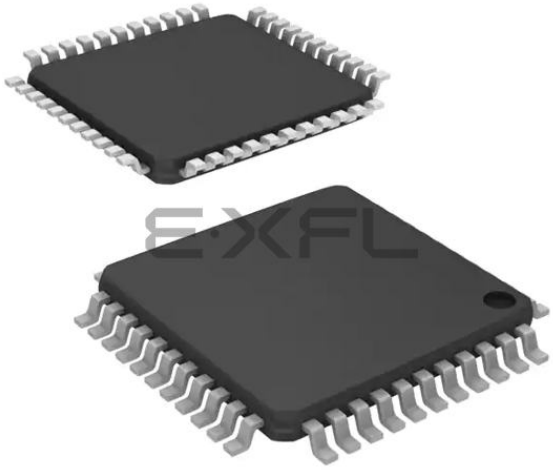


Welcome to E-XFL.COM

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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf45k42-e-pt

3.0 PIC18 CPU

This family of devices contains a PIC18 8-bit CPU core based on the modified Harvard architecture. The PIC18 CPU supports:

- System Arbitration, which decides memory access allocation depending on user priorities
- Vectored Interrupt capability with automatic two level deep context saving
- 31-level deep hardware stack with overflow and underflow reset capabilities
- Support Direct, Indirect, and Relative Addressing modes
- 8x8 Hardware Multiplier

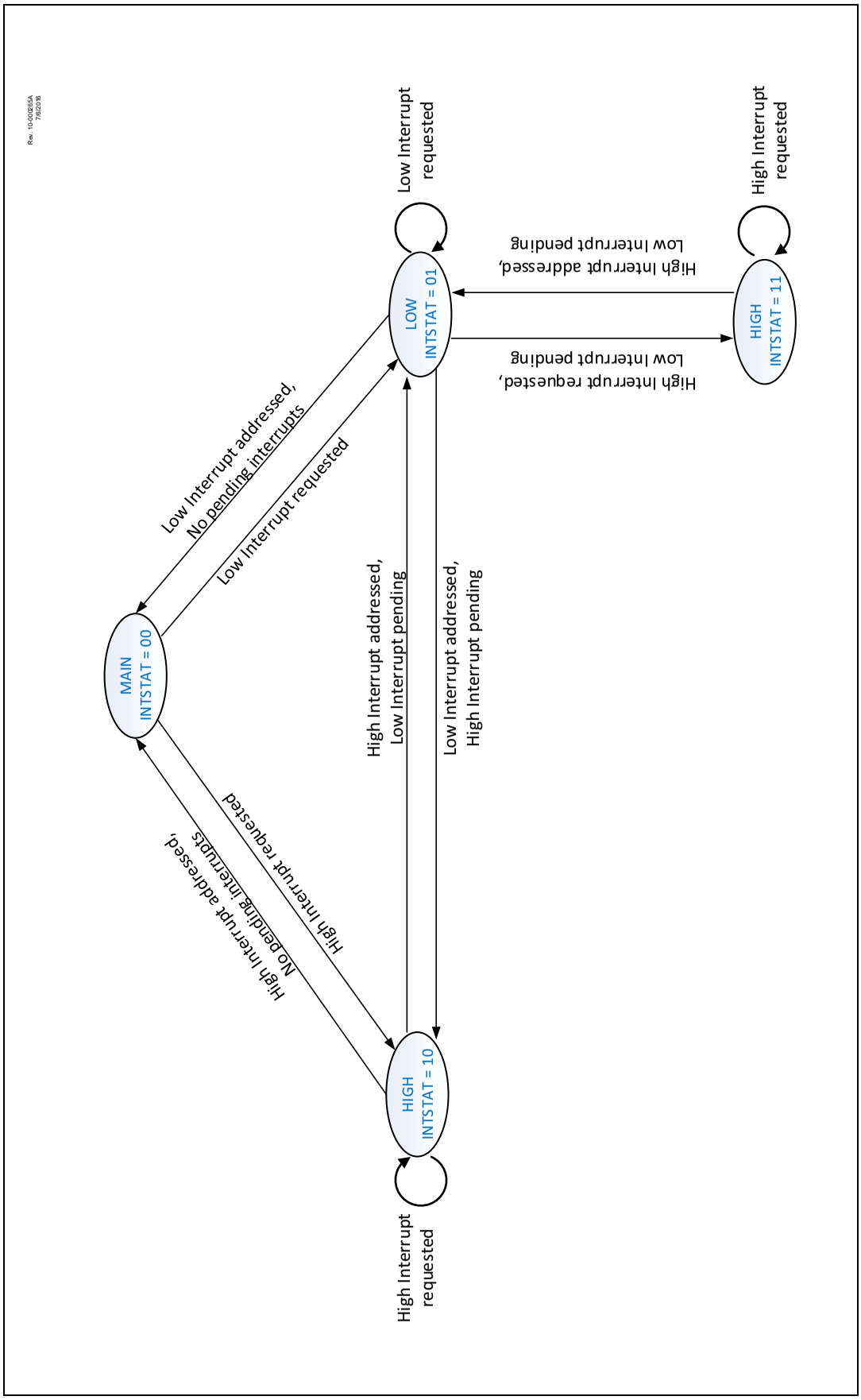


FIGURE 9-1: VECTORED INTERRUPTS STATE TRANSITION DIAGRAM

REGISTER 14-3: CRCDATH: CRC DATA HIGH BYTE REGISTER

R/W-xx	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
DATA<15:8>							
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 **DATA<15:8>**: CRC Input/Output Data bits

