



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 "<u>Embedded -</u> <u>Microcontrollers</u>"

Details

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 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/pic18f4520t-i-ml

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong

1.2 Other Special Features

- **Memory Endurance:** The Enhanced Flash cells for both program memory and data EEPROM are rated to last for many thousands of erase/write cycles – up to 100,000 for program memory and 1,000,000 for EEPROM. Data retention without refresh is conservatively estimated to be greater than 40 years.
- Self-Programmability: These devices can write to their own program memory spaces under internal software control. By using a bootloader routine located in the protected Boot Block at the top of program memory, it becomes possible to create an application that can update itself in the field.
- Extended Instruction Set: The PIC18F2420/ 2520/4420/4520 family introduces an optional extension to the PIC18 instruction set, which adds 8 new instructions and an Indexed Addressing mode. This extension, enabled as a device configuration option, has been specifically designed to optimize re-entrant application code originally developed in high-level languages, such as C.
- Enhanced CCP Module: In PWM mode, this module provides 1, 2 or 4 modulated outputs for controlling half-bridge and full-bridge drivers. Other features include auto-shutdown, for disabling PWM outputs on interrupt, or other select conditions, and auto-restart to reactivate outputs once the condition has cleared.
- Enhanced Addressable USART: This serial communication module is capable of standard RS-232 operation and provides support for the LIN bus protocol. Other enhancements include automatic baud rate detection and a 16-bit Baud Rate Generator for improved resolution. When the microcontroller is using the internal oscillator block, the EUSART provides stable operation for applications that talk to the outside world without using an external crystal (or its accompanying power requirement).
- **10-Bit A/D Converter:** This module incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period and thus, reducing code overhead.
- Extended Watchdog Timer (WDT): This enhanced version incorporates a 16-bit prescaler, allowing an extended time-out range that is stable across operating voltage and temperature. See Section 26.0 "Electrical Characteristics" for time-out periods.

1.3 Details on Individual Family Members

Devices in the PIC18F2420/2520/4420/4520 family are available in 28-pin and 40/44-pin packages. Block diagrams for the two groups are shown in Figure 1-1 and Figure 1-2.

The devices are differentiated from each other in five ways:

- 1. Flash program memory (16 Kbytes for PIC18F2420/4420 devices and 32 Kbytes for PIC18F2520/4520 devices).
- A/D channels (10 for 28-pin devices, 13 for 40/44-pin devices).
- 3. I/O ports (3 bidirectional ports on 28-pin devices, 5 bidirectional ports on 40/44-pin devices).
- CCP and Enhanced CCP implementation (28-pin devices have 2 standard CCP modules, 40/44-pin devices have one standard CCP module and one ECCP module).
- 5. Parallel Slave Port (present only on 40/44-pin devices).

All other features for devices in this family are identical. These are summarized in Table 1-1.

The pinouts for all devices are listed in Table 1-2 and Table 1-3.

Like all Microchip PIC18 devices, members of the PIC18F2420/2520/4420/4520 family are available as both standard and low-voltage devices. Standard devices with Enhanced Flash memory, designated with an "F" in the part number (such as PIC18F2420), accommodate an operating VDD range of 4.2V to 5.5V. Low-voltage parts, designated by "LF" (such as PIC18LF2420), function over an extended VDD range of 2.0V to 5.5V.







3.2.3 RC_RUN MODE

In RC_RUN mode, the CPU and peripherals are clocked from the internal oscillator block using the INTOSC multiplexer. In this mode, the primary clock is shut down. When using the INTRC source, this mode provides the best power conservation of all the Run modes while still executing code. It works well for user applications which are not highly timing sensitive or do not require high-speed clocks at all times.

