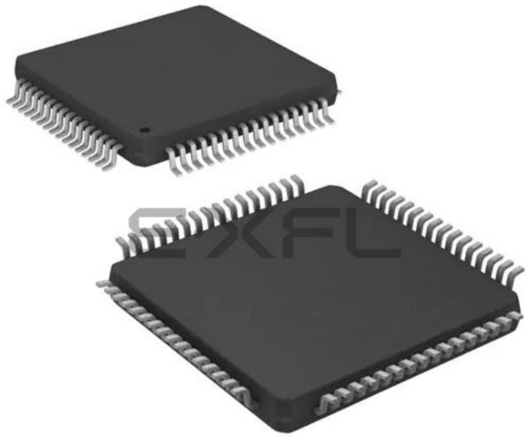


Welcome to E-XFL.COM

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, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	54
Program Memory Size	28KB (16K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 30x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1527-e-pt

TABLE 3-2: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

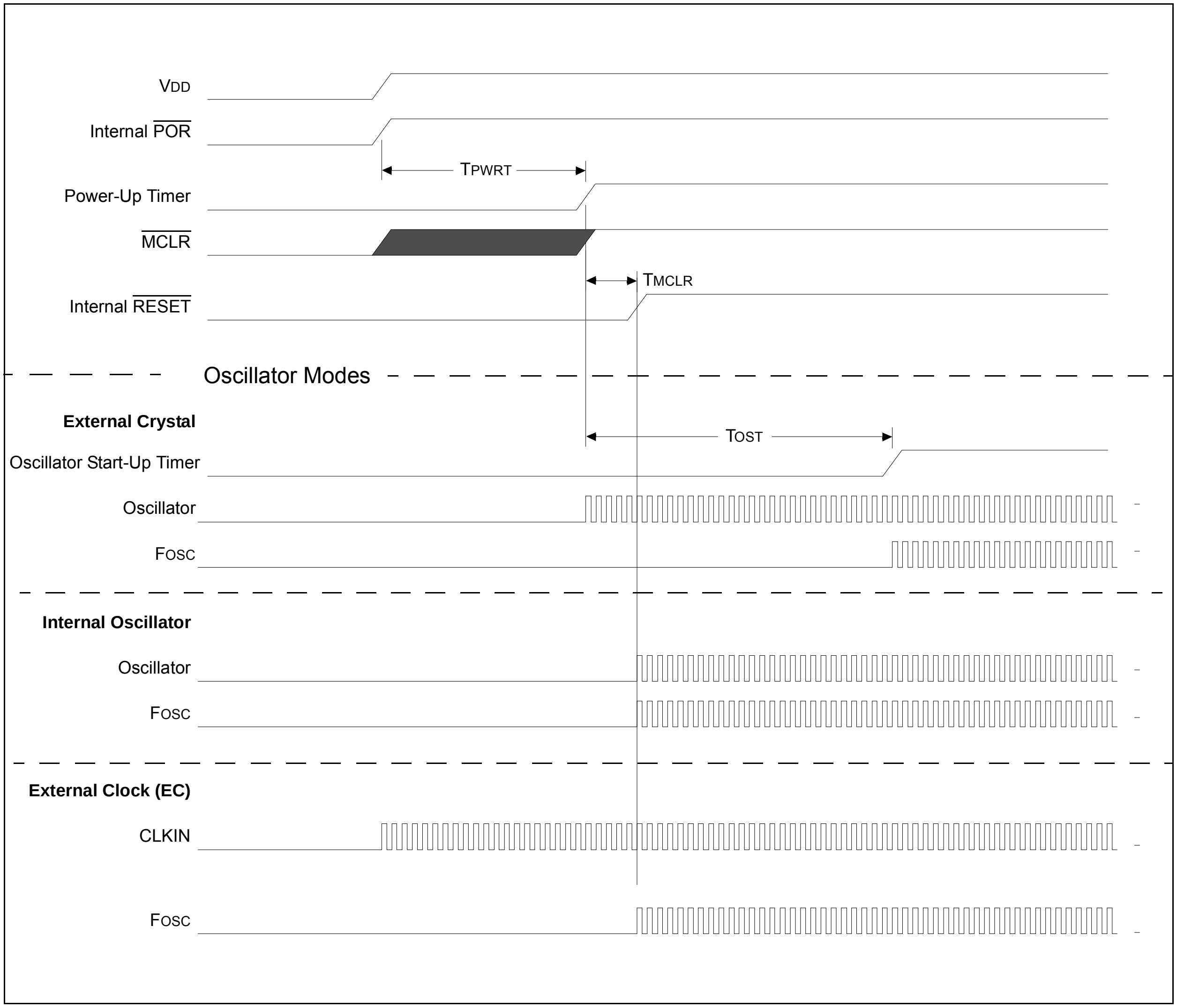
Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 4											
20Ch	—	Unimplemented								—	—
20Dh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	`1111 1111`	`1111 1111`
20Eh	—	Unimplemented								—	—
20Fh	WPUD	WPUD7	WPUD6	WPUD5	WPUD4	WPUD3	WPUD2	WPUD1	WPUD0	`1111 1111`	`1111 1111`
210h	WPUE	WPUE7	WPUE6	WPUE5	WPUE4	WPUE3	WPUE2	WPUE1	WPUE0	`1111 1111`	`1111 1111`
211h	SSP1BUF	Synchronous Serial Port Receive Buffer/Transmit Register								`xxxx xxxx`	`uuuu uuuu`
212h	SSP1ADD	Synchronous Serial Port (I^{2}C mode) Address Register								`0000 0000`	`0000 0000`
213h	SSP1MSK	Synchronous Serial Port (I^{2}C mode) Address Mask Register								`1111 1111`	`1111 1111`
214h	SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	`0000 0000`	`0000 0000`
215h	SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				`0000 0000`	`0000 0000`
216h	SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	`0000 0000`	`0000 0000`
217h	SSP1CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	`0000 0000`	`0000 0000`
218h	—	Unimplemented								—	—
219h	SSP2BUF	Synchronous Serial Port Receive Buffer/Transmit Register								`xxxx xxxx`	`uuuu uuuu`
21Ah	SSP2ADD	Synchronous Serial Port (I^{2}C mode) Address Register								`0000 0000`	`0000 0000`
21Bh	SSP2MSK	Synchronous Serial Port (I^{2}C mode) Address Mask Register								`1111 1111`	`1111 1111`
21Ch	SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	`0000 0000`	`0000 0000`
21Dh	SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				`0000 0000`	`0000 0000`
21Eh	SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	`0000 0000`	`0000 0000`
21Fh	SSP2CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	`0000 0000`	`0000 0000`
Bank 5											
28Ch	PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	`xxxx xxxx`	`uuuu uuuu`
28Dh	PORTG	—	—	RG5	RG4	RG3	RG2	RG1	RG0	`--xx xxxx`	`--uu uuuu`
28Eh	—	Unimplemented								—	—
28Fh	—	Unimplemented								—	—
290h	—	Unimplemented								—	—
291h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								`xxxx xxxx`	`uuuu uuuu`
292h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								`xxxx xxxx`	`uuuu uuuu`
293h	CCP1CON	—	—	DC1B<1:0>		CCP1M<3:0>				`--00 0000`	`--00 0000`
294h	—	Unimplemented								—	—
295h	—	Unimplemented								—	—
296h	—	Unimplemented								—	—
297h	—	Unimplemented								—	—
298h	CCPR2L	Capture/Compare/PWM Register 2 (LSB)								`xxxx xxxx`	`uuuu uuuu`
299h	CCPR2H	Capture/Compare/PWM Register 2 (MSB)								`xxxx xxxx`	`uuuu uuuu`
29Ah	CCP2CON	—	—	DC2B<1:0>		CCP2M<3:0>				`--00 0000`	`--00 0000`
29Bh	—	Unimplemented								—	—
29Ch	—	Unimplemented								—	—
29Dh	CCPTMRS0	C4TSEL<1:0>		C3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<1:0>		`0000 0000`	`0000 0000`
29Eh	CCPTMRS1	C8TSEL<1:0>		C7TSEL<1:0>		C6TSEL<1:0>		C5TSEL<1:0>		`0000 0000`	`0000 0000`
29Fh	CCPTMRS2	—	—	—	—	C10TSEL<1:0>		C9TSEL<1:0>		`---- 0000`	`---- 0000`

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

Note 1: PIC16F1526/7 only.