REGISTER 14-4: CRCDATL: CRC DATA LOW BYTE REGISTER

R/W-xx	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
DATA<7:0>							
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 7-0 **DATA<7:0>**: CRC Input/Output Data bits
Writing to this register fills the shifter.

REGISTER 14-5: CRCACCH: CRC ACCUMULATOR HIGH BYTE REGISTER

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
ACC<15:8>							
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 **ACC<15:8>**: CRC Accumulator Register bits

REGISTER 21-5: TMRxL: TIMERx LOW BYTE REGISTER

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
TMRxL<7:0>							
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 7-0 **TMRxL<7:0>:** Timerx Low Byte bits

REGISTER 21-6: TMRxH: TIMERx HIGH BYTE REGISTER

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
TMRxH<7:0>							
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 7-0 **TMRxH<7:0>:** Timerx High Byte bits

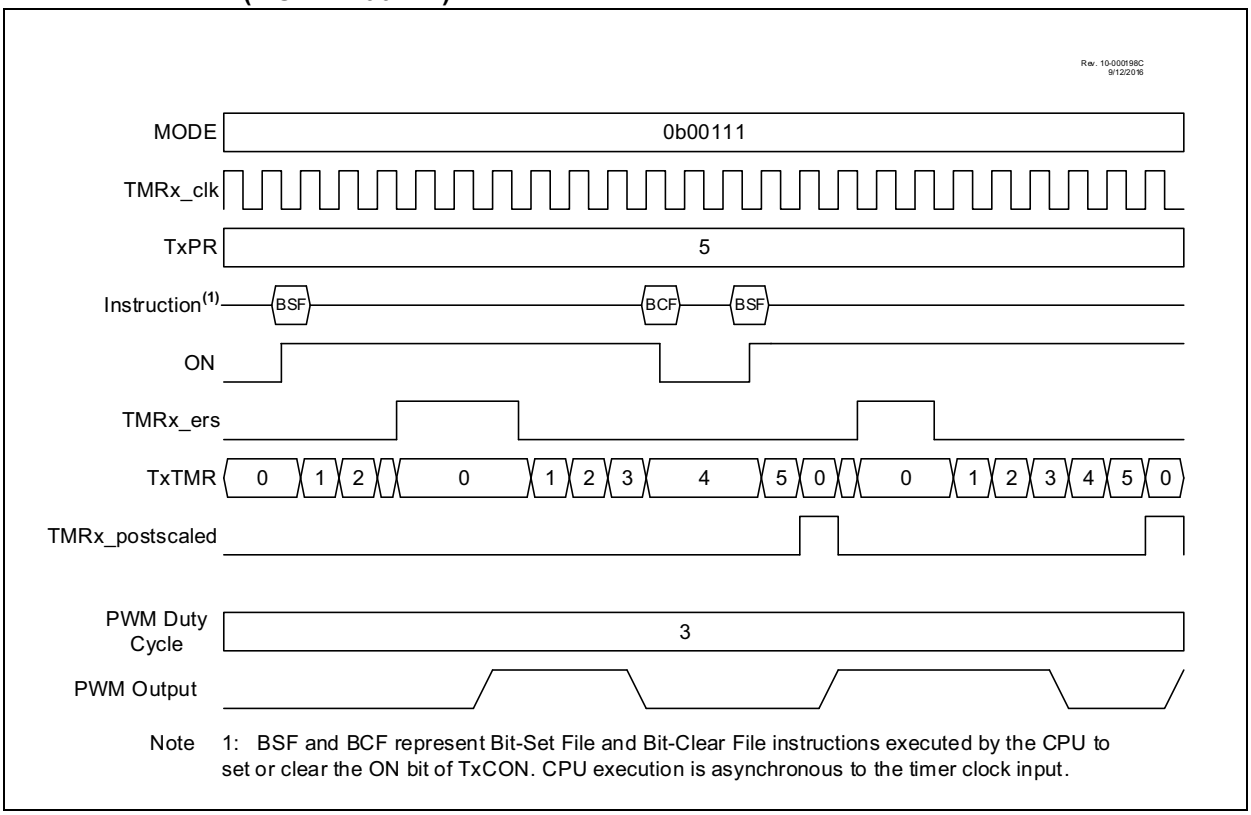
22.5.4 LEVEL-TRIGGERED HARDWARE LIMIT MODE

In the level triggered Hardware Limit Timer modes the counter is reset by high or low levels of the external signal TMR2_ers, as shown in Figure 22-7. Selecting MODE<4:0> = 00110 will cause the timer to reset on a low level external signal. Selecting MODE<4:0> = 00111 will cause the timer to reset on a high level external signal. In the example, the counter is reset while TMR2_ers = 1. ON is controlled by BSF and BCF instructions. When ON=0 the external signal is ignored.

