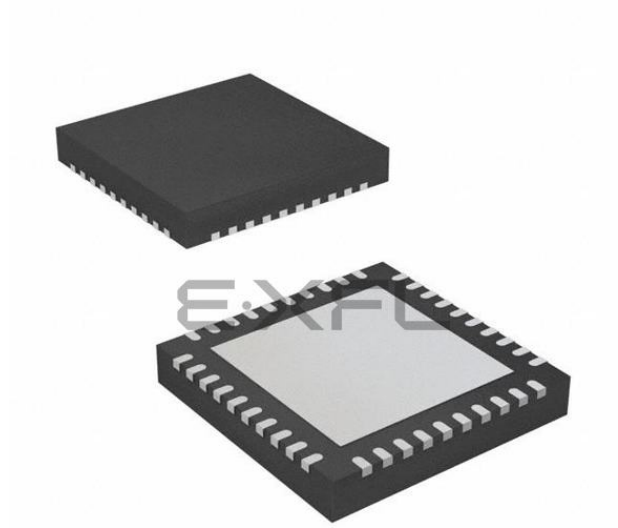


Welcome to E-XFL.COM

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)	2.3V ~ 5.5V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	40-UFQFN Exposed Pad
Supplier Device Package	40-UQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f47k42t-i-mv

4.7.2 DIRECT ADDRESSING

Direct addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of direct addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 4.5.2 "General Purpose Register File"**) or a location in the Access Bank (**Section 4.5.4 "Access Bank"**) as the data source for the instruction.

The Access RAM bit 'a' determines how the address is interpreted. When 'a' is '1', the contents of the BSR (**Section 4.5.1 "Bank Select Register (BSR)"**) are used with the address to determine the complete 14-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as MOVFFL, include the entire 14-bit address (either source or destination) in their opcodes. In these cases, the BSR is ignored entirely.

The destination of the operation's results is determined by the destination bit 'd'. When 'd' is '1', the results are stored back in the source register, overwriting its original contents. When 'd' is '0', the results are stored in the W register. Instructions without the 'd' argument have a destination that is implicit in the instruction; their destination is either the target register being operated on or the W register.

4.7.3 INDIRECT ADDRESSING

Indirect addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations which are to be read or written. Since the FSRs are themselves located in RAM as Special File Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures, such as tables and arrays in data memory.

The registers for indirect addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code, using loops, such as the example of clearing an entire RAM bank in Example 4-6.

EXAMPLE 4-6: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR   FSR0, 100h ;
NEXT    CLRF   POSTINC0   ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS  FSR0H, 1   ; All done with
                          ; Bank1?
        BRA    NEXT       ; NO, clear next
CONTINUE                  ; YES, continue
```

4.7.3.1 FSR Registers and the INDF Operand

At the core of indirect addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. Each FSR pair holds a 14-bit value, therefore, the two upper bits of the FSRnH register are not used. The 14-bit FSR value can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as "virtual" registers; they are mapped in the SFR space but are not physically implemented. Reading or writing to a particular INDF register actually accesses the data addressed by its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L. Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction's target. The INDF operand is just a convenient way of using the pointer.

Because indirect addressing uses a full 14-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

10.0 POWER-SAVING OPERATION MODES

The purpose of the Power-Down modes is to reduce power consumption. There are three Power-Down modes:

- Doze mode
- Sleep mode
- Idle mode

10.1 Doze Mode

Doze mode saves power by reducing CPU execution and program memory (PFM) access, without affecting peripheral operation.

10.1.1 DOZE OPERATION

When the Doze Enable bit is set (DOZEN = 1), the CPU executes one instruction cycle out of every N cycles as defined by the DOZE<2:0> bits of the CPUDOZE register. FOSC and FOSC/4 clock sources are unaffected in Doze mode and peripherals can continue using these sources.

10.1.2 INTERRUPTS DURING DOZE

When an interrupt occurs during Doze, the system behavior can be configured using the Recover-On-Interrupt bit (ROI) and the Doze-On-Exit bit (DOE). Refer to Table 10-2 for details about system behavior in all the cases for a transition from Main > ISR > Main. For PIC18(L)F26/27/45/46/47/55/56/57 devices, the transition from Main > ISR > Main always happens in Normal operation, regardless of the state of the DOZEN or DOE bits.

TABLE 10-1: SYSTEM BEHAVIOR FOR INTERRUPT DURING DOZE

DOZEN	ROI	Code Flow			
		Main	ISR (1)	Return to Main	
0	0	Normal operation	Normal operation and DOE = DOZEN (in hardware) DOZEN = 0 (unchanged)	If DOE = 1 when return from interrupt; Doze operation and DOZEN = 1 (in hardware)	If DOE = 0 when return from interrupt; Normal operation and DOZEN = 0 (in hardware)
0	1	Normal operation	Normal operation and DOE = DOZEN (in hardware) DOZEN = 0 (in hardware)		
1	0	Doze operation	Doze operation and DOE = DOZEN (in hardware) DOZEN = 1 (unchanged)		
1	1	Doze operation	Normal operation and DOE = DOZEN (in hardware) DOZEN = 0 (in hardware)		

Note 1: User software can change the DOE bit in ISR.

