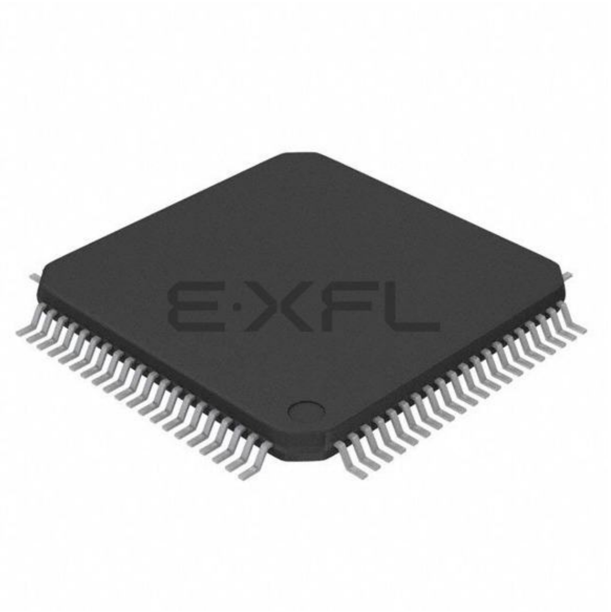
Microchip Technology - PIC18F8720-I/PT Datasheet



Welcome to E-XFL.COM

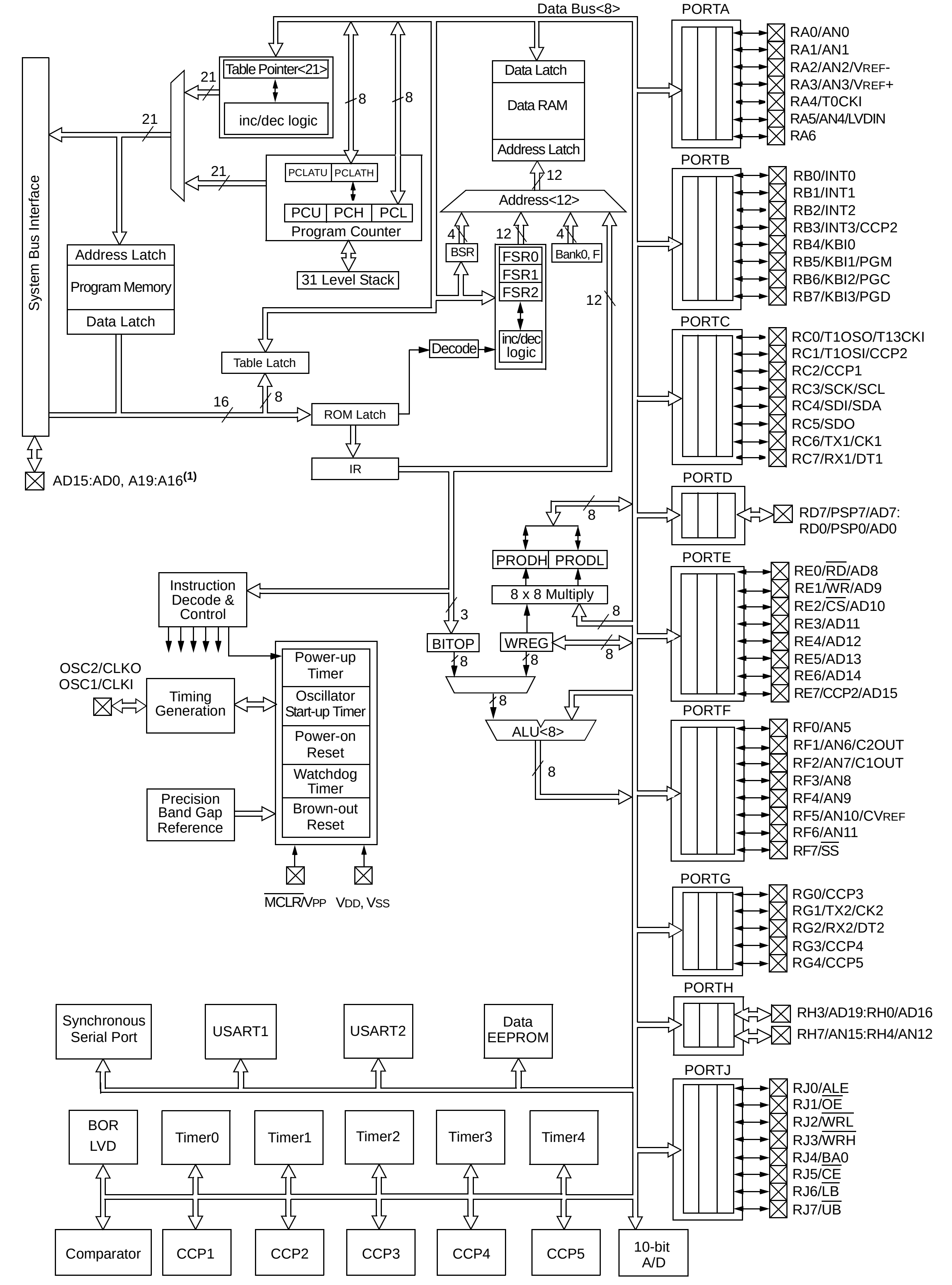
What is "Embedded - Microcontrollers"?

"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	25MHz
Connectivity	EBI/EMI, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	68
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.75K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-TQFP
Supplier Device Package	80-TQFP (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f8720-i-pt

FIGURE 1-2: PIC18F8X20 BLOCK DIAGRAM



Note 1: External memory interface pins are physically multiplexed with PORTD (AD7:AD0), PORTE (AD15:AD8) and PORTH (A19:A16).

TABLE 1-2: PIC18FXX20 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	PIC18F6X20	PIC18F8X20			
					PORTA is a bidirectional I/O port.
RA0/AN0	24	30			
RA0			I/O	TTL	Digital I/O.
AN0			I	Analog	Analog input 0.
RA1/AN1	23	29			
RA1			I/O	TTL	Digital I/O.
AN1			I	Analog	Analog input 1.
RA2/AN2/VREF-	22	28			
RA2			I/O	TTL	Digital I/O.
AN2			I	Analog	Analog input 2.
VREF-			I	Analog	A/D reference voltage (Low) input.
RA3/AN3/VREF+	21	27			
RA3			I/O	TTL	Digital I/O.
AN3			I	Analog	Analog input 3.
VREF+			I	Analog	A/D reference voltage (High) input.
RA4/T0CKI	28	34			
RA4			I/O	ST/OD	Digital I/O – Open-drain when configured as output.
T0CKI			I	ST	Timer0 external clock input.
RA5/AN4/LVDIN	27	33			
RA5			I/O	TTL	Digital I/O.
AN4			I	Analog	Analog input 4.
LVDIN			I	Analog	Low-Voltage Detect input.
RA6					See the OSC2/CLKO/RA6 pin.

