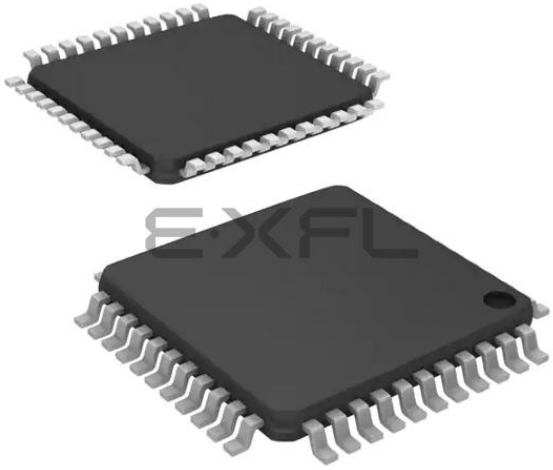


Welcome to E-XFL.COM

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	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°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/pic18lf47k42-i-pt

PIC18(L)F2X/4X/5XK42 FAMILY TYPES

Device	Data Sheet Index	Program Flash Memory (KB)	Data EEPROM (B)	Data SRAM (bytes)	I/O Pins	12-bit ADC[2] (ch)	5-bit DAC	Comparator	8-bit/ (with HLT) /16-bit Timer	Window Watchdog Timer (WWDT)	Signal Measurement Timer (SMT)	CCP/10-bit PWM	CWG	NCO	CLC	Zero-Cross Detect	Direct Memory Access (DMA) (ch)	Memory Access Partition	Vectored Interrupts	UART	I²C/SPI	Peripheral Pin Select	Peripheral Module Disable	Debug (1)
PIC18(L)F24K42	A	16	256	1024	25	24	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F25K42	A	32	256	2048	25	24	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F26K42	B	64	1024	4096	25	24	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F27K42	B	128	1024	8192	25	24	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F45K42	B	32	256	2048	36	35	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F46K42	B	64	1024	4096	36	35	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F47K42	B	128	1024	8192	36	35	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F55K42	B	32	256	2048	44	43	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F56K42	B	64	1024	4096	44	43	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I
PIC18(L)F57K42	B	128	1024	8192	44	43	1	2	3/4	Y	Y	4/4	3	1	4	Y	2	Y	Y	2	2/1	Y	Y	I

Note 1: I – Debugging integrated on chip.

Data Sheet Index:

Unshaded devices are not described in this document.

A: DS40001869 PIC18(L)F24/25K42 Data Sheet, 28-Pin

B: DS40001919 PIC18(L)F26/27/45/46/47/55/56/57K42 Data Sheet, 28/40/44/48-Pin

Note: For other small form-factor package availability and marking information, visit **http://www.microchip.com/packaging** or contact your local sales office.

9.6 Returning from Interrupt Service Routine (ISR)

The “Return from Interrupt” instruction (`RETFIE`) is used to mark the end of an ISR.

When `RETFIE 1` instruction is executed, the PC is loaded with the saved PC value from the top of the PC stack. Saved context is also restored with the execution of this instruction. Thus, execution returns to the previous state of operation that existed before the interrupt occurred.

When `RETFIE 0` instruction is executed, the saved context is not restored back to the registers.

9.7 Interrupt Latency

By assigning each interrupt with a vector address/ number (MVECEN = `1`), scanning of all interrupts is not necessary to determine the source of the interrupt.

When MVECEN = `1`, Vectored interrupt controller requires three clock cycles to vector to the ISR from main routine, thereby removing dependency of interrupt timing on compiled code.

There is a fixed latency of three instruction cycles between the completion of the instruction active when the interrupt occurred and the first instruction of the Interrupt Service Routine. Figure 9-7, Figure 9-8 and Figure 9-9 illustrate the sequence of events when a peripheral interrupt is asserted when the last executed instruction is one-cycle, two-cycle and three-cycle respectively, when MVECEN = `1`.

After the Interrupt Flag Status bit is set, the current instruction completes executing. In the first latency cycle, the contents of the PC, STATUS, WREG, BSR, FSR0/1/2, PRODL/H and PCLATH/U registers are context saved and the IVTBASE+ Vector number is calculated. In the second latency cycle, the PC is loaded with the calculated vector table address for the interrupt source and the starting address of the ISR is fetched. In the third latency cycle, the PC is loaded with the ISR address. All the latency cycles are executed as a `FNOP` instruction.

When MVECEN = `0`, Vectored interrupt controller requires two clock cycles to vector to the ISR from main routine. There is a latency of two instruction cycles plus the software latency between the completion of the instruction active when the interrupt occurred and the first instruction of the Interrupt Service Routine.