For example, if ROI = 1 and DOZE<2:0> = 001, the instruction cycle ratio is 1:4. The CPU and memory operate for one instruction cycle and stay idle for the next three instruction cycles. The Doze operation is illustrated in Figure 10-1.

13.1.2 CONTROL REGISTERS

Several control registers are used in conjunction with the TBLRD and TBLWT instructions. These include the following registers:

- NVMCON1 register
- NVMCON2 register
- TABLAT register
- TBLPTR registers

13.1.2.1 NVMCON1 and NVMCON2 Registers

The NVMCON1 register (Register 13-1) is the control register for memory accesses. The NVMCON2 register is not a physical register; it is used exclusively in the memory write and erase sequences. Reading NVMCON2 will read all '0's.

The REG<1:0> control bits determine if the access will be to Data EEPROM Memory locations. PFM locations or User IDs, Configuration bits, Rev ID and Device ID. When REG<1:0> = 00, any subsequent operations will operate on the Data EEPROM Memory. When REG<1:0> = 10, any subsequent operations will operate on the program memory. When REG<1:0> = x1, any subsequent operations will operate on the Configuration bits, User IDs, Rev ID and Device ID.

The FREE bit allows the program memory erase operation. When the FREE bit is set, an erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled. This bit is applicable only to the PFM and not to data EEPROM.

When set, the WREN bit will allow a program/erase operation. The WREN bit is cleared on power-up.

The WRERR bit is set by hardware when the WR bit is set and is cleared when the internal programming timer expires and the write operation is successfully complete.

The WR control bit initiates erase/write cycle operation when the REG<1:0> bits point to the Data EEPROM Memory location, and it initiates a write operation when the REG<1:0> bits point to the PFM location. The WR bit cannot be cleared by firmware; it can only be set by firmware. Then the WR bit is cleared by hardware at the completion of the write operation.

The NVMIF Interrupt Flag bit is set when the write is complete. The NVMIF flag stays set until cleared by firmware.

13.1.2.2 TABLAT – Table Latch Register

The Table Latch (TABLAT) is an 8-bit register mapped into the SFR space. The Table Latch register is used to hold 8-bit data during data transfers between program memory and data RAM.

13.1.2.3 TBLPTR – Table Pointer Register

The Table Pointer (TBLPTR) register addresses a byte within the program memory. The TBLPTR is comprised of three SFR registers: Table Pointer Upper Byte, Table Pointer High Byte and Table Pointer Low Byte (TBLPTRU:TBLPTRH:TBLPTRL). These three registers join to form a 22-bit wide pointer. The low-order 21 bits allow the device to address up to 2 Mbytes of program memory space. The 22nd bit allows access to the Device ID, the User ID and the Configuration bits.

The Table Pointer register, TBLPTR, is used by the TBLRD and TBLWT instructions. These instructions can update the TBLPTR in one of four ways based on the table operation. These operations on the TBLPTR affect only the low-order 21 bits.

13.1.2.4 Table Pointer Boundaries

TBLPTR is used in reads, writes and erases of the Program Flash Memory.

When a TBLRD is executed, all 22 bits of the TBLPTR determine which byte is read from program memory directly into the TABLAT register.

When a TBLWT is executed the byte in the TABLAT register is written, not to memory but, to a holding register in preparation for a program memory write. The holding registers constitute a write block which varies depending on the device (see Table 5-4).The 3, 4, or 5 LSbs of the TBLPTRL register determine which specific address within the holding register block is written to. The MSBs of the Table Pointer have no effect during TBLWT operations.

When a program memory write is executed the entire holding register block is written to the memory at the address determined by the MSbs of the TBLPTR. The 3, 4, or 5 LSBs are ignored during memory writes. For more detail, see **Section 13.1.6 "Writing to Program Flash Memory"**.

Figure 13-3 describes the relevant boundaries of TBLPTR based on Program Flash Memory operations.

15.10 Reset

The DMA registers are set to the default state on any Reset. The registers are also reset to the default state when the enable bit is cleared (DMA1CON1bits.EN=0).

15.11 Power Saving Mode Operation

The DMA utilizes system clocks and it is treated as a peripheral when it comes to power saving operations. Like other peripherals, the DMA also uses Peripheral Module Disable bits to further tailor its operation in low-power states.

15.11.1 SLEEP MODE

When the device enters Sleep mode, the system clock to the module is shut down, therefore no DMA operation is supported in Sleep. Once the system clock is disabled, the requisite read and write clocks are also disabled, without which the DMA cannot perform any of its tasks.

Any transfers that may be in progress are resumed on exiting from Sleep mode. Register contents are not affected by the device entering or leaving Sleep mode. It is recommended that DMA transactions be allowed to finish before entering Sleep mode.

15.11.2 IDLE MODE

In IDLE mode, all of the system clocks (including the read and write clocks) are still operating but the CPU is not using them to save power.

Therefore, every instruction cycle is available to the system arbiter and if the bubble is granted to the DMA, it may be utilized to move data.