Legend:	TTL	= TTL compatible input		CMOS	=	CMOS compatible input or output
	ST	= Schmitt Trigger input with CMOS levels		Analog	=	Analog input
	I	= Input		O	=	Output
	P	= Power		OD	=	Open-Drain (no P diode to VDD)

Note 1: Alternate assignment for CCP2 when CCP2MX is not selected (all operating modes except Microcontroller).

2: Default assignment when CCP2MX is set.

3: External memory interface functions are only available on PIC18F8X20 devices.

4: CCP2 is multiplexed with this pin by default when configured in Microcontroller mode. Otherwise, it is multiplexed with either RB3 or RC1.

5: PORTH and PORTJ are only available on PIC18F8X20 (80-pin) devices.

6: AVDD must be connected to a positive supply and AVSS must be connected to a ground reference for proper operation of the part in user or ICSP modes. See parameter D001A for details.

TABLE 3-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset `RESET` Instruction Stack Resets	Wake-up via WDT or Interrupt
PORTJ	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTH	PIC18F6X20	PIC18F8X20	`0000 xxxx`	`0000 uuuu`	`uuuu uuuu`
PORTG	PIC18F6X20	PIC18F8X20	`---x xxxx`	`uuuu uuuu`	`---u uuuu`
PORTF	PIC18F6X20	PIC18F8X20	`x000 0000`	`u000 0000`	`u000 0000`
PORTE	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTD	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTC	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTB	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
PORTA(5,6)	PIC18F6X20	PIC18F8X20	`-x0x 0000`(5)	`-u0u 0000`(5)	`-uuu uuuu`(5)
TMR4	PIC18F6X20	PIC18F8X20	`0000 0000`	`0000 0000`	`uuuu uuuu`
PR4	PIC18F6X20	PIC18F8X20	`1111 1111`	`1111 1111`	`uuuu uuuu`
T4CON	PIC18F6X20	PIC18F8X20	`-000 0000`	`-000 0000`	`-uuu uuuu`
CCPR4H	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
CCPR4L	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
CCP4CON	PIC18F6X20	PIC18F8X20	`0000 0000`	`0000 0000`	`uuuu uuuu`
CCPR5H	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
CCPR5L	PIC18F6X20	PIC18F8X20	`xxxx xxxx`	`uuuu uuuu`	`uuuu uuuu`
CCP5CON	PIC18F6X20	PIC18F8X20	`0000 0000`	`0000 0000`	`uuuu uuuu`
SPBRG2	PIC18F6X20	PIC18F8X20	`0000 0000`	`0000 0000`	`uuuu uuuu`
RCREG2	PIC18F6X20	PIC18F8X20	`0000 0000`	`0000 0000`	`uuuu uuuu`
TXREG2	PIC18F6X20	PIC18F8X20	`0000 0000`	`0000 0000`	`uuuu uuuu`
TXSTA2	PIC18F6X20	PIC18F8X20	`0000 -010`	`0000 -010`	`uuuu -uuu`
RCSTA2	PIC18F6X20	PIC18F8X20	`0000 000x`	`0000 000x`	`uuuu uuuu`

Legend: `u` = unchanged, `x` = unknown, - = unimplemented bit, read as ‘`0`’, `q` = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 3-2 for Reset value for specific condition.

5: Bit 6 of PORTA, LATA and TRISA are enabled in ECIO and RCIO Oscillator modes only. In all other oscillator modes, they are disabled and read ‘`0`’.

6: Bit 6 of PORTA, LATA and TRISA are not available on all devices. When unimplemented, they are read ‘`0`’.

TABLE 4-2: SPECIAL FUNCTION REGISTER MAP (CONTINUED)

Address	Name	Address	Name	Address	Name	Address	Name
F7Fh	—[1]	F5Fh	—[1]	F3Fh	—[1]	F1Fh	—[1]
F7Eh	—[1]	F5Eh	—[1]	F3Eh	—[1]	F1Eh	—[1]
F7Dh	—[1]	F5Dh	—[1]	F3Dh	—[1]	F1Dh	—[1]
F7Ch	—[1]	F5Ch	—[1]	F3Ch	—[1]	F1Ch	—[1]
F7Bh	—[1]	F5Bh	—[1]	F3Bh	—[1]	F1Bh	—[1]
F7Ah	—[1]	F5Ah	—[1]	F3Ah	—[1]	F1Ah	—[1]
F79h	—[1]	F59h	—[1]	F39h	—[1]	F19h	—[1]
F78h	TMR4	F58h	—[1]	F38h	—[1]	F18h	—[1]
F77h	PR4	F57h	—[1]	F37h	—[1]	F17h	—[1]
F76h	T4CON	F56h	—[1]	F36h	—[1]	F16h	—[1]
F75h	CCPR4H	F55h	—[1]	F35h	—[1]	F15h	—[1]
F74h	CCPR4L	F54h	—[1]	F34h	—[1]	F14h	—[1]
F73h	CCP4CON	F53h	—[1]	F33h	—[1]	F13h	—[1]
F72h	CCPR5H	F52h	—[1]	F32h	—[1]	F12h	—[1]
F71h	CCPR5L	F51h	—[1]	F31h	—[1]	F11h	—[1]
F70h	CCP5CON	F50h	—[1]	F30h	—[1]	F10h	—[1]
F6Fh	SPBRG2	F4Fh	—[1]	F2Fh	—[1]	F0Fh	—[1]
F6Eh	RCREG2	F4Eh	—[1]	F2Eh	—[1]	F0Eh	—[1]
F6Dh	TXREG2	F4Dh	—[1]	F2Dh	—[1]	F0Dh	—[1]
F6Ch	TXSTA2	F4Ch	—[1]	F2Ch	—[1]	F0Ch	—[1]
F6Bh	RCSTA2	F4Bh	—[1]	F2Bh	—[1]	F0Bh	—[1]
F6Ah	—[1]	F4Ah	—[1]	F2Ah	—[1]	F0Ah	—[1]
F69h	—[1]	F49h	—[1]	F29h	—[1]	F09h	—[1]
F68h	—[1]	F48h	—[1]	F28h	—[1]	F08h	—[1]
F67h	—[1]	F47h	—[1]	F27h	—[1]	F07h	—[1]
F66h	—[1]	F46h	—[1]	F26h	—[1]	F06h	—[1]
F65h	—[1]	F45h	—[1]	F25h	—[1]	F05h	—[1]
F64h	—[1]	F44h	—[1]	F24h	—[1]	F04h	—[1]
F63h	—[1]	F43h	—[1]	F23h	—[1]	F03h	—[1]
F62h	—[1]	F42h	—[1]	F22h	—[1]	F02h	—[1]
F61h	—[1]	F41h	—[1]	F21h	—[1]	F01h	—[1]
F60h	—[1]	F40h	—[1]	F20h	—[1]	F00h	—[1]

Note 1: Unimplemented registers are read as ‘0’.

2: This register is not available on PIC18F6X20 devices.

3: This is not a physical register.

4.10 Access Bank

The Access Bank is an architectural enhancement, which is very useful for C compiler code optimization. The techniques used by the C compiler may also be useful for programs written in assembly.

