
		Status Affected:	Z
		Encoding:	00 0001 1fff ffff
		Description:	The contents of register 'f' are cleared and the Z bit is set.
		Words:	1

-

### TABLE 17-2: COMPARATOR SPECIFICATIONS

Operating Conditions: 2.0V < VDD <5.5V, -40°C < TA < +125°C, unless otherwise stated.									
Param No.	Characteristics	Sym	Min	Тур	Мах	Units	Comments		
D300	Input Offset Voltage	VIOFF	_	±5.0	±10	mV			
D301	Input Common Mode Voltage	VICM	0		Vdd - 1.5*	V			
D302	Common Mode Rejection Ratio	CMRR	55*		_	db			
D303	Response Time <sup>(1)</sup>	TRESP		300	400*	ns	VDD = 3.0V to 5.5V -40° to +85°C		
			—	400	600*	ns	VDD = 3.0V to 5.5V -85° to +125°C		
			—	400	600*	ns	VDD = 2.0V to 3.0V -40° to +85°C		
D304	Comparator Mode Change to Output Valid	TMC2OV	—	300	10*	μ			

\* These parameters are characterized but not tested.

Note 1: Response time measured with one comparator input at (VDD – 1.5)/2, while the other input transitions from Vss to VDD.

### TABLE 17-3: VOLTAGE REFERENCE SPECIFICATIONS

Operating Conditions: 2.0V < VDD < 5.5V, -40°C < TA < +125°C, unless otherwise stated.									
Spec No.	Characteristics	Sym	Min	Тур	Мах	Units	Comments		
D310	Resolution	VRES	_	—	Vdd/24 Vdd/32	LSb LSb	Low Range (VRR = 1) High Range (VRR = 0)		
D311	Absolute Accuracy	Vraa	_		1/4 <sup>(2)</sup> * 1/2 <sup>(2)</sup> *	LSb LSb	Low Range (VRR = 1) High Range (VRR = 0)		
D312	Unit Resistor Value (R)	Vrur	—	2k*	_	Ω			
D313	Settling Time <sup>(1)</sup>	TSET	_	_	10*	μS			

\* These parameters are characterized but not tested.

Note 1: Settling time measured while VRR = 1 and VR<3:0> transitions from '0000' to '1111'.

2: When VDD is between 2.0V and 3.0V, the VREF output voltage levels on RA2 described by the equation:[VDD/2 ± (3 – VDD)/2] may cause the Absolute Accuracy (VRAA) of the VREF output signal on RA2 to be greater than the stated max.







FIGURE 18-10: TYPICAL INTERNAL OSCILLATOR FREQUENCY vs. TEMPERATURE VDD = 3 VOLTS



FIGURE 18-11: TYPICAL INTERNAL OSCILLATOR FREQUENCY vs. TEMPERATURE VDD = 2 VOLTS