15.11.3 DOZE MODE

Similar to the Idle mode, the CPU does not utilize all of the available instruction cycles slots that are available to it in order to save power. It only executes instructions based on its settings from the Doze settings.

Therefore, every instruction not used by the CPU is available for system arbitration and may be utilized by the DMA if granted by the arbiter.

15.11.4 PERIPHERAL MODULE DISABLE

The Peripheral Module Disable (PMD) registers provide a method to disable DMA by gating all clock sources supplied to it. The respective DMAxMD bit needs to be set in order to disable the DMA.

15.12 DMA Register Interfaces

The DMA can transfer data to any GPR or SFR location. For better user accessibility, some of the more commonly used SFR spaces have their Mirror registers placed in Bank 64 (0x4000-0x40FF). These Mirror registers can be only accessed through the DMA Source and Destination Address registers. Refer to Table 4-3 for details about Bank 64 Registers.

22.5 Operation Examples

Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPS and OUTPS bits in the T2CON register are cleared).
- The diagrams illustrate any clock except FOSC/4 and show clock-sync delays of at least two full cycles for both ON and T2TMR_ers. When using FOSC/4, the clock-sync delay is at least one instruction period for T2TMR_ers; ON applies in the next instruction period.
- ON and T2TMR_ers are somewhat generalized, and clock-sync delays may produce results that are slightly different than illustrated.
- The PWM Duty Cycle and PWM output are illustrated assuming that the timer is used for the PWM function of the CCP module as described in **Section 23.0 "Capture/ Compare/PWM Module"** and **Section 24.0 "Pulse-Width Modulation (PWM)"**. The signals are not a part of the T2TMR module.

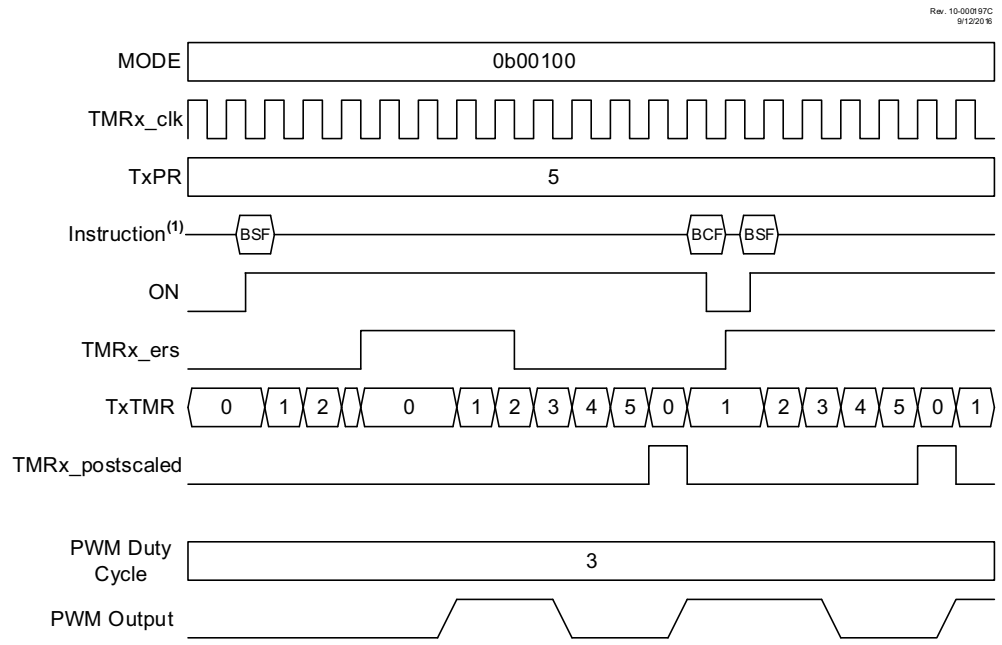
22.5.3 EDGE-TRIGGERED HARDWARE LIMIT MODE

In Hardware Limit mode the timer can be reset by the TMRx_ers external signal before the timer reaches the period count. Three types of Resets are possible:

- Reset on rising or falling edge (MODE<4:0> = 00011)
- Reset on rising edge (MODE<4:0> = 0010)
- Reset on falling edge (MODE<4:0> = 00101)

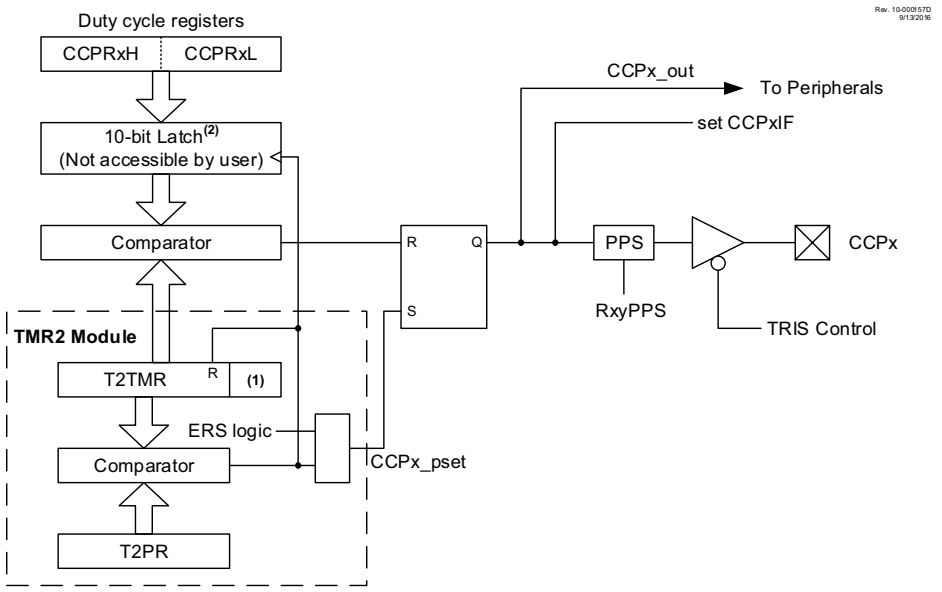
When the timer is used in conjunction with the CCP in PWM mode then an early Reset shortens the period and restarts the PWM pulse after a two clock delay. Refer to Figure 22-6.

FIGURE 22-6: EDGE TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE=00100)



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.

FIGURE 23-4: SIMPLIFIED PWM BLOCK DIAGRAM



Notes: 1. 8-bit timer is concatenated with two bits generated by Fosc or two bits of the internal prescaler to create 10-bit time-base.
2. The alignment of the 10 bits from the CCPR register is determined by the CCPxFMT bit.

REGISTER 26-2: CWGxCON1: CWG CONTROL REGISTER 1

U-0	U-0	R-x	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	IN	—	POLD	POLC	POLB	POLA
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	q = Value depends on condition

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **IN:** CWG Input Value bit (read-only)

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

bit 3 **POLD:** CWGxD Output Polarity bit
1 = Signal output is inverted polarity
0 = Signal output is normal polarity

bit 2 **POLC:** CWGxC Output Polarity bit
1 = Signal output is inverted polarity
0 = Signal output is normal polarity

bit 1 **POLB:** CWGxB Output Polarity bit
1 = Signal output is inverted polarity
0 = Signal output is normal polarity

bit 0 **POLA:** CWGxA Output Polarity bit
1 = Signal output is inverted polarity
0 = Signal output is normal polarity

REGISTER 27-2: CLCxPOL: SIGNAL POLARITY CONTROL REGISTER

R/W-0/0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
POL	—	—	—	G4POL	G3POL	G2POL	G1POL
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 **POL:** CLCxOUT Output Polarity Control bit
1 = The output of the logic cell is inverted
0 = The output of the logic cell is not inverted

bit 6-4 **Unimplemented:** Read as '0'

bit 3 **G4POL:** Gate 3 Output Polarity Control bit
1 = The output of gate 3 is inverted when applied to the logic cell
0 = The output of gate 3 is not inverted

bit 2 **G3POL:** Gate 2 Output Polarity Control bit
1 = The output of gate 2 is inverted when applied to the logic cell
0 = The output of gate 2 is not inverted

bit 1 **G2POL:** Gate 1 Output Polarity Control bit
1 = The output of gate 1 is inverted when applied to the logic cell
0 = The output of gate 1 is not inverted

bit 0 **G1POL:** Gate 0 Output Polarity Control bit
1 = The output of gate 0 is inverted when applied to the logic cell
0 = The output of gate 0 is not inverted

30.11 Register Definitions: Modulation Control

Long bit name prefixes for the Modulation peripheral is shown below. Refer to **Section 1.3.2.2 "Long Bit Names"** for more information.

Peripheral	Bit Name Prefix
MD1	MD1

REGISTER 30-1: MD1CON0: MODULATION CONTROL REGISTER 0

R/W-0/0	U-0	R-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
EN	—	OUT	OPOL	—	—	—	BIT
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 **EN:** Modulator Module Enable bit
1 = Modulator module is enabled and mixing input signals
0 = Modulator module is disabled and has no output

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

bit 5 **OUT:** Modulator Output bit
Displays the current output value of the Modulator module.(1)

bit 4 **OPOL:** Modulator Output Polarity Select bit
1 = Modulator output signal is inverted; idle high output
0 = Modulator output signal is not inverted; idle low output

bit 3-1 **Unimplemented:** Read as '0'

bit 0 **BIT:** Allows software to manually set modulation source input to module(2)
1 = Modulator selects Carrier High
0 = Modulator selects Carrier Low

Note 1: The modulated output frequency can be greater and asynchronous from the clock that updates this register bit, the bit value may not be valid for higher speed modulator or carrier signals.
2: BIT bit must be selected as the modulation source in the MD1SRC register for this operation.

31.3 Asynchronous Address Mode

A special Address Detection mode is available for use when multiple receivers share the same transmission line, such as in RS-485 systems.

When Asynchronous Address mode is enabled, all data is transmitted and received as 9-bit characters. The 9th bit determines whether the character is an address or data. When the 9th bit is set, the eight Least Significant bits are the address. When the 9th bit is clear, the Least Significant bits are data. In either case, the 9th bit is stored in PERIF when the byte is written to the receive FIFO. When PERIE is also set, the RXIF will be suppressed, thereby suspending DMA transfers allowing software to process the received address.

An address character will enable all receivers that match the address and disable all other receivers. Once a receiver is enabled, all non-address characters will be received until an address character is received that does not match.

31.3.1 ADDRESS MODE TRANSMIT

The UART transmitter is enabled for asynchronous address operation by configuring the following control bits:

- TXEN = 1
- MODE<3:0> = 0100
- UxBRGH:L = desired baud rate
- RxyPPS = code for desired output pin
- ON = 1

Addresses are sent by writing to the UxP1L register. This transmits the written byte with the 9th bit set, which indicates that the byte is an address.

Data is sent by writing to the UxTXB register. This transmits the written byte with the 9th bit cleared, which indicates that the byte is data.

To send data to a particular device on the transmission bus, first transmit the address of the intended device. All subsequent data will be accepted only by that device until an address of another device is transmitted.

Writes to UxP1L take precedence over writes to UxTXB. When both the UxP1L and UxTXB registers are written while the TSR is busy, the next byte to be transmitted will be from UxP1L.

To ensure that all data intended for one device is sent before the address is changed, wait until the TXMTIF bit is high before writing UxP1L with the new address.

31.3.2 ADDRESS MODE RECEIVE

The UART receiver is enabled for asynchronous address operation by configuring the following control bits:

- RXEN = 1
- MODE<3:0> = 0100
- UxBRGH:L = desired baud rate
- RXPPS = code for desired input pin
- Input pin ANSEL bit = 0
- UxP2L = receiver address
- UxP3L = address mask
- ON = 1

In Address mode, no data will be transferred to the input FIFO until a valid address is received. This is the default state. Any of the following conditions will cause the UART to revert to the default state:

- ON = 0
- RXEN = 0
- Received address does not match

When a character with the 9th bit set is received, the Least Significant eight bits of that character will be qualified by the values in the UxP2L and UxP3L registers.

The byte is XOR'd with UxP2L then AND'd with UxP3L. A match occurs when the result is 0h, in which case, the unaltered received character is stored in the receive FIFO, thereby setting the UxRXIF interrupt bit. The 9th bit is stored in the corresponding PERIF bit, identifying this byte as an address.

An address match also enables the receiver for all data such that all subsequent characters without the 9th bit set will be stored in the receive FIFO.

When the 9th bit is set and a match does not occur, the character is not stored in the receive FIFO and all subsequent data is ignored.

The UxP3L register mask allows a range of addresses to be accepted. Software can then determine the sub-address of the range by processing the received address character.

REGISTER 31-7: UxFIFO: UART FIFO STATUS REGISTER

R/W/S-0/0	R/W-0/0	R/W/S/C-1/1	R/S/C-0/0	R/S/C-1/1	S/C-1/1	R/W/S/C-1/1	R/S/C-0/0
TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF
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	S = Hardware set	C = Hardware clear

bit 7 **TXWRE:** Transmit Write Error Status bit (Must be cleared by software)

LIN Master mode:

1 = UxP1L was written when a master process was active

LIN Slave mode:

1 = UxTXB was written when UxP2 = 0 or more than UxP2 bytes have been written to UxTXB since last Break

Address Detect mode:

1 = UxP1L was written before the previous data in UxP1L was transferred to TX shifter

All modes:

1 = A new byte was written to UxTXB when the output FIFO was full
0 = No error

bit 6 **STPMD**: Stop Bit Detection Mode bit

1 = Assert UxRXIF at end of last Stop bit or end of first Stop bit when STP = 11
0 = Assert UxRXIF in middle of first Stop bit

bit 5 **TXBE:** Transmit Buffer Empty Status bit

1 = Transmit buffer is empty. Setting this bit will clear the transmit buffer and output shift register.
0 = Transmit buffer is not empty. Software cannot clear this bit.

bit 4 **TXBF**: Transmit Buffer Full Status bit

1 = Transmit buffer is full
0 = Transmit buffer is not full

bit 3 **RXIDL:** Receive Pin Idle Status bit

1 = Receive pin is in Idle state
0 = UART is receiving Start, Stop, Data, Auto-baud, or Break

bit 2 **XON:** Software Flow Control Transmit Enable Status bit

1 = Transmitter is enabled
0 = Transmitter is disabled

bit 1 **RXBE:** Receive Buffer Empty Status bit

1 = Receive buffer is empty. Setting this bit will clear the RX buffer(1)
0 = Receive buffer is not empty. Software cannot clear this bit.

bit 0 **RXBF:** Receive Buffer Full Status bit

1 = Receive buffer is full
0 = Receive buffer is not full

Note 1: The BSF instruction should not be used to set RXBE because doing so will clear a byte pending in the transmit shift register when the UxTXB register is empty. Instead, use the MOVWF instruction with a '0' in the TXBE bit location.

REGISTER 31-12: UxP1H: UART PARAMETER 1 HIGH REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P1<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 7-6 **Unimplemented:** Read as '0'

bit 0 **P1<8>:** Most Significant Bit of Parameter 1

DMX mode:
Most Significant bit of number of bytes to transmit between Start Code and automatic Break generation

DALI Control Device mode:
Most Significant bit of idle time delay after which a Forward Frame is sent. Measured in half-bit periods

DALI Control Gear mode:
Most Significant bit of delay between the end of a Forward Frame and the start of the Back Frame Measured in half-bit periods

Other modes:
Not used

REGISTER 31-13: UxP1L: UART PARAMETER 1 LOW 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
P1<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 **P1<7:0>:** Least Significant Bits of Parameter 1

DMX mode:
Least Significant Byte of number of bytes to transmit between Start Code and automatic Break generation

DALI Control Device mode:
Least Significant Byte of idle time delay after which a Forward Frame is sent. Measured in half-bit periods

DALI Control Gear mode:
Least Significant Byte of delay between the end of a Forward Frame and the start of the Back Frame Measured in half-bit periods

LIN mode:
PID to transmit (Only Least Significant 6 bits used)

Asynchronous Address mode:
Address to transmit (9th transmit bit automatically set to '1')

Other modes:
Not used

REGISTER 32-2: SPIxINTE: SPI INTERRUPT ENABLE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0
SRMTIE	TCZIE	SOSIE	EOSIE	—	RXOIE	TXUIE	—
bit 7							bit 0

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

bit 7 **SRMTIE**: Shift Register Empty Interrupt Enable bit
1 = Enables the Shift Register Empty Interrupt
0 = Disables the Shift Register Empty Interrupt

bit 6 **TCZIE**: Transfer Counter is Zero Interrupt Enable bit
1 = Enables the Transfer Counter is Zero Interrupt
0 = Disables the Transfer Counter is Zero Interrupt

bit 5 **SOSIE**: Start of Slave Select Interrupt Enable bit
1 = Enables the Start of Slave Select Interrupt
0 = Disables the Start of Slave Select Interrupt

bit 4 **EOSIE**: End of Slave Select Interrupt Enable bit
1 = Enables the End of Slave Select Interrupt
0 = Disables the End of Slave Select Interrupt

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

bit 2 **RXOIE**: Receiver Overflow Interrupt Enable bit
1 = Enables the Receiver Overflow Interrupt
0 = Disables the Receiver Overflow Interrupt

bit 1 **TXUIE**: Transmitter Underflow Interrupt Enable bit
1 = Enables the Transmitter Underflow Interrupt
0 = Disables the Transmitter Underflow Interrupt

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

REGISTER 32-3: SPIxTCNTL – SPI TRANSFER COUNTER LSB 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
TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
bit 7							bit 0

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

bit 7-0 TCNT<7:0>
BMODE = 0
Bits 10-3 of the Transfer Counter, counting the total number of bits to transfer
BMODE = 1
Bits 7-0 of the Transfer Counter, counting the total number of bytes to transfer

Note: This register should not be written to while a transfer is in progress (BUSY bit of SPIxCON2 is set).

39.2 HLVD Setup

To set up the HLVD module:

1. Select the desired HLVD trip point by writing the value to the SEL<3:0> bits of the HLVDCON1 register.
2. Depending on the application to detect high-voltage peaks or low-voltage drops or both, set the INTH or INTL bit appropriately.
3. Enable the HLVD module by setting the EN bit.
4. Clear the HLVD interrupt flag (PIR2 register), which may have been set from a previous interrupt.
5. If interrupts are desired, enable the HLVD interrupt by setting the HLVDIE in the PIE2 register and GIE bits.

   An interrupt will not be generated until the RDY bit is set.

Note:	Before changing any module settings (INTH, INTL, SEL<3:0>), first disable the module (EN = 0), make the changes and re-enable the module. This prevents the generation of false HLVD events.

39.3 Current Consumption

When the module is enabled, the HLVD comparator and voltage divider are enabled and consume static current. The total current consumption, when enabled, is specified in electrical specification Parameter **D206** (Table 44-4).

Depending on the application, the HLVD module does not need to operate constantly. To reduce current requirements, the HLVD circuitry may only need to be enabled for short periods where the voltage is checked. After such a check, the module could be disabled.

39.4 HLVD Start-up Time

The internal reference voltage of the HLVD module, specified in electrical specification (Table 44-19), may be used by other internal circuitry, such as the programmable Brown-out Reset. If the HLVD or other circuits using the voltage reference are disabled to lower the device's current consumption, the reference voltage circuit will require time to become stable before a low or high-voltage condition can be reliably detected. This start-up time, TFVRST, is an interval that is independent of device clock speed. It is specified in electrical specification (Table 44-19).

The HLVD interrupt flag is not enabled until TFVRST has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval (see Figure 39-2 or Figure 39-3).

NEGF **Negate f**

Syntax: NEGF f {,a}

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

Operation: $(\overline{f}) + 1 \rightarrow f$

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

Encoding:

0110	110a	ffff	ffff

Description: Location 'f' is negated using two's complement. The result is placed in the data memory location 'f'.
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

Q Cycle Activity:

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

Example:
```
NEGF    REG, 1
```
Before Instruction
REG = 0011 1010 [3Ah]
After Instruction
REG = 1100 0110 [C6h]

NOP **No Operation**

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example:

None.

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

SUBULNK Subtract Literal from FSR2 and Return

Syntax: SUBULNK k

Operands: $0 \le k \le 63$

Operation: FSR2 – k → FSR2

(TOS) → PC

Status Affected: None

Encoding:

1110	1001	11kk	kkkk

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR2. A RETURN is then executed by loading the PC with the TOS. The instruction takes two cycles to execute; a NOP is performed during the second cycle.
This may be thought of as a special case of the SUBFSR instruction, where f = 3 (binary '11'); it operates only on FSR2.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination
No Operation	No Operation	No Operation	No Operation

Example: SUBULNK 23h

Before Instruction
FSR2 = 03FFh
PC = 0100h

After Instruction
FSR2 = 03DCh
PC = (TOS)

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
3A98h	—	Reserved, maintain as '0'								
3A97h-3A95h	—	Unimplemented								
3A94h	INLVLF(3)	INLVLF7	INLVLF6	INLVLF5	INLVLF4	INLVLF3	INLVLF2	INLVLF1	INLVLF0	270
3A93h	SLRCONF(3)	SLRCONF7	SLRCONF6	SLRCONF5	SLRCONF4	SLRCONF3	SLRCONF2	SLRCONF1	SLRCONF0	269
3A92h	ODCONF(3)	ODCONF7	ODCONF6	ODCONF5	ODCONF4	ODCONF3	ODCONF2	ODCONF1	ODCONF0	268
3A91h	WPUF(3)	WPUF7	WPUF6	WPUF5	WPUF4	WPUF3	WPUF2	WPUF1	WPUF0	267
3A90h	ANSELF(3)	ANSELF7	ANSELF6	ANSELF5	ANSELF4	ANSELF3	ANSELF2	ANSELF1	ANSELF0	266
3A8Fh-3A8Ah	—	Unimplemented								
3A89h	—	Reserved, maintain as '0'								
3A88h	—	Reserved, maintain as '0'								
3A87h	IOCEF	—	—	—	—	IOCEF3	—	—	—	287
3A86h	IOCEN	—	—	—	—	IOCEN3	—	—	—	287
3A85h	IOCEP	—	—	—	—	IOCEP3	—	—	—	287
3A84h	INLVLE	—	—	—	—	INLVLE3	INLVLE2(2)	INLVLE1(2)	INLVLE0(2)	270
3A83h	SLRCONE(2)	—	—	—	—	—	SRLE2(2)	SRLE1(2)	SRLE0(2)	269
3A82h	ODCONE(2)	—	—	—	—	—	ODCE2(2)	ODCE1(2)	ODCE0(2)	268
3A81h	WPUE	—	—	—	—	WPUE3	WPUE2(2)	WPUE1(2)	WPUE0(2)	267
3A80h	ANSELE(2)	ANSELE7	ANSELE6	ANSELE5	ANSELE4	ANSELE3	ANSELE2	ANSELE1	ANSELE0	266
3A7Fh-3A7CH	—	Unimplemented								
3A7Bh	RD1I2C(2)	—	IOCEN3	PU		—	—	TH		263
3A7Ah	RD0I2C(2)	—	IOCEN3	PU		—	—	TH		263
3A79h	—	Reserved, maintain as '0'								
3A78h	—	Reserved, maintain as '0'								
3A77h-3A75h	—	Unimplemented								
3A74h	INLVLD(2)	INLVLD7	INLVLD6	INLVLD5	INLVLD4	INLVLD3	INLVLD2	INLVLD1	INLVLD0	270
3A73h	SLRCOND(2)	SRLD7	SRLD6	SRLD5	SRLD4	SRLD3	SRLD2	SRLD1	SRLD0	269
3A72h	ODCOND(2)	ODCD7	ODCD6	ODCD5	ODCD4	ODCD3	ODCD2	ODCD1	ODCD0	268
3A71h	WPUD(2)	WPUD7	WPUD6	WPUD5	WPUD4	WPUD3	WPUD2	WPUD1	WPUD0	267
3A70h	ANSELD(2)	ANSELD7	ANSELD6	ANSELD5	ANSELD4	ANSELD3	ANSELD2	ANSELD1	ANSELD0	266
3A6Fh-3A6Ch	—	Unimplemented								
3A6Bh	RC4I2C	—	SLEW	PU		—	—	TH		263
3A6Ah	RC3I2C	—	SLEW	PU		—	—	TH		263
3A69h	—	Reserved, maintain as '0'								
3A68h	—	Reserved, maintain as '0'								
3A67h	IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0	287
3A66h	IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0	287
3A65h	IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0	287
3A64h	INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4	INLVLC3	INLVLC2	INLVLC1	INLVLC0	270
3A63h	SLRCONC	SLRC7	SLRC6	SLRC5	SLRC4	SLRC3	SLRC2	SLRC1	SLRC0	269
3A62h	ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	268
3A61h	WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0	267
3A60h	ANSELC	ANSELC7	ANSELC6	ANSELC5	ANSELC4	ANSELC3	ANSELC2	ANSELC1	ANSELC0	266
3A5Fh - 3A5Ch	—	Unimplemented								

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.

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
39DAh	OSCCON2	—	COSC			CDIV				105
39D9h	OSCCON1	—	NOSC			NDIV				104
39D8h	CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE			177
39D7h - 39D2h	—	Unimplemented								
39D1h	VREGCON(1)	—	—	—	—	—	—	VREGPM	—	176
39D0h	BORCON	SBOREN	—	—	—	—	—	—	BORRDY	85
39CFh - 39C8h	—	Unimplemented								
39C7h	PMD7	—	—	—	—	—	—	DMA2MD	DMA1MD	297
39C6h	PMD6	—	—	SMT1MD	CLC4MD	CLC3MD	CLC2MD	CLC1MD	DSMMD	296
39C5h	PMD5	—	—	U2MD	U1MD	—	SPI1MD	I2C2MD	I2C1MD	295
39C4h	PMD4	CWG3MD	CWG2MD	CWG1MD	—	—	—	—	—	294
39C3h	PMD3	PWM8MD	PWM7MD	PWM6MD	PWM5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	293
39C2h	PMD2	—	DACMD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	292
39C1h	PMD1	NCO1MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	291
39C0h	PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	290
39BFh - 39ABh	—	Unimplemented								
39AAh	PIR10	—	—	—	—	—	—	CLC4IF	CCP4IF	146
39A9h	PIR9	—	—	—	—	CLC3IF	CWG3IF	CCP3IF	TMR6IF	145
39A8h	PIR8	TMR5GIF	TMR5IF	—	—	—	—	—	—	145
39A7h	PIR7	—	—	INT2IF	CLC2IF	CWG2IF	—	CCP2IF	TMR4IF	144
39A6h	PIR6	TMR3GIF	TMR3IF	U2IF	U2EIF	U2TXIF	U2RXIF	I2C2EIF	I2C2IF	143
39A5h	PIR5	I2C2TXIF	I2C2RXIF	DMA2AIF	DMA2ORIF	DMA2DCN-TIF	DMA2SCN-TIF	C2IF	INT1IF	142
39A4h	PIR4	CLC1IF	CWG1IF	NCO1IF	—	CCP1IF	TMR2IF	TMR1GIF	TMR1IF	141
39A3h	PIR3	TMR0IF	U1IF	U1EIF	U1TXIF	U1RXIF	I2C1EIF	I2C1IF	I2C1TXIF	140
39A2h	PIR2	I2C1RXIF	SPI1IF	SPI1TXIF	SPI1RXIF	DMA1AIF	DMA1ORIF	DMA1DCN-TIF	DMA1SCNTIF	138
39A1h	PIR1	SMT1PWAIF	SMT1PRAIF	SMT1IF	C1IF	ADTIF	ADIF	ZCDIF	INT0IF	138
39A0h	PIR0	IOCIF	CRCIF	SCANIF	NVMIF	CSWIF	OSFIF	HLVDIF	SWIF	137
399Fh - 399Bh	—	Unimplemented								
399Ah	PIE10	—	—	—	—	—	—	CLC4IE	CCP4IE	156
3999h	PIE9	—	—	—	—	CLC3IE	CWG3IE	CCP3IE	TMR6IE	155
3998h	PIE8	TMR5GIE	TMR5IE	—	—	—	—	—	—	155
3997h	PIE7	—	—	INT2IE	CLC2IE	CWG2IE	—	CCP2IE	TMR4IE	154
3996h	PIE6	TMR3GIE	TMR3IE	U2IE	U2EIE	U2TXIE	U2RXIE	I2C2EIE	I2C2IE	153
3995h	PIE5	I2C2TXIE	I2C2RXIE	DMA2AIE	DMA2ORIE	DMA2DCN-TIE	DMA2SCN-TIE	C2IE	INT1IE	152
3994h	PIE4	CLC1IE	CWG1IE	NCO1IE	—	CCP1IE	TMR2IE	TMR1GIE	TMR1IE	151
3993h	PIE3	TMR0IE	U1IE	U1EIE	U1TXIE	U1RXIE	I2C1EIE	I2C1IE	I2C1TXIE	150
3992h	PIE2	I2C1RXIE	SPI1IE	SPI1TXIE	SPI1RXIE	DMA1AIE	DMA1ORIE	DMA1DCN-TIE	DMA1SCNTIE	149
3991h	PIE1	SMT1PWAIE	SMT1PRAIE	SMT1IE	C1IE	ADTIE	ADIE	ZCDIE	INT0IE	148
3990h	PIE0	IOCIE	CRCIE	SCANIE	NVMIE	CSWIE	OSFIE	HLVDIE	SWIE	147
398Fh - 398Bh	—	Unimplemented								
398Ah	IPR10	—	—	—	—	—	—	CLC4IP	CCP4IP	165

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.