If the primary clock source is the internal oscillator block (either INTRC or INTOSC), there are no distinguishable differences between PRI_RUN and RC_RUN modes during execution. However, a clock switch delay will occur during entry to and exit from RC_RUN mode. Therefore, if the primary clock source is the internal oscillator block, the use of RC_RUN mode is not recommended. This mode is entered by setting the SCS1 bit to '1'. Although it is ignored, it is recommended that the SCS0 bit also be cleared; this is to maintain software compatibility with future devices. When the clock source is switched to the INTOSC multiplexer (see Figure 3-3), the primary oscillator is shut down and the OSTS bit is cleared. The IRCF bits may be modified at any time to immediately change the clock speed.

Note: Caution should be used when modifying a single IRCF bit. If VDD is less than 3V, it is possible to select a higher clock speed than is supported by the low VDD. Improper device operation may result if the VDD/Fosc specifications are violated.

4.6 Reset State of Registers

Most registers are unaffected by a Reset. Their status is unknown on POR and unchanged by all other Resets. The other registers are forced to a "Reset state" depending on the type of Reset that occurred.

Most registers are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. Status bits from the RCON register, \overline{RI} , \overline{TO} , \overline{PD} , \overline{POR} and \overline{BOR} , are set or cleared differently in different Reset situations, as indicated in Table 4-3. These bits are used in software to determine the nature of the Reset. Table 4-4 describes the Reset states for all of the Special Function Registers. These are categorized by Power-on and Brown-out Resets, Master Clear and WDT Resets and WDT wake-ups.

TABLE 4-3:	STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION
	FOR RCON REGISTER

Condition	Program	RCON Register					STKPTR Register	
Condition	Counter	RI	то	PD	POR	BOR	STKFUL	STKUNF
Power-on Reset	0000h	1	1	1	0	0	0	0
RESET Instruction	0000h	0	u	u	u	u	u	u
Brown-out Reset	0000h	1	1	1	u	0	u	u
MCLR Reset during Power-Managed Run Modes	0000h	u	1	u	u	u	u	u
MCLR Reset during Power-Managed Idle Modes and Sleep Mode	0000h	u	1	0	u	u	u	u
WDT Time-out during Full Power or Power-Managed Run Mode	0000h	u	0	u	u	u	u	u
MCLR Reset during Full-Power Execution	0000h	u	u	u	u	u	u	u
Stack Full Reset (STVREN = 1)	0000h	u	u	u	u	u	1	u
Stack Underflow Reset (STVREN = 1)	0000h	u	u	u	u	u	u	1
Stack Underflow Error (not an actual Reset, STVREN = 0)	0000h	u	u	u	u	u	u	1
WDT Time-out during Power-Managed Idle or Sleep Modes	PC + 2	u	0	0	u	u	u	u
Interrupt Exit from Power-Managed Modes	PC + 2 ⁽¹⁾	u	u	0	u	u	u	u

Legend: u = unchanged

Note 1: When the wake-up is due to an interrupt and the GIEH or GIEL bits are set, the PC is loaded with the interrupt vector (008h or 0018h).

6.0 FLASH PROGRAM MEMORY

The Flash program memory is readable, writable and erasable during normal operation over the entire VDD range.

A read from program memory is executed on one byte at a time. A write to program memory is executed on blocks of 32 bytes at a time. Program memory is erased in blocks of 64 bytes at a time. A bulk erase operation may not be issued from user code.

Writing or erasing program memory will cease instruction fetches until the operation is complete. The program memory cannot be accessed during the write or erase, therefore, code cannot execute. An internal programming timer terminates program memory writes and erases.

A value written to program memory does not need to be a valid instruction. Executing a program memory location that forms an invalid instruction results in a NOP.

6.1 Table Reads and Table Writes

In order to read and write program memory, there are two operations that allow the processor to move bytes between the program memory space and the data RAM:

- Table Read (TBLRD)
- Table Write (TBLWT)

The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and places it into the data RAM space. Figure 6-1 shows the operation of a table read with program memory and data RAM.