When the CCP uses the timer as the PWM time base then the PWM output will be set high when the timer starts counting and then set low only when the timer count matches the CCPRx value. The timer is reset when either the timer count matches the T2PR value or two clock periods after the external Reset signal goes true and stays true.

The timer starts counting, and the PWM output is set high, on either the clock following the T2PR match or two clocks after the external Reset signal relinquishes the Reset. The PWM output will remain high until the timer counts up to match the CCPRx pulse width value. If the external Reset signal goes true while the PWM output is high then the PWM output will remain high until the Reset signal is released allowing the timer to count up to match the CCPRx value.

FIGURE 22-7: LEVEL TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00111)



Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

TABLE 23-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 20 MHz)

PWM Frequency	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
Timer Prescale	16	4	1	1	1	1
T2PR Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

TABLE 23-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 8 MHz)

PWM Frequency	1.22 kHz	4.90 kHz	19.61 kHz	76.92 kHz	153.85 kHz	200.0 kHz
Timer Prescale	16	4	1	1	1	1
T2PR Value	0x65	0x65	0x65	0x19	0x0C	0x09
Maximum Resolution (bits)	8	8	8	6	5	5

23.4.7 OPERATION IN SLEEP MODE

In Sleep mode, the T2TMR register will not increment and the state of the module will not change. If the CCPx pin is driving a value, it will continue to drive that value. When the device wakes up, T2TMR will continue from its previous state.

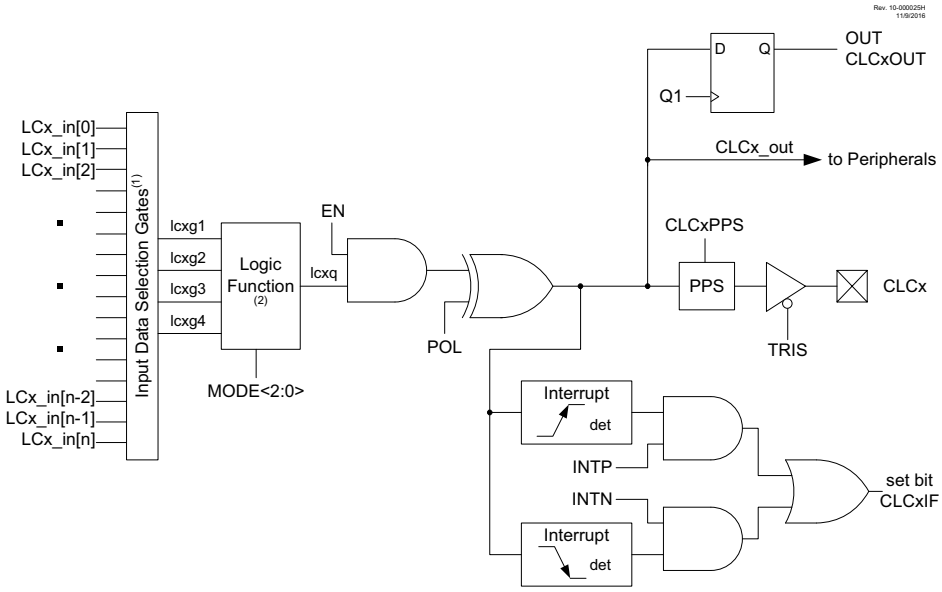
23.4.8 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency. Any changes in the system clock frequency will result in changes to the PWM frequency. See **Section 7.0 "Oscillator Module (with Fail-Safe Clock Monitor)"** for additional details.

23.4.9 EFFECTS OF RESET

Any Reset will force all ports to Input mode and the CCP registers to their Reset states.

FIGURE 27-1: CLCx SIMPLIFIED BLOCK DIAGRAM



Note 1: See Figure 27-2: Input Data Selection and Gating

2: See Figure 27-3: Programmable Logic Functions.

27.1 CLCx Setup

Programming the CLCx module is performed by configuring the four stages in the logic signal flow. The four stages are:

- Data selection
- Data gating
- Logic function selection
- Output polarity

Each stage is setup at run time by writing to the corresponding CLCx Special Function Registers. This has the added advantage of permitting logic reconfiguration on-the-fly during program execution.

27.1.1 DATA SELECTION

There are 32 signals available as inputs to the configurable logic. Four 32-input multiplexers are used to select the inputs to pass on to the next stage.

Data selection is through four multiplexers as indicated on the left side of Figure 27-2. Data inputs in the figure are identified by a generic numbered input name.

Table 27-1 correlates the generic input name to the actual signal for each CLC module. The column labeled 'DyS<4:0> Value' indicates the MUX selection code for the selected data input. DyS is an abbreviation for the MUX select input codes: D1S<4:0> through D4S<4:0>.