This data memory region can be used for:

- Intermediate computational values
- Local variables of subroutines
- Faster context saving/switching of variables
- Common variables
- Faster evaluation/control of SFRs (no banking)

The Access Bank is comprised of the upper 160 bytes in Bank 15 (SFRs) and the lower 96 bytes in Bank 0. These two sections will be referred to as Access RAM High and Access RAM Low, respectively. Figure 4-7 indicates the Access RAM areas.

A bit in the instruction word specifies if the operation is to occur in the bank specified by the BSR register or in the Access Bank. This bit is denoted by the 'a' bit (for access bit).

When forced in the Access Bank (a = 0), the last address in Access RAM Low is followed by the first address in Access RAM High. Access RAM High maps the Special Function Registers, so that these registers can be accessed without any software overhead. This is useful for testing status flags and modifying control bits.

4.11 Bank Select Register (BSR)

The need for a large general purpose memory space dictates a RAM banking scheme. The data memory is partitioned into sixteen banks. When using direct addressing, the BSR should be configured for the desired bank.

BSR<3:0> holds the upper 4 bits of the 12-bit RAM address. The BSR<7:4> bits will always read '0's and writes will have no effect.

A MOVLB instruction has been provided in the instruction set to assist in selecting banks.

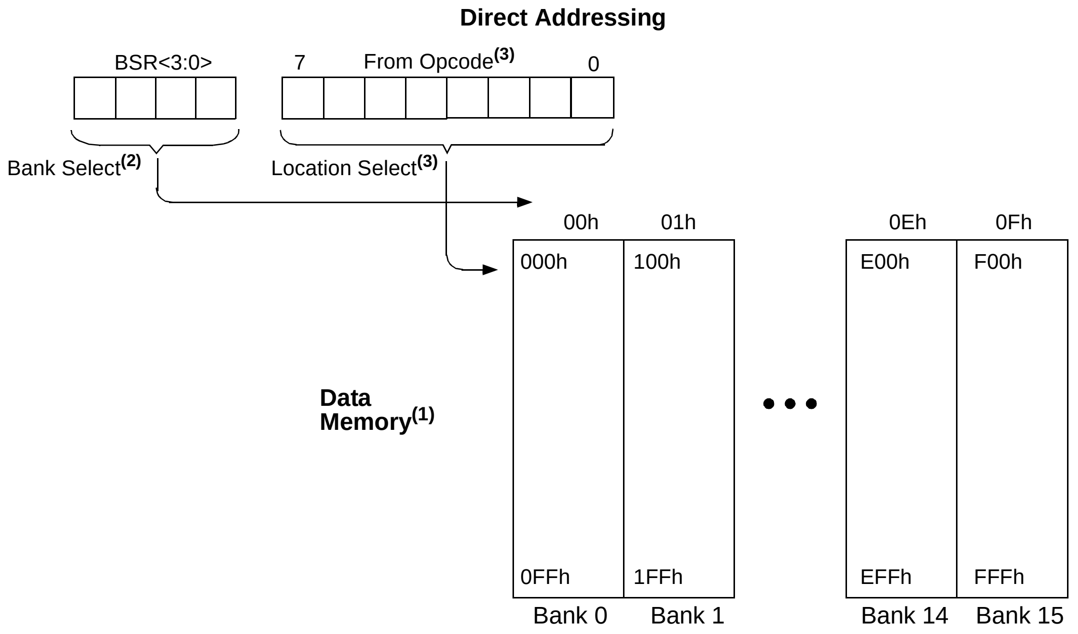
If the currently selected bank is not implemented, any read will return all '0's and all writes are ignored. The Status register bits will be set/cleared as appropriate for the instruction performed.

Each Bank extends up to FFh (256 bytes). All data memory is implemented as static RAM.

A MOVFF instruction ignores the BSR, since the 12-bit addresses are embedded into the instruction word.

Section 4.12 "Indirect Addressing, INDF and FSR Registers" provides a description of indirect addressing, which allows linear addressing of the entire RAM space.

FIGURE 4-8: DIRECT ADDRESSING



Note 1: For register file map detail, see Table 4-2.

2: The access bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

3: The MOVFF instruction embeds the entire 12-bit address in the instruction.

REGISTER 5-1: EECON1 REGISTER (ADDRESS FA6h)

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7 **EEPGD:** Flash Program or Data EEPROM Memory Select bit

1 = Access Flash program memory
0 = Access data EEPROM memory

bit 6 **CFGS:** Flash Program/Data EEPROM or Configuration Select bit

1 = Access configuration registers
0 = Access Flash program or data EEPROM memory

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

bit 4 **FREE:** Flash Row Erase Enable bit

1 = Erase the program memory row addressed by TBLPTR on the next WR command (cleared by completion of erase operation)
0 = Perform write only

bit 3 **WRERR:** Flash Program/Data EEPROM Error Flag bit

1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation)
0 = The write operation completed

Note: When a WRERR occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the error condition.

bit 2 **WREN:** Flash Program/Data EEPROM Write Enable bit

1 = Allows write cycles to Flash program/data EEPROM
0 = Inhibits write cycles to Flash program/data EEPROM

bit 1 **WR:** Write Control bit

1 = Initiates a data EEPROM erase/write cycle or a program memory erase cycle or write cycle. (The operation is self-timed and the bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.)
0 = Write cycle to the EEPROM is complete

bit 0 **RD:** Read Control bit

1 = Initiates an EEPROM read (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1.)
0 = Does not initiate an EEPROM read

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

6.0 EXTERNAL MEMORY INTERFACE

Note:	The External Memory Interface is not implemented on PIC18F6X20 (64-pin) devices.

The External Memory Interface is a feature of the PIC18F8X20 devices that allows the controller to access external memory devices (such as Flash, EPROM, SRAM, etc.) as program or data memory.

The physical implementation of the interface uses 27 pins. These pins are reserved for external address/data bus functions; they are multiplexed with I/O port pins on four ports. Three I/O ports are multiplexed with the address/data bus, while the fourth port is multiplexed with the bus control signals. The I/O port functions are enabled when the EBDIS bit in the MEMCON register is set (see Register 6-1). A list of the multiplexed pins and their functions is provided in Table 6-1.

As implemented in the PIC18F8X20 devices, the interface operates in a similar manner to the external memory interface introduced on PIC18C601/801 microcontrollers. The most notable difference is that the interface on PIC18F8X20 devices only operates in 16-bit modes. The 8-bit mode is not supported.

For a more complete discussion of the operating modes that use the external memory interface, refer to **Section 4.1.1 "PIC18F8X20 Program Memory Modes"**.

6.1 Program Memory Modes and the External Memory Interface

As previously noted, PIC18F8X20 controllers are capable of operating in any one of four program memory modes, using combinations of on-chip and external program memory. The functions of the multiplexed port pins depend on the program memory mode selected, as well as the setting of the EBDIS bit.

In **Microprocessor Mode**, the external bus is always active and the port pins have only the external bus function.

In **Microcontroller Mode,** the bus is not active and the pins have their port functions only. Writes to the MEMCOM register are not permitted.