2: Unimplemented, read as ‘`1`’.

FIGURE 6-3: RESET START-UP SEQUENCE



12.0 I/O PORTS

In general, when a peripheral is enabled on a port pin, that pin cannot be used as a general purpose output. However, the pin can still be read.

Each port has three standard registers for its operation. These registers are:

- TRISx registers (data direction)
- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)

Some ports may have one or more of the following additional registers. These registers are:

- ANSELx (analog select)
- WPUx (weak pull-up)

TABLE 12-1: PORT AVAILABILITY PER DEVICE

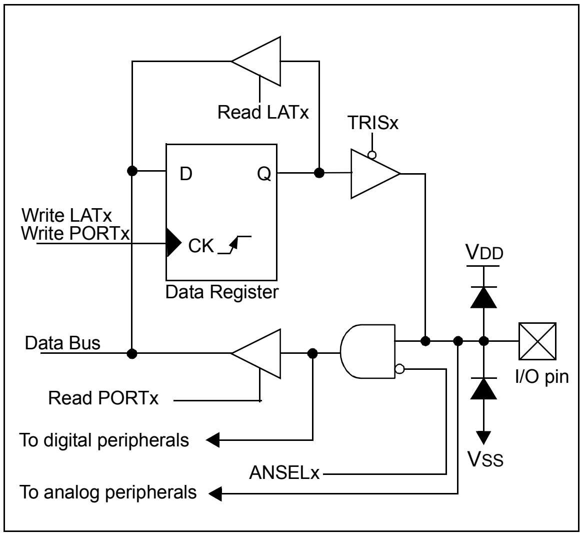
Device	PORTA	PORTB	PORTC	PORTD	PORTE	PORTF	PORTG
PIC16(L)F1526	●	●	●	●	●	●	●
PIC16(L)F1527	●	●	●	●	●	●	●

The Data Latch (LATA register) is useful for read-modify-write operations on the value that the I/O pins are driving.

A write operation to the LATA register has the same effect as a write to the corresponding PORTA register. A read of the LATA register reads of the values held in the I/O PORT latches, while a read of the PORTA register reads the actual I/O pin value.

Ports that support analog inputs have an associated ANSELx register. When an ANSEL bit is set, the digital input buffer associated with that bit is disabled. Disabling the input buffer prevents analog signal levels on the pin between a logic high and low from causing excessive current in the logic input circuitry. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 12-1.

FIGURE 12-1: GENERIC I/O PORT OPERATION



12.6 Register Definitions: PORTB

REGISTER 12-6: PORTB: PORTB REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
bit 7							bit 0

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

bit 7-0 **RB<7:0>**: PORTB I/O Pin bits[(1)]
`1` = Port pin is $\geq V_{IH}$
`0` = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

REGISTER 12-7: TRISB: PORTB TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
bit 7							bit 0

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

bit 7-0 **TRISB<7:0>:** PORTB Tri-State Control bits
`1` = PORTB pin configured as an input (tri-stated)
`0` = PORTB pin configured as an output

REGISTER 12-8: LATB: PORTB DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
bit 7							bit 0

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

bit 7-0 **LATB<7:0>**: PORTB Output Latch Value bits[(1)]

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

12.7 PORTC Registers

12.7.1 DATA REGISTER

PORTC is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISC (Register 12-12). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 12-11) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

12.7.2 DIRECTION CONTROL

The TRISC register (Register 12-12) controls the PORTC pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISC register are maintained set when using them as analog inputs. I/O pins configured as analog input always read ‘0’.

12.7.3 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each PORTC pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-7.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in the priority list.

TABLE 12-7: PORTC OUTPUT PRIORITY

Pin Name	Function Priority[1]
RC0	SOSCO RC0
RC1	SOSCI CCP2 RC1
RC2	CCP1 RC2
RC3	SCL1 SCK1 RC3[2]
RC4	SDA1 RC4[2]
RC5	SDO1 RC5
RC6	CK1 TX1 RC6
RC7	DT1 RC7

Note 1: Priority listed from highest to lowest.

2: RC3 and RC4 read the I^2C ST input when I^2C mode is enabled.

12.12 Register Definitions: PORTE

REGISTER 12-19: PORTE: PORTE REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
bit 7							bit 0

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

bit 7-0 **RE<7:0>**: PORTE General Purpose I/O Pin bits[(1)]
1 = Port pin is ≥ V_{IH}
0 = Port pin is ≤ V_{IL}

Note 1: Writes to PORTE are actually written to corresponding LATE register. Reads from PORTE register is return of actual I/O pin values.

REGISTER 12-20: TRISE: PORTE TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0
bit 7							bit 0

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

bit 7-0 **TRISE<7:0>:** PORTE Tri-State Control bits
1 = PORTE pin configured as an input (tri-stated)
0 = PORTE pin configured as an output

REGISTER 12-21: LATE: PORTE DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0
bit 7							bit 0

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

bit 7-0 **LATE<7:0>**: PORTE Output Latch Value bits[(1)]

Note 1: Writes to PORTE are actually written to corresponding LATE register. Reads from PORTE register is return of actual I/O pin values.

REGISTER 12-27: ANSELF: PORTF ANALOG SELECT REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSF7	ANSF6	ANSF5	ANSF4	ANSF3	ANSF2	ANSF1	ANSF0
bit 7							bit 0

Legend:

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

bit 7-0 **ANSF<7:0>**: Analog Select between Analog or Digital Function on pins RF<7:0>, respectively
1 = Analog input. Pin is assigned as analog input(1). Digital input buffer disabled.
0 = Digital I/O. Pin is assigned to port or digital special function.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

TABLE 12-14: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELF	ANSF7	ANSF6	ANSF5	ANSF4	ANSF3	ANSF2	ANSF1	ANSF0	130
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	129
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	129
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	129

Legend: x = unknown, u = unchanged, - = unimplemented locations read as ‘0’. Shaded cells are not used by PORTF.

TABLE 12-15: SUMMARY OF CONFIGURATION WORD WITH PORTF

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG2	13:8	—		LVP	$\overline{\text{DEBUG}}$	$\overline{\text{LPBOR}}$	BORV	STVREN	—	45
	7:0	—	—	—	$\overline{\text{VCAPEN}}$(1)	—	—	WRT<1:0>		

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

Note 1: PIC16F1526/7 only.