Table write operations store data from the data memory space into holding registers in program memory. The procedure to write the contents of the holding registers into program memory is detailed in **Section 6.5** "**Writing to Flash Program Memory**". Figure 6-2 shows the operation of a table write with program memory and data RAM.

Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions will need to be word aligned.

FIGURE 6-1:





7.3 Reading the Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADR register, clear the EEPGD control bit (EECON1<7>) and then set control bit, RD (EECON1<0>). The data is available on the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation, or until it is written to by the user (during a write operation).

The basic process is shown in Example 7-1.

7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADR register and the data written to the EEDATA register. The sequence in Example 7-2 must be followed to initiate the write cycle.

The write will not begin if this sequence is not exactly followed (write 55h to EECON2, write 0AAh to EECON2, then set WR bit) for each byte. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in EECON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code execution (i.e., runaway programs). The WREN bit should be kept clear at all times, except when updating the EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, EECON1, EEADR and EEDATA cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. Both WR and WREN cannot be set with the same instruction.

At the completion of the write cycle, the WR bit is cleared in hardware and the EEPROM Interrupt Flag bit, EEIF, is set. The user may either enable this interrupt or poll this bit. EEIF must be cleared by software.

7.5 Write Verify

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

EXAMPLE 7-1: DATA EEPROM READ

MOVLW	DATA_EE_ADI	DR ;	
MOVWF	EEADR	;	Data Memory Address to read
BCF	EECON1, EEF	GD ;	Point to DATA memory
BCF	EECON1, CFG	ss ;	Access EEPROM
BSF	EECON1, RD	;	EEPROM Read
MOVF	EEDATA, W	;	W = EEDATA

EXAMPLE 7-2: DATA EEPROM WRITE

	MOVLW	DATA_EE_ADDR	i
	MOVWF	EEADR	; Data Memory Address to write
	MOVLW	DATA_EE_DATA	i
	MOVWF	EEDATA	; Data Memory Value to write
	BCF	EECON1, EEPGD	; Point to DATA memory
	BCF	EECON1, CFGS	; Access EEPROM
	BSF	EECON1, WREN	; Enable writes
	BCF	INTCON, GIE	; Disable Interrupts
	MOVLW	55h	;
Required	MOVWF	EECON2	; Write 55h
Sequence	MOVLW	0AAh	;
	MOVWF	EECON2	; Write OAAh
	BSF	EECON1, WR	; Set WR bit to begin write
	BSF	INTCON, GIE	; Enable Interrupts
			; User code execution
	BCF	EECON1, WREN	; Disable writes on write complete (EEIF set)

PIC18F2420/2520/4420/4520

REGISTER 9-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

]
R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	CMIF	—	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF
bit 7							bit 0
Legend:							
R = Readable	e bit	W = Writable	bit	U = Unimpler	nented bit, rea	d as '0'	
-n = Value at	POR	'1' = Bit is set		'0' = Bit is cle	ared	x = Bit is unkr	nown
bit 7	OSCFIF: Osc	illator Fail Inter	rupt Flag bit				
	1 = Device os	scillator failed,	clock input has	changed to IN	ITOSC (must b	be cleared in so	ftware)
	0 = Device cl	ock operating					
bit 6	CMIF: Compa	arator Interrupt	Flag bit				
	1 = Compara	tor input has cl	nanged (must l	be cleared in so	oftware)		
		tor input has no	ot changed				
DIT 5	Unimplement	ted: Read as ()'				
bit 4	EEIF: Data El	=PROM/Flash	Write Operatio	n Interrupt Flag	g bit		
	1 = The write	operation is co	omplete (must	be cleared in s	oftware)		
hit 2	BCLIE: Buc C		nt Elog hit	nas not been s			
bit 5	1 = A bus col	lision occurred	(must be clea	red in software	`		
	0 = No bus co	ollision occurre	d)		
bit 2	HLVDIF: High	/Low-Voltage	Detect Interrup	t Flag bit			
	1 = A high/low-voltage condition occurred (direction determined by VDIRMAG bit. HI VDCON<7>)						
	0 = A high/low-voltage condition has not occurred						
bit 1	TMR3IF: TMF	R3 Overflow Int	errupt Flag bit				
	1 = TMR3 register overflowed (must be cleared in software)						
	0 = TMR3 reg	gister did not ov	verflow				
bit 0	CCP2IF: CCF	2 Interrupt Flag	g bit				
	Capture mode	<u>e:</u>			<i></i>		
	1 = A TMR1	register capture	e occurred (mu	ist be cleared in	n software)		
		register captu	re occurred				
	1 = A TMR1	<u>register compa</u>	re match occu	rred (must be c	leared in softw	vare)	
	0 = No TMR1	register comp	are match occ	urred			
	PWM mode:						
	Unused in this	s mode.					