In **Microprocessor with Boot Block** or **Extended Microcontroller Mode,** the external program memory bus shares I/O port functions on the pins. When the device is fetching or doing table read/table write operations on the external program memory space, the pins will have the external bus function. If the device is fetching and accessing internal program memory locations only, the EBDIS control bit will change the pins from external memory to I/O port functions. When EBDIS = 0, the pins function as the external bus. When EBDIS = 1, the pins function as I/O ports.

Note:	Maximum FOSC for the PIC18FX520 is limited to 25 MHz when using the external memory interface.

REGISTER 6-1: MEMCON REGISTER

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0

bit7 bit0

bit 7 **EBDIS:** External Bus Disable bit

1 = External system bus disabled, all external bus drivers are mapped as I/O ports
0 = External system bus enabled and I/O ports are disabled

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

bit 5-4 **WAIT<1:0>:** Table Reads and Writes Bus Cycle Wait Count bits

11 = Table reads and writes will wait 0 TCY
10 = Table reads and writes will wait 1 TCY
01 = Table reads and writes will wait 2 TCY
00 = Table reads and writes will wait 3 TCY

bit 3-2 **Unimplemented:** Read as '0'

bit 1-0 **WM<1:0>:** TBLWRT Operation with 16-bit Bus bits

1x = Word Write mode: TABLAT<0> and TABLAT<1> word output, $\overline{\text{WRH}}$ active when TABLAT<1> written
01 = Byte Select mode: TABLAT data copied on both MSB and LSB, $\overline{\text{WRH}}$ and ($\overline{\text{UB}}$ or $\overline{\text{LB}}$) will activate
00 = Byte Write mode: TABLAT data copied on both MSB and LSB, $\overline{\text{WRH}}$ or $\overline{\text{WRL}}$ will activate

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

REGISTER 9-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

U-0	R/W-1	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	CMIP	—	EEIP	BCLIP	LVDIP	TMR3IP	CCP2IP
bit 7							bit 0

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

bit 6 **CMIP:** Comparator Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 4 **EEIP:** Data EEPROM/Flash Write Operation Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **BCLIP:** Bus Collision Interrupt Priority bit
1 = High priority
0 = Low priority

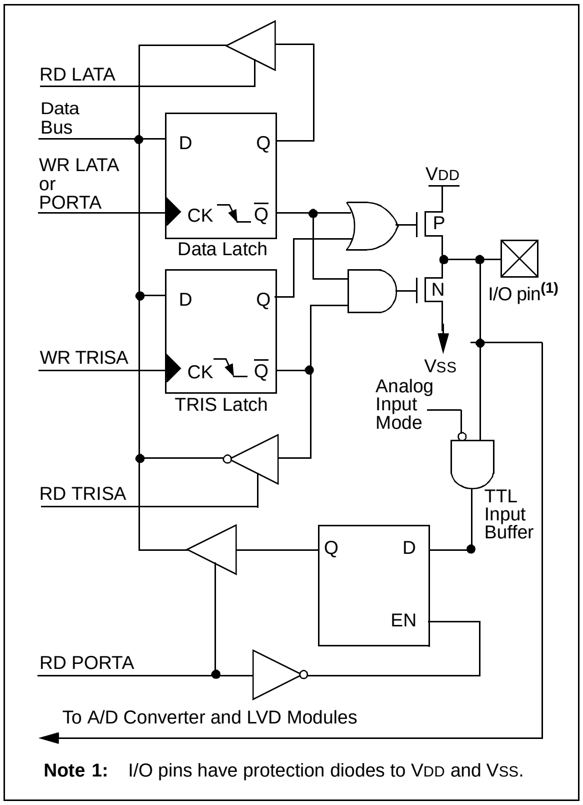
bit 2 **LVDIP:** Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR3IP:** TMR3 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **CCP2IP:** CCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

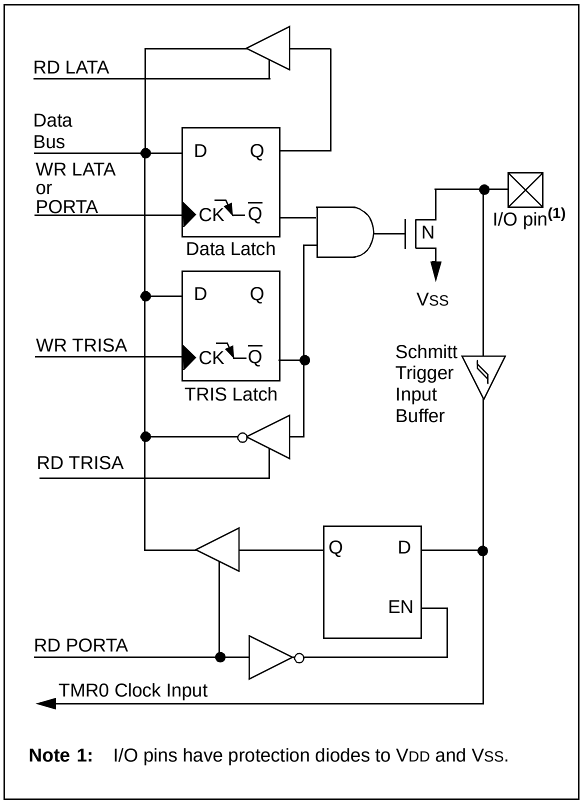
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

FIGURE 10-2: BLOCK DIAGRAM OF RA3:RA0 AND RA5 PINS



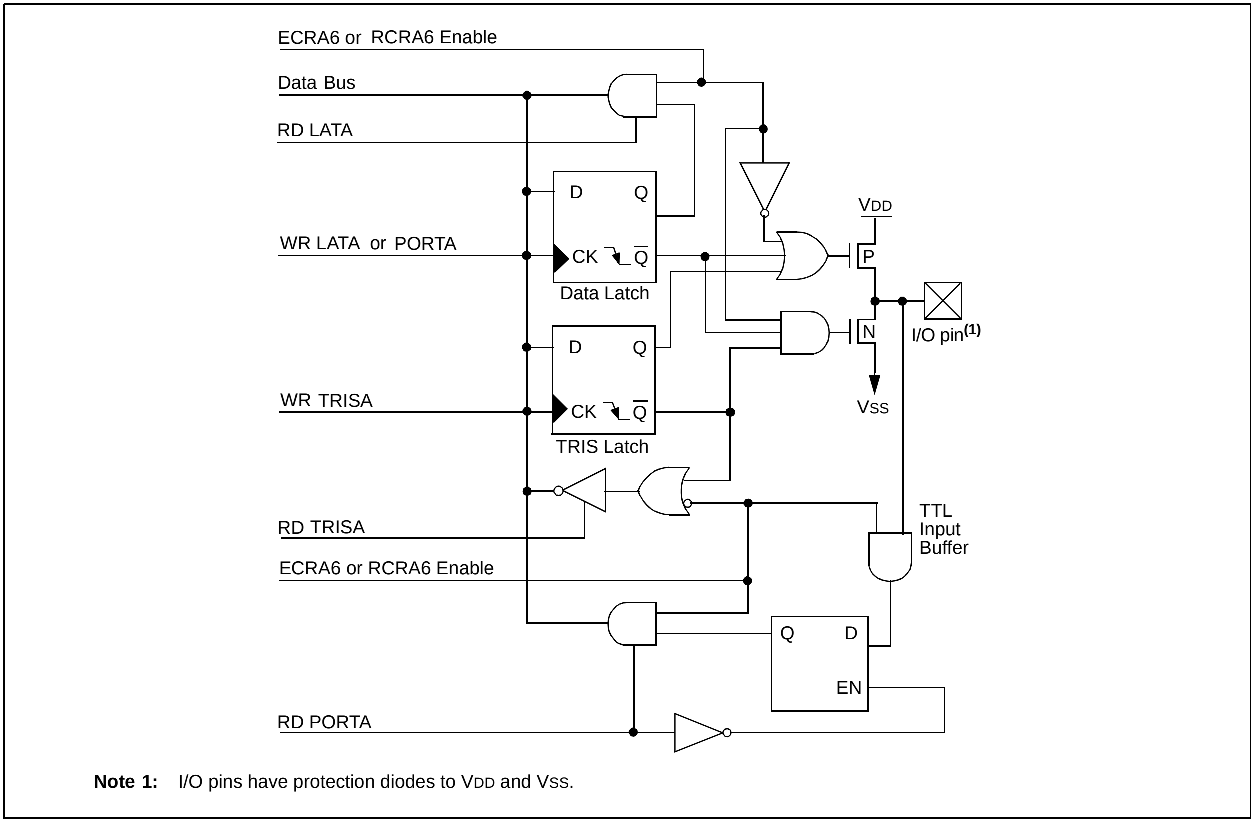
Note 1: I/O pins have protection diodes to VDD and VSS.

FIGURE 10-3: BLOCK DIAGRAM OF RA4/T0CKI PIN



Note 1: I/O pins have protection diodes to VDD and VSS.

FIGURE 10-4: BLOCK DIAGRAM OF RA6 PIN (WHEN ENABLED AS I/O)



Note 1: I/O pins have protection diodes to VDD and VSS.

13.0 TIMER2 MODULE

The Timer2 module timer has the following features:

- 8-bit timer (TMR2 register)
- 8-bit period register (PR2)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match of PR2
- SSP module optional use of TMR2 output to generate clock shift

Timer2 has a control register shown in Register 13-1. Timer2 can be shut-off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption. Figure 13-1 is a simplified block diagram of the Timer2 module. Register 13-1 shows the Timer2 Control register. The prescaler and postscaler selection of Timer2 are controlled by this register.

13.1 Timer2 Operation

Timer2 can be used as the PWM time base for the PWM mode of the CCP module. The TMR2 register is readable and writable and is cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, T2CKPS1:T2CKPS0 (T2CON<1:0>). The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt (latched in flag bit, TMR2IF (PIR1<1>)).

The prescaler and postscaler counters are cleared when any of the following occurs:

- a write to the TMR2 register
- a write to the T2CON register
- any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as ‘0’

bit 6-3 **T2OUTPS3:T2OUTPS0:** Timer2 Output Postscale Select bits

0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit

1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS1:T2CKPS0:** Timer2 Clock Prescale Select bits

00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

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

19.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 12 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable. There are seven possible options for TAD:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal RC oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time of 1.6 μs.

Table 19-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

19.3 Configuring Analog Port Pins

The ADCON1, TRISA, TRISF and TRISH registers control the operation of the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CHS3:CHS0 bits and the TRIS bits.

Note 1: When reading the port register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert as an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.

2: Analog levels on any pin defined as a digital input may cause the input buffer to consume current out of the device's specification limits.

TABLE 19-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency	
Operation	**ADCS2:ADCS0**	**PIC18FXX20**	**PIC18LFXX20**
2 TOSC	`000`	1.25 MHz	666 kHz
4 TOSC	`100`	2.50 MHz	1.33 MHz
8 TOSC	`001`	5.00 MHz	2.67 MHz
16 TOSC	`101`	10.0 MHz	5.33 MHz
32 TOSC	`010`	20.0 MHz	10.67 MHz
64 TOSC	`110`	40.0 MHz	21.33 MHz
RC	`x11`	—	—

REGISTER 23-7: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
CP7[1]	CP6[1]	CP5[1]	CP4[1]	CP3	CP2	CP1	CP0
bit 7							bit 0

bit 7 **CP7:** Code Protection bit[1]
1 = Block 7 (01C000-01FFFFh) not code-protected
0 = Block 7 (01C000-01FFFFh) code-protected

bit 6 **CP6:** Code Protection bit[1]
1 = Block 6 (018000-01BFFFh) not code-protected
0 = Block 6 (018000-01BFFFh) code-protected

bit 5 **CP5:** Code Protection bit[1]
1 = Block 5 (014000-017FFFh) not code-protected
0 = Block 5 (014000-017FFFh) code-protected

bit 4 **CP4:** Code Protection bit[1]
1 = Block 4 (010000-013FFFh) not code-protected
0 = Block 4 (010000-013FFFh) code-protected

bit 3 **CP3:** Code Protection bit
For PIC18FX520 devices:
1 = Block 3 (006000-007FFFh) not code-protected
0 = Block 3 (006000-007FFFh) code-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 3 (00C000-00FFFFh) not code-protected
0 = Block 3 (00C000-00FFFFh) code-protected

bit 2 **CP2:** Code Protection bit
For PIC18FX520 devices:
1 = Block 2 (004000-005FFFh) not code-protected
0 = Block 2 (004000-005FFFh) code-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 2 (008000-00BFFFh) not code-protected
0 = Block 2 (008000-00BFFFh) code-protected

bit 1 **CP1:** Code Protection bit
For PIC18FX520 devices:
1 = Block 1 (002000-003FFFh) not code-protected
0 = Block 1 (002000-003FFFh) code-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 1 (004000-007FFFh) not code-protected
0 = Block 1 (004000-007FFFh) code-protected

bit 0 **CP0:** Code Protection bit
For PIC18FX520 devices:
1 = Block 0 (000800-001FFFh) not code-protected
0 = Block 0 (000800-001FFFh) code-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 0 (000200-003FFFh) not code-protected
0 = Block 0 (000200-003FFFh) code-protected

Note 1: Unimplemented in PIC18FX520 and PIC18FX620 devices; maintain this bit set.