Data inputs are selected with CLCxSEL0 through CLCxSEL3 registers (Register 27-3 through Register 27-6).

Note: Data selections are undefined at power-up.

29.10 Register Definitions: ZCD Control

REGISTER 29-1: ZCDCON: ZERO-CROSS DETECT CONTROL REGISTER

R/W-0/0	U-0	R-x	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
SEN	—	OUT	POL	—	—	INTP	INTN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **SEN:** Zero-Cross Detect Software Enable bit
This bit is ignored when ZCDSEN configuration bit is set.
1= Zero-cross detect is enabled.
0= Zero-cross detect is disabled. ZCD pin operates according to PPS and TRIS controls.

bit 6 **Unimplemented:** Read as '0'

bit 5 **OUT:** Zero-Cross Detect Data Output bit
ZCDPOL bit = 0:
1 = ZCD pin is sinking current
0 = ZCD pin is sourcing current
ZCDPOL bit = 1:
1 = ZCD pin is sourcing current
0 = ZCD pin is sinking current

bit 4 **POL:** Zero-Cross Detect Polarity bit
1 = ZCD logic output is inverted
0 = ZCD logic output is not inverted

bit 3-2 **Unimplemented:** Read as '0'

bit 1 **INTP:** Zero-Cross Detect Positive-Going Edge Interrupt Enable bit
1 = ZCDIF bit is set on low-to-high ZCD_output transition
0 = ZCDIF bit is unaffected by low-to-high ZCD_output transition

bit 0 **INTN:** Zero-Cross Detect Negative-Going Edge Interrupt Enable bit
1 = ZCDIF bit is set on high-to-low ZCD_output transition
0 = ZCDIF bit is unaffected by high-to-low ZCD_output transition

TABLE 29-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE ZCD MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
ZCDCON	SEN	—	OUT	POL	—	—	INTP	INTN	462

Legend: — = unimplemented, read as '0'. Shaded cells are unused by the ZCD module.

The Master process is started by writing the PID to the UxP1L register when UxP2 is '0' and the UART is idle. The UxTXIF will not be set in this case. Only the six Least Significant bits of UxP1L are used in the PID transmission.

The two Most Significant bits of the transmitted PID are PID parity bits. PID<6> is the exclusive-or of PID bits 0,1,2,and 4. PID<7> is the inverse of the exclusive-or of PID bits 1,3,4,and 5.

The UART calculates and inserts these bits in the serial stream.

Writing UxP1L automatically clears the UxTXCHK and UxRXCHK registers and generates the Break, delimiter bit, Sync character (55h), and PID transmission portion of the transaction. The data portion of the transaction that follows, if there is one, is a Slave process. See **Section 31.5.2 "LIN Slave Mode"** for more details of that process. The Master receives it's own PID when RXEN is set. Software performs the Slave process corresponding to the PID that was sent and received. Attempting to write UxP1L before an active master process is complete will not succeed. Instead, the TXWRE bit will be set.

31.5.2 LIN SLAVE MODE

LIN Slave mode is configured by the following settings:

- MODE<3:0> = 1011
- TXEN = 1
- RXEN = 1
- UxP2 = Number of data bytes to transmit
- UxP3 = Number of data bytes to receive
- UxBRGH:L = Value to achieve default baud rate
- TXPOL = 0 (for high Idle state)
- STP = desired Stop bits selection
- C0EN = desired checksum mode
- RxyPPS = TX pin selection code
- TX pin TRIS control = 0
- ON = 1

The Slave process starts upon detecting a Break on the RX pin. The Break clears the UxTXCHK, UxRXCHK, UxP2, and UxP3 registers. At the end of the Break, the auto-baud circuity is activated and the baud rate is automatically set using the Sync character following the Break. The character following the Sync character is received as the PID code and is saved in the receive FIFO. The UART computes the two PID parity bits from the six Least Significant bits of the PID. If either parity bit does not match the corresponding bit of the received PID code, the PERIF flag is set and saved at the same FIFO location as the PID code. The UxRXIF bit is set indicating that the PID is available.

Software retrieves the PID by reading the UxRXB register and determines the Slave process to execute from that. The checksum method, number of data bytes, and whether to send or receive data, is defined by software according to the PID code.

31.5.2.1 LIN Slave Receiver

When the Slave process is a receiver, the software performs the following tasks:

- UxP3 register is written with a value equal to the number of data bytes to receive.
- C0EN bit is set or cleared to select the appropriate checksum. This must be completed before the Start bit of the checksum byte is received.
- Each byte of the process response is read from UxRXB when UxRXIF is set.

The UART updates the checksum on each received byte. When the last data byte is received, the computed checksum total is stored in the UxRXCHK register. The next received byte is saved in the receive FIFO and added with the value in UxRXCHK. The result of this addition is not accessible. However, if the result is not all '1's, the CERIF bit in the UxERRIR is set. The CERIF flag persists until cleared by software. Software needs to read UxRXB to remove the checksum byte from the FIFO, but the byte can be discarded if not needed for any other purpose.

After the checksum is received, the UART ignores all activity on the RX pin until a Break starts the next transaction.

31.5.2.2 LIN Slave Transmitter

When the Slave process is a transmitter, software performs the following tasks in the order shown:

- UxP2 register is written with a value equal to the number of bytes to transmit. This will enable TXIF flag which is disabled when UxP2 is '0'.
- C0EN bit is set or cleared to select the appropriate checksum
- Inter-byte delay is performed
- Each byte of the process response is written to UxTXB when UxTXIF is set

The UART accumulates the checksum as each byte is written to UxTXB. After the last byte is written, the UART stores the calculated checksum in the UxTXCHK register and transmits the inverted result as the last byte in the response.

The TXIF flag is disabled when UxP2 bytes have been written. Any writes to UxTXB that exceed the UxP2 count will be ignored and set the TXWRE flag in the UxFIFO register.

33.4.3.5 Slave Transmission (10-bit Addressing Mode)

This section describes the sequence of events for the I^2C module configured as an I^2C slave in 10-bit Addressing mode and is transmitting data. Figure 33-12 is used as a visual reference for this description.

1. Master asserts Start condition (can also be a restart) on the bus. Start condition Interrupt Flag (SCIF) in I2CxPIR register is set. If Start condition interrupt is enabled (SCIE bit is set), generic interrupt I2CxIF is set.
2. Master transmits high address byte with R/W = 0.
3. The received high address is compared with the values in I2CxADR1 and I2CxADR3 registers.
4. If high address matches; R/W is copied to R/W bit, D/A bit is cleared, high address data is copied to I2CxADB1. If the address does not match; module becomes idle.
5. If Address hold interrupt is enabled (ADRIE = 1), CSTR is set. I2CxIF is set.
6. Slave software can read high address from I2CxADB1 and set/clear ACKDT before releasing SCL.
7. ACKDT value is copied out to SDA for ACK pulse. SCL line is released by clearing CSTR.
8. Master sends ninth SCL pulse for ACK.
9. Slave can force a NACK at this point due to previous error not being cleared. E.g. Receive buffer overflow or transmit buffer underflow errors. In these cases the Slave hardware forces a NACK and the module becomes idle.
10. Master transmits low address data byte.
11. If the low address matches; SMA is set, ADRIF is set, R/W is copied to R/W bit, D/A bit is cleared, low address data is copied to I2CxADB0, and ACTDT is copied to SDA. If the address does not match; module becomes idle.
12. If address hold interrupt is enabled, the CSTR bit is set as mentioned in step 6. Slave software can read low address byte from I2CxADB0 register and change ACKDT value before releasing SCL.
13. Master sends 9th SCL pulse for ACK.
14. If the Acknowledge interrupt and hold is enabled (ACKTIE = 1), CSTR is set, I2CxIF is set.
15. Slave software can read address from I2CxADB0 and I2CxADB1 registers and change the value of ACKDT before releasing SCL by clearing CSTR.
16. Master asserts Restart condition (cannot be Start) on the bus. Restart Condition Interrupt Flag (RSCIF) is set. If the Restart Condition Interrupt is enabled, generic interrupt I2CxIF is set.
17. Master transmits high address byte with R/W = 1.
18. If SMA = 1, and if high address matches; R/W is copied to R/W bit, D/A bit is cleared, high address data is copied to I2CxADB1, and ACTDT is output to SDA. If the address does not match or SMA = 0; module become idle.
19. If ADRIE = 1, CSTR is set. I2CIF is set. Slave software can read address from I2CxADB0/1 and set/clear ACKDT. The ACKDT value is copied out to SDA. SCL is released by clearing CSTR bit.
20. If TXBE = 1 and I2CCNT!= 0 (I2CTXIF = 1), CSTR is set. Slave software must load data into I2CxTXB to release SCL.
21. Master sends SCL pulse for ACK. If I2CCNT = 0, CNTIF is set.
22. If NACK; NACKIF is set, slave goes idle.
23. If ACKTIE = 1, CSTR is set, I2CIF is set. Slave software can read address from I2CxADB0/1 before releasing SCL by clearing CSTR.
24. Master sends eight SCL pulses to clock out data.
25. Go to step 20.

bit 3 **MDR:** Master Data Request *(Master pause)*
1 = Master state mechine pauses until data is read/written to proceed (SCL is output held low)
0 = Master clocking of data is enabled.

MMA = 1 & RXBF = 1
pause_for_rx - Set by hardware on 7th falling SCL edge
- User must read from I2CRXB to release SCL
MMA = 1 & TXBE = 1 & I2CCNT!= 0
pause_for_tx - Set by hardware on 8th falling SCL edge
- User must write to I2CTXB to release SCL
ADB = 1
- I2CCNT is ignored for the high and low address in 10-bit mode
pause_for_restart - Set by hardware on 9th falling SCL edge
RSEN = 1 & MMA = 1 && I2CCNT = 0 || ACKSTAT = 1
- User must set START or write to I2CTXB to release SCL and shift Restart onto bus

bit 2-0 **MODE<2:0>:** I^2C Mode Select bits
111 = I^2C Muti-Master mode (SMBus 2.0 Host), **(5)**
Works as both mode<2:0> = 001 and mode<2:0> = 100
110 = I^2C Muti-Master mode (SMBus 2.0 Host), **(5)**
Works as both mode<2:0> = 000 and mode<2:0> = 100
101 = I^2C Master mode, 10-bit address
100 = I^2C Master mode, 7-bit address
011 = I^2C Slave mode, one 10-bit address with masking
010 = I^2C Slave mode, two 10-bit address
001 = I^2C Slave mode, two 7-bit address with masking
000 = I^2C Slave mode, four 7-bit address

Note 1: SDA and SCL pins must be configured for open-drain with internal or external pull-up
2: SDA and SCL pins must be selected as both input and output in PPS
3: CSTR can be set by more than one hardware source, all sources must be addressed by user software before the SCL line is released. CSTR is a module status bit, and does not show the true bus state.
4: SMA is set on the same SCL edge as CSTR for a matching received address
5: In this mode, ADRIE should be set, this allows an interrupt to clear the BCLIF condition and allow the ACK of matching address.
6: In 10-bit Slave mode, when ADB = 1, CSTR will set when the high address has not been read out of I2CxRXB before the low address is shifted in.

REGISTER 33-5: I2CxBTO: I²C BUS TIMEOUT SELECTION REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	BTO<2:0>		
bit 7							bit 0

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

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **BTO<2:0>:** I²C Bus Timeout Selection bits

BTO<2:0>	**I²Cx Bus Timeout Selection**
111	CLC4OUT
110	CLC3OUT
101	CLC2OUT
100	CLC1OUT
011	TMR6 post scaled output
010	TMR4 post scaled output
001	TMR2 post scaled output
000	Reserved

REGISTER 33-13: I2CxADR1: I²C ADDRESS 1 REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
ADR14	ADR13	ADR12	ADR11	ADR10	ADR9	ADR8	—
bit 7							bit 0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	—
bit 7							bit 0

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

bit 7-1 **ADR[7-1]:** Address or Divider bits

MODE<2:0> = 000 | 110 - 7-bit Slave/Multi-Master Modes

ADR<7:1>: 7-bit Slave Address

ADR<0>: Unused in this mode; bit state is a don't care

MODE<2:0> = 001 | 111 - 7-bit Slave/Multi-Master modes w/Masking

MSK0<7:1>: 7-bit Slave Address

MSK0<0>: Unused in this mode; bit state is a don't care

MODE<2:0> = 01x - 10-bit Slave Modes

ADR<14-10>: Bit pattern sent by master is fixed by I²C specification and must be equal to '11110'. However, these bit values are compared by hardware to the received data to determine a match. It is up to the user to set these bits as '11110'.

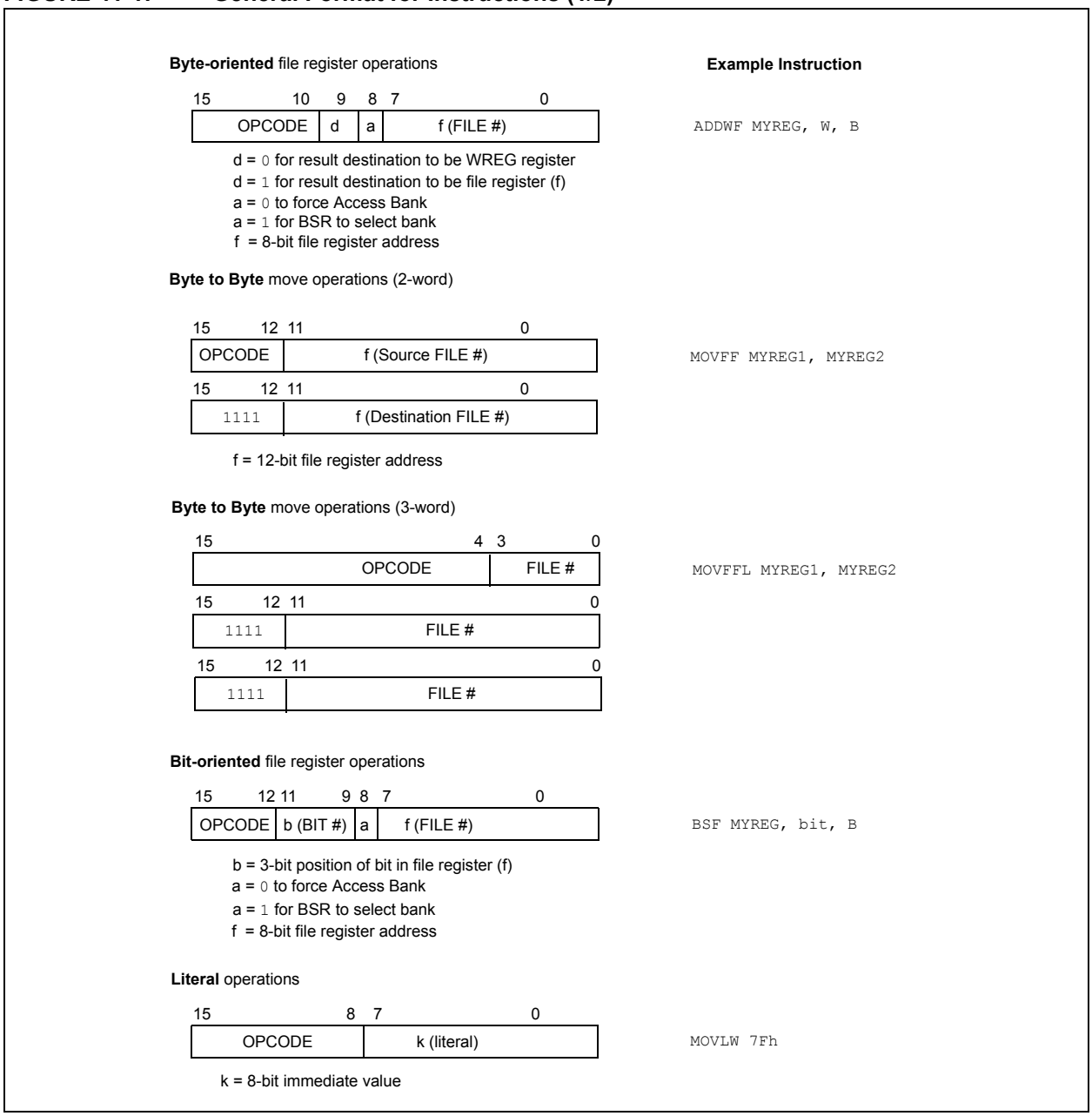
ADR<9-8>: Two Most Significant bits of 10-bit address

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

TABLE 41-1: OPCODE FIELD DESCRIPTIONS (CONTINUED)

Field	Description
< >	Register bit field
∈	In the set of
italics	User defined term (font is courier)

FIGURE 41-1: General Format for Instructions (1/2)



POP **Pop Top of Return Stack**

Syntax: 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: 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 pushing 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 = 345Ah
PC = 0124h

After Instruction
PC = 0126h
TOS = 0126h
Stack (1 level down) = 345Ah

TSTFSZ **Test f, skip if 0**

Syntax: TSTFSZ f {,a}

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

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, making this a 2-cycle instruction.
If 'a' is '0', the Access Bank is selected.
If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 00h,
PC = Address (ZERO)
If CNT ≠ 00h,
PC = Address (NZERO)

XORLW **Exclusive OR literal with W**

Syntax: XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XORed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:
```
XORLW   0AFh
```
Before Instruction
W = B5h
After Instruction
W = 1Ah

MOVSF **Move Indexed to f**

Syntax: MOVSF [z_s], f_d

Operands: $0 \le z_s \le 127$
$0 \le f_d \le 4095$

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

1st word (source)	1110	1011	0zzz	$zzzz_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of the source register are moved to destination register 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset 'z_s' in the first word to the value of FSR2. The address of the destination register is specified by the 12-bit literal 'f_d' in the second word. Both addresses can be anywhere in the 4096-byte data space (000h to FFFh).
MOVSF has curtailed the destination range to the lower 4 Kbyte space in memory (Banks 1 through 15). For everything else, use MOVSFL.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVSF [05h], REG2

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 33h

MOVSFL **Move Indexed to f (Long Range)**

Syntax: MOVSFL [z_s], f_d

Operands: $0 \le z_s \le 127$
$0 \le f_d \le 16383$

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

1st word (opcode)	0000	0000	0110	0010
2nd word (source)	1111	xxxz	zzzz	zz_sff
3rd word (full destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of the source register are moved to destination register 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset 'z_s' in the first word to the value of FSR2 (14 bits). The address of the destination register is specified by the 14-bit literal 'f_d' in the second word. Both addresses can be anywhere in the 16 Kbyte data space (0000h to 3FFFh). The MOVSFL instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register. If the resultant source address points to an indirect addressing register, the value returned will be 00h.

Words: 3

Cycles: 3

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
Decode	Read register "z" (src.)	Process data	No operation
Decode	No operation No dummy read	No operation	Write register "f" (dest.)

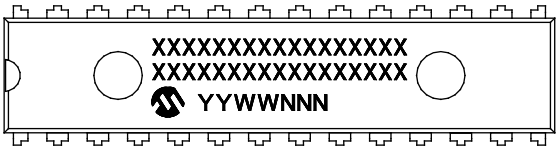
Example: MOVSFL [05h], REG2

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h

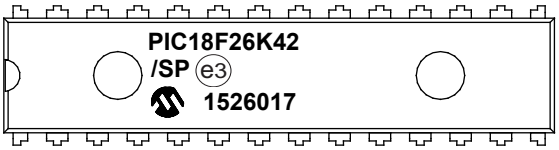
46.0 PACKAGING INFORMATION

Package Marking Information

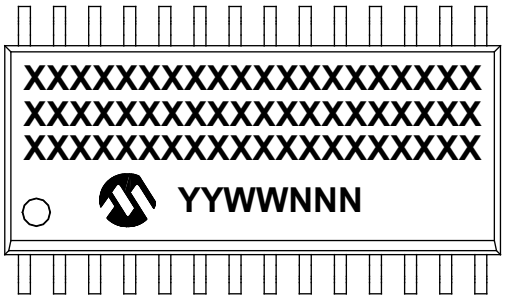
28-Lead SPDIP (.300”)



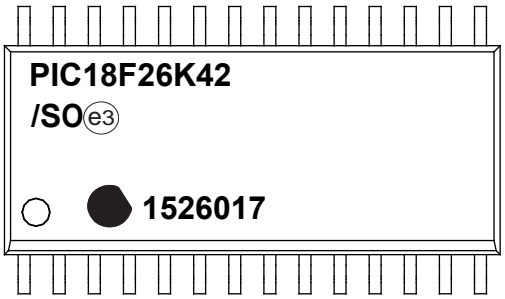
Example



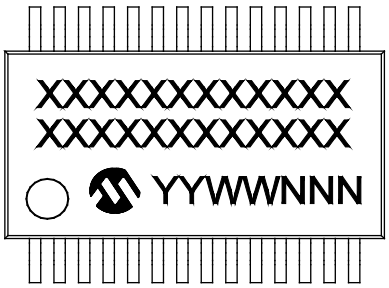
28-Lead SOIC (7.50 mm)



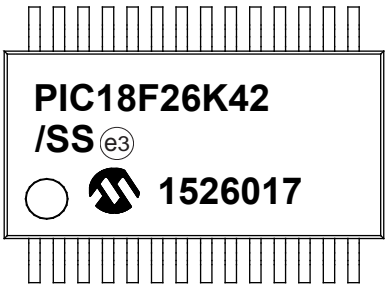
Example



28-Lead SSOP (5.30 mm)



Example

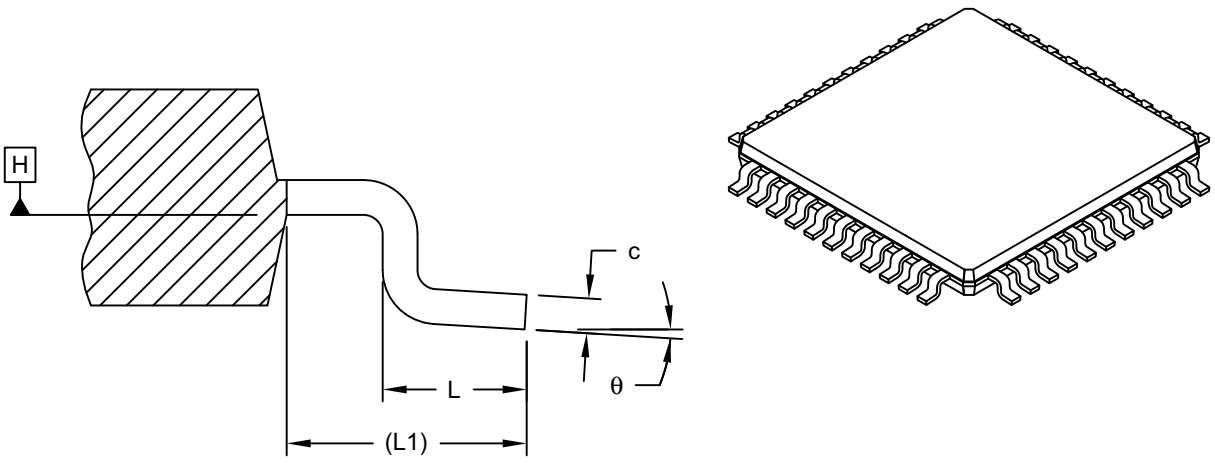


Legend:	XX...X	Customer-specific information or Microchip part number
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC® designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package.

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

44-Lead Plastic Thin Quad Flatpack (PT) - 10x10x1.0 mm Body [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



SECTION A-A

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		44	
Lead Pitch	e		0.80 BSC	
Overall Height	A	-	-	1.20
Standoff	A1	0.05	-	0.15
Molded Package Thickness	A2	0.95	1.00	1.05
Overall Width	E		12.00 BSC	
Molded Package Width	E1		10.00 BSC	
Overall Length	D		12.00 BSC	
Molded Package Length	D1		10.00 BSC	
Lead Width	b	0.30	0.37	0.45
Lead Thickness	c	0.09	-	0.20
Lead Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	θ	0°	3.5°	7°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Exact shape of each corner is optional.
3. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076C Sheet 2 of 2

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820