



Welcome to [E-XFL.COM](https://www.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	32MHz
Connectivity	I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	17
Program Memory Size	7KB (4K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	20-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16f1828-i-ss

PIC16(L)F1824/8

TABLE 1-3: PIC16(L)F1828 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RC3/AN7/CPS7/C12IN3-/P2A ^(1,2) /CCP2 ^(1,2) /P1C ^(1,2) /MDMIN	RC3	TTL	CMOS	General purpose I/O.
	AN7	AN	—	A/D Channel 7 input.
	CPS7	AN	—	Capacitive sensing input 7.
	C12IN3-	AN	—	Comparator C1 or C2 negative input.
	P2A	—	CMOS	PWM output.
	CCP2	AN	—	Capture/Compare/PWM2.
	P1C	—	CMOS	PWM output.
RC4/C2OUT/SRNQ/P1B/TX ⁽¹⁾ /CK ⁽¹⁾ /MDOUT	RC4	TTL	CMOS	General purpose I/O.
	C2OUT	—	CMOS	Comparator C2 output.
	SRNQ	—	CMOS	SR latch inverting output.
	P1B	—	CMOS	PWM output.
	TX	—	CMOS	USART asynchronous transmit.
	CK	ST	CMOS	USART synchronous clock.
	MDOUT	—	CMOS	Modulator output.
RC5/P1A/CCP1/RX ⁽¹⁾ /DT ⁽¹⁾ /MDCIN2	RC5	TTL	CMOS	General purpose I/O.
	P1A	—	CMOS	PWM output.
	CCP1	ST	CMOS	Capture/Compare/PWM1.
	RX	ST	—	USART asynchronous input.
	DT	ST	CMOS	USART synchronous data.
RC6/AN8/CPS8/CCP4/ \overline{SS}	RC6	TTL	CMOS	General purpose I/O.
	AN8	AN	—	A/D Channel 8 input.
	CPS8	AN	—	Capacitive sensing input 8.
	CCP4	AN	—	Capture/Compare/PWM4.
	\overline{SS}	ST	—	Slave Select input.
RC7/AN9/CPS9/SDO	RC7	TTL	CMOS	General purpose I/O.
	AN9	AN	—	A/D Channel 9 input.
	CPS9	AN	—	Capacitive sensing input 9.
	SDO	—	CMOS	SPI data output.
VDD	VDD	Power	—	Positive supply.
VSS	VSS	Power	—	Ground reference.

Legend: AN = Analog input or output CMOS = CMOS compatible input or output OD = Open Drain
TTL = TTL compatible input ST = Schmitt Trigger input with CMOS levels I²C™ = Schmitt Trigger input with I²C levels
HV = High Voltage XTAL = Crystal

Note 1: Pin functions can be moved using the APFCON0 and APFCON1 registers (Register 12-1 and Register 12-2).

2: Default function location.

3.2.2 SPECIAL FUNCTION REGISTER

The Special Function Registers are registers used by the application to control the desired operation of peripheral functions in the device. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh). The registers associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

3.2.3 GENERAL PURPOSE RAM

There are up to 80 bytes of GPR in each data memory bank. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh).

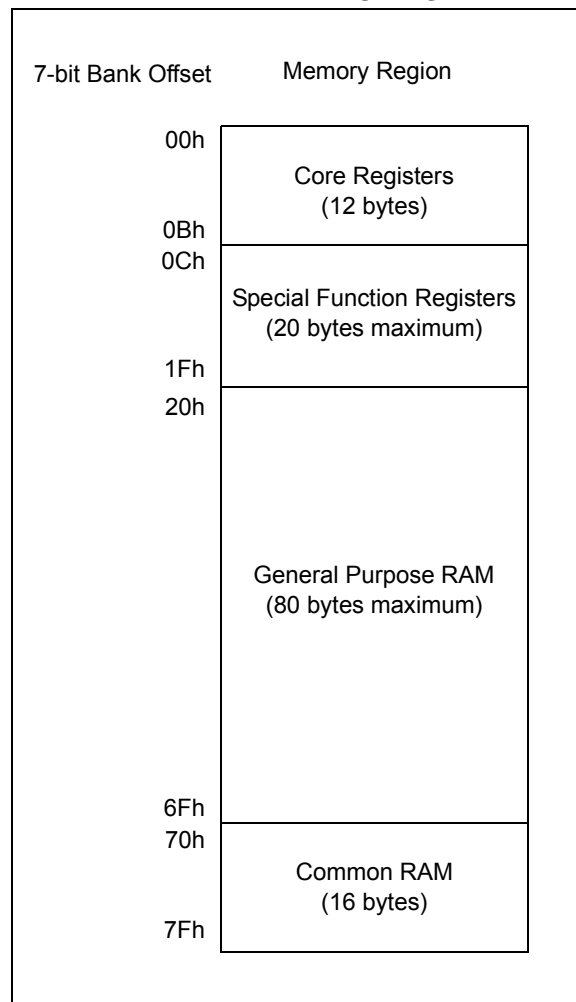
3.2.3.1 Linear Access to GPR

The general purpose RAM can be accessed in a non-banked method via the FSRs. This can simplify access to large memory structures. See **Section 3.5.2 “Linear Data Memory”** for more information.

3.2.4 COMMON RAM

There are 16 bytes of common RAM accessible from all banks.

FIGURE 3-2: BANKED MEMORY PARTITIONING



3.2.5 DEVICE MEMORY MAPS

The memory maps for the device family are as shown in Table 3-3.

TABLE 3-3: MEMORY MAP TABLES

Device	Banks	Table No.
PIC16(L)F1824	0-7	Table 3-4
PIC16(L)F1828	8-15	Table 3-5
	16-23	Table 3-6
	24-31	Table 3-7
	31	Table 3-8

PIC16(L)F1824/8

FIGURE 3-5: ACCESSING THE STACK EXAMPLE 2

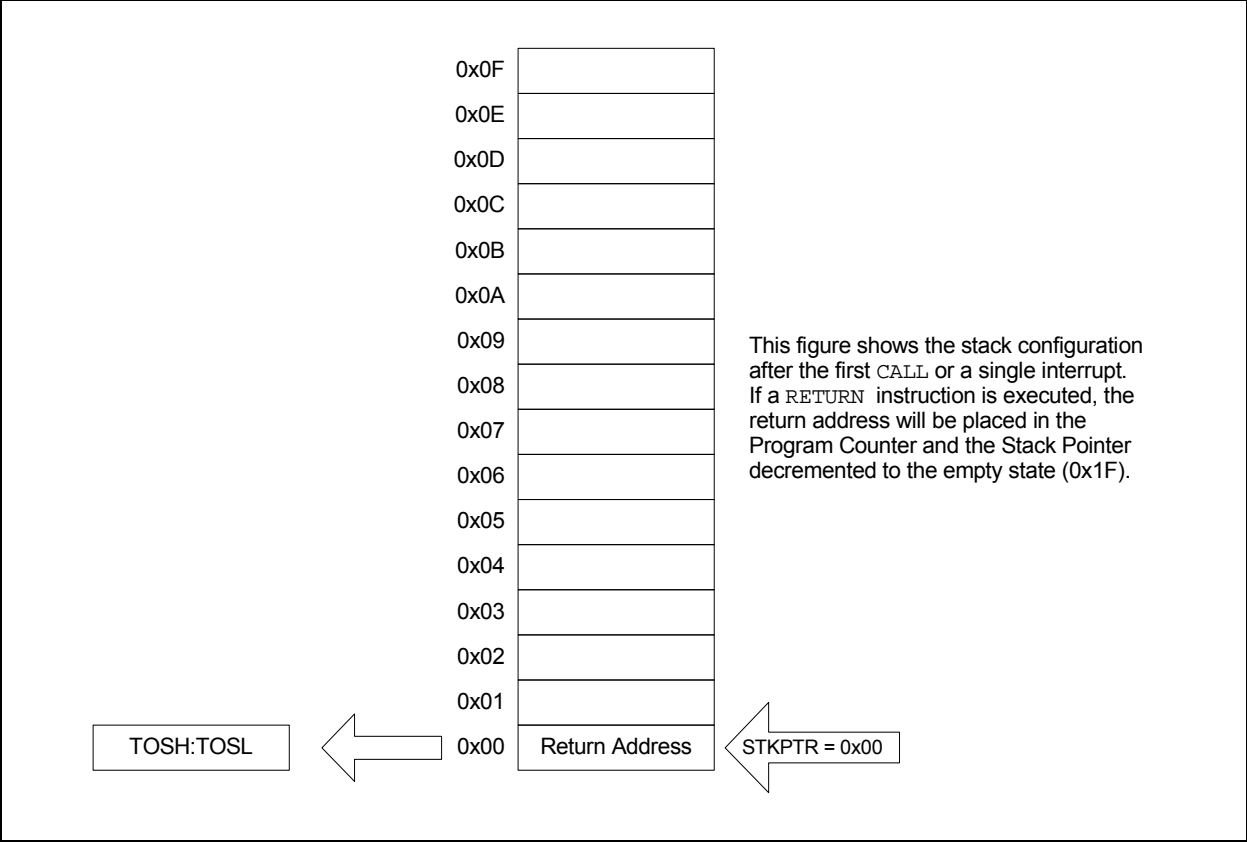
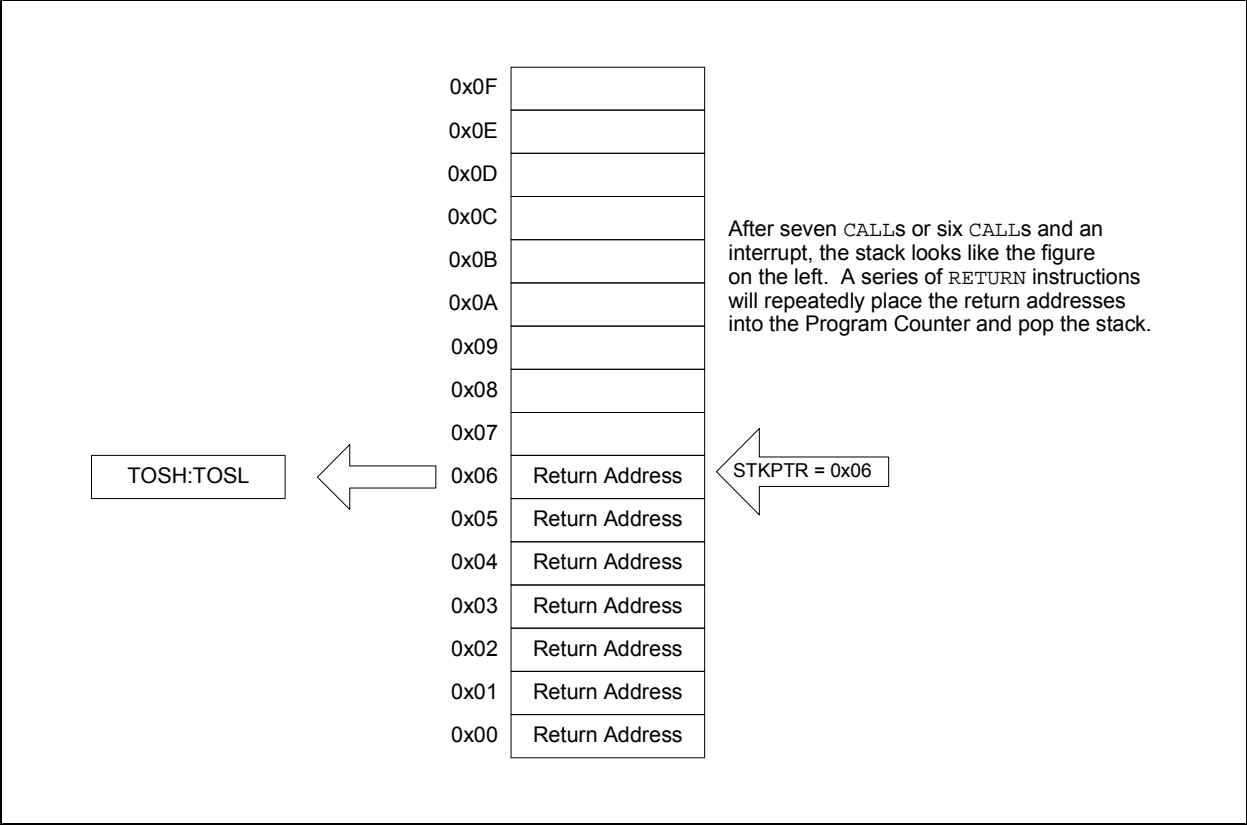


FIGURE 3-6: ACCESSING THE STACK EXAMPLE 3



PIC16(L)F1824/8

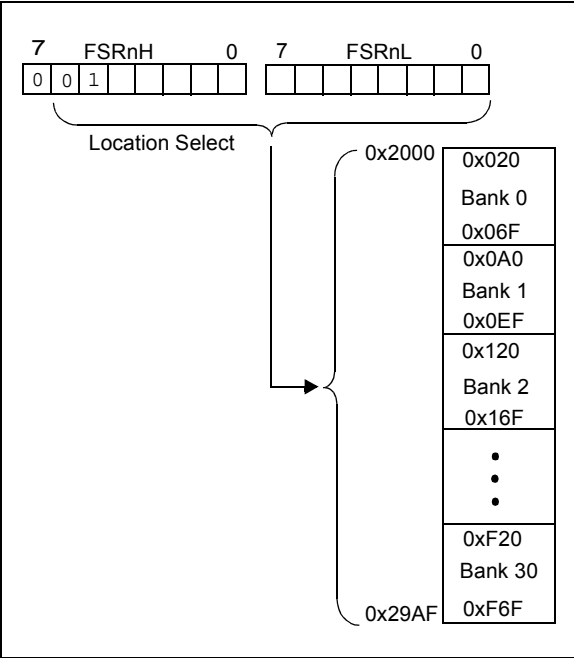
3.5.2 LINEAR DATA MEMORY

The linear data memory is the region from FSR address 0x2000 to FSR address 0x29AF. This region is a virtual region that points back to the 80-byte blocks of GPR memory in all the banks.

Unimplemented memory reads as 0x00. Use of the linear data memory region allows buffers to be larger than 80 bytes because incrementing the FSR beyond one bank will go directly to the GPR memory of the next bank.

The 16 bytes of common memory are not included in the linear data memory region.

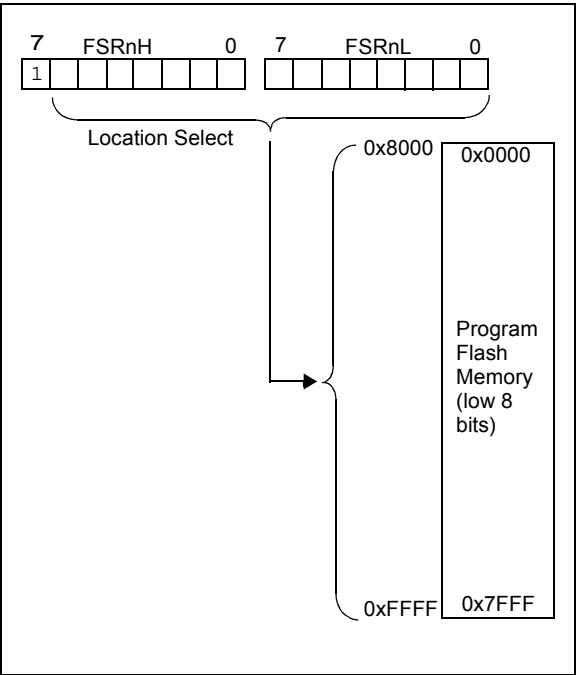
FIGURE 3-10: LINEAR DATA MEMORY MAP



3.5.3 PROGRAM FLASH MEMORY

To make constant data access easier, the entire program Flash memory is mapped to the upper half of the FSR address space. When the MSB of FSRnH is set, the lower 15 bits are the address in program memory which will be accessed through INDF. Only the lower eight bits of each memory location is accessible via INDF. Writing to the program Flash memory cannot be accomplished via the FSR/INDF interface. All instructions that access program Flash memory via the FSR/INDF interface will require one additional instruction cycle to complete.

FIGURE 3-11: PROGRAM FLASH MEMORY MAP



REGISTER 10-1: WDTCON: WATCHDOG TIMER CONTROL REGISTER

U-0	U-0	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-1/1	R/W-0/0
—	—	WDTPS4	WDTPS3	WDTPS2	WDTPS1	WDTPS0	SWDTEN
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

-m/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 5-1

WDTPS<4:0>: Watchdog Timer Period Select bits

Bit Value = Prescale Rate

00000 = 1:32 (Interval 1 ms typ)

00001 = 1:64 (Interval 2 ms typ)

00010 = 1:128 (Interval 4 ms typ)

00011 = 1:256 (Interval 8 ms typ)

00100 = 1:512 (Interval 16 ms typ)

00101 = 1:1024 (Interval 32 ms typ)

00110 = 1:2048 (Interval 64 ms typ)

00111 = 1:4096 (Interval 128 ms typ)

01000 = 1:8192 (Interval 256 ms typ)

01001 = 1:16384 (Interval 512 ms typ)

01010 = 1:32768 (Interval 1s typ)

01011 = 1:65536 (Interval 2s typ) (Reset value)

01100 = 1:131072 (2^{17}) (Interval 4s typ)

01101 = 1:262144 (2^{18}) (Interval 8s typ)

01110 = 1:524288 (2^{19}) (Interval 16s typ)

01111 = 1:1048576 (2^{20}) (Interval 32s typ)

10000 = 1:2097152 (2^{21}) (Interval 64s typ)

10001 = 1:4194304 (2^{22}) (Interval 128s typ)

10010 = 1:8388608 (2^{23}) (Interval 256s typ)

10011 = Reserved. Results in minimum interval (1:32)

•
•
•

11111 = Reserved. Results in minimum interval (1:32)

bit 0

SWDTEN: Software Enable/Disable for Watchdog Timer bit

If WDTE<1:0> = 00:

This bit is ignored.

If WDTE<1:0> = 01:

1 = WDT is turned on

0 = WDT is turned off

If WDTE<1:0> = 1x:

This bit is ignored.

11.3.2 ERASING FLASH PROGRAM MEMORY

While executing code, program memory can only be erased by rows. To erase a row:

1. Load the EEADRH:EEADRL register pair with the address of the new row to be erased.
2. Clear the CFGS bit of the EECON1 register.
3. Set the EEPGD, FREE and WREN bits of the EECON1 register.
4. Write 55h, then AAh, to EECON2 (Flash programming unlock sequence).
5. Set control bit WR of the EECON1 register to begin the erase operation.
6. Poll the FREE bit in the EECON1 register to determine when the row erase has completed.

See Example 11-4.

After the “BSF EECON1, WR” instruction, the processor requires two cycles to set up the erase operation. The user must place two NOP instructions after the WR bit is set. The processor will halt internal operations for the typical 2ms erase time. This is not Sleep mode, as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the EECON1 write instruction.

11.3.3 WRITING TO FLASH PROGRAM MEMORY

Program memory is programmed using the following steps:

1. Load the starting address of the word(s) to be programmed.
2. Load the write latches with data.
3. Initiate a programming operation.
4. Repeat steps 1 through 3 until all data is written.

Before writing to program memory, the word(s) to be written must be erased or previously unwritten. Program memory can only be erased one row at a time. No automatic erase occurs upon the initiation of the write.

Program memory can be written one or more words at a time. The maximum number of words written at one time is equal to the number of write latches. See Figure 11-2 (block writes to program memory with 32 write latches) for more details. The write latches are aligned to the address boundary defined by EEADRL as shown in Table 11-1. Write operations do not cross these boundaries. At the completion of a program memory write operation, the write latches are reset to contain 0x3FFF.

The following steps should be completed to load the write latches and program a block of program memory. These steps are divided into two parts. First, all write latches are loaded with data except for the last program memory location. Then, the last write latch is loaded and the programming sequence is initiated. A special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. This unlock sequence should not be interrupted.

1. Set the EEPGD and WREN bits of the EECON1 register.
2. Clear the CFGS bit of the EECON1 register.
3. Set the LWLO bit of the EECON1 register. When the LWLO bit of the EECON1 register is ‘1’, the write sequence will only load the write latches and will not initiate the write to Flash program memory.
4. Load the EEADRH:EEADRL register pair with the address of the location to be written.
5. Load the EEDATH:EEDATL register pair with the program memory data to be written.
6. Write 55h, then AAh, to EECON2, then set the WR bit of the EECON1 register (Flash programming unlock sequence). The write latch is now loaded.
7. Increment the EEADRH:EEADRL register pair to point to the next location.
8. Repeat steps 5 through 7 until all but the last write latch has been loaded.
9. Clear the LWLO bit of the EECON1 register. When the LWLO bit of the EECON1 register is ‘0’, the write sequence will initiate the write to Flash program memory.
10. Load the EEDATH:EEDATL register pair with the program memory data to be written.
11. Write 55h, then AAh, to EECON2, then set the WR bit of the EECON1 register (Flash programming unlock sequence). The entire latch block is now written to Flash program memory.

It is not necessary to load the entire write latch block with user program data. However, the entire write latch block will be written to program memory.

An example of the complete write sequence for eight words is shown in Example 11-5. The initial address is loaded into the EEADRH:EEADRL register pair; the eight words of data are loaded using indirect addressing.

Note: If the number of write latches is smaller than the erase block size, the code sequence provided in Example 11-5 may be repeated multiple times to fully program an erased program memory row.

PIC16(L)F1824/8

12.0 I/O PORTS

Depending on the device selected and peripherals enabled, there are up to two ports available. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

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

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

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

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

Ports with analog functions also have an ANSELx register which can disable the digital input and save power. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 12-1.

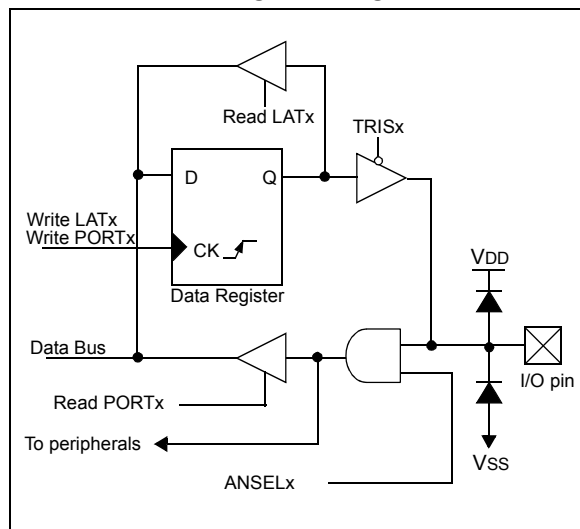
12.1 Alternate Pin Function

The Alternate Pin Function Control 0 (APFCON0) and Alternate Pin Function Control 1 (APFCON1) registers are used to steer specific peripheral input and output functions between different pins. The APFCON0 and APFCON1 registers are shown in Register 12-1 and Register 12-2. For this device family, the following functions can be moved between different pins.

- RX/DT/TX/CK
- SDO
- \overline{SS} (Slave Select)
- T1G
- P1B/P1C/P1D/P2B
- CCP1/P1A/CCP2

These bits have no effect on the values of any TRIS register. PORT and TRIS overrides will be routed to the correct pin. The unselected pin will be unaffected.

FIGURE 12-1: GENERIC I/O PORT OPERATION



REGISTER 12-1: APFCON0: ALTERNATE PIN FUNCTION CONTROL REGISTER 0

R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0	U-0
RXDTSEL	SDOSEL ⁽¹⁾	SSSEL ⁽¹⁾	—	T1GSEL	TXCKSEL	—	—
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 **RXDTSEL:** Pin Selection
(PIC16(L)F1824)
0 = RX/DT function is on RC5
1 = RX/DT function is on RA1
(PIC16(L)F1828)
0 = RX/DT function is on RB5
1 = RX/DT function is on RC5
- bit 6 **SDOSEL:** Pin Selection
(PIC16(L)F1824)
0 = SDO function is on RC2
1 = SDO function is on RA4
(PIC16(L)F1828)
Bit is Read-Only, '0'
SDO function is always on RC7
- bit 5 **SSSEL:** Pin Selection
(PIC16(L)F1824)
0 = SS function is on RC3
1 = SS function is on RA3
(PIC16(L)F1828)
Bit is Read-Only, '0'
SS function is always on RC6
- bit 4 **Unimplemented:** Read as '0'
- bit 3 **T1GSEL:** Pin Selection
0 = T1G function is on RA4
1 = T1G function is on RA3
- bit 2 **TXCKSEL:** Pin Selection
(PIC16(L)F1824)
0 = TX/CK function is on RC4
1 = TX/CK function is on RA0
(PIC16(L)F1828)
0 = TX/CK function is on RB7
1 = TX/CK function is on RC4
- bit 1-0 **Unimplemented:** Read as '0'

Note 1: PIC16(L)F1824 only.

17.4 Low-Power Voltage State

In order for the DAC module to consume the least amount of power, one of the two voltage reference input sources to the resistor ladder must be disconnected. Either the positive voltage source, (VSRC+), or the negative voltage source, (VSRC-) can be disabled.

The negative voltage source is disabled by setting the DACLPS bit in the DACCON0 register. Clearing the DACLPS bit in the DACCON0 register disables the positive voltage source.

17.4.1 OUTPUT CLAMPED TO POSITIVE VOLTAGE SOURCE

The DAC output voltage can be set to VSRC+ with the least amount of power consumption by performing the following:

- Clearing the DACEN bit in the DACCON0 register.
- Setting the DACLPS bit in the DACCON0 register.
- Configuring the DACPSS bits to the proper positive source.
- Configuring the DACR<4:0> bits to '11111' in the DACCON1 register.

This is also the method used to output the voltage level from the FVR to an output pin. See Figure 17-2 for more information.

Reference Figure 17-3 for output clamping examples.

17.4.2 OUTPUT CLAMPED TO NEGATIVE VOLTAGE SOURCE

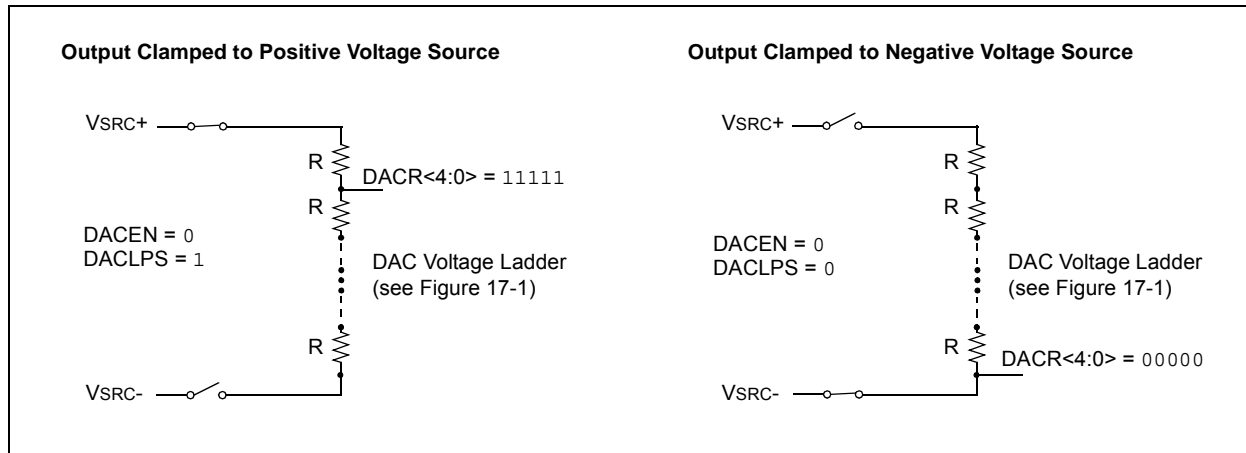
The DAC output voltage can be set to VSRC- with the least amount of power consumption by performing the following:

- Clearing the DACEN bit in the DACCON0 register.
- Clearing the DACLPS bit in the DACCON0 register.
- Configuring the DACNSS bits to the proper negative source.
- Configuring the DACR<4:0> bits to '00000' in the DACCON1 register.

This allows the comparator to detect a zero-crossing while not consuming additional current through the DAC module.

Reference Figure 17-3 for output clamping examples.

FIGURE 17-3: OUTPUT VOLTAGE CLAMPING EXAMPLES



17.5 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the DACCON0 register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

17.6 Effects of a Reset

A device Reset affects the following:

- DAC is disabled.
- DAC output voltage is removed from the DACOUT pin.
- The DACR<4:0> range select bits are cleared.

18.0 SR LATCH

The module consists of a single SR latch with multiple Set and Reset inputs as well as separate latch outputs. The SR latch module includes the following features:

- Programmable input selection
- SR latch output is available externally
- Separate Q and \bar{Q} outputs
- Firmware Set and Reset

The SR latch can be used in a variety of analog applications, including oscillator circuits, one-shot circuit, hysteretic controllers, and analog timing applications.

18.1 Latch Operation

The latch is a Set-Reset latch that does not depend on a clock source. Each of the Set and Reset inputs are active-high. The latch can be Set or Reset by:

- Software control (SRPS and SRPR bits)
- Comparator C1 output (sync_C1OUT)
- Comparator C2 output (sync_C2OUT)
- SRI pin
- Programmable clock (SRCLK)

The SRPS and the SRPR bits of the SRCON0 register may be used to Set or Reset the SR latch, respectively. The latch is Reset-dominant. Therefore, if both Set and Reset inputs are high, the latch will go to the Reset state. Both the SRPS and SRPR bits are self resetting which means that a single write to either of the bits is all that is necessary to complete a latch Set or Reset operation.

The output from Comparator C1 or C2 can be used as the Set or Reset inputs of the SR latch. The output of either Comparator can be synchronized to the Timer1 clock source. See **Section 19.0 “Comparator Module”** and **Section 21.0 “Timer1 Module with Gate Control”** for more information.

An external source on the SRI pin can be used as the Set or Reset inputs of the SR latch.

An internal clock source is available that can periodically set or reset the SR latch. The SRCLK<2:0> bits in the SRCON0 register are used to select the clock source period. The SRSCKE and SRRCKE bits of the SRCON1 register enable the clock source to Set or Reset the SR latch, respectively.

18.2 Latch Output

The SRQEN and SRNQEN bits of the SRCON0 register control the Q and \bar{Q} latch outputs. Both of the SR latch outputs may be directly output to an I/O pin at the same time.

The applicable TRIS bit of the corresponding port must be cleared to enable the port pin output driver.

18.3 Effects of a Reset

Upon any device Reset, the SR latch output is not initialized to a known state. The user's firmware is responsible for initializing the latch output before enabling the output pins.

PIC16(L)F1824/8

24.4.6 PWM STEERING MODE

In Single Output mode, PWM steering allows any of the PWM pins to be the modulated signal. Additionally, the same PWM signal can be simultaneously available on multiple pins.

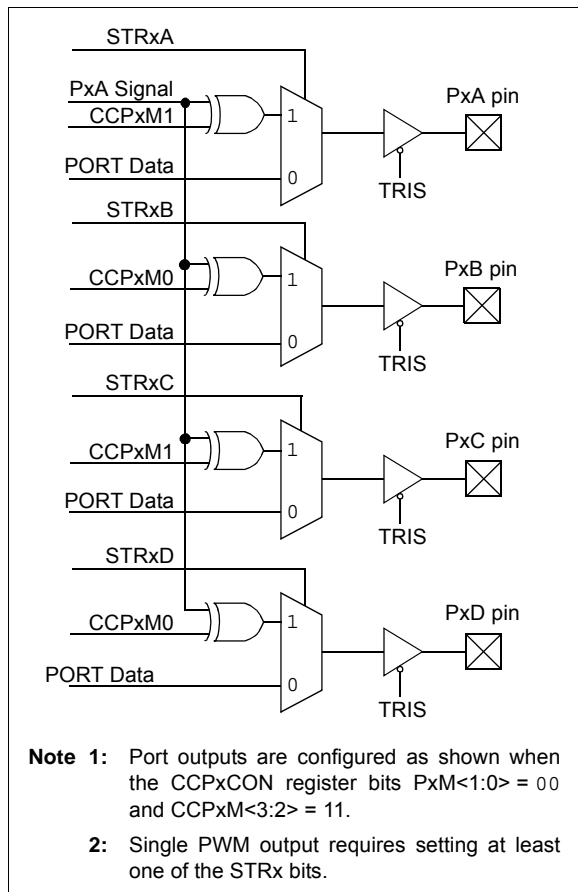
Once the Single Output mode is selected ($CCPxM<3:2> = 11$ and $PxM<1:0> = 00$ of the $CCPxCON$ register), the user firmware can bring out the same PWM signal to one, two, three or four output pins by setting the appropriate $STRx<D:A>$ bits of the $PSTRxCON$ register, as shown in Table 24-9.

Note: The associated TRIS bits must be set to output ('0') to enable the pin output driver in order to see the PWM signal on the pin.

While the PWM Steering mode is active, $CCPxM<1:0>$ bits of the $CCPxCON$ register select the PWM output polarity for the $Px<D:A>$ pins.

The PWM auto-shutdown operation also applies to PWM Steering mode as described in **Section 24.4.3 “Enhanced PWM Auto-shutdown mode”**. An auto-shutdown event will only affect pins that have PWM outputs enabled.

FIGURE 24-18: SIMPLIFIED STEERING BLOCK DIAGRAM



PIC16(L)F1824/8

REGISTER 24-2: CCPTMRS0: PWM TIMER SELECTION CONTROL 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
C4TSEL<1:0>		C3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<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 **C4TSEL<1:0>**: CCP4 Timer Selection bits
00 = CCP4 is based off Timer2 in PWM mode
01 = CCP4 is based off Timer4 in PWM mode
10 = CCP4 is based off Timer6 in PWM mode
11 = Reserved
- bit 5-4 **C3TSEL<1:0>**: CCP3 Timer Selection bits
00 = CCP3 is based off Timer2 in PWM mode
01 = CCP3 is based off Timer4 in PWM mode
10 = CCP3 is based off Timer6 in PWM mode
11 = Reserved
- bit 3-2 **C2TSEL<1:0>**: CCP2 Timer Selection bits
00 = CCP2 is based off Timer2 in PWM mode
01 = CCP2 is based off Timer4 in PWM mode
10 = CCP2 is based off Timer6 in PWM mode
11 = Reserved
- bit 1-0 **C1TSEL<1:0>**: CCP1 Timer Selection bits
00 = CCP1 is based off Timer2 in PWM mode
01 = CCP1 is based off Timer4 in PWM mode
10 = CCP1 is based off Timer6 in PWM mode
11 = Reserved

26.1.2.4 Receive Framing Error

Each character in the receive FIFO buffer has a corresponding framing error Status bit. A framing error indicates that a Stop bit was not seen at the expected time. The framing error status is accessed via the FERR bit of the RCSTA register. The FERR bit represents the status of the top unread character in the receive FIFO. Therefore, the FERR bit must be read before reading the RCREG.

The FERR bit is read-only and only applies to the top unread character in the receive FIFO. A framing error (FERR = 1) does not preclude reception of additional characters. It is not necessary to clear the FERR bit. Reading the next character from the FIFO buffer will advance the FIFO to the next character and the next corresponding framing error.

The FERR bit can be forced clear by clearing the SPEN bit of the RCSTA register which resets the EUSART. Clearing the CREN bit of the RCSTA register does not affect the FERR bit. A framing error by itself does not generate an interrupt.

Note: If all receive characters in the receive FIFO have framing errors, repeated reads of the RCREG will not clear the FERR bit.

26.1.2.5 Receive Overrun Error

The receive FIFO buffer can hold two characters. An overrun error will be generated if a third character, in its entirety, is received before the FIFO is accessed. When this happens the OERR bit of the RCSTA register is set. The characters already in the FIFO buffer can be read but no additional characters will be received until the error is cleared. The error must be cleared by either clearing the CREN bit of the RCSTA register or by resetting the EUSART by clearing the SPEN bit of the RCSTA register.

26.1.2.6 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCSTA register is set the EUSART will shift nine bits into the RSR for each character received. The RX9D bit of the RCSTA register is the ninth and Most Significant data bit of the top unread character in the receive FIFO. When reading 9-bit data from the receive FIFO buffer, the RX9D data bit must be read before reading the eight Least Significant bits from the RCREG.

26.1.2.7 Address Detection

A special Address Detection mode is available for use when multiple receivers share the same transmission line, such as in RS-485 systems. Address detection is enabled by setting the ADDEN bit of the RCSTA register.

Address detection requires 9-bit character reception. When address detection is enabled, only characters with the ninth data bit set will be transferred to the receive FIFO buffer, thereby setting the RCIF interrupt bit. All other characters will be ignored.

Upon receiving an address character, user software determines if the address matches its own. Upon address match, user software must disable address detection by clearing the ADDEN bit before the next Stop bit occurs. When user software detects the end of the message, determined by the message protocol used, software places the receiver back into the Address Detection mode by setting the ADDEN bit.

26.3 EUSART Baud Rate Generator (BRG)

The Baud Rate Generator (BRG) is an 8-bit or 16-bit timer that is dedicated to the support of both the asynchronous and synchronous EUSART operation. By default, the BRG operates in 8-bit mode. Setting the BRG16 bit of the BAUDCON register selects 16-bit mode.

The SPBRGH, SPBRGL register pair determines the period of the free running baud rate timer. In Asynchronous mode the multiplier of the baud rate period is determined by both the BRGH bit of the TXSTA register and the BRG16 bit of the BAUDCON register. In Synchronous mode, the BRGH bit is ignored.

Table 26-3 contains the formulas for determining the baud rate. Example 26-1 provides a sample calculation for determining the baud rate and baud rate error.

Typical baud rates and error values for various asynchronous modes have been computed for your convenience and are shown in Table 26-3. It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG (BRG16 = 1) to reduce the baud rate error. The 16-bit BRG mode is used to achieve slow baud rates for fast oscillator frequencies.

Writing a new value to the SPBRGH, SPBRGL register pair causes the BRG timer to be reset (or cleared). This ensures that the BRG does not wait for a timer overflow before outputting the new baud rate.

If the system clock is changed during an active receive operation, a receive error or data loss may result. To avoid this problem, check the status of the RCIDL bit to make sure that the receive operation is Idle before changing the system clock.

EXAMPLE 26-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$\text{Desired Baud Rate} = \frac{F_{OSC}}{64(SPBRGH:SPBRGL + 1)}$$

Solving for SPBRGH:SPBRGL:

$$X = \frac{\frac{F_{OSC}}{\text{Desired Baud Rate}}}{64} - 1$$

$$= \frac{\frac{16000000}{9600}}{64} - 1$$

$$= [25.042] = 25$$

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

$$= 9615$$

$$\text{Error} = \frac{\text{Calc. Baud Rate} - \text{Desired Baud Rate}}{\text{Desired Baud Rate}}$$

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

PIC16(L)F1824/1828

CALL Call Subroutine

Syntax: [*label*] CALL k

Operands: $0 \leq k \leq 2047$

Operation: (PC)+1 → TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Status Affected: None

Description: Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

CALLW Subroutine Call With W

Syntax: [*label*] CALLW

Operands: None

Operation: (PC) + 1 → TOS,
(W) → PC<7:0>,
(PCLATH<6:0>) → PC<14:8>

Status Affected: None

Description: Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction.

CLRF Clear f

Syntax: [*label*] CLRF f

Operands: $0 \leq f \leq 127$

Operation: 00h → (f)
1 → Z

Status Affected: Z

Description: The contents of register 'f' are cleared and the Z bit is set.

CLRW Clear W

Syntax: [*label*] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Description: W register is cleared. Zero bit (Z) is set.

CLRWDTClear Watchdog Timer

Syntax: [*label*] CLRWDTClear Watchdog Timer

Operands: None

Operation: 00h → WDT
0 → WDT prescaler,
1 → $\overline{\text{TO}}$
1 → $\overline{\text{PD}}$

Status Affected: $\overline{\text{TO}}$, $\overline{\text{PD}}$

Description: CLRWDTClear Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{\text{TO}}$ and $\overline{\text{PD}}$ are set.

COMF Complement f

Syntax: [*label*] COMF f,d

Operands: $0 \leq f \leq 127$
d ∈ [0,1]

Operation: (\bar{f}) → (destination)

Status Affected: Z

Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

DECF Decrement f

Syntax: [*label*] DECF f,d

Operands: $0 \leq f \leq 127$
d ∈ [0,1]

Operation: (f) - 1 → (destination)

Status Affected: Z

Description: Decrement register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

PIC16(L)F1824/8

TABLE 30-12: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)							
Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$							
Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
US120	TckH2DTV	SYNC XMIT (Master and Slave)	3.0-5.5V	—	80	ns	
		Clock high to data-out valid	1.8-5.5V	—	100	ns	
US121	TckRF	Clock out rise time and fall time (Master mode)	3.0-5.5V	—	45	ns	
			1.8-5.5V	—	50	ns	
US122	TdTRF	Data-out rise time and fall time	3.0-5.5V	—	45	ns	
			1.8-5.5V	—	50	ns	

FIGURE 30-15: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING

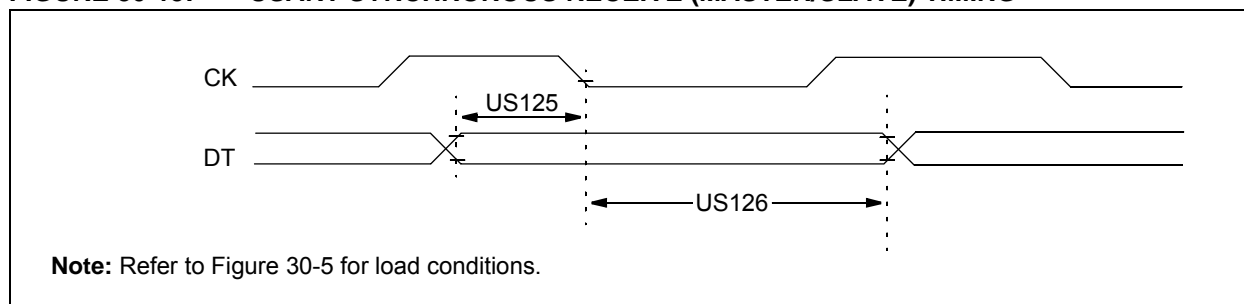


TABLE 30-13: USART SYNCHRONOUS RECEIVE REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)						
Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$						
Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
US125	TdtV2ckL	SYNC RCV (Master and Slave) Data-hold before CK ↓ (DT hold time)	10	—	ns	
US126	TckL2DTL	Data-hold after CK ↓ (DT hold time)	15	—	ns	

FIGURE 31-54: COMPARATOR HYSTERESIS, NORMAL-POWER MODE ($CxSP = 1$, $CxHYS = 1$)

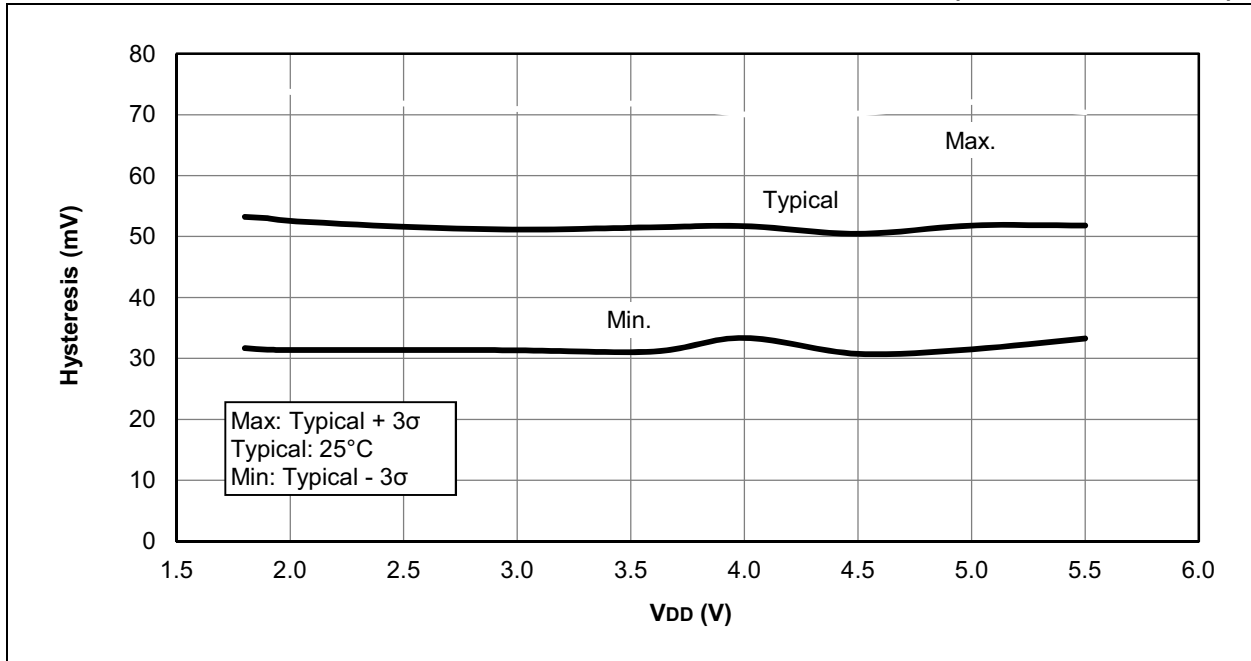
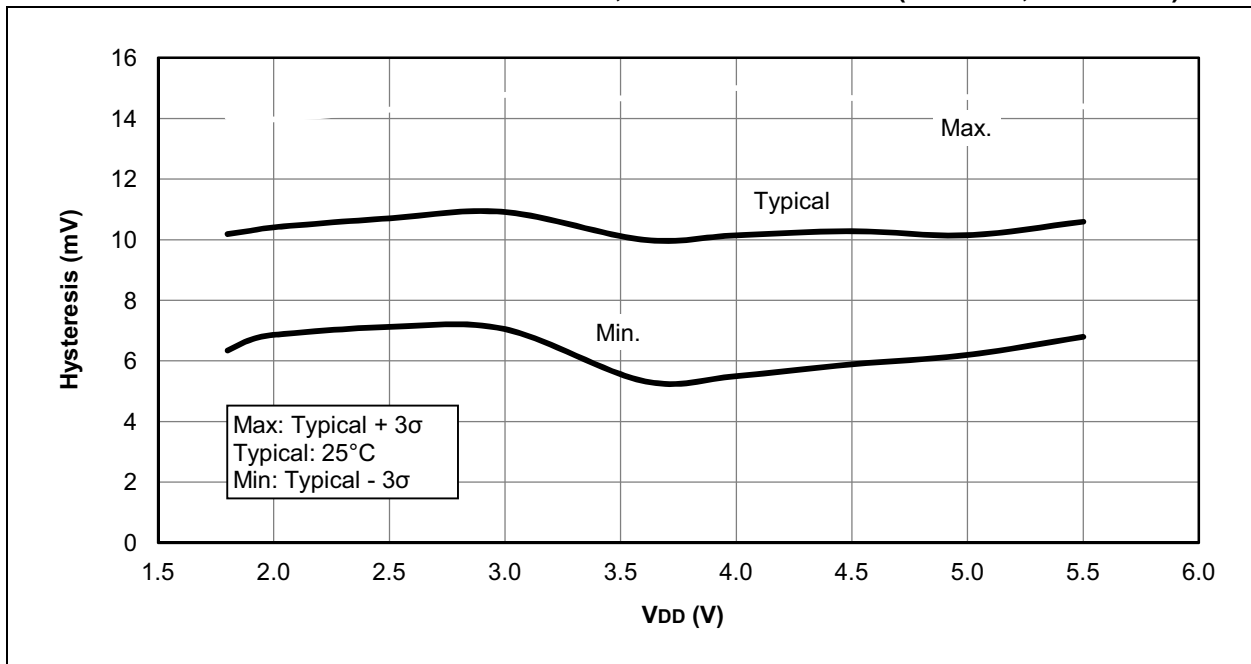


FIGURE 31-55: COMPARATOR HYSTERESIS, LOW-POWER MODE ($CxSP = 0$, $CxHYS = 1$)



32.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip's 8, 16, and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

32.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB X IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multipurpose source files
- Directives that allow complete control over the assembly process

32.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/librarian features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

32.5 MPLAB Assembler, Linker and Librarian for Various Device Families

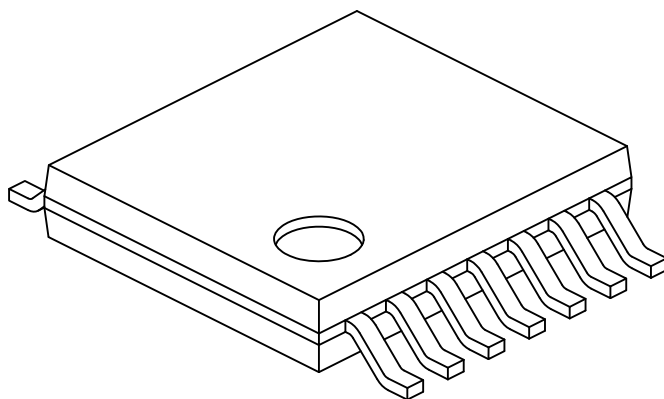
MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

PIC16(L)F1824/8

14-Lead Plastic Thin Shrink Small Outline (ST) - 4.4 mm Body [TSSOP]

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



		Units	MILLIMETERS		
Dimension Limits			MIN	NOM	MAX
Number of Pins	N		14		
Pitch	e		0.65 BSC		
Overall Height	A	-	-	-	1.20
Molded Package Thickness	A2	0.80	1.00	1.05	
Standoff	A1	0.05	-	-	0.15
Overall Width	E		6.40 BSC		
Molded Package Width	E1	4.30	4.40	4.50	
Molded Package Length	D	4.90	5.00	5.10	
Foot Length	L	0.45	0.60	0.75	
Footprint	(L1)		1.00 REF		
Foot Angle	ϕ	0°	-	-	8°
Lead Thickness	c	0.09	-	-	0.20
Lead Width	b	0.19	-	-	0.30

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M

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

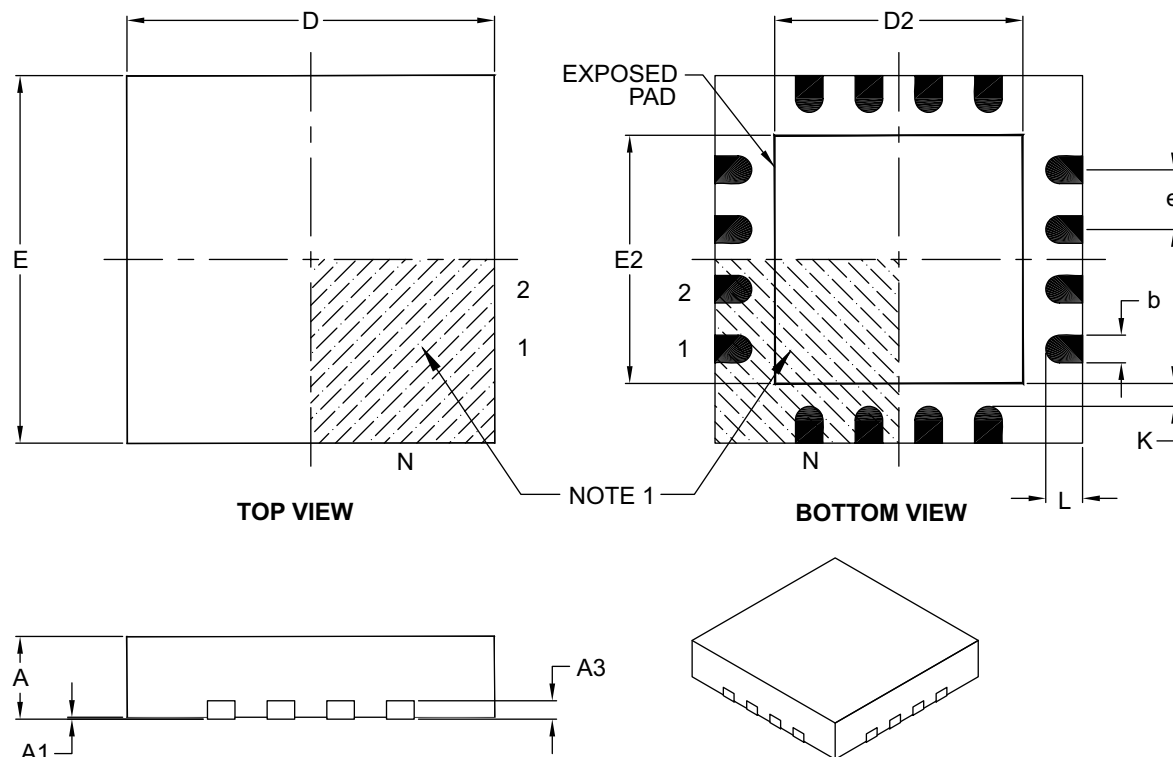
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing No. C04-087C Sheet 2 of 2

PIC16(L)F1824/8

16-Lead Plastic Quad Flat, No Lead Package (ML) – 4x4x0.9 mm Body [QFN]

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



		Units	MILLIMETERS		
Dimension Limits			MIN	NOM	MAX
Number of Pins	N		16		
Pitch	e		0.65 BSC		
Overall Height	A		0.80	0.90	1.00
Standoff	A1		0.00	0.02	0.05
Contact Thickness	A3		0.20 REF		
Overall Width	E		4.00 BSC		
Exposed Pad Width	E2		2.50	2.65	2.80
Overall Length	D		4.00 BSC		
Exposed Pad Length	D2		2.50	2.65	2.80
Contact Width	b		0.25	0.30	0.35
Contact Length	L		0.30	0.40	0.50
Contact-to-Exposed Pad	K		0.20	—	—

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- 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 C04-127B