17.1.3 SOFTWARE PROGRAMMABLE PRESCALER

A software programmable prescaler is available for exclusive use with Timer0. The prescaler is enabled by clearing the PSA bit of the OPTION_REG register.

Note:	The Watchdog Timer (WDT) uses its own independent prescaler.

There are 8 prescaler options for the Timer0 module ranging from 1:2 to 1:256. The prescale values are selectable via the PS<2:0> bits of the OPTION_REG register. In order to have a 1:1 prescaler value for the Timer0 module, the prescaler must be disabled by setting the PSA bit of the OPTION_REG register.

The prescaler is not readable or writable. All instructions writing to the TMR0 register will clear the prescaler.

17.1.4 TIMER0 INTERRUPT

Timer0 will generate an interrupt when the TMR0 register overflows from FFh to 00h. The TMR0IF interrupt flag bit of the INTCON register is set every time the TMR0 register overflows, regardless of whether or not the Timer0 interrupt is enabled. The TMR0IF bit can only be cleared in software. The Timer0 interrupt enable is the TMR0IE bit of the INTCON register.

Note:	The Timer0 interrupt cannot wake the processor from Sleep since the timer is frozen during Sleep.

17.1.5 8-BIT COUNTER MODE SYNCHRONIZATION

When in 8-bit Counter mode, the incrementing edge on the T0CKI pin must be synchronized to the instruction clock. Synchronization can be accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the instruction clock. The high and low periods of the external clocking source must meet the timing requirements as shown in **Section 25.0 “Electrical Specifications”**.

17.1.6 OPERATION DURING SLEEP

Timer0 cannot operate while the processor is in Sleep mode. The contents of the TMR0 register will remain unchanged while the processor is in Sleep mode.

18.11 Register Definitions: Timer1/3/5 Control

REGISTER 18-1: TxCON: TIMER1/3/5 CONTROL REGISTER

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

Legend:

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

bit 7-6 **TMRxCS<1:0>:** Timer1/3/5 Clock Source Select bits

11 = Timer1/3/5 clock source is LFINTOSC
10 = Timer1/3/5 clock source is pin or oscillator:
If SOSCEN = 0:
External clock from TxCKI pin (on the rising edge)
If SOSCEN = 1:
Crystal oscillator on SOSCI/SOSCO pins
01 = Timer1/3/5 clock source is system clock (FOSC)
00 = Timer1/3/5 clock source is instruction clock (FOSC/4)

bit 5-4 **TxCKPS<1:0>:** Timer1/3/5 Input Clock Prescale Select bits

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

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

1 = Dedicated secondary oscillator circuit enabled
0 = Dedicated secondary oscillator circuit disabled

bit 2 **$\overline{\text{TxSYNC}}$:** Timer1/3/5 External Clock Input Synchronization Control bit

TMRxCS<1:0> = 1X:
1 = Do not synchronize external clock input
0 = Synchronize external clock input with system clock (FOSC)

TMRxCS<1:0> = 0X:
This bit is ignored.

bit 1 **Unimplemented:** Read as ‘0’

bit 0 **TMRxON:** Timer1/3/5 On bit

1 = Enables Timer1/3/5
0 = Stops Timer1/3/5
Clears Timer1/3/5 gate flip-flop

REGISTER 20-3: CCPTMRS1: CCP TIMER SELECTION CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
C8TSEL<1:0>		C7TSEL<1:0>		C6TSEL<1:0>		C5TSEL<1:0>	
bit 7							bit 0

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

bit 7-6 **C8TSEL<1:0>:** CCP8 Timer Selection bits

When in Capture/Compare mode:

x1 = CCP8 is based off Timer5 in Capture/Compare mode
x0 = CCP8 is based off Timer1 in Capture/Compare mode

When in PWM mode:

11 = Reserved
10 = CCP8 is based off Timer10 in PWM mode
01 = CCP8 is based off Timer8 in PWM mode
00 = CCP8 is based off Timer2 in PWM mode

bit 5-4 **C7TSEL<1:0>:** CCP7 Timer Selection bits

When in Capture/Compare mode:

x1 = CCP7 is based off Timer5 in Capture/Compare mode
x0 = CCP7 is based off Timer1 in Capture/Compare mode

When in PWM mode:

11 = Reserved
10 = CCP7 is based off Timer8 in PWM mode
01 = CCP7 is based off Timer6 in PWM mode
00 = CCP7 is based off Timer2 in PWM mode

bit 3-2 **C6TSEL<1:0>:** CCP6 Timer Selection bits

When in Capture/Compare mode:

x1 = CCP6 is based off Timer5 in Capture/Compare mode
x0 = CCP6 is based off Timer1 in Capture/Compare mode

When in PWM mode:

11 = Reserved
10 = CCP6 is based off Timer8 in PWM mode
01 = CCP6 is based off Timer6 in PWM mode
00 = CCP6 is based off Timer2 in PWM mode

bit 1-0 **C5TSEL<1:0>:** CCP5 Timer Selection bits

When in Capture/Compare mode:

x1 = CCP5 is based off Timer5 in Capture/Compare mode
x0 = CCP5 is based off Timer1 in Capture/Compare mode

When in PWM mode:

11 = Reserved
10 = CCP5 is based off Timer8 in PWM mode
01 = CCP5 is based off Timer6 in PWM mode
00 = CCP5 is based off Timer2 in PWM mode

FIGURE 21-14: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 0, DHEN = 0)

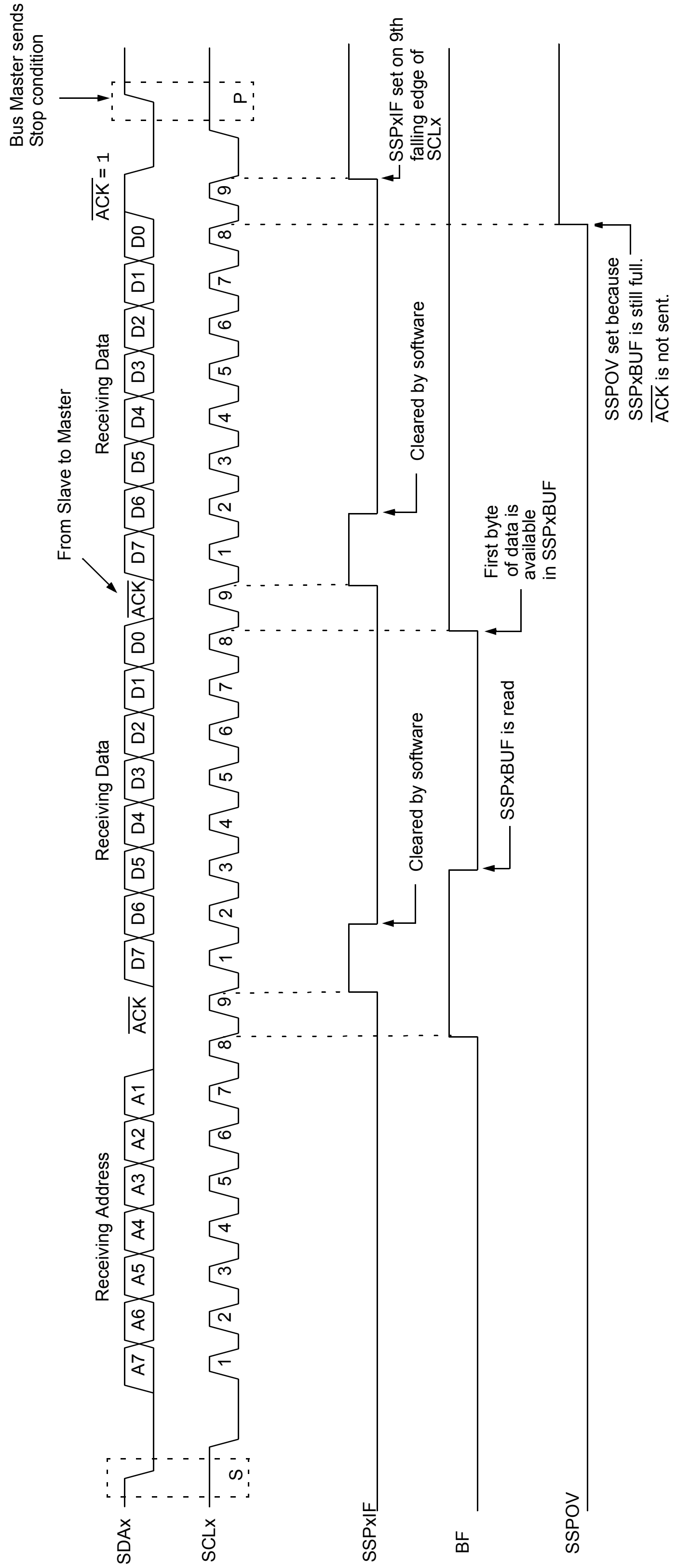
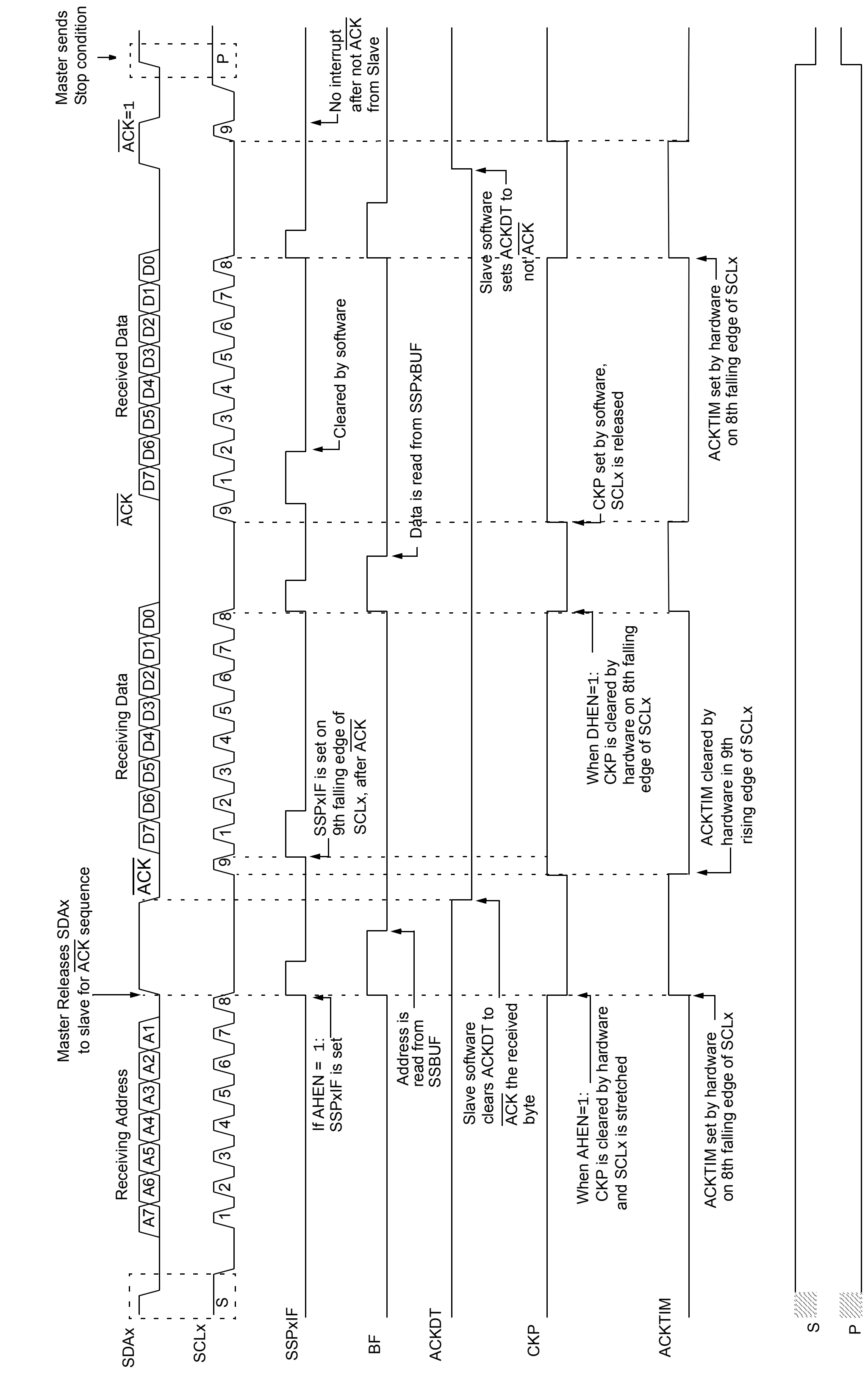


FIGURE 21-16: I^{2}C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)


SDAx
SCLx
SSPxIF
BF
ACKDT
CKP
ACKTIM
S
P
S
Receiving Address
A7
A6
A5
A4
A3
A2
A1
1
2
3
4
5
6
7
8
9
Master Releases SDAx to slave for ACK sequence
ACK
Receiving Data
D7
D6
D5
D4
D3
D2
D1
D0
ACK
Received Data
D7
D6
D5
D4
D3
D2
D1
D0
ACK=1
P
Master sends Stop condition
If AHEN = 1: SSPxIF is set
SSPxIF is set on 9th falling edge of SCLx, after ACK
Cleared by software
No interrupt after not ACK from Slave
Address is read from SSBUF
Data is read from SSPxBUF
Slave software clears ACKDT to ACK the received byte
Slave software sets ACKDT to not ACK
When AHEN=1: CKP is cleared by hardware and SCLx is stretched
When DHEN=1: CKP is cleared by hardware on 8th falling edge of SCLx
CKP set by software, SCLx is released
ACKTIM set by hardware on 8th falling edge of SCLx
ACKTIM cleared by hardware in 9th rising edge of SCLx
ACKTIM set by hardware on 8th falling edge of SCLx

22.1 EUSART Asynchronous Mode

The EUSART transmits and receives data using the standard non-return-to-zero (NRZ) format. NRZ is implemented with two levels: a VOH mark state which represents a '1' data bit, and a VOL space state which represents a '0' data bit. NRZ refers to the fact that consecutively transmitted data bits of the same value stay at the output level of that bit without returning to a neutral level between each bit transmission. An NRZ transmission port idles in the mark state. Each character transmission consists of one Start bit followed by eight or nine data bits and is always terminated by one or more Stop bits. The Start bit is always a space and the Stop bits are always marks. The most common data format is 8 bits. Each transmitted bit persists for a period of 1/(Baud Rate). An on-chip dedicated 8-bit/16-bit Baud Rate Generator is used to derive standard baud rate frequencies from the system oscillator. See Table 22-5 for examples of baud rate configurations.