REGISTER 9-21: PIE7: PERIPHERAL INTERRUPT ENABLE REGISTER 7

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0
—	—	INT2IE	CLC2IE	CWG2IE	—	CCP2IE	TMR4IE
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 **Unimplemented:** Read as '0'

bit 5 **INT2IE:** External Interrupt 2 Enable bit
1 = Enabled
0 = Disabled

bit 4 **CLC2IE:** CLC2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **CWG2IE:** CWG2 Interrupt Enable bit
1 = Enabled
0 = Disabled

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

bit 1 **CCP2IE:** CCP2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **TMR4IE:** TMR4 Interrupt Enable bit
1 = Enabled
0 = Disabled

REGISTER 9-29: IPR4: PERIPHERAL INTERRUPT PRIORITY REGISTER 4

R/W-1/1	R/W-1/1	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
CLC1IP	CWG1IP	NCO1IP	—	CCP1IP	TMR2IP	TMR1GIP	TMR1IP
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 **CLC1IP:** CLC1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **CWG1IP:** CWG1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **NCO1IP:** NCO1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **Unimplemented:** Read as '0'

bit 3 **CCP1IP:** CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **TMR2IP:** TMR2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR1GIP:** TMR1 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Interrupt Priority bit
1 = High priority
0 = Low priority

11.6 Operation During Sleep

When the device enters Sleep, the WWDT is cleared. If the WWDT is enabled during Sleep, the WWDT resumes counting. When the device exits Sleep, the WWDT is cleared again.

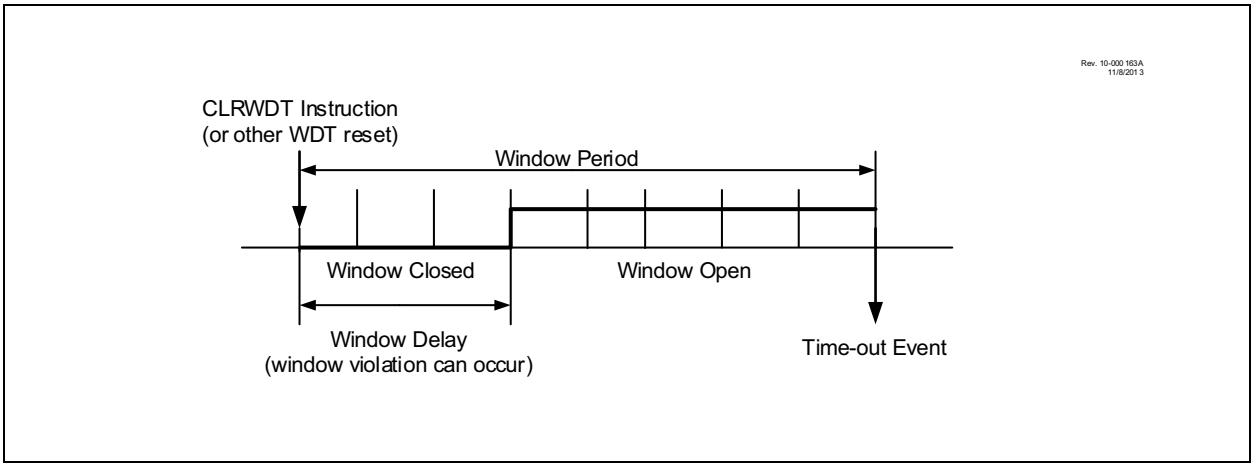
The WWDT remains clear until the Oscillator Start-up Timer (OST) completes, if enabled. See **Section 7.2.1.3 "Oscillator Start-up Timer (OST)"** for more information on the OST.

When a WWDT time-out occurs while the device is in Sleep, no Reset is generated. Instead, the device wakes up and resumes operation. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register are changed to indicate the event. The $\overline{RWDT}$ bit in the PCON0 register can also be used. See **Section 4.0 "Memory Organization"** for more information.

TABLE 11-2: WWDT CLEARING CONDITIONS

Conditions	WWDT
WDTE<1:0> = 00	Cleared
WDTE<1:0> = 01 and SEN = 0	
WDTE<1:0> = 10 and enter Sleep	
CLRWDT Command	
Oscillator Fail Detected	
Exit Sleep + System Clock = SOSC, EXTRC, INTOSC, EXTCLK	
Exit Sleep + System Clock = XT, HS, LP	Cleared until the end of OST
Change INTOSC divider (IRCF bits)	Unaffected

FIGURE 11-2: WINDOW PERIOD AND DELAY



Depending on the priority of the DMA with respect to CPU execution (Refer to **Section 3.2 "Memory Access Scheme"** for more information), the DMA Controller can move data through two methods:

- Stalling the CPU execution until it has completed its transfers (DMA has higher priority over the CPU in this mode of operation)
- Utilizing unused CPU cycles for DMA transfers (CPU has higher priority over the DMA in this mode of operation). Unused CPU cycles are referred to as bubbles which are instruction cycles available for use by the DMA to perform read and write operations. In this way, the effective bandwidth for handling data is increased; at the same time, DMA operations can proceed without causing a processor stall.

15.4 DMA Interface

The DMA module transfers data from the source to the destination one byte at a time, this smallest data movement is called a DMA data transaction. A DMA Message refers to one or more DMA data transactions.

Each DMA data transaction consists of two separate actions:

- Reading the Source Address Memory and storing the value in the DMA Buffer register
- Writing the contents of the DMA Buffer register to the Destination Address Memory

Note:	DMA data movement is a two-cycle operation.

The XIP bit (DMAxCON0 register) is a status bit to indicate whether or not the data in the DMAxBUF register has been written to the destination address. If the bit is set then data is waiting to be written to the destination. If clear, it means that either data has been written to the destination or that no source read has occurred.

The DMA has read access to PFM, Data EEPROM, and SFR/GPR space, and write access to SFR/GPR space. Based on these memory access capabilities, the DMA can support the following memory transactions:

TABLE 15-1: DMA MEMORY ACCESS

Read Source	Write Destination
Program Flash Memory	GPR
Program Flash Memory	SFR
Data EE	GPR
Data EE	SFR
GPR	GPR
SFR	GPR
GPR	SFR
SFR	SFR

Note:	Even though the DMA module has access to all memory and peripherals that are also available to the CPU, it is recommended that the DMA does not access any register that is part of the System arbitration. The DMA, as a system arbitration client should not be read or written by itself or by another DMA instantiation.

The following sections discuss the various control interfaces required for DMA data transfers.

15.4.1 DMA ADDRESSING

The start addresses for the source read and destination write operations are set using the DMAxSSA <21:0> and DMAxDSA <15:0> registers, respectively.

When the DMA Message transfers are in progress, the DMAxSPTR <21:0> and DMAxDPTR <15:0> registers contain the current address pointers for each source read and destination write operation, these registers are modified after each transaction based on the Address mode selection bits.

The SMODE and DMODE bits in the DMAxCON1 control register determine the address modes of operation by controlling how the DMAxSPTR <21:0> and DMAxDPTR <15:0> bits are updated after every DMA data transaction combination (Figure 15-2).

Each address can be separately configured to:

- Remain unchanged
- Increment by 1
- Decrement by 1

FIGURE 15-3: DMA COUNTERS BLOCK DIAGRAM

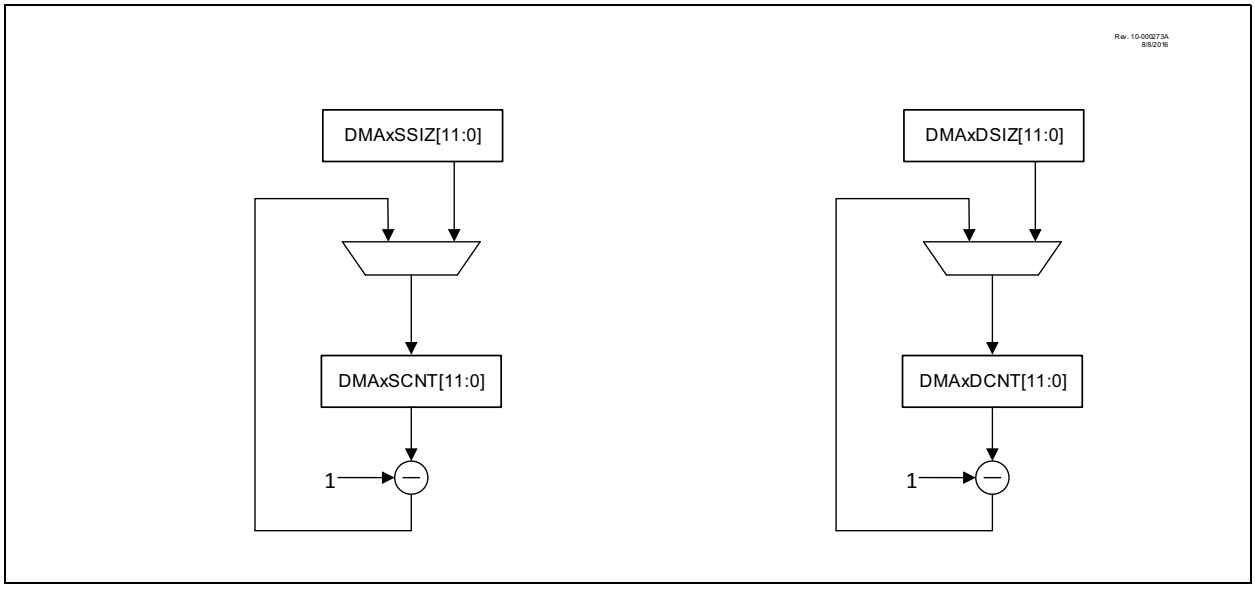
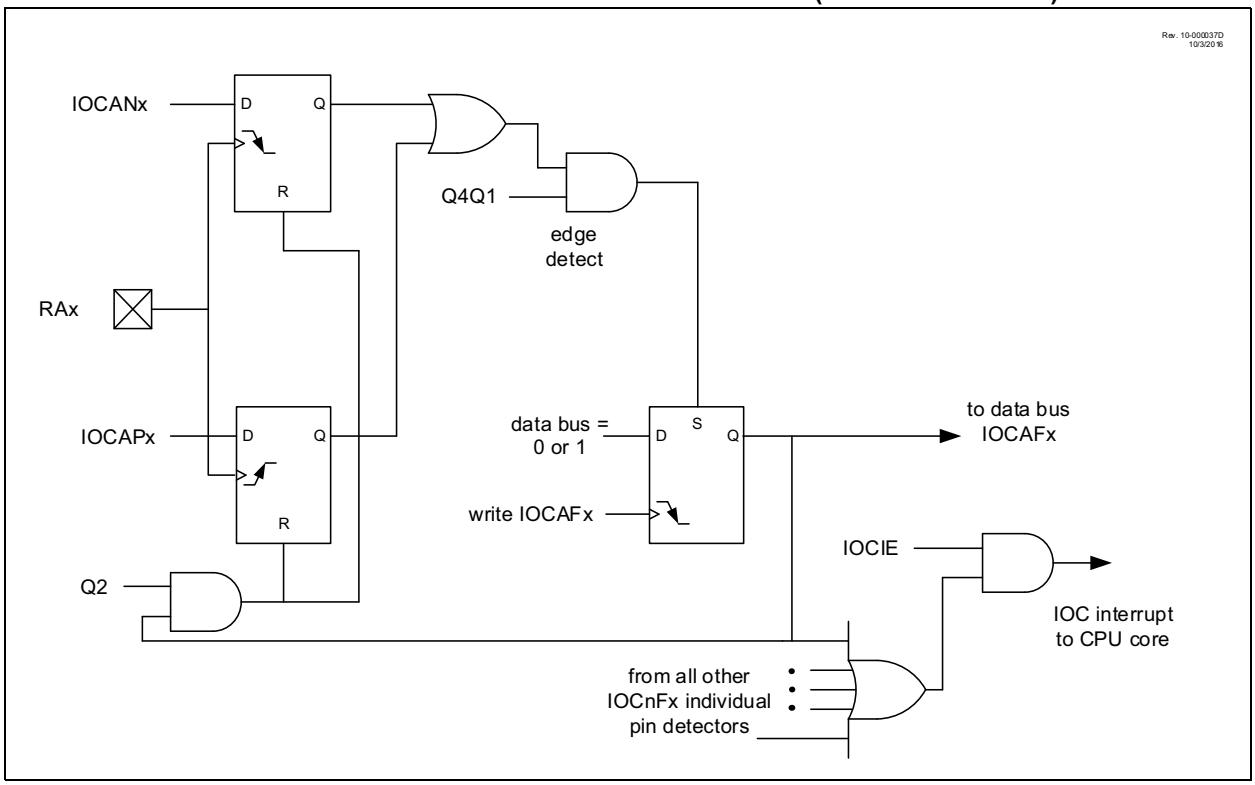


Table 15-2 has a few examples of configuring DMA Message sizes.

TABLE 15-2: EXAMPLE MESSAGE SIZE TABLE

Operation	Example	SCNT	DCNT	Comments
Read from single SFR location to RAM	U1RXB	1	N	N equals the number of bytes desired in the destination buffer. N >= 1.
Write to single SFR location from RAM	U1TXB	N	1	N equals the number of bytes desired in the source buffer. N >= 1.
Read from multiple SFR location	ADRES[H:L]	2	2*N	N equals the number of ADC results to be stored in memory. N>= 1
	TMR1[H:L]	2	2*N	N equals the number of TMR1 Acquisition results to be stored in memory. N>= 1
	SMT1CPR[U:H:L]	3	3*N	N equals the number of Capture Pulse Width measurements to be stored in memory. N>= 1
Write to Multiple SFR registers	PWMDC[H:L]	2*N	2	N equals the number of PWM duty cycle values to be loaded from a memory table. N>= 1
	All ADC registers	N*31	31	Using the DMA to transfer a complete ADC context from RAM to the ADC registers.N>= 1

FIGURE 18-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM (PORTA EXAMPLE)



18.6 Register Definitions: Interrupt-on-Change Control

REGISTER 18-1: IOCxP: INTERRUPT-ON-CHANGE POSITIVE EDGE REGISTER EXAMPLE

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
IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0
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 **IOCxP<7:0>:** Interrupt-on-Change Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the IOCx pin for a positive-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

REGISTER 18-2: IOCxN: INTERRUPT-ON-CHANGE NEGATIVE EDGE REGISTER EXAMPLE

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
IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0
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 **IOCxN<7:0>:** Interrupt-on-Change Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the IOCx pin for a negative-going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin

REGISTER 18-3: IOCxF: INTERRUPT-ON-CHANGE FLAG REGISTER EXAMPLE

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0
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 - Bit is set in hardware

bit 7-0 **IOCxF<7:0>:** Interrupt-on-Change Flag bits
1 = A enabled change was detected on the associated pin. Set when IOCP[n] = 1 and a positive edge was detected on the IOCn pin, or when IOCN[n] = 1 and a negative edge was detected on the IOCn pin
0 = No change was detected, or the user cleared the detected change

REGISTER 25-2: SMT1CON1: SMT CONTROL REGISTER 1

R/W/HC-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
GO	REPEAT	—	—	MODE<3:0>			
bit 7							bit 0

Legend:		
HC = Bit is cleared by hardware		HS = Bit is set by hardware
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	q = Value depends on condition

bit 7 **GO:** GO Data Acquisition bit
1 = Incrementing, acquiring data is enabled
0 = Incrementing, acquiring data is disabled

bit 6 **REPEAT:** SMT Repeat Acquisition Enable bit
1 = Repeat Data Acquisition mode is enabled
0 = Single Acquisition mode is enabled

bit 5-4 **Unimplemented**: Read as '0'

bit 3-0 **MODE<3:0>** SMT Operation Mode Select bits
1111 = Reserved
•
•
•
1011 = Reserved
1010 = Windowed counter
1001 = Gated counter
1000 = Counter
0111 = Capture
0110 = Time of flight
0101 = Gated windowed measure
0100 = Windowed measure
0011 = High and low time measurement
0010 = Period and Duty-Cycle Acquisition
0001 = Gated Timer
0000 = Timer

REGISTER 27-9: CLCxGLS2: GATE 2 LOGIC SELECT 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
G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N
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 **G3D4T:** Gate 2 Data 4 True (noninverted) bit
1 = CLCIN3 (true) is gated into CLCx Gate 2
0 = CLCIN3 (true) is not gated into CLCx Gate 2

bit 6 **G3D4N:** Gate 2 Data 4 Negated (inverted) bit
1 = CLCIN3 (inverted) is gated into CLCx Gate 2
0 = CLCIN3 (inverted) is not gated into CLCx Gate 2

bit 5 **G3D3T:** Gate 2 Data 3 True (noninverted) bit
1 = CLCIN2 (true) is gated into CLCx Gate 2
0 = CLCIN2 (true) is not gated into CLCx Gate 2

bit 4 **G3D3N:** Gate 2 Data 3 Negated (inverted) bit
1 = CLCIN2 (inverted) is gated into CLCx Gate 2
0 = CLCIN2 (inverted) is not gated into CLCx Gate 2

bit 3 **G3D2T:** Gate 2 Data 2 True (noninverted) bit
1 = CLCIN1 (true) is gated into CLCx Gate 2
0 = CLCIN1 (true) is not gated into CLCx Gate 2

bit 2 **G3D2N:** Gate 2 Data 2 Negated (inverted) bit
1 = CLCIN1 (inverted) is gated into CLCx Gate 2
0 = CLCIN1 (inverted) is not gated into CLCx Gate 2

bit 1 **G3D1T:** Gate 2 Data 1 True (noninverted) bit
1 = CLCIN0 (true) is gated into CLCx Gate 2
0 = CLCIN0 (true) is not gated into CLCx Gate 2

bit 0 **G3D1N:** Gate 2 Data 1 Negated (inverted) bit
1 = CLCIN0 (inverted) is gated into CLCx Gate 2
0 = CLCIN0 (inverted) is not gated into CLCx Gate 2

REGISTER 32-8: SPIxCON1: SPI CONFIGURATION REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-1/1	R/W-0/0	R/W-0/0
SMP	CKE	CKP	FST	—	SSP	SDIP	SDOP
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'

bit 7 **SMP**: SPI Input Sample Phase Control bit

Slave mode:

1 = Reserved

0 = SDI input is sampled in the middle of data output time

Master mode:

1 = SDI input is sampled at the end of data output time

0 = SDI input is sampled in the middle of data output time

bit 6 **CKE**: Clock Edge Select bit

1 = Output data changes on transition from active to idle clock state

0 = Output data changes on transition from idle to active clock state

bit 5 **CKP**: Clock Polarity Select bit

1 = Idle state for SCK is high level

0 = Idle state for SCK is low level

bit 4 **FST**: Fast Start Enable bit

Slave mode:

This bit is ignored

Master mode:

1 = Delay to first SCK may be less than ½ baud period

0 = Delay to first SCK will be at least ½ baud period

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

bit 2 **SSP**: SS Input/Output Polarity Control bit

1 = SS is active-low

0 = SS is active-high

bit 1 **SDIP**: SDI Input Polarity Control bit

1 = SDI input is active-low

0 = SDI input is active-high

bit 0 **SDOP**: SDI Output Polarity Control bit

1 = SDO output is active-low

0 = SDO output is active-high

33.3.2 BYTE FORMAT

All communication in I^2C is done in 9-bit segments. A byte is sent from a master to a slave or vice-versa, followed by an Acknowledge bit sent by the receiver. After the 8th falling edge of the SCL line, the device transmitting data on the SDA line releases control of that pin to an input, and reads in an acknowledge value on the next clock pulse. The clock signal is provided by the master. Data is valid to change while the SCL line is low, and sampled on the rising edge of the clock. Changes on the SDA line while the SCL line is high define Start and Stop conditions on the bus which are explained further in the chapter.

33.3.3 SDA AND SCL PINS

The user must configure these pins as open-drain inputs. This is done by clearing the appropriate TRIS bits and setting the appropriate and ODCON bits. The user may also select the input threshold, slew-rate and internal pull-up settings using the RxyI2C control registers (Register 16-9).

33.3.4 SDA HOLD TIME

The hold time of the SDA pin is selected by the SDAHT<1:0> bits of the I2CxCON2 register. Hold time is the time SDA is held valid after the falling edge of SCL. A longer hold time setting may help on buses with large capacitance.

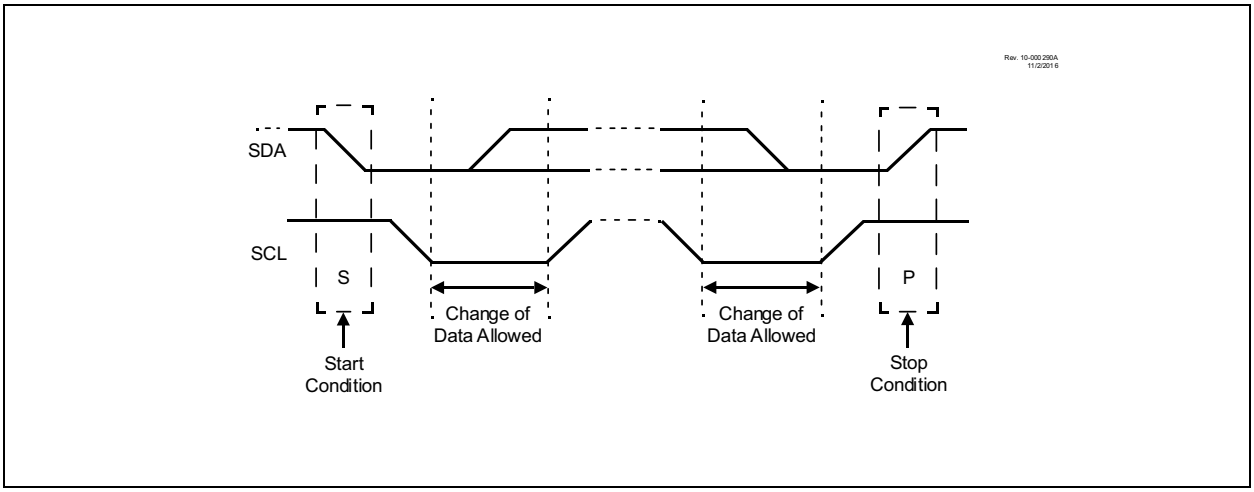
33.3.5 START CONDITION

The I^2C specification defines a Start condition as a transition of SDA line from a high to a low state while SCL line is high. A Start condition is always generated by the master and signifies the transition of the bus from an Idle to an Active state. Figure 33-3 shows waveforms for Start conditions. Master hardware waits for the BFRE bit of I2CxSTAT0 to be set, before asserting a Start condition on the SCL and SDA lines. If two masters assert a start at the same time, a collision will occur during the addressing phase.

33.3.6 STOP CONDITION

A Stop condition is a transition of the SDA line from low to high while the SCL line is high. Figure 33-3 shows waveforms for Stop conditions.

FIGURE 33-3: START AND STOP CONDITIONS



Note:	At least one SCL low time must appear before a Stop is valid. Therefore if the SDA line goes low then high again while the SCL line is high, only the Start condition is detected.

REGISTER 36-27: ADSTPTH: ADC THRESHOLD SETPOINT REGISTER HIGH

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
STPT<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 **STPT<15:8>**: ADC Threshold Setpoint MSB. Upper byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ERR, see Register 36-29 for more details.

REGISTER 36-28: ADSTPTL: ADC THRESHOLD SETPOINT REGISTER LOW

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
STPT<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 **STPT<7:0>**: ADC Threshold Setpoint LSB. Lower byte of ADC threshold setpoint, depending on ADCALC, may be used to determine ERR, see Register 36-30 for more details.

SUBFWB **Subtract f from W with borrow**

Syntax: SUBFWB f {,d {,a}}

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

Operation: (W) – (f) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	01da	ffff	ffff

Description: Subtract register 'f' and CARRY flag (borrow) from W (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored in register 'f' (default).
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 $f \le 95$ (5Fh). See **Section 41.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1:
```
SUBFWB   REG, 1, 0
```
Before Instruction
REG = 3
W = 2
C = 1
After Instruction
REG = FF
W = 2
C = 0
Z = 0
N = 1 ; result is negative

Example 2:
```
SUBFWB   REG, 0, 0
```
Before Instruction
REG = 2
W = 5
C = 1
After Instruction
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; result is positive

Example 3:
```
SUBFWB   REG, 1, 0
```
Before Instruction
REG = 1
W = 2
C = 0
After Instruction
REG = 0
W = 2
C = 1
Z = 1 ; result is zero
N = 0

SUBLW **Subtract W from literal**

Syntax: SUBLW k

Operands: $0 \le k \le 255$

Operation: k – (W) → W

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1000	kkkk	kkkk

Description: W is subtracted from 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 1:
```
SUBLW   02h
```
Before Instruction
W = 01h
C = ?
After Instruction
W = 01h
C = 1 ; result is positive
Z = 0
N = 0

Example 2:
```
SUBLW   02h
```
Before Instruction
W = 02h
C = ?
After Instruction
W = 00h
C = 1 ; result is zero
Z = 1
N = 0

Example 3:
```
SUBLW   02h
```
Before Instruction
W = 03h
C = ?
After Instruction
W = FFh ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

TABLE 42-1: REGISTER FILE SUMMARY FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3FD1h - 3FD0h	—	Unimplemented								
3FCFh	PORTF(3)	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	263
3FCEh	PORTE	—	—	—	—	RE0	RE2(2)	RE1(2)	RE1(2)	263
3FCDh	PORTD(2)	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	263
3FCCh	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	263
3FCBh	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	263
3FCAh	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	263
3FC9h - 3FC8h	—	Unimplemented								
3FB7h	TRISF(3)	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	264
3FB6h	TRISE(2)	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	264
3FB5h	TRISD(2)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	264
3FC4h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	264
3FC3h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	264
3FC2h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	264
3FC1h - 3FC0h	—	Unimplemented								
3FBFh	LATF(3)	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	265
3FBEh	LATE(2)	LATE7	LATE7	LATE7	LATE7	LATE7	LATE7	LATE7	LATE7	265
3FBDh	LATD(2)	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	265
3FBCh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	265
3FBBh	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	265
3FBAh	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	265
3FB9h	T0CON1	CS<2:0>			ASYNC	CKPS<3:0>				302
3FB8h	T0CON0	EN	—	OUT	MD16	OUTPS				301
3FB7h	TMR0H	TMR0H								303
3FB6h	TMR0L	TMR0L								303
3FB5h	T1CLK	CS								315
3FB4h	T1GATE	GSS								316
3FB3h	T1GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	314
3FB2h	T1CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	338
3FB1h	TMR1H	TMR1H								317
3FB0h	TMR1L	TMR1L								317
3FAFh	T2RST	—	—	—	RSEL					336
3FAEh	T2CLK	—	—	—	—	CS				315
3FADh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE					339
3FACh	T2CON	ON	CKPS		OUTPS					313
3FABh	T2PR	PR2								337
3FAAh	T2TMR	TMR2								337
3FA9h	T3CLK	CS								315
3FA8h	T3GATE	GSS								316
3FA7h	T3GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	314
3FA6h	T3CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	338
3FA5h	TMR3H	TMR3H								317
3FA4h	TMR3L	TMR3L								317
3FA3h	T4RST	—	—	—	RSEL					336
3FA2h	T4CLK	—	—	—	—	CS				335

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition

Note 1: Unimplemented in LF devices.

2: Unimplemented in PIC18(L)F26/27K42.

3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.

4: Unimplemented in PIC18(L)F45/55K42.

43.0 DEVELOPMENT SUPPORT

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

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

43.1 MPLAB X Integrated Development Environment Software

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

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

Feature-Rich Editor:

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

User-Friendly, Customizable Interface:

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

Project-Based Workspaces:

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

File History and Bug Tracking:

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

TABLE 44-9: EXTERNAL CLOCK/OSCILLATOR TIMING REQUIREMENTS (CONTINUED)

Standard Operating Conditions (unless otherwise stated)							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
OS21	F_{CY}	Instruction Frequency	—	FOSC/4	—	MHz	
OS22	T_{CY}	Instruction Period	62.5	$1/F_{CY}$	—	ns	

* 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.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min" values with an external clock applied to OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

2: The system clock frequency (FOSC) is selected by the "main clock switch controls" as described in **Section 10.0 "Power-Saving Operation Modes"**.

3: The system clock frequency (FOSC) must meet the voltage requirements defined in the **Section 44.2 "Standard Operating Conditions"**.

4: LP, XT and HS oscillator modes require an appropriate crystal or resonator to be connected to the device. For clocking the device with the external square wave, one of the EC mode selections must be used.

Note: Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

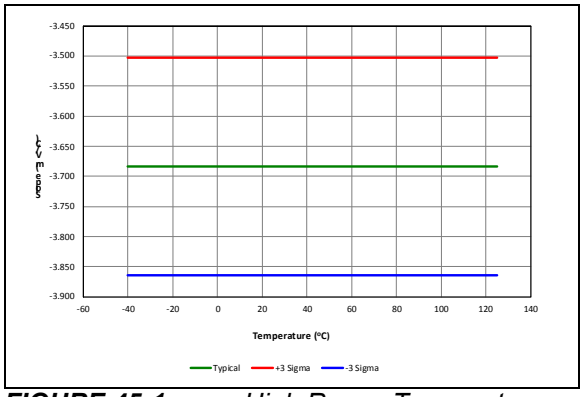


FIGURE 45-1: *High Range Temperature Indicator Voltage Sensitivity Across Temperature*

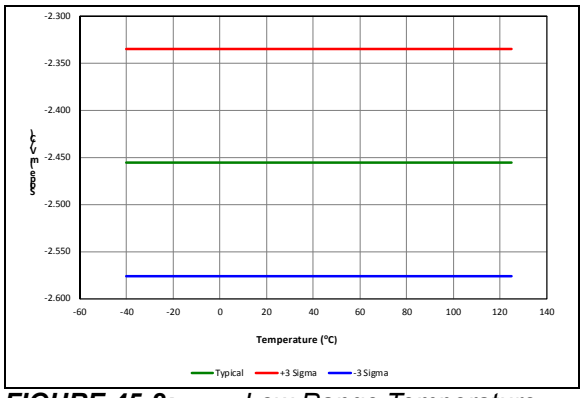


FIGURE 45-2: *Low Range Temperature Indicator Voltage Sensitivity Across Temperature*

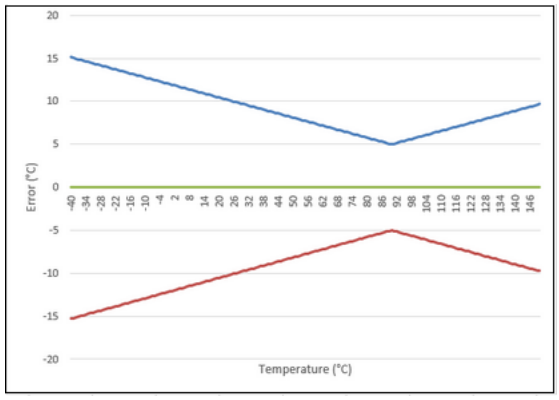
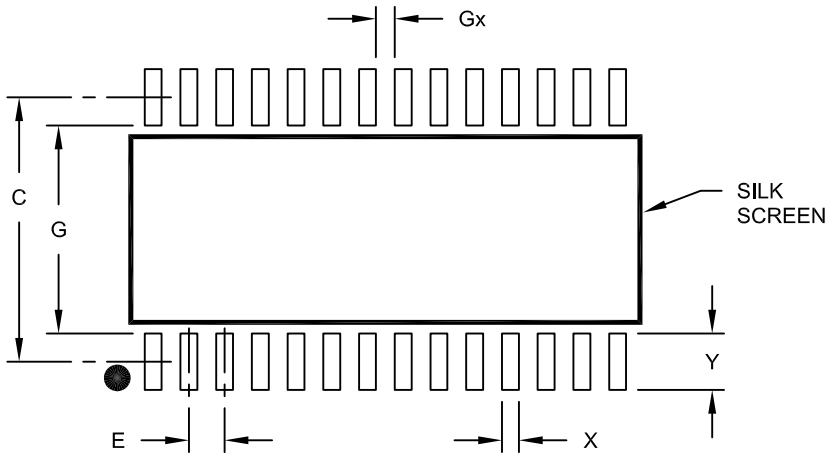


FIGURE 45-3: *Temperature Indicator Performance Over Temperature*

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		1.27 BSC	
Contact Pad Spacing	C		9.40	
Contact Pad Width (X28)	X			0.60
Contact Pad Length (X28)	Y			2.00
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	7.40		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

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

Microchip Technology Drawing No. C04-2052A