9.4 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are two Peripheral Interrupt Priority registers (IPR1 and IPR2). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 9-8: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

PSPIP ⁽¹⁾ ADIP RCIP TXIP SSPIP CCP1IP TMR2IP TMR2	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
bit 7	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP
	bit 7							bit 0

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

bit 7	<pre>PSPIP: Parallel Slave Port Read/Write Interrupt Priority bit⁽¹⁾ 1 = High priority 0 = Low priority</pre>
bit 6	ADIP: A/D Converter Interrupt Priority bit 1 = High priority 0 = Low priority
bit 5	RCIP: EUSART Receive Interrupt Priority bit 1 = High priority 0 = Low priority
bit 4	TXIP: EUSART Transmit Interrupt Priority bit1 = High priority0 = Low priority
bit 3	SSPIP: Master Synchronous Serial Port Interrupt Priority bit 1 = High priority 0 = Low priority
bit 2	CCP1IP: CCP1 Interrupt Priority bit 1 = High priority 0 = Low priority
bit 1	TMR2IP: TMR2 to PR2 Match Interrupt Priority bit 1 = High priority 0 = Low priority
bit 0	TMR1IP: TMR1 Overflow Interrupt Priority bit 1 = High priority 0 = Low priority



14.1 Timer3 Operation

Timer3 can operate in one of three modes:

- Timer
- Synchronous Counter
- Asynchronous Counter

The operating mode is determined by the clock select bit, TMR3CS (T3CON<1>). When TMR3CS is cleared (= 0), Timer3 increments on every internal instruction cycle (Fosc/4). When the bit is set, Timer3 increments on every rising edge of the Timer1 external clock input or the Timer1 oscillator, if enabled.

As with Timer1, the RC1/T1OSI and RC0/T1OSO/ T13CKI pins become inputs when the Timer1 oscillator is enabled. This means the values of TRISC<1:0> are ignored and the pins are read as '0'.



FIGURE 14-2: TIMER3 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



17.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPCON1<5:0> and SSPSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCK is the clock output)
- · Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)

EVAMDI E 17-1.

· Slave Select mode (Slave mode only)

The MSSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPBUF holds the data that was written to the SSPSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPBUF register. Then, the Buffer Full detect bit, BF (SSPSTAT<0>) and the interrupt flag bit, SSPIF, are set. This double-buffering of the received data (SSPBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPBUF register during transmission/reception of data

will be ignored and the write collision detect bit, WCOL (SSPCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPBUF should be read before the next byte of data to transfer is written to the SSPBUF. The Buffer Full bit, BF (SSPSTAT<0>), indicates when SSPBUF has been loaded with the received data (transmission is complete). When the SSPBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSP interrupt is used to determine when the transmission/reception has completed. The SSPBUF must be read and/or written. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 17-1 shows the loading of the SSPBUF (SSPSR) for data transmission.

The SSPSR is not directly readable or writable and can only be accessed by addressing the SSPBUF register. Additionally, the MSSP Status register (SSPSTAT) indicates the various status conditions.

Note: The SSPBUF register cannot be used with read-modify-write instructions such as BCF, BTFSC and COMF, etc.

		. LOADING	
LOOP	BTFSS	SSPSTAT, BF	;Has data been received (transmit complete)?
	BRA	LOOP	;No
	MOVF	SSPBUF, W	;WREG reg = contents of SSPBUF
	MOVWF	RXDATA	;Save in user RAM, if data is meaningful
	MOVF	TXDATA, W	;W reg = contents of TXDATA
	MOVWF	SSPBUF	;New data to xmit

I OADING THE SODDIE (SODO) DECISTED

Note:	To av of the	void lost o SSPBU	data in F must	Master i be perfo	node, a rmed to	read clear
	the	Buffer	Full	(BF)	detect	bit
	(SSF	STAT<0>	>)	betwee	n	each
	trans	mission.				

17.3.3 ENABLING SPI I/O

To enable the serial port, MSSP Enable bit, SSPEN (SSPCON1<5>), must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, reinitialize the SSPCON registers and then set the SSPEN bit. This configures the SDI, SDO, SCK and SS pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

- · SDI is automatically controlled by the SPI module
- SDO must have TRISC<5> bit cleared
- SCK (Master mode) must have TRISC<3> bit cleared
- SCK (Slave mode) must have TRISC<3> bit set
- SS must have TRISA<5> bit set

Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

17.3.4 TYPICAL CONNECTION

Figure 17-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCK signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- · Master sends data Slave sends dummy data
- · Master sends data Slave sends data
- · Master sends dummy data Slave sends data



FIGURE 17-2: SPI MASTER/SLAVE CONNECTION

17.4.8 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPCON2<0>). If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL is high is the Start condition and causes the S bit (SSPSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPCON2<0>) will be automatically cleared by hardware; the Baud Rate Generator is suspended, leaving the SDA line held low and the Start condition is complete.

Note: If, at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

17.4.8.1 WCOL Status Flag

If the user writes the SSPBUF when a Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note: Because queueing of events is not allowed, writing to the lower 5 bits of SSPCON2 is disabled until the Start condition is complete.



FIGURE 17-19: FIRST START BIT TIMING

EUSART ASYNCHRONOUS 18.2.2 RECEIVER

The receiver block diagram is shown in Figure 18-6. The data is received on the RX pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at Fosc. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

- Initialize the SPBRGH:SPBRG registers for the 1. appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
- 2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
- 3. If interrupts are desired, set enable bit, RCIE.
- If 9-bit reception is desired, set bit, RX9. 4
- 5. Enable the reception by setting bit, CREN.
- 6. Flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCIE, was set.
- 7. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
- 8. Read the 8-bit received data by reading the RCREG register.
- 9. If any error occurred, clear the error by clearing enable bit, CREN.
- 10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 18-6:

SETTING UP 9-BIT MODE WITH 18.2.3 ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

- Initialize the SPBRGH:SPBRG registers for the 1. appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
- 2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
- If interrupts are required, set the RCEN bit and 3. select the desired priority level with the RCIP bit.
- Set the RX9 bit to enable 9-bit reception. 4.
- 5. Set the ADDEN bit to enable address detect.
- Enable reception by setting the CREN bit. 6.
- 7. The RCIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCIE and GIE bits are set.
- Read the RCSTA register to determine if any 8. error occurred during reception, as well as read bit 9 of data (if applicable).
- Read RCREG to determine if the device is being addressed.
- 10. If any error occurred, clear the CREN bit.
- 11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.



19.1 A/D Acquisition Requirements

For the A/D Converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 19-3. The source impedance (Rs) and the internal sampling switch (Rss) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (Rss) impedance varies over the device voltage (VDD). The source impedance affects the offset voltage at the analog input (due to pin leakage current). The maximum recommended impedance for analog sources is 2.5 k Ω . After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

Note: When the conversion is started, the holding capacitor is disconnected from the input pin.

EQUATION 19-1: ACQUISITION TIME

TACQ = Amplifier Settling Time + Holding Capacitor Charging Time + Temperature Coefficient = TAMP + TC + TCOFF

EQUATION 19-2: A/D MINIMUM CHARGING TIME

VHOLD	=	$(\text{VREF} - (\text{VREF}/2048)) \bullet (1 - e^{(-\text{TC/CHOLD}(\text{Ric} + \text{Rss} + \text{Rs}))})$
or		
TC	=	-(CHOLD)(RIC + RSS + RS) ln(1/2048)

EQUATION 19-3: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

TACQ	=	TAMP + TC + TCOFF					
TAMP	=	0.2 µs					
TCOFF	=	(Temp – 25°C)(0.02 μs/°C) (85°C – 25°C)(0.02 μs/°C) 1.2 μs					
Tempera	Temperature coefficient is only required for temperatures > 25°C. Below 25°C, TCOFF = 0 μ s.						
ТС	=	-(ChOLD)(RIC + Rss + Rs) $\ln(1/2047) \ \mu s$ -(25 pF) (1 k Ω + 2 k Ω + 2.5 k Ω) ln(0.0004883) μs 1.05 μs					
TACQ	=	$0.2 \ \mu s + 1 \ \mu s + 1.2 \ \mu s$ 2.4 \ \ \ \ \ u s					

To calculate the minimum acquisition time, Equation 19-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 19-3 shows the calculation of the minimum required acquisition time TACQ. This calculation is based on the following application system assumptions:

CHOLD	=	25 pF
Rs	=	2.5 kΩ
Conversion Error	\leq	1/2 LSb
Vdd	=	$5V ightarrow Rss$ = 2 k Ω
Temperature	=	85°C (system max.)

24.0 INSTRUCTION SET SUMMARY

PIC18F2420/2520/4420/4520 devices incorporate the standard set of 75 PIC18 core instructions, as well as an extended set of 8 new instructions, for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

24.1 Standard Instruction Set

The standard PIC18 instruction set adds many enhancements to the previous PIC[®] MCU instruction sets, while maintaining an easy migration from these PIC MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- · Byte-oriented operations
- **Bit-oriented** operations
- · Literal operations
- Control operations

The PIC18 instruction set summary in Table 24-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 24-1 shows the opcode field descriptions.

Most byte-oriented instructions have three operands:

- 1. The file register (specified by 'f')
- 2. The destination of the result (specified by 'd')
- 3. The accessed memory (specified by 'a')

The file register designator 'f' specifies which file register is to be used by the instruction. The destination designator 'd' specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the WREG register. If 'd' is one, the result is placed in the file register specified in the instruction.

All bit-oriented instructions have three operands:

- 1. The file register (specified by 'f')
- 2. The bit in the file register (specified by 'b')
- 3. The accessed memory (specified by 'a')

The bit field designator 'b' selects the number of the bit affected by the operation, while the file register designator 'f' represents the number of the file in which the bit is located. The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by 'k')
- The desired FSR register to load the literal value into (specified by 'f')
- No operand required (specified by '—')

The **control** instructions may use some of the following operands:

- A program memory address (specified by 'n')
- The mode of the CALL or RETURN instructions (specified by 's')
- The mode of the table read and table write instructions (specified by 'm')
- No operand required (specified by '—')

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the 4 MSbs are '1's. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles, with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μ s. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μ s. Two-word branch instructions (if true) would take 3 μ s.

Figure 24-1 shows the general formats that the instructions can have. All examples use the convention 'nnh' to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 24-2, lists the standard instructions recognized by the Microchip Assembler (MPASMTM).

Section 24.1.1 "Standard Instruction Set" provides a description of each instruction.

PIC18F2420/2520/4420/4520

POF	0	Рор Тор	Pop Top of Return Stack					
Syntax:		POP	POP					
Oper	ands:	None	None					
Oper	ation:	$(TOS) \rightarrow b$	it bucket					
Statu	is Affected:	None	None					
Enco	oding:	0000	0000	000	0	0110		
Description:		The TOS v stack and i then becor was pushe This instru the user to stack to inc	The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack. This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.					
Word	ds:	1						
Cycle	es:	1						
QC	ycle Activity:							
	Q1	Q2	Q3		Q4			
	Decode	No operation	POP 1 valu	TOS le	op	No peration		
Example:		POP GOTO	NEW					
Before Instructio TOS Stack (1 le		tion evel down)	= C = C)031A2)14332	2h 2h			
After Instruction TOS PC		n	= C = N)14332 NEW	2h			

PUS	H	Push Top	Push Top of Return Stack				
Synta	ax:	PUSH	PUSH				
Oper	ands:	None	None				
Oper	ation:	$(PC + 2) \rightarrow$	$(PC + 2) \rightarrow TOS$				
Statu	s Affected:	None	None				
Enco	ding:	0000	0000	000	0	0101	
Desc	ription:	The PC + 2 the return s value is pus This instruc software sta then pushin	The PC + 2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows implementing a software stack by modifying TOS and then pushing it onto the return stack.				
Word	ls:	1					
Cycle	es:	1					
Q Cycle Activity:							
	Q1	Q2	Q	3		Q4	
	Decode	PUSH PC + 2 onto return stack	No operation		No operation		
<u>Exan</u>	nple:	PUSH					
Before Instructio TOS PC		tion	= =	345Ah 0124h			
	After Instructio PC TOS Stack (1	on level down)	= = =	0126h 0126h 345Ah			





TABLE 26-14: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param No.	Symbol	Characteristi	Min	Max	Units	Conditions	
70	TssL2scH, TssL2scL	$\overline{SS} \downarrow$ to SCK \downarrow or SCK \uparrow Input	Тсү	—	ns		
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input t	20	_	ns		
73A	Tb2b	Last Clock Edge of Byte 1 to th of Byte 2	1.5 Tcy + 40	_	ns	(Note 2)	
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	-	ns	
75	TdoR	SDO Data Output Rise Time PIC18FXXXX		—	25	ns	
			PIC18LFXXXX		45	ns	VDD = 2.0V
76	TdoF	SDO Data Output Fall Time			25	ns	
78	TscR	scR SCK Output Rise Time	PIC18FXXXX	—	25	ns	
		(Master mode)	PIC18LFXXXX	—	45	ns	VDD = 2.0V
79	TscF	SCK Output Fall Time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	scH2doV, SDO Data Output Valid after			50	ns	
		TscL2doV SCK Edge		PIC18LFXXXX	_	100	ns

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

28.2 Package Details

The following sections give the technical details of the packages.

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

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



	INCHES				
Dimensior	n Limits	MIN	NOM	MAX	
Number of Pins	Ν	28			
Pitch	е	.100 BSC			
Top to Seating Plane	А	-	-	.200	
Molded Package Thickness	A2	.120	.135	.150	
Base to Seating Plane	A1	.015	-	-	
Shoulder to Shoulder Width	E	.290	.310	.335	
Molded Package Width	E1	.240	.285	.295	
Overall Length	D	1.345	1.365	1.400	
Tip to Seating Plane	L	.110	.130	.150	
Lead Thickness	с	.008	.010	.015	
Upper Lead Width	b1	.040	.050	.070	
Lower Lead Width	b	.014	.018	.022	
Overall Row Spacing §	eB	-	-	.430	

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.

2. § Significant Characteristic.

3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.

4. Dimensioning and tolerancing per ASME Y14.5M.

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

Microchip Technology Drawing C04-070B

PIC18F2420/2520/4420/4520

NOTES: