



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](#)"

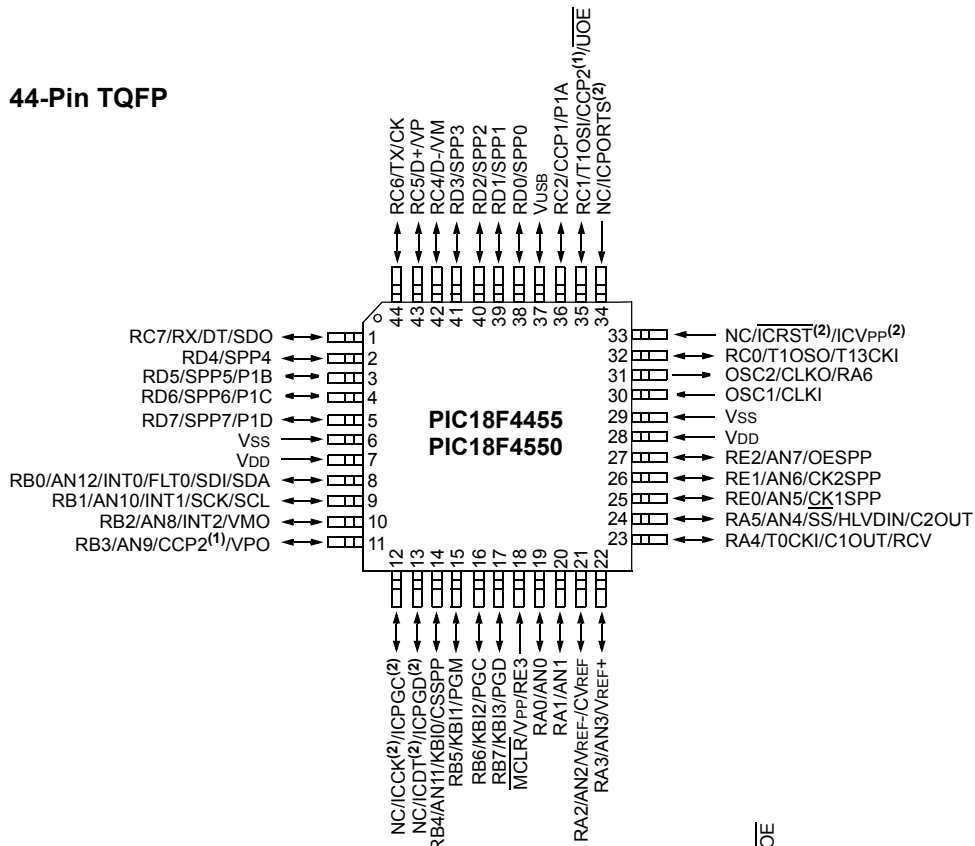
Details

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	48MHz
Connectivity	I ² C, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	35
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 5.5V
Data Converters	A/D 13x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf4550t-i-ml

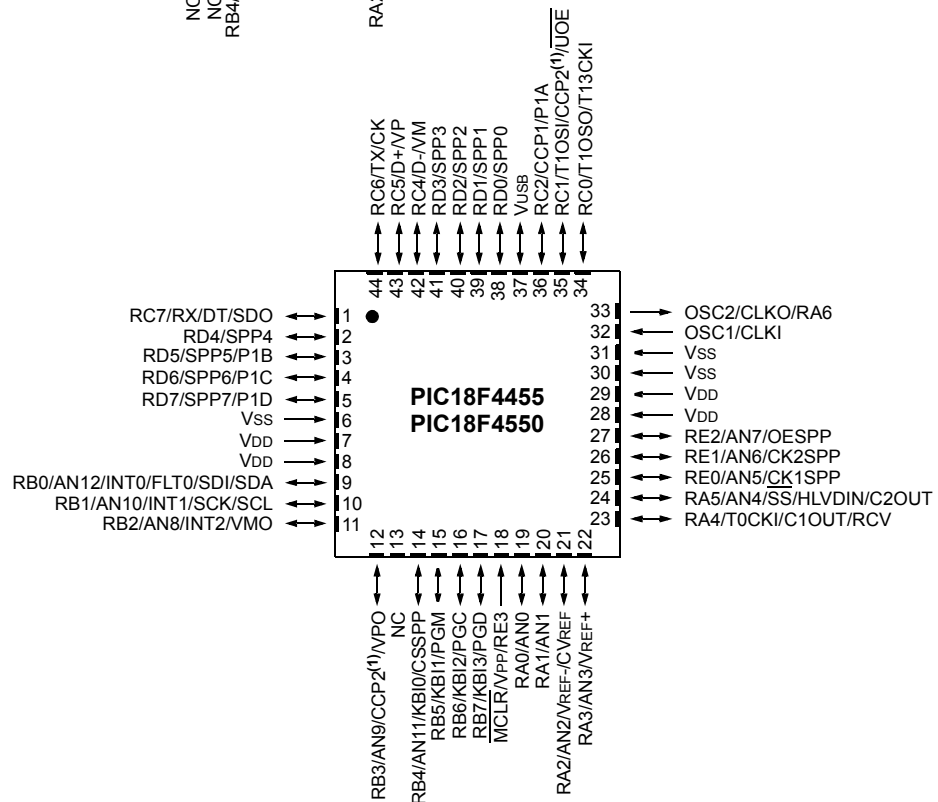
PIC18F2455/2550/4455/4550

Pin Diagrams (Continued)

44-Pin TQFP



44-Pin QFN

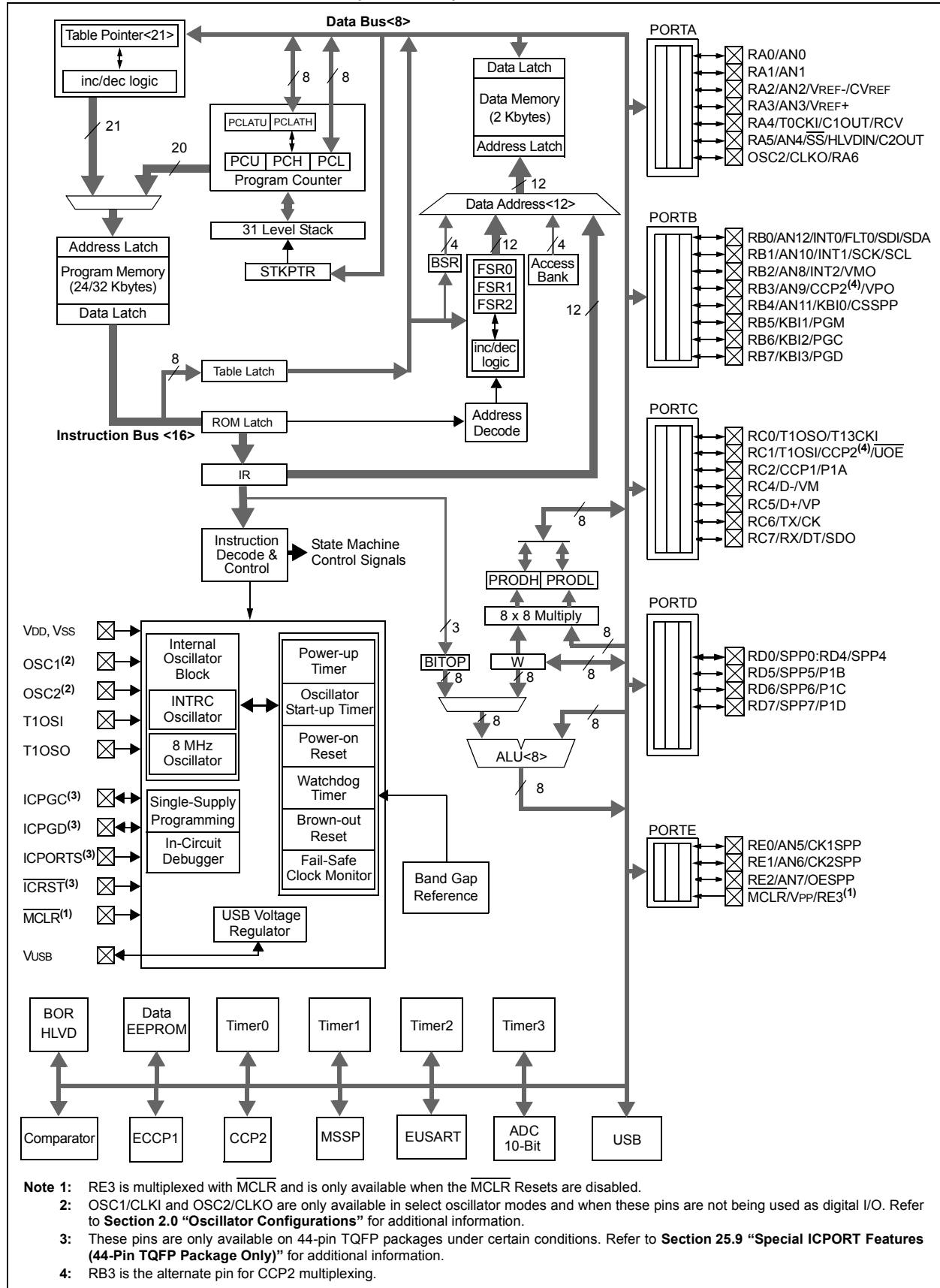


Note 1: RB3 is the alternate pin for CCP2 multiplexing.

2: Special ICPORT features available in select circumstances. See **Section 25.9 "Special ICPORT Features (44-Pin TQFP Package Only)"** for more information.

PIC18F2455/2550/4455/4550

FIGURE 1-2: PIC18F4455/4550 (40/44-PIN) BLOCK DIAGRAM



2.2.5.4 Compensating for INTOSC Drift

It is possible to adjust the INTOSC frequency by modifying the value in the OSCTUNE register. This has no effect on the INTRC clock source frequency.

Tuning the INTOSC source requires knowing when to make the adjustment, in which direction it should be made and in some cases, how large a change is needed. When using the EUSART, for example, an adjustment may be required when it begins to generate framing errors or receives data with errors while in Asynchronous mode. Framing errors indicate that the device clock frequency is too high; to adjust for this, decrement the value in OSCTUNE to reduce the clock frequency. On the other hand, errors in data may suggest that the clock speed is too low; to compensate, increment OSCTUNE to increase the clock frequency.

It is also possible to verify device clock speed against a reference clock. Two timers may be used: one timer is clocked by the peripheral clock, while the other is clocked by a fixed reference source, such as the Timer1 oscillator. Both timers are cleared but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value is greater than expected, then the internal oscillator block is running too fast. To adjust for this, decrement the OSCTUNE register.

Finally, a CCP module can use free-running Timer1 (or Timer3), clocked by the internal oscillator block and an external event with a known period (i.e., AC power frequency). The time of the first event is captured in the CCPRxH:CCPRxL registers and is recorded for use later. When the second event causes a capture, the time of the first event is subtracted from the time of the second event. Since the period of the external event is known, the time difference between events can be calculated.

If the measured time is much greater than the calculated time, the internal oscillator block is running too fast; to compensate, decrement the OSCTUNE register. If the measured time is much less than the calculated time, the internal oscillator block is running too slow; to compensate, increment the OSCTUNE register.

4.5 Device Reset Timers

PIC18F2455/2550/4455/4550 devices incorporate three separate on-chip timers that help regulate the Power-on Reset process. Their main function is to ensure that the device clock is stable before code is executed. These timers are:

- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- PLL Lock Time-out

4.5.1 POWER-UP TIMER (PWRT)

The Power-up Timer (PWRT) of the PIC18F2455/2550/4455/4550 devices is an 11-bit counter which uses the INTRC source as the clock input. This yields an approximate time interval of $2048 \times 32 \mu\text{s} = 65.6 \text{ ms}$. While the PWRT is counting, the device is held in Reset.

The power-up time delay depends on the INTRC clock and will vary from chip to chip due to temperature and process variation. See DC parameter 33 (Table 28-12) for details.

The PWRT is enabled by clearing the $\overline{\text{PWRTE}}\text{N}$ Configuration bit.

4.5.2 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over (parameter 33, Table 28-12). This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, HS and HSPLL modes and only on Power-on Reset or on exit from most power-managed modes.

4.5.3 PLL LOCK TIME-OUT

With the PLL enabled in its PLL mode, the time-out sequence following a Power-on Reset is slightly different from other oscillator modes. A separate timer is used to provide a fixed time-out that is sufficient for the PLL to lock to the main oscillator frequency. This PLL lock time-out (T_{PLL}) is typically 2 ms and follows the oscillator start-up time-out.

4.5.4 TIME-OUT SEQUENCE

On power-up, the time-out sequence is as follows:

1. After the POR condition has cleared, PWRT time-out is invoked (if enabled).
2. Then, the OST is activated.

The total time-out will vary based on oscillator configuration and the status of the PWRT. Figure 4-3, Figure 4-4, Figure 4-5, Figure 4-6 and Figure 4-7 all depict time-out sequences on power-up, with the Power-up Timer enabled and the device operating in HS Oscillator mode. Figures 4-3 through 4-6 also apply to devices operating in XT mode. For devices in RC mode and with the PWRT disabled, on the other hand, there will be no time-out at all.

Since the time-outs occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, all time-outs will expire. Bringing $\overline{\text{MCLR}}$ high will begin execution immediately (Figure 4-5). This is useful for testing purposes or to synchronize more than one PIC18FXXXX device operating in parallel.

TABLE 4-2: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up ⁽²⁾ and Brown-out		Exit from Power-Managed Mode
	$\overline{\text{PWRTE}}\text{N} = 0$	$\overline{\text{PWRTE}}\text{N} = 1$	
HS, XT	$66 \text{ ms}^{(1)} + 1024 T_{\text{osc}}$	$1024 T_{\text{osc}}$	$1024 T_{\text{osc}}$
HSPLL, XTPLL	$66 \text{ ms}^{(1)} + 1024 T_{\text{osc}} + 2 \text{ ms}^{(2)}$	$1024 T_{\text{osc}} + 2 \text{ ms}^{(2)}$	$1024 T_{\text{osc}} + 2 \text{ ms}^{(2)}$
EC, ECIO	$66 \text{ ms}^{(1)}$	—	—
ECPLL, ECPIO	$66 \text{ ms}^{(1)} + 2 \text{ ms}^{(2)}$	$2 \text{ ms}^{(2)}$	$2 \text{ ms}^{(2)}$
INTIO, INTCKO	$66 \text{ ms}^{(1)}$	—	—
INTHS, INTXT	$66 \text{ ms}^{(1)} + 1024 T_{\text{osc}}$	$1024 T_{\text{osc}}$	$1024 T_{\text{osc}}$

Note 1: 66 ms (65.5 ms) is the nominal Power-up Timer (PWRT) delay.

Note 2: 2 ms is the nominal time required for the PLL to lock.

PIC18F2455/2550/4455/4550

5.4 Data Addressing Modes

Note: The execution of some instructions in the core PIC18 instruction set are changed when the PIC18 extended instruction set is enabled. See **Section 5.6 “Data Memory and the Extended Instruction Set”** for more information.

While the program memory can be addressed in only one way – through the program counter – information in the data memory space can be addressed in several ways. For most instructions, the addressing mode is fixed. Other instructions may use up to three modes, depending on which operands are used and whether or not the extended instruction set is enabled.

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

An additional addressing mode, Indexed Literal Offset, is available when the extended instruction set is enabled (XINST Configuration bit = 1). Its operation is discussed in greater detail in **Section 5.6.1 “Indexed Addressing with Literal Offset”**.

5.4.1 INHERENT AND LITERAL ADDRESSING

Many PIC18 control instructions do not need any argument at all; they either perform an operation that globally affects the device or they operate implicitly on one register. This addressing mode is known as Inherent Addressing. Examples include `SLEEP`, `RESET` and `DAW`.

Other instructions work in a similar way but require an additional explicit argument in the opcode. This is known as Literal Addressing mode because they require some literal value as an argument. Examples include `ADDLW` and `MOVLW`, which respectively, add or move a literal value to the W register. Other examples include `CALL` and `GOTO`, which include a 20-bit program memory address.

5.4.2 DIRECT ADDRESSING

Direct Addressing mode 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 5.3.4 “General**

Purpose Register File”) or a location in the Access Bank (**Section 5.3.3 “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 5.3.2 “Bank Select Register (BSR)”**) are used with the address to determine the complete 12-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 `MOVFF`, include the entire 12-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.

5.4.3 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 to be read or written to. Since the FSRs are themselves located in RAM as Special Function 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 5-5.

EXAMPLE 5-5: 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

12.7 Considerations in Asynchronous Counter Mode

Following a Timer1 interrupt and an update to the TMR1 registers, the Timer1 module uses a falling edge on its clock source to trigger the next register update on the rising edge. If the update is completed after the clock input has fallen, the next rising edge will not be counted.

If the application can reliably update TMR1 before the timer input goes low, no additional action is needed. Otherwise, an adjusted update can be performed

following a later Timer1 increment. This can be done by monitoring TMR1L within the interrupt routine until it increments, and then updating the TMR1H:TMR1L register pair while the clock is low, or one-half of the period of the clock source. Assuming that Timer1 is being used as a Real-Time Clock, the clock source is a 32.768 kHz crystal oscillator; in this case, one-half period of the clock is 15.25 μ s.

The Real-Time Clock application code in Example 12-1 shows a typical ISR for Timer1, as well as the optional code required if the update cannot be done reliably within the required interval.

EXAMPLE 12-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```

RTCinit
    MOVLW    80h                ; Preload TMR1 register pair
    MOVWF    TMR1H              ; for 1 second overflow
    CLRF     TMR1L
    MOVLW    b'00001111'       ; Configure for external clock,
    MOVWF    T1CON              ; Asynchronous operation, external oscillator
    CLRF     secs               ; Initialize timekeeping registers
    CLRF     mins
    MOVLW    .12
    MOVWF    hours
    BSF      PIE1, TMR1IE      ; Enable Timer1 interrupt
    RETURN

RTCisr
                                ; Insert the next 4 lines of code when TMR1
                                ; can not be reliably updated before clock pulse goes low
    BTFSC    TMR1L, 0          ; wait for TMR1L to become clear
    BRA      $-2               ; (may already be clear)
    BTFSS    TMR1L, 0          ; wait for TMR1L to become set
    BRA      $-2               ; TMR1 has just incremented
                                ; If TMR1 update can be completed before clock pulse goes low
                                ; Start ISR here
    BSF      TMR1H, 7          ; Preload for 1 sec overflow
    BCF      PIR1, TMR1IF      ; Clear interrupt flag
    INCF     secs, F           ; Increment seconds
    MOVLW    .59               ; 60 seconds elapsed?
    CPFSGT   secs
    RETURN                                ; No, done
    CLRF     secs              ; Clear seconds
    INCF     mins, F           ; Increment minutes
    MOVLW    .59               ; 60 minutes elapsed?
    CPFSGT   mins
    RETURN                                ; No, done
    CLRF     mins              ; clear minutes
    INCF     hours, F          ; Increment hours
    MOVLW    .23               ; 24 hours elapsed?
    CPFSGT   hours
    RETURN                                ; No, done
    CLRF     hours             ; Reset hours
    RETURN                                ; Done
    
```

PIC18F2455/2550/4455/4550

NOTES:

PIC18F2455/2550/4455/4550

15.1 CCP Module Configuration

Each Capture/Compare/PWM module is associated with a control register (generically, CCPxCON) and a data register (CCPRx). The data register, in turn, is comprised of two 8-bit registers: CCPRxL (low byte) and CCPRxH (high byte). All registers are both readable and writable.

15.1.1 CCP MODULES AND TIMER RESOURCES

The CCP modules utilize Timers 1, 2 or 3, depending on the mode selected. Timer1 and Timer3 are available to modules in Capture or Compare modes, while Timer2 is available for modules in PWM mode.

TABLE 15-1: CCP MODE – TIMER RESOURCE

CCP/ECCP Mode	Timer Resource
Capture Compare PWM	Timer1 or Timer3 Timer1 or Timer3 Timer2

The assignment of a particular timer to a module is determined by the Timer to CCP enable bits in the T3CON register (Register 14-1). Both modules may be active at any given time and may share the same timer resource if they are configured to operate in the same mode (Capture/Compare or PWM) at the same time. The interactions between the two modules are summarized in Figure 15-2. In Timer1 in Asynchronous Counter mode, the capture operation will not work.

15.1.2 CCP2 PIN ASSIGNMENT

The pin assignment for CCP2 (capture input, compare and PWM output) can change, based on device configuration. The CCP2MX Configuration bit determines which pin CCP2 is multiplexed to. By default, it is assigned to RC1 (CCP2MX = 1). If the Configuration bit is cleared, CCP2 is multiplexed with RB3.

Changing the pin assignment of CCP2 does not automatically change any requirements for configuring the port pin. Users must always verify that the appropriate TRIS register is configured correctly for CCP2 operation, regardless of where it is located.

TABLE 15-2: INTERACTIONS BETWEEN CCP1 AND CCP2 FOR TIMER RESOURCES

CCP1 Mode	CCP2 Mode	Interaction
Capture	Capture	Each module can use TMR1 or TMR3 as the time base. The time base can be different for each CCP.
Capture	Compare	CCP2 can be configured for the Special Event Trigger to reset TMR1 or TMR3 (depending upon which time base is used). Automatic A/D conversions on trigger event can also be done. Operation of CCP1 could be affected if it is using the same timer as a time base.
Compare	Capture	CCP1 be configured for the Special Event Trigger to reset TMR1 or TMR3 (depending upon which time base is used). Operation of CCP2 could be affected if it is using the same timer as a time base.
Compare	Compare	Either module can be configured for the Special Event Trigger to reset the time base. Automatic A/D conversions on CCP2 trigger event can be done. Conflicts may occur if both modules are using the same time base.
Capture	PWM ⁽¹⁾	None
Compare	PWM ⁽¹⁾	None
PWM ⁽¹⁾	Capture	None
PWM ⁽¹⁾	Compare	None
PWM ⁽¹⁾	PWM	Both PWMs will have the same frequency and update rate (TMR2 interrupt).

Note 1: Includes standard and Enhanced PWM operation.

17.2.5 USB ADDRESS REGISTER (UADDR)

The USB Address register contains the unique USB address that the peripheral will decode when active. UADDR is reset to 00h when a USB Reset is received, indicated by URSTIF, or when a Reset is received from the microcontroller. The USB address must be written by the microcontroller during the USB setup phase (enumeration) as part of the Microchip USB firmware support.

17.2.6 USB FRAME NUMBER REGISTERS (UFRMH:UFRML)

The Frame Number registers contain the 11-bit frame number. The low-order byte is contained in UFRML, while the three high-order bits are contained in UFRMH. The register pair is updated with the current frame number whenever a SOF token is received. For the microcontroller, these registers are read-only. The Frame Number register is primarily used for isochronous transfers.

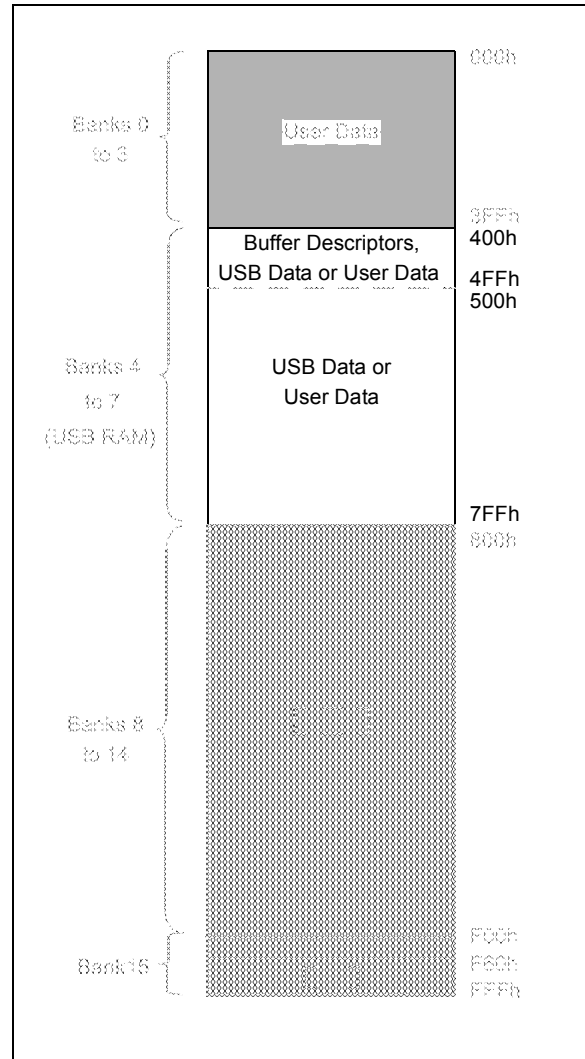
17.3 USB RAM

USB data moves between the microcontroller core and the SIE through a memory space known as the USB RAM. This is a special dual port memory that is mapped into the normal data memory space in Banks 4 through 7 (400h to 7FFh) for a total of 1 Kbyte (Figure 17-5).

Bank 4 (400h through 4FFh) is used specifically for endpoint buffer control, while Banks 5 through 7 are available for USB data. Depending on the type of buffering being used, all but 8 bytes of Bank 4 may also be available for use as USB buffer space.

Although USB RAM is available to the microcontroller as data memory, the sections that are being accessed by the SIE should not be accessed by the microcontroller. A semaphore mechanism is used to determine the access to a particular buffer at any given time. This is discussed in **Section 17.4.1.1 "Buffer Ownership"**.

FIGURE 17-5: IMPLEMENTATION OF USB RAM IN DATA MEMORY SPACE



PIC18F2455/2550/4455/4550

19.4.6 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in SSPCON1 and by setting the SSPEN bit. In Master mode, the SCL and SDA lines are manipulated by the MSSP hardware if the TRIS bits are set.

Master mode operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSP module is disabled. Control of the I²C bus may be taken when the P bit is set or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I²C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options:

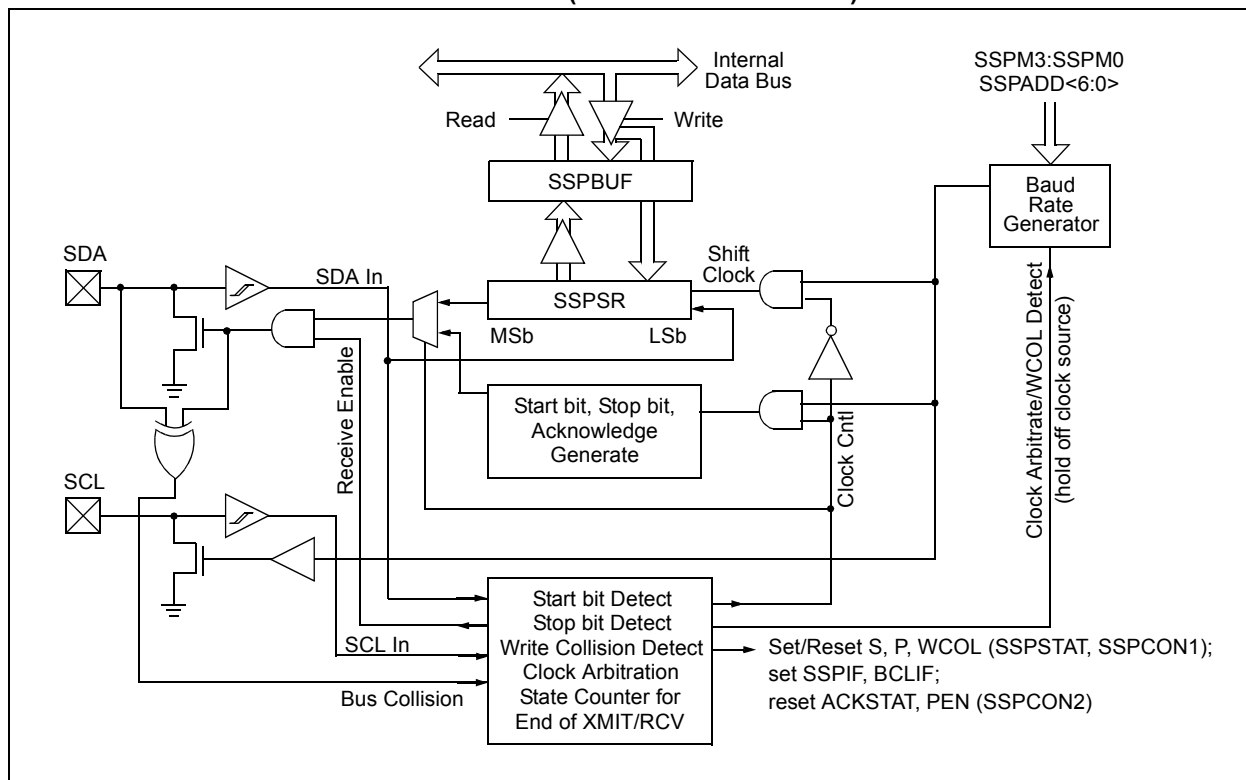
1. Assert a Start condition on SDA and SCL.
2. Assert a Repeated Start condition on SDA and SCL.
3. Write to the SSPBUF register initiating transmission of data/address.
4. Configure the I²C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDA and SCL.

Note: The MSSP module, when configured in I²C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPBUF register to initiate transmission before the Start condition is complete. In this case, the SSPBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPBUF did not occur.

The following events will cause the MSSP Interrupt Flag bit, SSPIF, to be set (and MSSP interrupt, if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received
- Acknowledge transmit
- Repeated Start

FIGURE 19-18: MSSP BLOCK DIAGRAM (I²C™ MASTER MODE)



20.2.5 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. The Break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The Frame Break character is sent whenever the SENDB and TXEN bits (TXSTA<3> and TXSTA<5>) are set while the Transmit Shift Register is loaded with data. Note that the value of data written to TXREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

Note that the data value written to the TXREG for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 20-10 for the timing of the Break character sequence.

20.2.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to set up the Break character.
3. Load the TXREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware. The Sync character now transmits in the preconfigured mode.

When the TXREG becomes empty, as indicated by the TXIF, the next data byte can be written to TXREG.

20.2.6 RECEIVING A BREAK CHARACTER

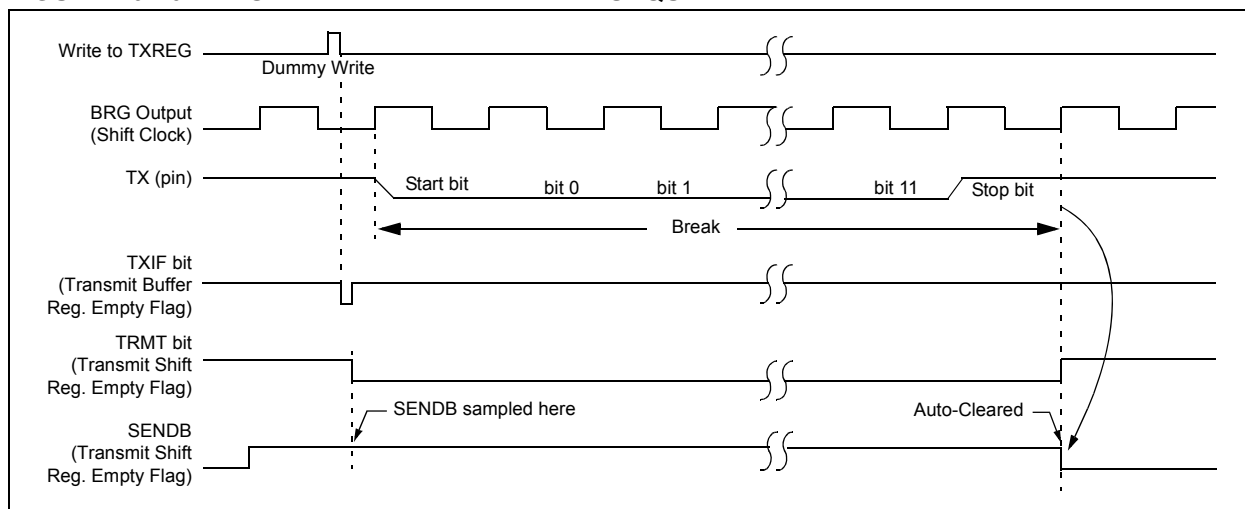
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for Break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 20.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF interrupt and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Rate Detect feature. For both methods, the user can set the ABD bit once the TXIF interrupt is observed.

FIGURE 20-10: SEND BREAK CHARACTER SEQUENCE



PIC18F2455/2550/4455/4550

FIGURE 23-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE

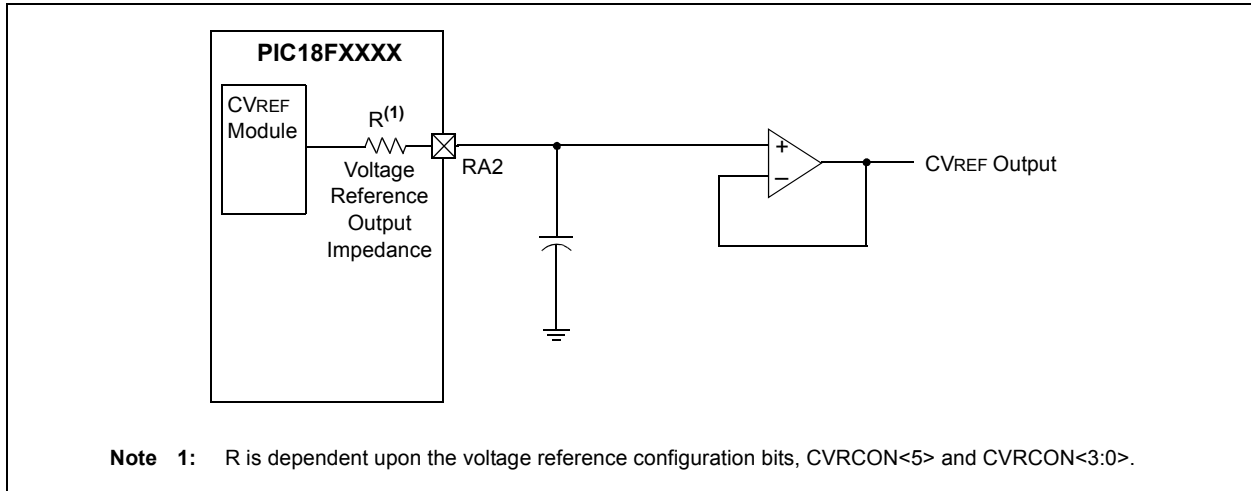


TABLE 23-1: REGISTERS ASSOCIATED WITH COMPARATOR VOLTAGE REFERENCE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	55
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	55
TRISA	—	TRISA6 ⁽¹⁾	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	56

Legend: Shaded cells are not used with the comparator voltage reference.

Note 1: PORTA<6> and its direction and latch bits are individually configured as port pins based on various oscillator modes. When disabled, these bits read as '0'.

PIC18F2455/2550/4455/4550

REGISTER 25-9: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	WRT3 ⁽¹⁾	WRT2	WRT1	WRT0
bit 7							bit 0

Legend:

R = Readable bit

C = Clearable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **WRT3:** Write Protection bit⁽¹⁾

1 = Block 3 (006000-007FFFh) is not write-protected

0 = Block 3 (006000-007FFFh) is write-protected

bit 2 **WRT2:** Write Protection bit

1 = Block 2 (004000-005FFFh) is not write-protected

0 = Block 2 (004000-005FFFh) is write-protected

bit 1 **WRT1:** Write Protection bit

1 = Block 1 (002000-003FFFh) is not write-protected

0 = Block 1 (002000-003FFFh) is write-protected

bit 0 **WRT0:** Write Protection bit

1 = Block 0 (000800-001FFFh) or (001000-001FFFh) is not write-protected

0 = Block 0 (000800-001FFFh) or (001000-001FFFh) is write-protected

Note 1: Unimplemented in PIC18FX455 devices; maintain this bit set.

REGISTER 25-10: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)

R/C-1	R/C-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC ⁽¹⁾	—	—	—	—	—
bit 7							bit 0

Legend:

R = Readable bit

C = Clearable bit

U = Unimplemented bit, read as '0'

-n = Value when device is unprogrammed

u = Unchanged from programmed state

bit 7 **WRTD:** Data EEPROM Write Protection bit

1 = Data EEPROM is not write-protected

0 = Data EEPROM is write-protected

bit 6 **WRTB:** Boot Block Write Protection bit

1 = Boot block (000000-0007FFFh) is not write-protected

0 = Boot block (000000-0007FFFh) is write-protected

bit 5 **WRTC:** Configuration Register Write Protection bit⁽¹⁾

1 = Configuration registers (300000-3000FFFh) are not write-protected

0 = Configuration registers (300000-3000FFFh) are write-protected

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

Note 1: This bit is read-only in normal execution mode; it can be written only in Program mode.

PIC18F2455/2550/4455/4550

FIGURE 25-7: EXTERNAL BLOCK TABLE READ (EBTRx) DISALLOWED

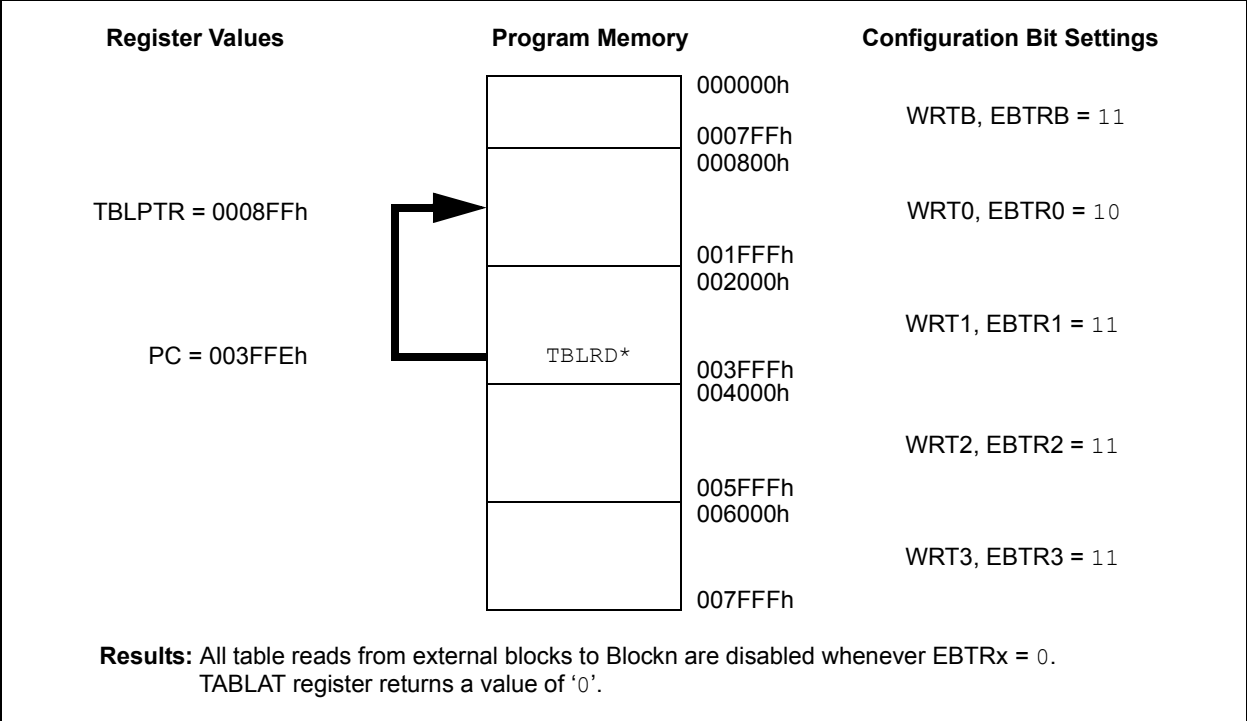
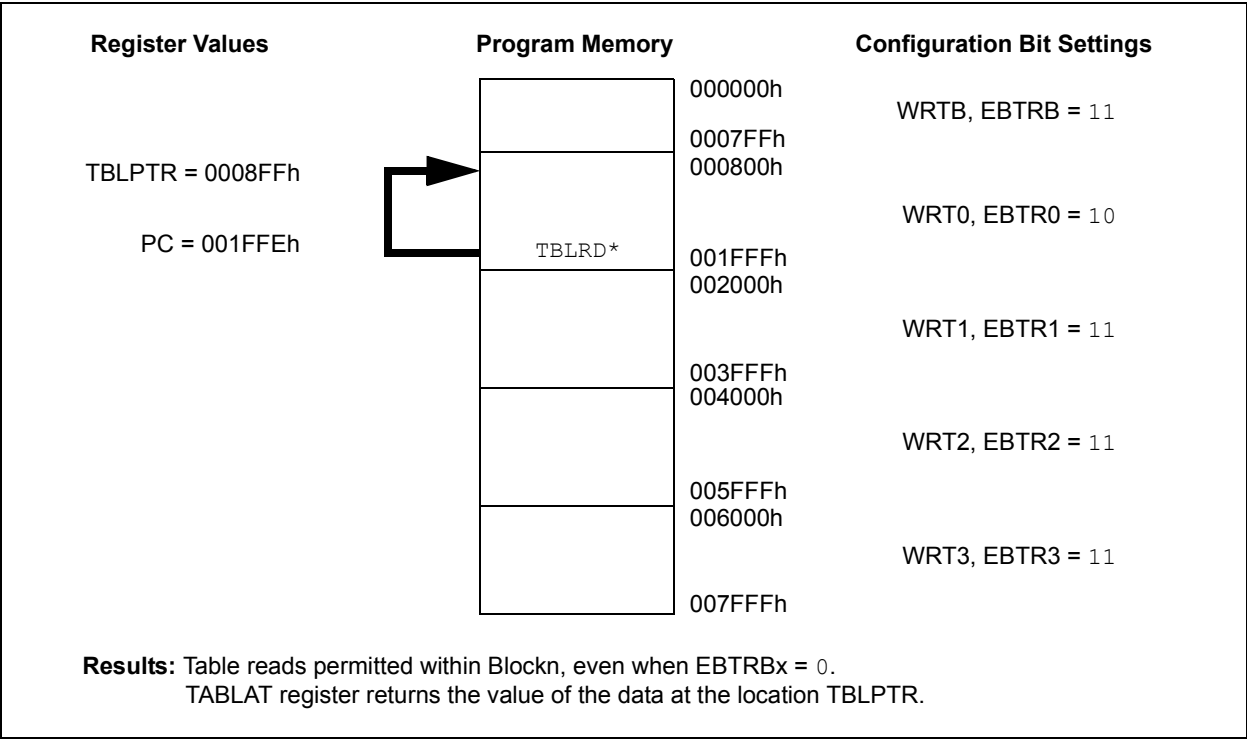


FIGURE 25-8: EXTERNAL BLOCK TABLE READ (EBTRx) ALLOWED



PIC18F2455/2550/4455/4550

TABLE 26-1: OPCODE FIELD DESCRIPTIONS

Field	Description
a	RAM access bit a = 0: RAM location in Access RAM (BSR register is ignored) a = 1: RAM bank is specified by BSR register
bbb	Bit address within an 8-bit file register (0 to 7).
BSR	Bank Select Register. Used to select the current RAM bank.
C, DC, Z, OV, N	ALU Status bits: C arry, D igit Carry, Z ero, O verflow, N egative.
d	Destination select bit d = 0: store result in WREG d = 1: store result in file register f
dest	Destination: either the WREG register or the specified register file location.
f	8-bit register file address (00h to FFh) or 2-bit FSR designator (0h to 3h).
f _s	12-bit register file address (000h to FFFh). This is the source address.
f _d	12-bit register file address (000h to FFFh). This is the destination address.
GIE	Global Interrupt Enable bit.
k	Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value).
label	Label name.
mm	The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions:
*	No change to register (such as TBLPTR with table reads and writes)
*+	Post-Increment register (such as TBLPTR with table reads and writes)
*-	Post-Decrement register (such as TBLPTR with table reads and writes)
++	Pre-Increment register (such as TBLPTR with table reads and writes)
n	The relative address (2's complement number) for relative branch instructions or the direct address for Call/Branch and Return instructions.
PC	Program Counter.
PCL	Program Counter Low Byte.
PCH	Program Counter High Byte.
PCLATH	Program Counter High Byte Latch.
PCLATU	Program Counter Upper Byte Latch.
PD	Power-Down bit.
PRODH	Product of Multiply High Byte.
PRODL	Product of Multiply Low Byte.
s	Fast Call/Return mode select bit s = 0: do not update into/from shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode)
TBLPTR	21-bit Table Pointer (points to a program memory location).
TABLAT	8-bit Table Latch.
TO	Time-out bit.
TOS	Top-of-Stack.
u	Unused or unchanged.
WDT	Watchdog Timer.
WREG	Working register (accumulator).
x	Don't care ('0' or '1'). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
z _s	7-bit offset value for indirect addressing of register files (source).
z _d	7-bit offset value for indirect addressing of register files (destination).
{ }	Optional argument.
[text]	Indicates an indexed address.
(text)	The contents of text.
[expr]<n>	Specifies bit n of the register indicated by the pointer expr.
→	Assigned to.
< >	Register bit field.
∈	In the set of.
<i>italics</i>	User-defined term (font is Courier New).

PIC18F2455/2550/4455/4550

GOTO Unconditional Branch

Syntax: GOTO k

Operands: $0 \leq k \leq 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word ($k<7:0>$)

2nd word ($k<19:8>$)

1110	1111	k_7kkk	$kkkk_0$
1111	$k_{19}kkk$	$kkkk$	$kkkk_8$

Description: GOTO allows an unconditional branch anywhere within the entire 2-Mbyte memory range. The 20-bit value 'k' is loaded into PC<20:1>. GOTO is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: GOTO THERE

After Instruction

PC = Address (THERE)

INCF Increment f

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

Operands: $0 \leq f \leq 255$

$d \in [0,1]$

$a \in [0,1]$

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

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

Encoding:

0010	10da	ffff	ffff
------	------	------	------

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back 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 (default). If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \leq 95$ (5Fh). See **Section 26.2.3 "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: INCF CNT, 1, 0

Before Instruction

CNT = FFh
Z = 0
C = ?
DC = ?

After Instruction

CNT = 00h
Z = 1
C = 1
DC = 1

PIC18F2455/2550/4455/4550

MOVFF Move f to f

Syntax: MOVFF f_s, f_d

Operands: $0 \leq f_s \leq 4095$
 $0 \leq f_d \leq 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1100	ffff	ffff	ffff _s
1111	ffff	ffff	ffff _d

1st word (source)

2nd word (destin.)

Description: The contents of source register ' f_s ' are moved to destination register ' f_d '. Location of source ' f_s ' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination ' f_d ' can also be anywhere from 000h to FFFh. Either source or destination can be W (a useful special situation). MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port). The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVFF REG1, REG2

Before Instruction
 REG1 = 33h
 REG2 = 11h

After Instruction
 REG1 = 33h
 REG2 = 33h

MOVLB Move Literal to Low Nibble in BSR

Syntax: MOVLW k

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow \text{BSR}$

Status Affected: None

Encoding:

0000	0001	kkkk	kkkk
------	------	------	------

Description: The eight-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '0' regardless of the value of k_{7:k₄}.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: MOVLB 5

Before Instruction
 BSR Register = 02h

After Instruction
 BSR Register = 05h

PIC18F2455/2550/4455/4550

DC and AC Characteristics		
Graphs and Tables	407	
DC Characteristics	379	
Power-Down and Supply Current	370	
Supply Voltage	369	
DCFSNZ	333	
DECF	332	
DECFSZ	333	
Dedicated ICD/ICSP Port	311	
Development Support	363	
Device Differences	419	
Device Overview	7	
Features (table)	9	
New Core Features	7	
Other Special Features	8	
Device Reset Timers	49	
Oscillator Start-up Timer (OST)	49	
PLL Lock Time-out	49	
Power-up Timer (PWRT)	49	
Direct Addressing	75	
E		
Effect on Standard PIC MCU Instructions	77, 360	
Electrical Characteristics	367	
Enhanced Capture/Compare/PWM (ECCP)	151	
Associated Registers	164	
Capture and Compare Modes	152	
Capture Mode. <i>See</i> Capture (ECCP Module).		
Outputs and Configuration	152	
Pin Configurations for ECCP1	152	
PWM Mode. <i>See</i> PWM (ECCP Module).		
Standard PWM Mode	152	
Timer Resources	152	
Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART). <i>See</i> EUSART.		
Equations		
A/D Acquisition Time	270	
A/D Minimum Charging Time	270	
Calculating the Minimum Required A/D Acquisition Time	270	
Errata	5	
EUSART		
Asynchronous Mode	253	
12-Bit Break Transmit and Receive	259	
Associated Registers, Receive	257	
Associated Registers, Transmit	255	
Auto-Wake-up on Sync Break Character	258	
Receiver	256	
Setting up 9-Bit Mode with Address Detect	256	
Transmitter	253	
Baud Rate Generator		
Operation in Power-Managed Modes	247	
Baud Rate Generator (BRG)	247	
Associated Registers	248	
Auto-Baud Rate Detect	251	
Baud Rate Error, Calculating	248	
Baud Rates, Asynchronous Modes	249	
High Baud Rate Select (BRGH Bit)	247	
Sampling	247	
Synchronous Master Mode	260	
Associated Registers, Receive	262	
Associated Registers, Transmit	261	
Reception	262	
Transmission	260	
Synchronous Slave Mode	263	
Associated Registers, Receive	264	
Associated Registers, Transmit	263	
Reception	264	
Transmission	263	
Extended Instruction Set	355	
ADDFSR	356	
ADDULNK	356	
and Using MPLAB IDE Tools	362	
CALLW	357	
Considerations for Use	360	
MOVSF	357	
MOVSS	358	
PUSHL	358	
SUBFSR	359	
SUBULNK	359	
Syntax	355	
External Clock Input	26	
F		
Fail-Safe Clock Monitor	291, 306	
Exiting the Operation	306	
Interrupts in Power-Managed Modes	307	
POR or Wake-up from Sleep	307	
WDT During Oscillator Failure	306	
Fast Register Stack	62	
Firmware Instructions	313	
Flash Program Memory	81	
Associated Registers	89	
Control Registers	82	
EECON1 and EECON2	82	
TABLAT (Table Latch) Register	84	
TBLPTR (Table Pointer) Register	84	
Erase Sequence	86	
Erasing	86	
Operation During Code-Protect	89	
Protection Against Spurious Writes	89	
Reading	85	
Table Pointer		
Boundaries Based on Operation	84	
Table Pointer Boundaries	84	
Table Reads and Table Writes	81	
Unexpected Termination of Write	89	
Write Sequence	87	
Write Verify	89	
Writing To	87	
FSCM. <i>See</i> Fail-Safe Clock Monitor.		
G		
GOTO	334	
H		
Hardware Multiplier	97	
Introduction	97	
Operation	97	
Performance Comparison	97	

PIC18F2455/2550/4455/4550

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

To: Technical Publications Manager
RE: Reader Response
From: Name _____
Company _____
Address _____
City/State/ZIP/Country _____
Telephone: (____) _____ - _____ FAX: (____) _____ - _____

Application (optional):

Would you like a reply? __Y__ __N__

Device: PIC18F2455/2550/4455/4550 Literature Number: DS39632E

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this document easy to follow? If not, why?

4. What additions to the document do you think would enhance the structure and subject?

5. What deletions from the document could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?