The EUSART transmits and receives the LSb first. The EUSART's transmitter and receiver are functionally independent, but share the same data format and baud rate. Parity is not supported by the hardware, but can be implemented in software and stored as the ninth data bit.

22.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 22-1. The heart of the transmitter is the serial Transmit Shift Register (TSR), which is not directly accessible by software. The TSR obtains its data from the transmit buffer, which is the TXxREG register.

22.1.1.1 Enabling the Transmitter

The EUSART transmitter is enabled for asynchronous operations by configuring the following three control bits:

- TXEN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the TXEN bit of the TXxSTA register enables the transmitter circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the TXx/CKx I/O pin as an output. If the TXx/CKx pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

Note:	The TXxIF transmitter interrupt flag is set when the TXEN enable bit is set.

22.1.1.2 Transmitting Data

A transmission is initiated by writing a character to the TXxREG register. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXxREG is immediately transferred to the TSR register. If the TSR still contains all or part of a previous character, the new character data is held in the TXxREG until the Stop bit of the previous character has been transmitted. The pending character in the TXxREG is then transferred to the TSR in one TCY immediately following the Stop bit transmission. The transmission of the Start bit, data bits and Stop bit sequence commences immediately following the transfer of the data to the TSR from the TXxREG.

22.1.1.3 Transmit Data Polarity

The polarity of the transmit data can be controlled with the SCKP bit of the BAUDxCON register. The default state of this bit is '0' which selects high true transmit idle and data bits. Setting the SCKP bit to '1' will invert the transmit data resulting in low true idle and data bits. The SCKP bit controls transmit data polarity only in Asynchronous mode. In Synchronous mode the SCKP bit has a different function.

22.1.1.4 Transmit Interrupt Flag

The TXxIF interrupt flag bit of the PIR1/PIR4 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXxIF bit is only clear when the TSR is busy with a character and a new character has been queued for transmission in the TXxREG. The TXxIF flag bit is not cleared immediately upon writing TXxREG. TXxIF becomes valid in the second instruction cycle following the write execution. Polling TXxIF immediately following the TXxREG write will return invalid results. The TXxIF bit is read-only, it cannot be set or cleared by software.

The TXxIF interrupt can be enabled by setting the TXxIE interrupt enable bit of the PIE1/PIE4 register. However, the TXxIF flag bit will be set whenever the TXxREG is empty, regardless of the state of TXxIE enable bit.

To use interrupts when transmitting data, set the TXxIE bit only when there is more data to send. Clear the TXxIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

22.1.2.8 Asynchronous Reception Set-up:

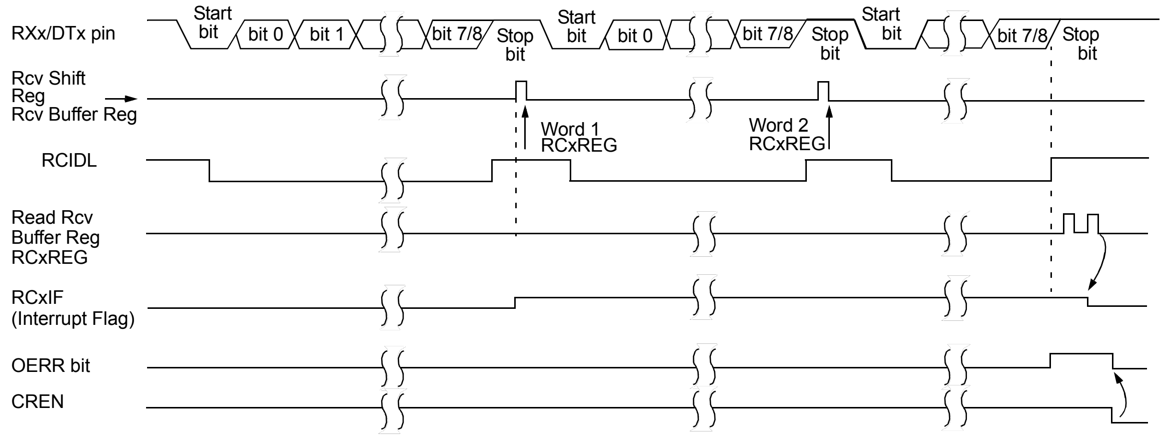
1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 22.4 “EUSART Baud Rate Generator (BRG)”**).
2. Set the RXx/DTx and TXx/CKx TRIS controls to ‘1’.
3. Enable the serial port by setting the SPEN bit and the RXx/DTx pin TRIS bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCxIE interrupt enable bit and set the GIE and PEIE bits of the INTCON register.
5. If 9-bit reception is desired, set the RX9 bit.
6. Enable reception by setting the CREN bit.
7. The RCxIF interrupt flag bit will be set when a character is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
8. Read the RCxSTA register to get the error flags and, if 9-bit data reception is enabled, the ninth data bit.
9. Get the received 8 Least Significant data bits from the receive buffer by reading the RCxREG register.
10. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.

22.1.2.9 9-bit Address Detection Mode Set-up

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 22.4 “EUSART Baud Rate Generator (BRG)”**).
2. Set the RXx/DTx and TXx/CKx TRIS controls to ‘1’.
3. Enable the serial port by setting the SPEN bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCxIE interrupt enable bit and set the GIE and PEIE bits of the INTCON register.
5. Enable 9-bit reception by setting the RX9 bit.
6. Enable address detection by setting the ADDEN bit.
7. Enable reception by setting the CREN bit.
8. The RCxIF interrupt flag bit will be set when a character with the ninth bit set is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
9. Read the RCxSTA register to get the error flags. The ninth data bit will always be set.
10. Get the received 8 Least Significant data bits from the receive buffer by reading the RCxREG register. Software determines if this is the device’s address.
11. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.
12. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and generate interrupts.

FIGURE 22-5: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RXx/DTx input. The RCxREG (receive buffer) is read after the third word, causing the OERR (overrun) bit to be set.

22.4.4 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 TXxSTA register. The Break character transmission is then initiated by a write to the TXxREG. The value of data written to TXxREG 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 TXxSTA register indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 22-9 for the timing of the Break character sequence.

22.4.4.1 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 TXxREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXxREG 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 TXxREG becomes empty, as indicated by the TXxIF, the next data byte can be written to TXxREG.

22.4.5 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 RCxSTA register and the Received data as indicated by RCxREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

A Break character has been received when;

- RCxIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in **Section 22.4.3 "Auto-Wake-up on Break"**. By enabling this feature, the EUSART will sample the next two transitions on RXx/DTx, cause an RCxIF 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 BAUDxCON register before placing the EUSART in Sleep mode.

FIGURE 22-9: SEND BREAK CHARACTER SEQUENCE

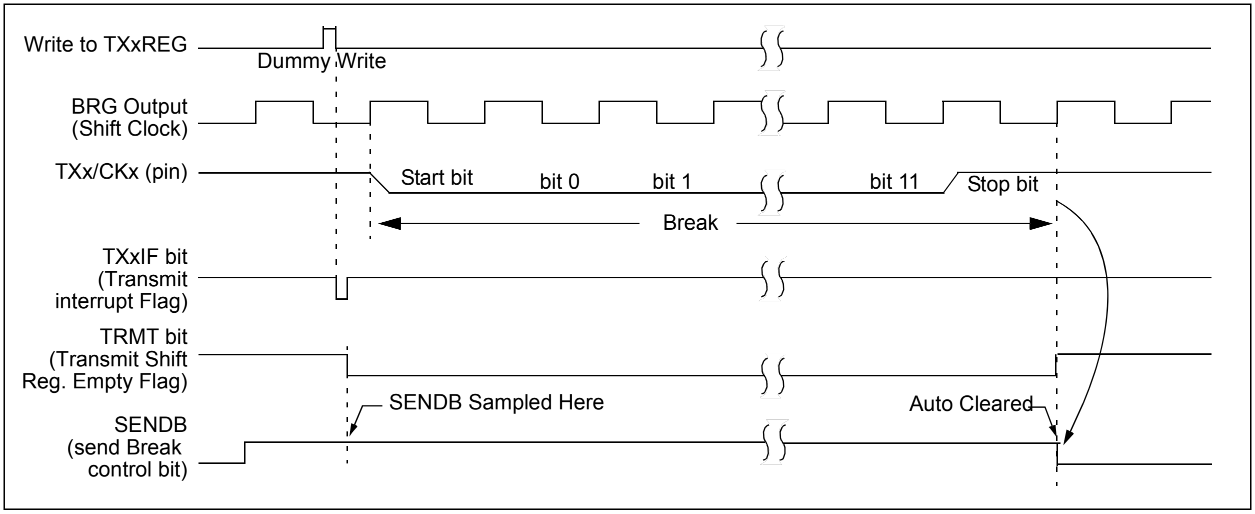


FIGURE 25-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

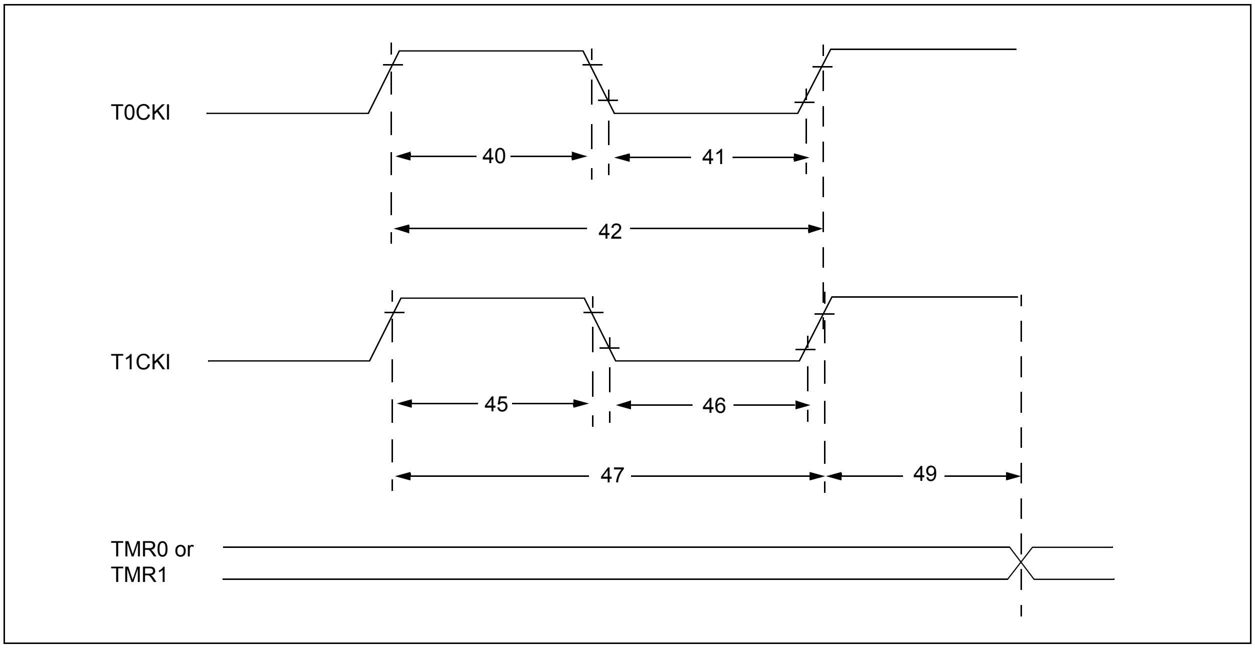


TABLE 25-5: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)
Operating Temperature -40°C ≤ TA ≤ +125°C

Param No.	Sym.	Characteristic		Min.	Typ†	Max.	Units	Conditions
40*	TT0H	T0CKI High Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
41*	TT0L	T0CKI Low Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
42*	TT0P	T0CKI Period		Greater of: 20 or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value
45*	TT1H	T1CKI High Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
46*	TT1L	T1CKI Low Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
47*	TT1P	T1CKI Input Period	Synchronous	Greater of: 30 or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value
			Asynchronous	60	—	—	ns	
48	FT1	Timer1 Oscillator Input Frequency Range (oscillator enabled by setting bit SOSCEN)		32.4	32.768	33.1	kHz	
49*	TCKEZTMR1	Delay from External Clock Edge to Timer Increment		2 TOSC	—	7 TOSC	—	Timers in Sync mode

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 26-31: IPD BASE, SLEEP MODE, PIC16LF1526 ONLY

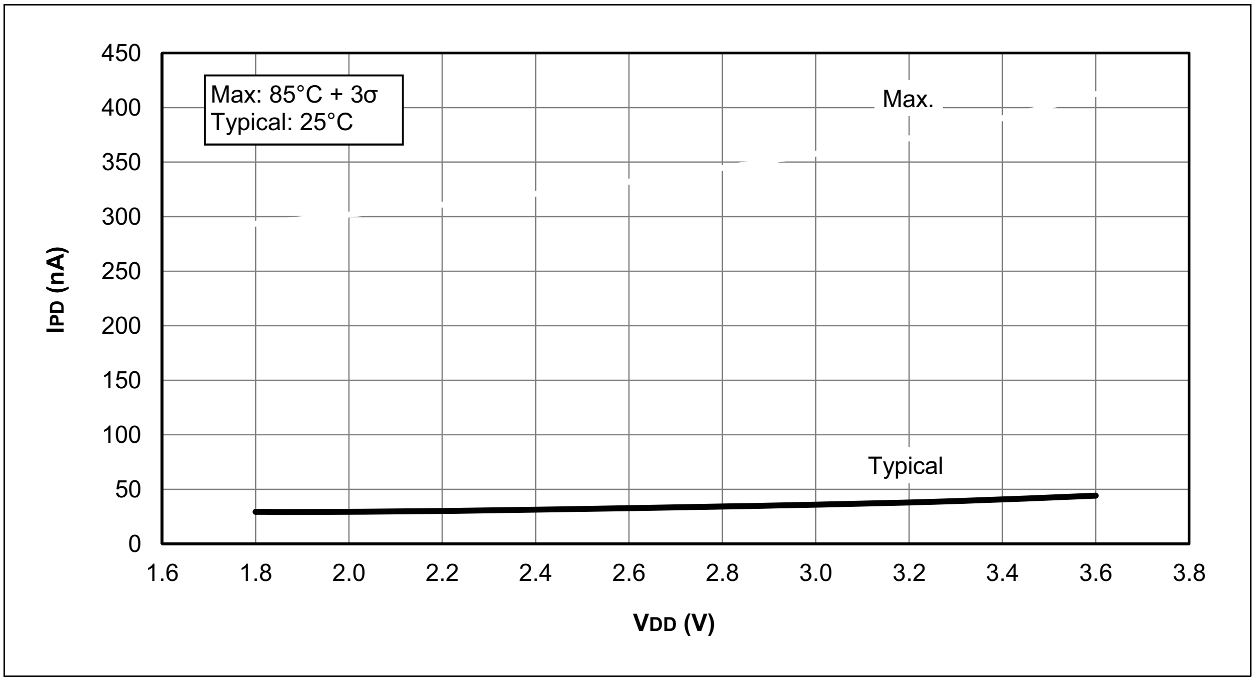


FIGURE 26-32: IPD BASE, LOW-POWER SLEEP MODE, VREGPM = 1, PIC16F1526/7 ONLY

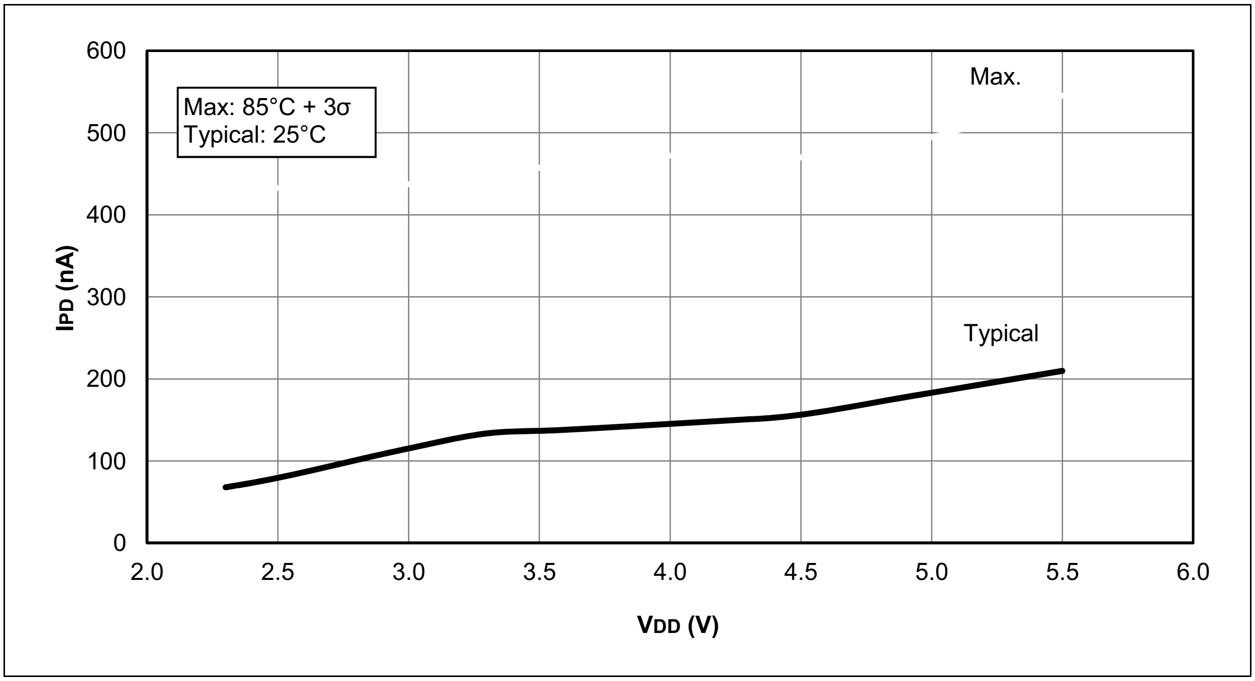


FIGURE 26-43: V_{OH} vs. I_{OH} OVER TEMPERATURE, V_{DD} = 3.0V

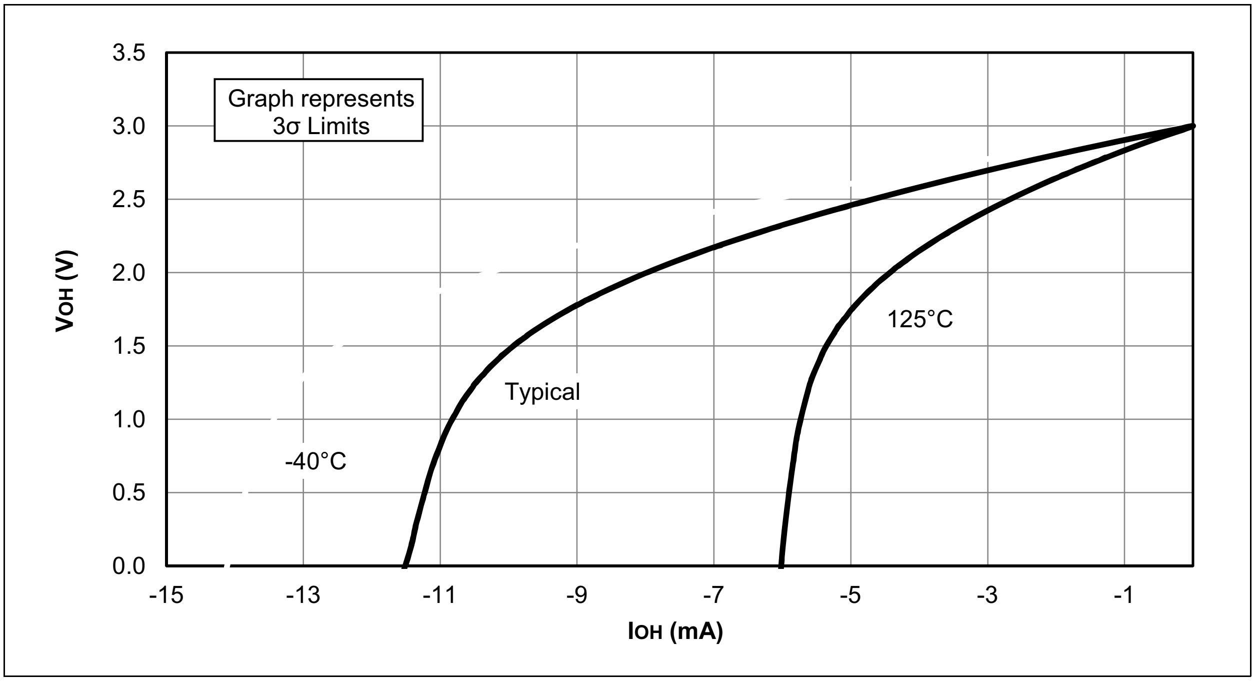


FIGURE 26-44: V_{OL} vs. I_{OL} OVER TEMPERATURE, V_{DD} = 3.0V

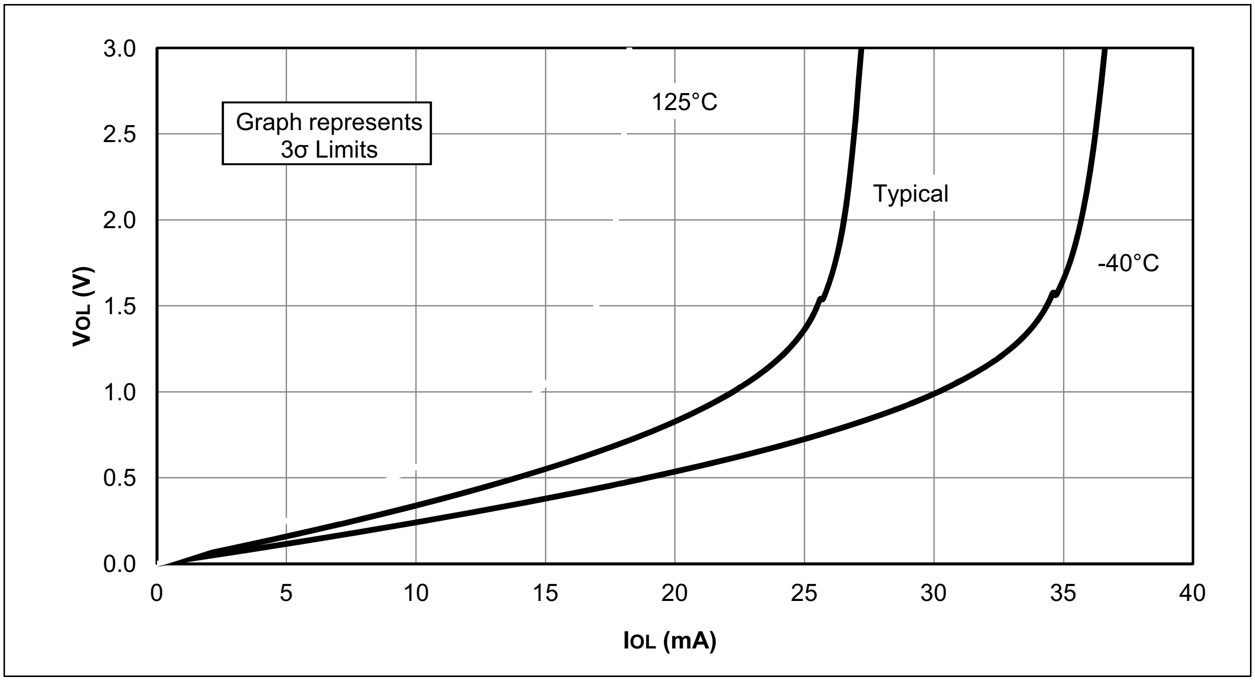


FIGURE 26-47: POR RELEASE VOLTAGE

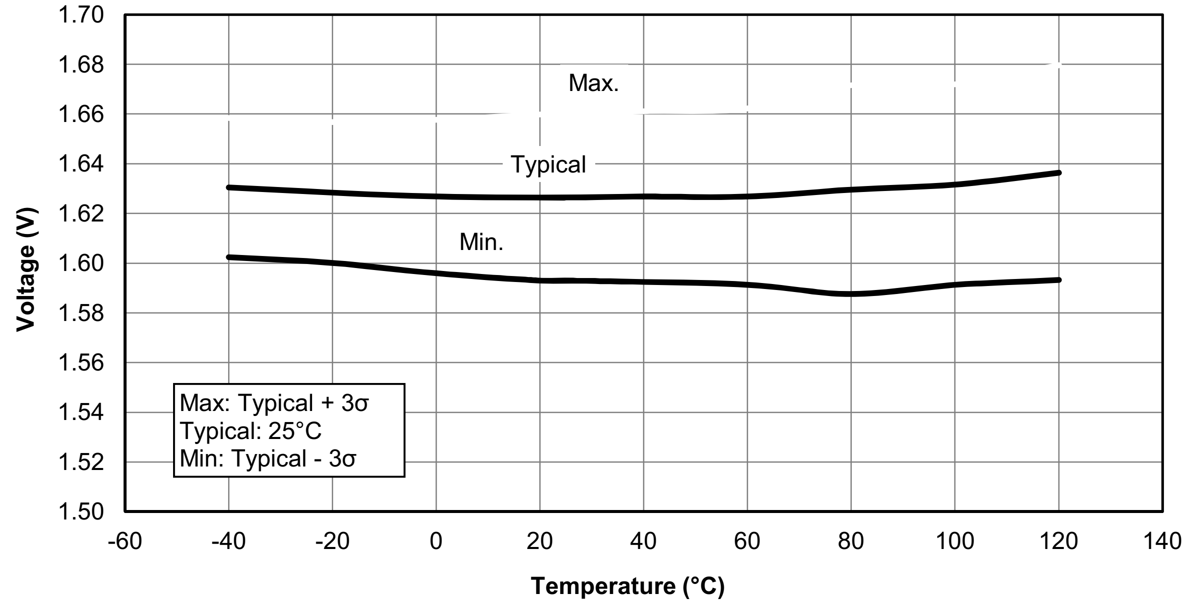
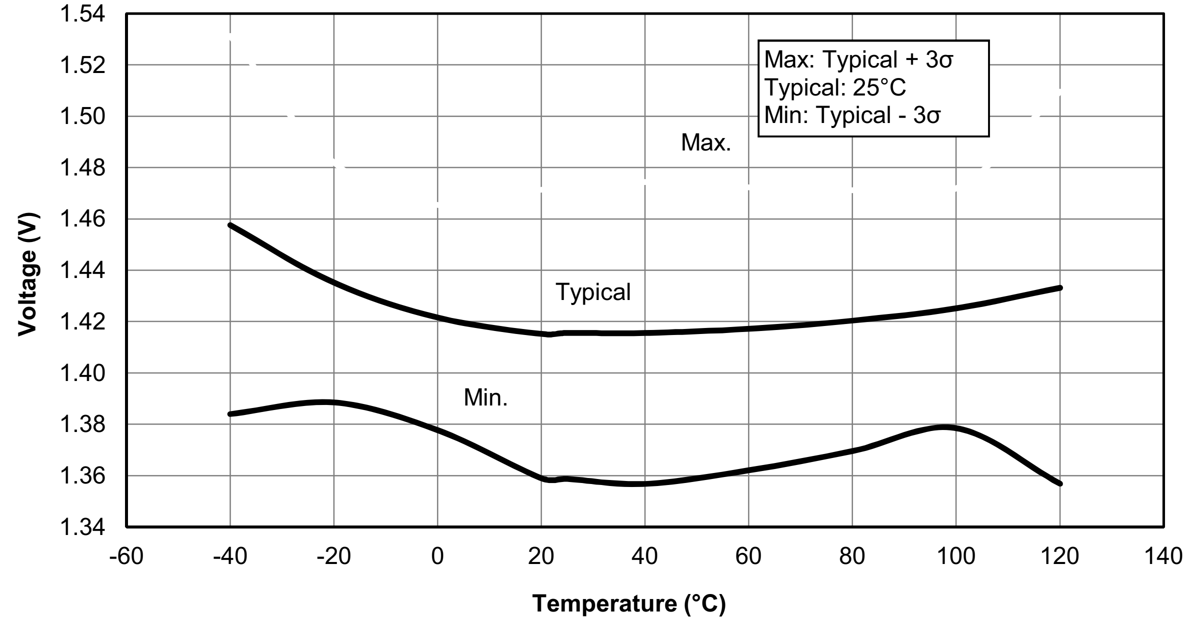


FIGURE 26-48: POR REARM VOLTAGE, PIC16F1526/7 ONLY



APPENDIX A: DATA SHEET REVISION HISTORY

Revision A (01/2011)

Original release.

Revision B (05/2011)

Electrical Spec updates.

Revision C (01/2013)

Updated Electrical Spec and added Characterization Data Graphs.

Revision D (09/2015)

Updated chapters High-Performance RISC CPU, Device Overview, Memory Organization, Device Configuration, Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART), Packaging Information. Other minor corrections.