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed		u = Unchanged from programmed state

REGISTER 23-8: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—
bit 7							bit 0

bit 7 **CPD:** Data EEPROM Code Protection bit

`1` = Data EEPROM not code-protected
`0` = Data EEPROM code-protected

bit 6 **CPB:** Boot Block Code Protection bit

For PIC18FX520 devices:
`1` = Boot Block (000000-0007FFh) not code-protected
`0` = Boot Block (000000-0007FFh) code-protected

For PIC18FX620 and PIC18FX720 devices:
`1` = Boot Block (000000-0001FFh) not code-protected
`0` = Boot Block (000000-0001FFh) code-protected

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

REGISTER 23-9: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
WRT7(1)	WRT6(1)	WRT5(1)	WRT4(1)	WRT3	WRT2	WRT1	WRT0
bit 7							bit 0

bit 7 **WR7:** Write Protection bit(1)
1 = Block 7 (01C000-01FFFFh) not write-protected
0 = Block 7 (01C000-01FFFFh) write-protected

bit 6 **WR6:** Write Protection bit(1)
1 = Block 6 (018000-01BFFFh) not write-protected
0 = Block 6 (018000-01BFFFh) write-protected

bit 5 **WR5:** Write Protection bit(1)
1 = Block 5 (014000-017FFFh) not write-protected
0 = Block 5 (014000-017FFFh) write-protected

bit 4 **WR4:** Write Protection bit(1)
1 = Block 4 (010000-013FFFh) not write-protected
0 = Block 4 (010000-013FFFh) write-protected

bit 3 **WR3:** Write Protection bit
For PIC18FX520 devices:
1 = Block 3 (006000-007FFFh) not write-protected
0 = Block 3 (006000-007FFFh) write-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 3 (00C000-00FFFFh) not write-protected
0 = Block 3 (00C000-00FFFFh) write-protected

bit 2 **WR2:** Write Protection bit
For PIC18FX520 devices:
1 = Block 2 (004000-005FFFh) not write-protected
0 = Block 2 (004000-005FFFh) write-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 2 (008000-00BFFFh) not write-protected
0 = Block 2 (008000-00BFFFh) write-protected

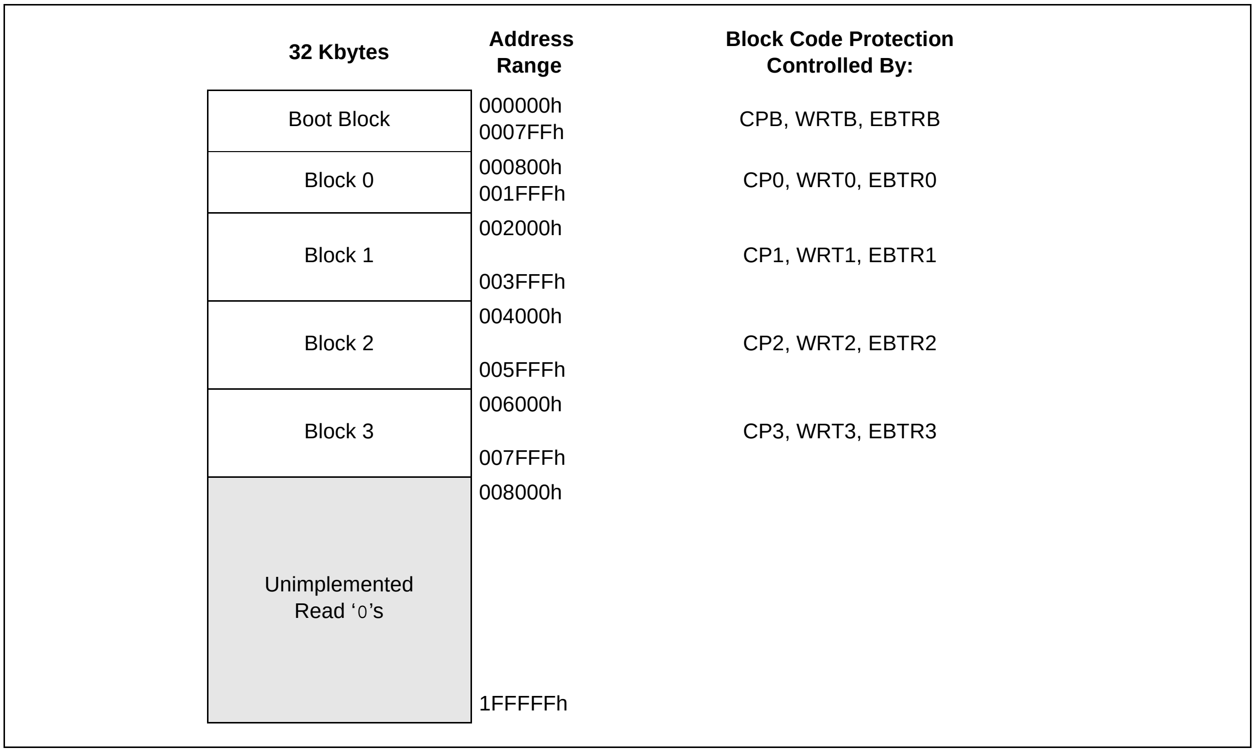
bit 1 **WR1:** Write Protection bit
For PIC18FX520 devices:
1 = Block 1 (002000-003FFFh) not write-protected
0 = Block 1 (002000-003FFFh) write-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 1 (004000-007FFFh) not write-protected
0 = Block 1 (004000-007FFFh) write-protected

bit 0 **WR0:** Write Protection bit
For PIC18FX520 devices:
1 = Block 0 (000800-001FFFh) not write-protected
0 = Block 0 (000800-001FFFh) write-protected
For PIC18FX620 and PIC18FX720 devices:
1 = Block 0 (000200-003FFFh) not write-protected
0 = Block 0 (000200-003FFFh) write-protected

Note 1: Unimplemented in PIC18FX520 and PIC18FX620 devices; maintain this bit set.

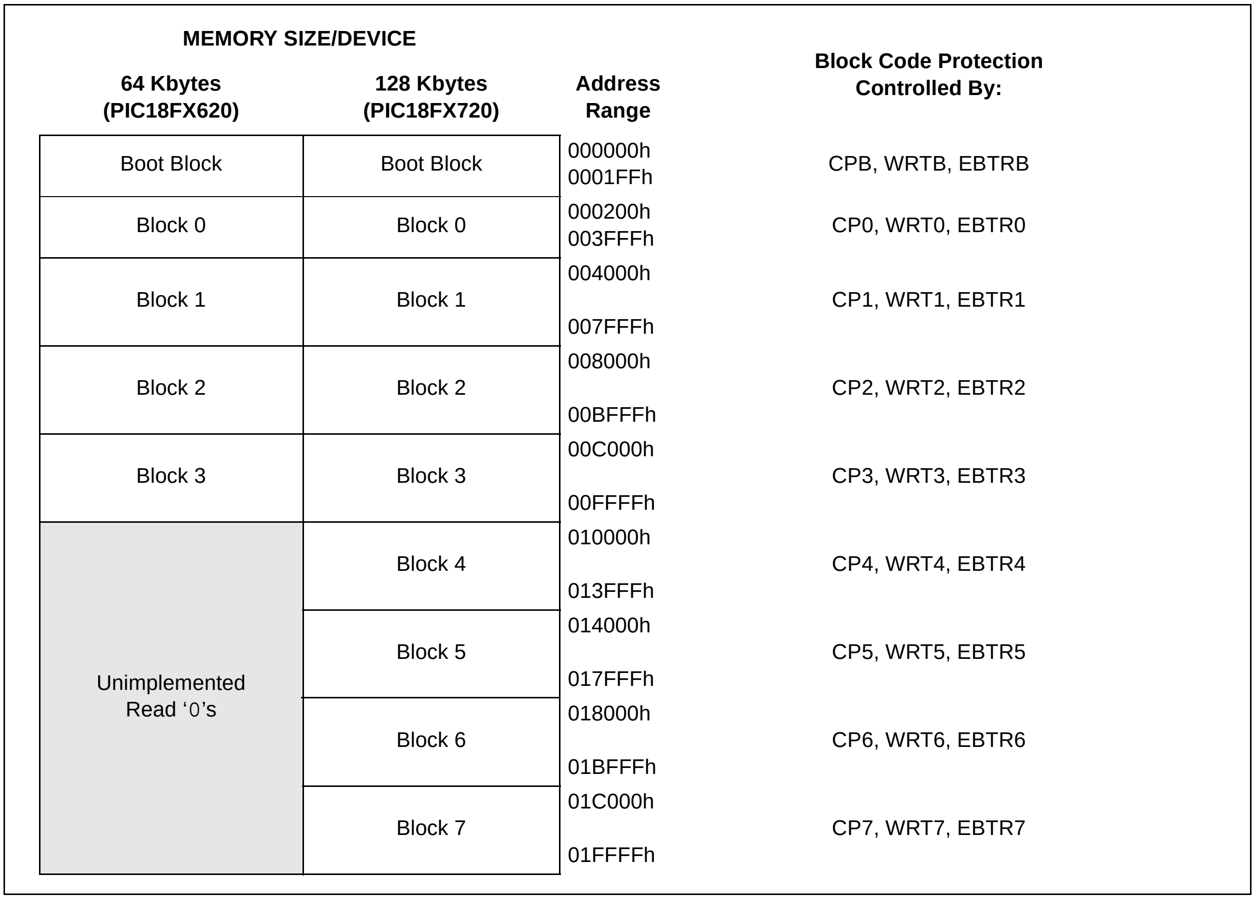
Legend:
R = Readable bit P = Programmable bit U = Unimplemented bit, read as ‘0’
- n = Value when device is unprogrammed u = Unchanged from programmed state

FIGURE 23-3: CODE-PROTECTED PROGRAM MEMORY FOR PIC18FX520 DEVICES



32 Kbytes	Address Range	Block Code Protection Controlled By:
Boot Block	000000h 0007FFh	CPB, WRTB, EBTRB
Block 0	000800h 001FFFh	CP0, WRT0, EBTR0
Block 1	002000h 003FFFh	CP1, WRT1, EBTR1
Block 2	004000h 005FFFh	CP2, WRT2, EBTR2
Block 3	006000h 007FFFh	CP3, WRT3, EBTR3
Unimplemented Read ‘0’s	008000h 1FFFFFh	

FIGURE 23-4: CODE-PROTECTED PROGRAM MEMORY FOR PIC18FX620/X720 DEVICES



MEMORY SIZE/DEVICE			
64 Kbytes (PIC18FX620)	128 Kbytes (PIC18FX720)	Address Range	Block Code Protection Controlled By:
Boot Block	Boot Block	000000h 0001FFh	CPB, WRTB, EBTRB
Block 0	Block 0	000200h 003FFFh	CP0, WRT0, EBTR0
Block 1	Block 1	004000h 007FFFh	CP1, WRT1, EBTR1
Block 2	Block 2	008000h 00BFFFh	CP2, WRT2, EBTR2
Block 3	Block 3	00C000h 00FFFFh	CP3, WRT3, EBTR3
Unimplemented Read ‘0’s	Block 4	010000h 013FFFh	CP4, WRT4, EBTR4
	Block 5	014000h 017FFFh	CP5, WRT5, EBTR5
	Block 6	018000h 01BFFFh	CP6, WRT6, EBTR6
	Block 7	01C000h 01FFFFh	CP7, WRT7, EBTR7

GOTO Unconditional Branch

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 1048575$

Operation: k → PC<20:1>

Status Affected: None

Encoding:				
1st word (k<7:0>)	1110	1111	k$_7$kkk	kkkk$_0$
2nd word(k<19:8>)	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: [*label*] INCF f [,d [,a]

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

Operation: (f) + 1 → 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 will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value (default).

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 = 0xFF
Z = 0
C = ?
DC = ?

After Instruction
CNT = 0x00
Z = 1
C = 1
DC = 1

SUBWFB Subtract W from f with Borrow

Syntax: [*label*] SUBWFB f [,d [,a]

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

Operation: $(f) - (W) - (\overline{C}) \rightarrow dest$

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

Encoding:

`0101`	`10da`	`ffff`	`ffff`

Description: Subtract W and the Carry flag (borrow) from register ‘f’ (2’s complement method). If ‘d’ is ‘`0`’, the result is stored in W. If ‘d’ is ‘`1`’, the result is stored back in register ‘f’ (default). If ‘a’ is ‘`0`’, the Access Bank will be selected, overriding the BSR value. If ‘a’ is ‘`1`’, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `SUBWFB REG, 1, 0`

Before Instruction
```
REG   =  0x19   (0001 1001)
W     =  0x0D   (0000 1101)
C     =  1
```
After Instruction
```
REG   =  0x0C   (0000 1011)
W     =  0x0D   (0000 1101)
C     =  1
Z     =  0
N     =  0      ; result is positive
```

Example 2: `SUBWFB REG, 0, 0`

Before Instruction
```
REG   =  0x1B   (0001 1011)
W     =  0x1A   (0001 1010)
C     =  0
```
After Instruction
```
REG   =  0x1B   (0001 1011)
W     =  0x00
C     =  1
Z     =  1      ; result is zero
N     =  0
```

Example 3: `SUBWFB REG, 1, 0`

Before Instruction
```
REG   =  0x03   (0000 0011)
W     =  0x0E   (0000 1101)
C     =  1
```
After Instruction
```
REG   =  0xF5   (1111 0100)
                ; [2's comp]
W     =  0x0E   (0000 1101)
C     =  0
Z     =  0
N     =  1      ; result is negative
```

SWAPF Swap f

Syntax: [*label*] SWAPF f [,d [,a]

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

Operation: $(f<3:0>) \rightarrow dest<7:4>$,
$(f<7:4>) \rightarrow dest<3:0>$

Status Affected: None

Encoding:

`0011`	`10da`	`ffff`	`ffff`

Description: The upper and lower nibbles of register ‘f’ are exchanged. If ‘d’ is ‘`0`’, the result is placed in W. If ‘d’ is ‘`1`’, the result is placed in register ‘f’ (default). If ‘a’ is ‘`0`’, the Access Bank will be selected, overriding the BSR value. If ‘a’ is ‘`1`’, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `SWAPF REG, 1, 0`

Before Instruction
```
REG   =  0x53
```
After Instruction
```
REG   =  0x35
```

FIGURE 27-15: TYPICAL AND MAXIMUM IPD vs. VDD OVER TEMPERATURE (TIMER1 AS MAIN OSCILLATOR, 32.768 kHz, C1 AND C2 = 47 pF)

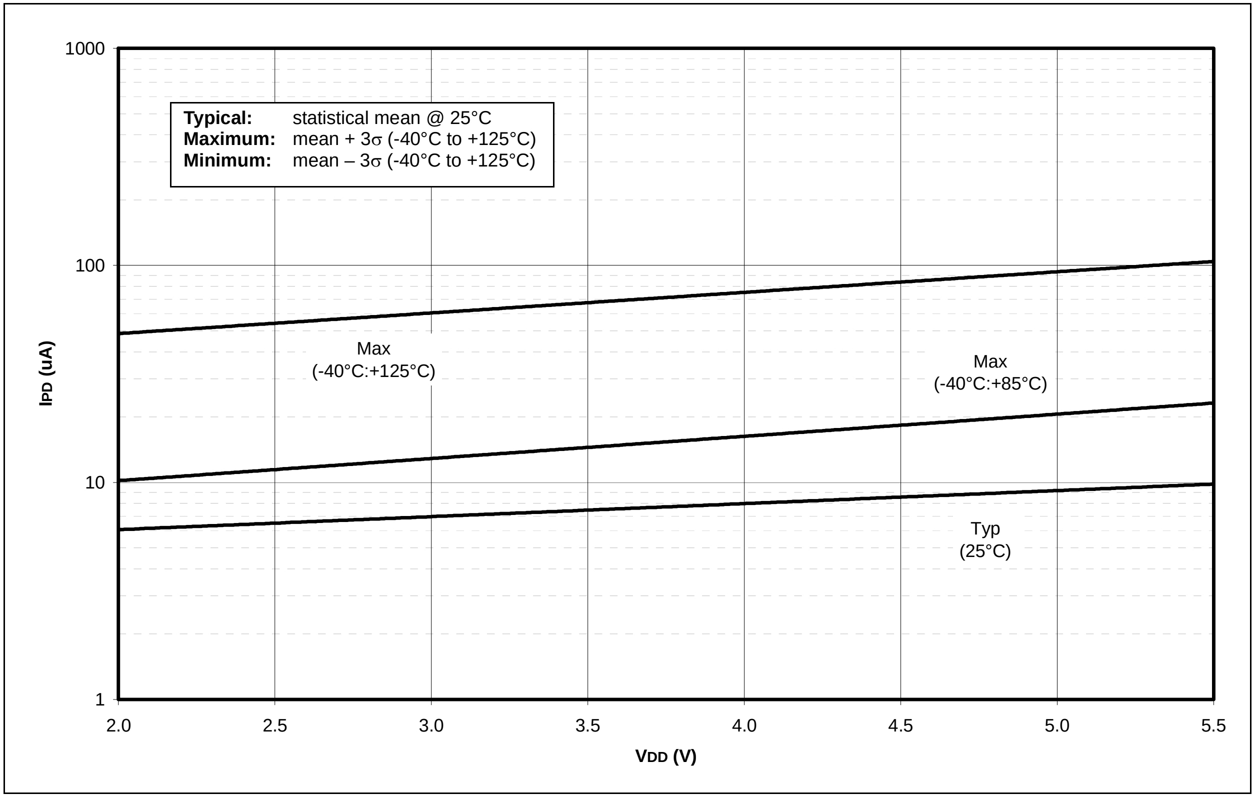
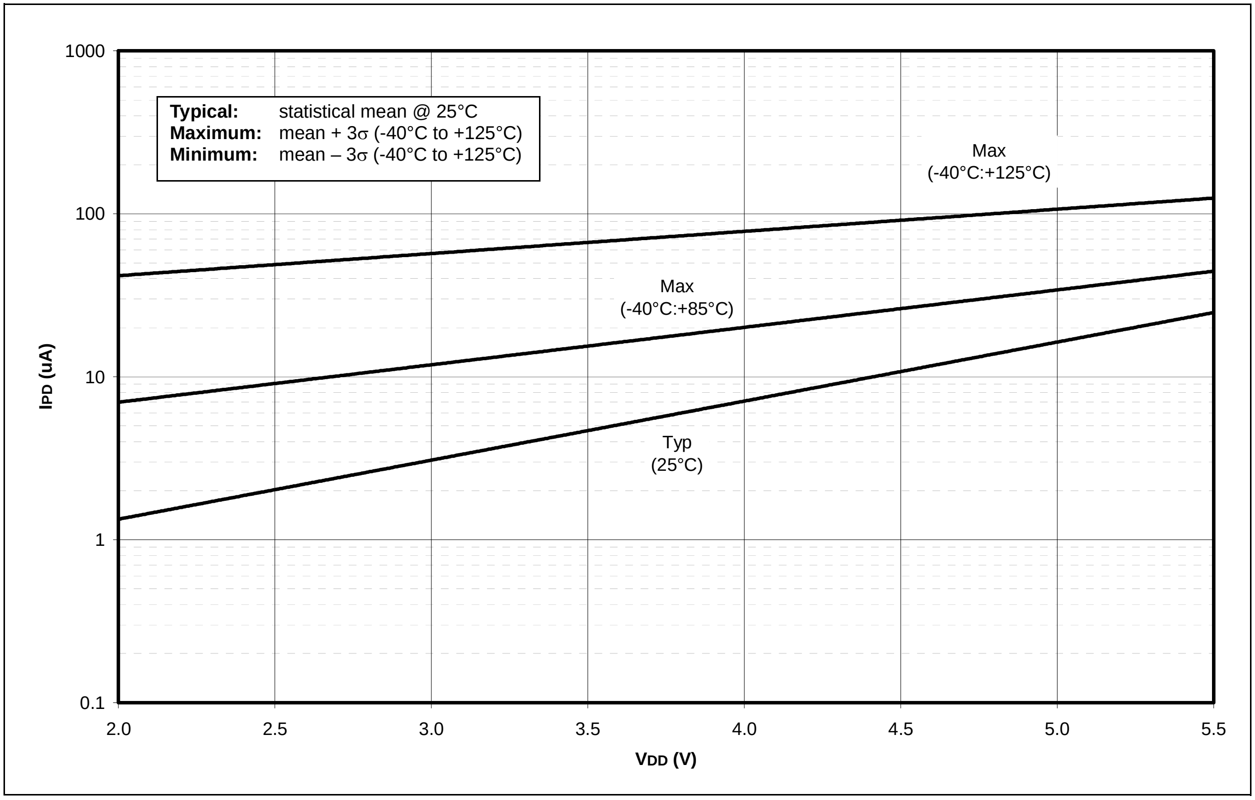


FIGURE 27-16: TYPICAL AND MAXIMUM ΔIWDT vs. VDD OVER TEMPERATURE (WDT ENABLED)



APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC17C756 to a PIC18F8720.

Not Currently Available

APPENDIX D: MIGRATION FROM MID-RANGE TO ENHANCED DEVICES

A detailed discussion of the differences between the mid-range MCU devices (i.e., PIC16CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in *AN716, "Migrating Designs from PIC16C74A/74B to PIC18C442"*. The changes discussed, while device specific, are generally applicable to all mid-range to enhanced device migrations.

This Application Note is available as Literature Number DS00716.

NOTES: