Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	40MHz
Connectivity	UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SSOP (0.209", 5.30mm Width)
Supplier Device Package	20-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f1320-i-ss

Table of Contents

TABLE 1-2: PIC18F1220/1320 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP/ SOIC	SSOP	QFN			
						PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/AN4/INT0	8	9	9			
RB0				I/O	TTL	Digital I/O.
AN4				I	Analog	Analog input 4.
INT0				I	ST	External interrupt 0.
RB1/AN5/TX/CK/INT1	9	10	10			
RB1				I/O	TTL	Digital I/O.
AN5				I	Analog	Analog input 5.
TX				O	—	EUSART asynchronous transmit.
CK				I/O	ST	EUSART synchronous clock (see related RX/DT).
INT1				I	ST	External interrupt 1.
RB2/P1B/INT2	17	19	23			
RB2				I/O	TTL	Digital I/O.
P1B				O	—	Enhanced CCP1/PWM output.
INT2				I	ST	External interrupt 2.
RB3/CCP1/P1A	18	20	24			
RB3				I/O	TTL	Digital I/O.
CCP1				I/O	ST	Capture 1 input/Compare 1 output/PWM 1 output.
P1A				O	—	Enhanced CCP1/PWM output.
RB4/AN6/RX/DT/KBI0	10	11	12			
RB4				I/O	TTL	Digital I/O.
AN6				I	Analog	Analog input 6.
RX				I	ST	EUSART asynchronous receive.
DT				I/O	ST	EUSART synchronous data (see related TX/CK).
KBI0				I	TTL	Interrupt-on-change pin.
RB5/PGM/KBI1	11	12	13			
RB5				I/O	TTL	Digital I/O.
PGM				I/O	ST	Low-Voltage ICSP™ Programming enable pin.
KBI1				I	TTL	Interrupt-on-change pin.
RB6/PGC/T1OSO/ T13CKI/P1C/KBI2	12	13	15			
RB6				I/O	TTL	Digital I/O.
PGC				I/O	ST	In-Circuit Debugger and ICSP programming clock pin.
T1OSO				O	—	Timer1 oscillator output.
T13CKI				I	ST	Timer1/Timer3 external clock output.
P1C				O	—	Enhanced CCP1/PWM output.
KBI2				I	TTL	Interrupt-on-change pin.
RB7/PGD/T1OSI/ P1D/KBI3	13	14	16			
RB7				I/O	TTL	Digital I/O.
PGD				I/O	ST	In-Circuit Debugger and ICSP programming data pin.
T1OSI				I	CMOS	Timer1 oscillator input.
P1D				O	—	Enhanced CCP1/PWM output.
KBI3				I	TTL	Interrupt-on-change pin.
V_{SS}	5	5, 6	3, 5	P	—	Ground reference for logic and I/O pins.
V_{DD}	14	15, 16	17, 19	P	—	Positive supply for logic and I/O pins.
NC	—	—	18	—	—	No connect.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels; I = Input
O = Output; P = Power
OD = Open-drain (no P diode to V_{DD})

3.4.3 RC_RUN MODE

In RC_RUN mode, the CPU and peripherals are clocked from the internal oscillator block using the INTOSC multiplexer and 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 of the INTIO1 or INTIO2 oscillators), 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 clearing the IDLEN bit, setting SCS1 (SCS0 is ignored) and executing a SLEEP instruction. The IRCF bits may select the clock frequency before the SLEEP instruction is executed. When the clock source is switched to the INTOSC multiplexer (see Figure 3-10), 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 system clock speed. Executing a SLEEP instruction is not required to select a new clock frequency from the INTOSC multiplexer.

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

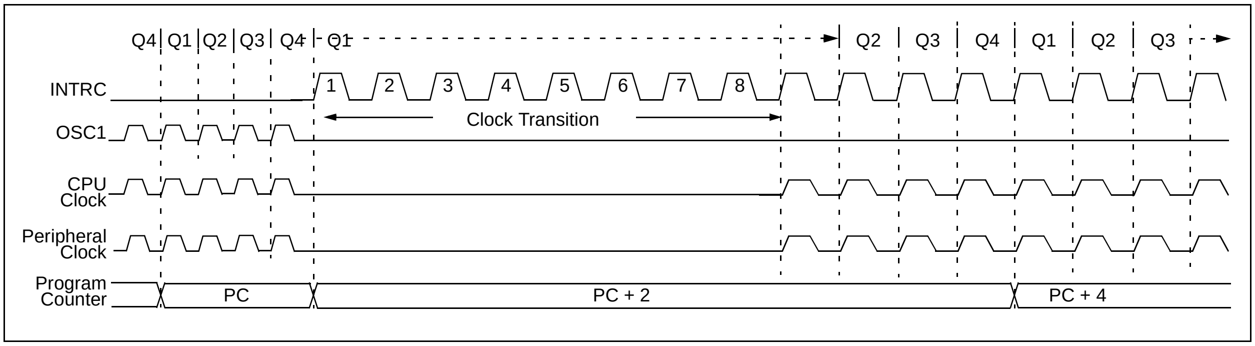
If the IRCF bits are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the system clocks.

If the IRCF bits are changed from all clear (thus, enabling the INTOSC output), the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the system continue while the INTOSC source stabilizes, in approximately 1 ms.

If the IRCF bits were previously at a non-zero value before the SLEEP instruction was executed and the INTOSC source was already stable, the IOFS bit will remain set.

When a wake event occurs, the system continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 3-8). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock is providing the system clock. The IDLEN and SCS bits are not affected by the wake-up. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

FIGURE 3-10: TIMING TRANSITION TO RC_RUN MODE



8.0 8 x 8 HARDWARE MULTIPLIER

8.1 Introduction

An 8 x 8 hardware multiplier is included in the ALU of the PIC18F1220/1320 devices. By making the multiply a hardware operation, it completes in a single instruction cycle. This is an unsigned multiply that gives a 16-bit result. The result is stored into the 16-bit product register pair (PRODH:PRODL). The multiplier does not affect any flags in the Status register.

Making the 8 x 8 multiplier execute in a single cycle gives the following advantages:

- Higher computational throughput
- Reduces code size requirements for multiply algorithms

The performance increase allows the device to be used in applications previously reserved for Digital Signal Processors.

Table 8-1 shows a performance comparison between Enhanced devices using the single-cycle hardware multiply and performing the same function without the hardware multiply.

TABLE 8-1: PERFORMANCE COMPARISON

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time		
				@ 40 MHz	@ 10 MHz	@ 4 MHz
8 x 8 unsigned	Without hardware multiply	13	69	6.9 μs	27.6 μs	69 μs
	Hardware multiply	1	1	100 ns	400 ns	1 μs
8 x 8 signed	Without hardware multiply	33	91	9.1 μs	36.4 μs	91 μs
	Hardware multiply	6	6	600 ns	2.4 μs	6 μs
16 x 16 unsigned	Without hardware multiply	21	242	24.2 μs	96.8 μs	242 μs
	Hardware multiply	28	28	2.8 μs	11.2 μs	28 μs
16 x 16 signed	Without hardware multiply	52	254	25.4 μs	102.6 μs	254 μs
	Hardware multiply	35	40	4 μs	16 μs	40 μs

8.2 Operation

Example 8-1 shows the sequence to do an 8 x 8 unsigned multiply. Only one instruction is required when one argument of the multiply is already loaded in the WREG register.

Example 8-2 shows the sequence to do an 8 x 8 signed multiply. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 8-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W     ;
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
```

EXAMPLE 8-2: 8 x 8 SIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
BTFSC   ARG2, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;       - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;       - ARG2
```

REGISTER 9-7: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0/0	U-0	U-0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0
OSCFIE	—	—	EEIE	—	LVDIE	TMR3IE	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **OSCFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **EEIE:** Data EEPROM/Flash Write Operation Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **LVDIE:** Low-Voltage Detect Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **TMR3IE:** TMR3 Overflow Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **Unimplemented:** Read as ‘0’

15.5.6 PROGRAMMABLE DEAD-BAND DELAY

In half-bridge applications where all power switches are modulated at the PWM frequency at all times, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period of time until one switch completely turns off. During this brief interval, a very high current (shoot-through current) may flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In the Half-Bridge Output mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 15-6 for an illustration. The lower seven bits of the PWM1CON register (Register 15-2) sets the delay period in terms of microcontroller instruction cycles (T_{CY} or 4 T_{OSC}).

15.5.7 ENHANCED PWM AUTO-SHUTDOWN

When the ECCP is programmed for any of the Enhanced PWM modes, the active output pins may be configured for auto-shutdown. Auto-shutdown immediately places the Enhanced PWM output pins into a defined shutdown state when a shutdown event occurs.

A shutdown event can be caused by the INT0, INT1 or INT2 pins (or any combination of these three sources). The auto-shutdown feature can be disabled by not selecting any auto-shutdown sources. The auto-shutdown sources to be used are selected using the ECCPAS2:ECCPAS0 bits (bits <6:4> of the ECCPAS register).

When a shutdown occurs, the output pins are asynchronously placed in their shutdown states, specified by the PSSAC1:PSSAC0 and PSSBD1:PSSBD0 bits (ECCPAS<3:0>). Each pin pair (P1A/P1C and P1B/P1D) may be set to drive high, drive low or be tri-stated (not driving). The ECCPASE bit (ECCPAS<7>) is also set to hold the Enhanced PWM outputs in their shutdown states.

The ECCPASE bit is set by hardware when a shutdown event occurs. If automatic restarts are not enabled, the ECCPASE bit is cleared by firmware when the cause of the shutdown clears. If automatic restarts are enabled, the ECCPASE bit is automatically cleared when the cause of the auto-shutdown has cleared.

If the ECCPASE bit is set when a PWM period begins, the PWM outputs remain in their shutdown state for that entire PWM period. When the ECCPASE bit is cleared, the PWM outputs will return to normal operation at the beginning of the next PWM period.

Note:	Writing to the ECCPASE bit is disabled while a shutdown condition is active.

REGISTER 15-3: ECCPAS: ENHANCED CAPTURE/COMPARE/PWM/AUTO-SHUTDOWN CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **ECCPASE:** ECCP Auto-Shutdown Event Status bit
0 = ECCP outputs are operating
1 = A shutdown event has occurred; ECCP outputs are in shutdown state

bit 6 **ECCPAS2:** ECCP Auto-Shutdown bit 2
0 = INT0 pin has no effect
1 = INT0 pin low causes shutdown

bit 5 **ECCPAS1:** ECCP Auto-Shutdown bit 1
0 = INT2 pin has no effect
1 = INT2 pin low causes shutdown

bit 4 **ECCPAS0:** ECCP Auto-Shutdown bit 0
0 = INT1 pin has no effect
1 = INT1 pin low causes shutdown

bit 3-2 **PSSACn:** Pins A and C Shutdown State Control bits
00 = Drive Pins A and C to ‘0’
01 = Drive Pins A and C to ‘1’
1x = Pins A and C tri-state

bit 1-0 **PSSBDn:** Pins B and D Shutdown State Control bits
00 = Drive Pins B and D to ‘0’
01 = Drive Pins B and D to ‘1’
1x = Pins B and D tri-state

TABLE 15-5: REGISTERS ASSOCIATED WITH ENHANCED PWM AND TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 11qq	0--q qquu
PIR1	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	-000 -000	-000 -000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	-000 -000	-000 -000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	-111 -111	-111 -111
TMR2	Timer2 Module Register								0000 0000	0000 0000
PR2	Timer2 Module Period Register								1111 1111	1111 1111
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
TRISB	PORTB Data Direction Register								1111 1111	1111 1111
CCPR1H	Enhanced Capture/Compare/PWM Register 1 High Byte								xxxx xxxx	uuuu uuuu
CCPR1L	Enhanced Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx	uuuu uuuu
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	0000 0000
ECCPAS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	0000 0000	0000 0000
PWM1CON	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0	0000 0000	uuuu uuuu
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0000 qq00	0000 qq00

Legend: x = unknown, u = unchanged, – = unimplemented, read as '0'.
Shaded cells are not used by the ECCP module in Enhanced PWM mode.

16.5 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the RB1/AN5/TX/CK/INT1 pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

16.5.1 EUSART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical, except in the case of the Sleep mode.

If two words are written to the TXREG and then the `SLEEP` instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREG register.
c) Flag bit, TXIF, will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit, TXIF, will now be set.
e) If enable bit, TXIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. Clear bits CREN and SREN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting enable bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 16-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	-000 -000	-000 -000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	-000 -000	-000 -000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	-111 -111	-111 -111
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
TXREG	EUSART Transmit Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCTL	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGH	Baud Rate Generator Register High Byte								0000 0000	0000 0000
SPBRG	Baud Rate Generator Register Low Byte								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as '0'. Shaded cells are not used for synchronous slave transmission.

REGISTER 17-1: ADCON0: A/D CONTROL REGISTER 0

R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
VCFG1	VCFG0	—	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **VCFG<1:0>:** Voltage Reference Configuration bits

	A/D VREF+	**A/D VREF-**
00	AVDD	AVSS
01	External VREF+	AVSS
10	AVDD	External VREF-
11	External VREF+	External VREF-

bit 5 **Unimplemented:** Read as ‘0’

bit 4-2 **CHS<2:0>:** Analog Channel Select bits
000 = Channel 0 (AN0)
001 = Channel 1 (AN1)
010 = Channel 2 (AN2)
011 = Channel 3 (AN3)
100 = Channel 4 (AN4)
101 = Channel 5 (AN5)
110 = Channel 6 (AN6)
111 = Unimplemented[(1)]

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion in progress
0 = A/D Idle

bit 0 **ADON:** A/D On bit
1 = A/D converter module is enabled
0 = A/D converter module is disabled
Performing a conversion on unimplemented channels returns full-scale results.

Note 1: Performing a conversion on unimplemented channels returns full-scale results.

19.0 SPECIAL FEATURES OF THE CPU

PIC18F1220/1320 devices include several features intended to maximize system reliability, minimize cost through elimination of external components and offer code protection. These are:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- ID Locations
- In-Circuit Serial Programming

Several oscillator options are available to allow the part to fit the application. The RC oscillator option saves system cost, while the LP crystal option saves power. These are discussed in detail in **Section 2.0 "Oscillator Configurations"**.

A complete discussion of device Resets and interrupts is available in previous sections of this data sheet.

In addition to their Power-up and Oscillator Start-up Timers provided for Resets, PIC18F1220/1320 devices have a Watchdog Timer, which is either permanently enabled via the Configuration bits, or software controlled (if configured as disabled).

The inclusion of an internal RC oscillator also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up, while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate Configuration register bits.

19.1 Configuration Bits

The Configuration bits can be programmed (read as '0'), or left unprogrammed (read as '1'), to select various device configurations. These bits are mapped starting at program memory location 300000h.

The user will note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h-3FFFFFh), which can only be accessed using table reads and table writes.

Programming the Configuration registers is done in a manner similar to programming the Flash memory. The EECON1 register WR bit starts a self-timed write to the Configuration register. In normal operation mode, a `TBLWT` instruction, with the TBLPTR pointing to the Configuration register, sets up the address and the data for the Configuration register write. Setting the WR bit starts a long write to the Configuration register. The Configuration registers are written a byte at a time. To write or erase a configuration cell, a `TBLWT` instruction can write a '1' or a '0' into the cell. For additional details on Flash programming, refer to **Section 6.5 "Writing to Flash Program Memory"**.

TABLE 19-1: CONFIGURATION BITS AND DEVICE IDS

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300001h	CONFIG1H	IESO	FSCM	—	—	FOSC3	FOSC2	FOSC1	FOSC0	11-- 1111
300002h	CONFIG2L	—	—	—	—	BORV1	BORV0	BOR	$\overline{\text{PWRTEN}}$	---- 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDT	---1 1111
300005h	CONFIG3H	MCLRE	—	—	—	—	—	—	—	1--- ----
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVR	1--- -1-1
300008h	CONFIG5L	—	—	—	—	—	—	CP1	CP0	---- --11
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	—	—	—	—	—	—	WRT1	WRT0	---- --11
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	—	—	—	—	—	—	EBTR1	EBTR0	---- --11
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1(1)	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx(1)
3FFFFFh	DEVID2(1)	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 0111

Legend: x = unknown, u = unchanged, – = unimplemented. Shaded cells are unimplemented, read as '0'.

Note 1: See Register 19-12 for DEVID1 values. DEVID registers are read-only and cannot be programmed by the user.

19.5.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM, regardless of the protection bit settings.

19.5.3 CONFIGURATION REGISTER PROTECTION

The Configuration registers can be write-protected. The WRTC bit controls protection of the Configuration registers. In normal execution mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

19.6 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are both readable and writable during normal execution through the TBLRD and TBLWT instructions, or during program/verify. The ID locations can be read when the device is code-protected.

19.7 In-Circuit Serial Programming

PIC18F1220/1320 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed (see Table 19-4).

Note:	The Timer1 oscillator shares the T1OSI and T1OSO pins with the PGD and PGC pins used for programming and debugging. When using the Timer1 oscillator, In-Circuit Serial Programming (ICSP) may not function correctly (high voltage or low voltage), or the In-Circuit Debugger (ICD) may not communicate with the controller. As a result of using either ICSP or ICD, the Timer1 crystal may be damaged. If ICSP or ICD operations are required, the crystal should be disconnected from the circuit (disconnect either lead), or installed after programming. The oscillator loading capacitors may remain in-circuit during ICSP or ICD operation.

TABLE 19-4: ICSP/ICD CONNECTIONS

Signal	Pin	Notes
PGD	RB7/PGD/T1OSI/P1D/KBI3	Shared with T1OSC – protect crystal
PGC	RB6/PGC/T1OSO/T13CKI/P1C/KBI2	Shared with T1OSC – protect crystal
$\overline{MCLR}$	$\overline{MCLR}$/VPP/RA5	
VDD	VDD	
VSS	VSS	
PGM	RB5/PGM/KBI1	Optional – pull RB5 low is LVP enabled

19.8 In-Circuit Debugger

When the DEBUG bit in Configuration register, CONFIG4L, is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 19-5 shows which resources are required by the background debugger.

TABLE 19-5: DEBUGGER RESOURCES

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	512 bytes
Data Memory:	10 bytes

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{MCLR}$/VPP, VDD, VSS, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip, or one of the third party development tool companies (see the note following **Section 19.7 "In-Circuit Serial Programming"** for more information).

20.0 INSTRUCTION SET SUMMARY

The PIC18 instruction set adds many enhancements to the previous PIC instruction sets, while maintaining an easy migration from these PIC instruction sets.

Most instructions are a single program memory word (16 bits), but there are three 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 20-1 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 20-1 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 three double-word instructions. These three instructions were made double-word instructions so that all the required information is available in these 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 μs. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 20-1 shows the general formats that the instructions can have.

All examples use the format ‘`nnh`’ to represent a hexadecimal number, where ‘`h`’ signifies a hexadecimal digit.

The Instruction Set Summary, shown in Table 20-1, lists the instructions recognized by the Microchip Assembler (MPASM™). **Section 20.2 “Instruction Set”** provides a description of each instruction.

20.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified and the result is stored according to either the instruction or the destination designator ‘d’. A read operation is performed on a register even if the instruction writes to that register.

For example, a “`BCF PORTB,1`” instruction will read PORTB, clear bit 1 of the data, then write the result back to PORTB. The read operation would have the unintended result that any condition that sets the RBIF flag would be cleared. The R-M-W operation may also copy the level of an input pin to its corresponding output latch.

DAW — Decimal Adjust W Register

Syntax: [*label*] DAW

Operands: None

Operation: If [W<3:0> > 9] or [DC = 1] then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>;

If [W<7:4> > 9] or [C = 1] then
(W<7:4>) + 6 → W<7:4>;
else
(W<7:4>) → W<7:4>;

Status Affected: C, DC

Encoding:

0000	0000	0000	0111

Description: `DAW` adjusts the 8-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result. The Carry bit may be set by `DAW` regardless of its setting prior to the `DAW` instruction.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example 1: `DAW`

Before Instruction
W = 0xA5
C = 0
DC = 0

After Instruction
W = 0x05
C = 1
DC = 0

Example 2:

Before Instruction
W = 0xCE
C = 0
DC = 0

After Instruction
W = 0x34
C = 1
DC = 0

DECF — Decrement f

Syntax: [*label*] DECF 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:

0000	01da	ffff	ffff

Description: Decrement register ‘f’. 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’ = 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: `DECF CNT`

Before Instruction
CNT = 0x01
Z = 0

After Instruction
CNT = 0x00
Z = 1

INCFSZ **Increment f, skip if 0**

Syntax:	[*label*] INCFSZ f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) + 1 → dest, skip if result = 0
Status Affected:	None
Encoding:	0011 11da 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 the result is '0', the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a 2-cycle instruction. 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(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE    INCFSZ   CNT
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address (ZERO)
If CNT ≠ 0;
PC = Address (NZERO)

INFSNZ **Increment f, skip if not 0**

Syntax:	[*label*] INFSNZ f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) + 1 → dest, skip if result ≠ 0
Status Affected:	None
Encoding:	0100 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 the result is not '0', the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a 2-cycle instruction. 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(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

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

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE    INFSNZ REG
ZERO
NZERO
```

Before Instruction
PC = Address (HERE)

After Instruction
REG = REG + 1
If REG ≠ 0;
PC = Address (NZERO)
If REG = 0;
PC = Address (ZERO)

SLEEP — Enter Sleep mode

Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT postscaler, 1 → $\overline{\text{TO}}$, 0 → $\overline{\text{PD}}$
Status Affected:	$\overline{\text{TO}}$, $\overline{\text{PD}}$
Encoding:	`0000` `0000` `0000` `0011`
Description:	The Power-down Status bit ($\overline{\text{PD}}$) is cleared. The Time-out status bit ($\overline{\text{TO}}$) is set. The Watchdog Timer and its postscaler are cleared. The processor is put into Sleep mode with the oscillator stopped.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to Sleep

Example: `SLEEP`

Before Instruction
- $\overline{\text{TO}}$ = ?
- $\overline{\text{PD}}$ = ?

After Instruction
- $\overline{\text{TO}}$ = 1 †
- $\overline{\text{PD}}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB — Subtract f from W with borrow

Syntax:	[*label*] SUBFWB f [,d [,a]]
Operands:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	(W) – (f) – ($\overline{\text{C}}$) → dest
Status Affected:	N, OV, C, DC, Z
Encoding:	`0101` `01da` `ffff` `ffff`
Description:	Subtract register 'f' and Carry flag (borrow) from W (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored 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: `SUBFWB REG`

Before Instruction
- REG = 0x03
- W = 0x02
- C = 0x01

After Instruction
- REG = 0xFF
- W = 0x02
- C = 0x00
- Z = 0x00
- N = 0x01 ; result is negative

Example 2: `SUBFWB REG, 0, 0`

Before Instruction
- REG = 2
- W = 5
- C = 1

After Instruction
- REG = 2
- W = 3
- C = 1
- Z = 0
- N = 0 ; result is positive

Example 3: `SUBFWB REG, 1, 0`

Before Instruction
- REG = 1
- W = 2
- C = 0

After Instruction
- REG = 0
- W = 2
- C = 1
- Z = 1 ; result is zero
- N = 0

TSTFSZ **Test f, skip if 0**

Syntax: [*label*] TSTFSZ f [,a]

Operands: 0 ≤ f ≤ 255
a ∈ [0,1]

Operation: skip if f = `0`

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If ‘f’ = `0`, the next instruction, fetched during the current instruction execution is discarded and a `NOP` is executed, making this a 2-cycle instruction. If ‘a’ is ‘`0`’, the Access Bank 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(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    TSTFSZ CNT
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)

After Instruction
If CNT = 0x00,
PC = Address (ZERO)
If CNT ≠ 0x00,
PC = Address (NZERO)

XORLW **Exclusive OR literal with W**

Syntax: [*label*] XORLW k

Operands: 0 ≤ k ≤ 255

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XOR’ed with the 8-bit literal ‘k’. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write to W

Example: `XORLW 0xAF`

Before Instruction
W = 0xB5

After Instruction
W = 0x1A

23.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

Note: The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.

“Typical” represents the mean of the distribution at 25°C. “Maximum” or “minimum” represents (mean + 3σ) or (mean – 3σ) respectively, where σ is a standard deviation, over the whole temperature range.

FIGURE 23-1: TYPICAL IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, +25°C

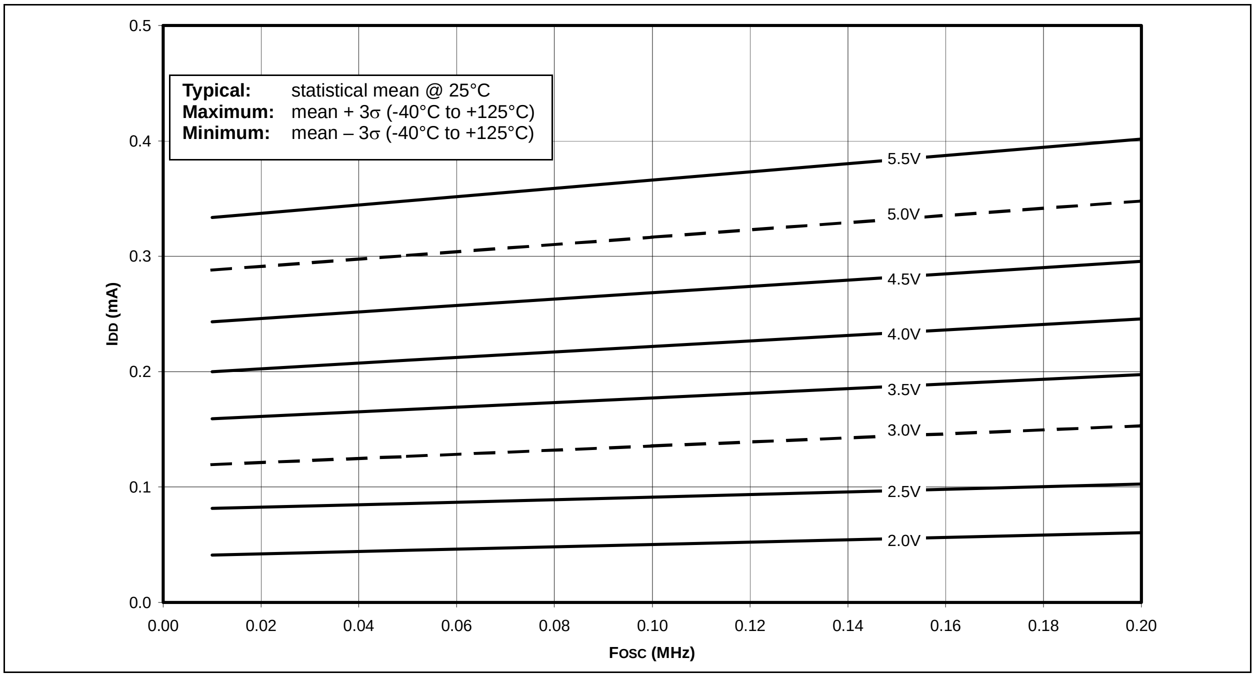


FIGURE 23-2: MAXIMUM IDD vs. FOSC OVER VDD PRI_RUN, EC MODE, -40°C TO +85°C

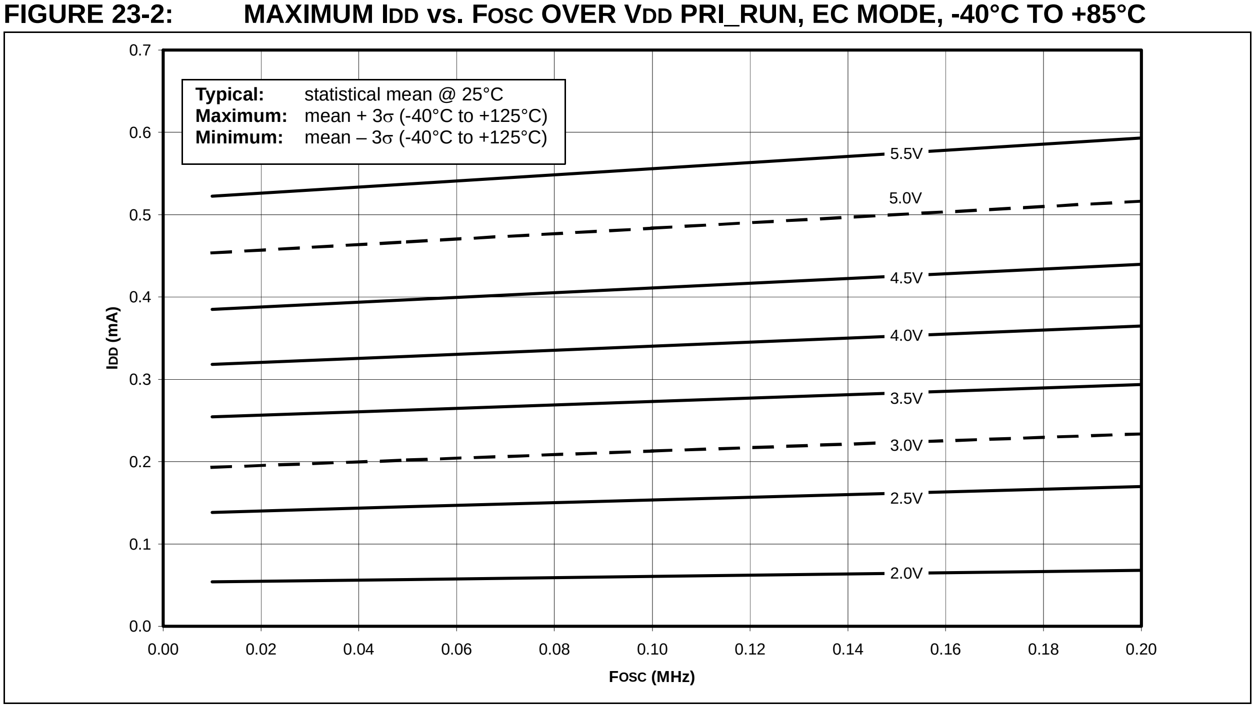


FIGURE 23-31: ΔIPD LVD vs. VDD SLEEP MODE, LVDL3:LVDL0 = 0001 (2V)

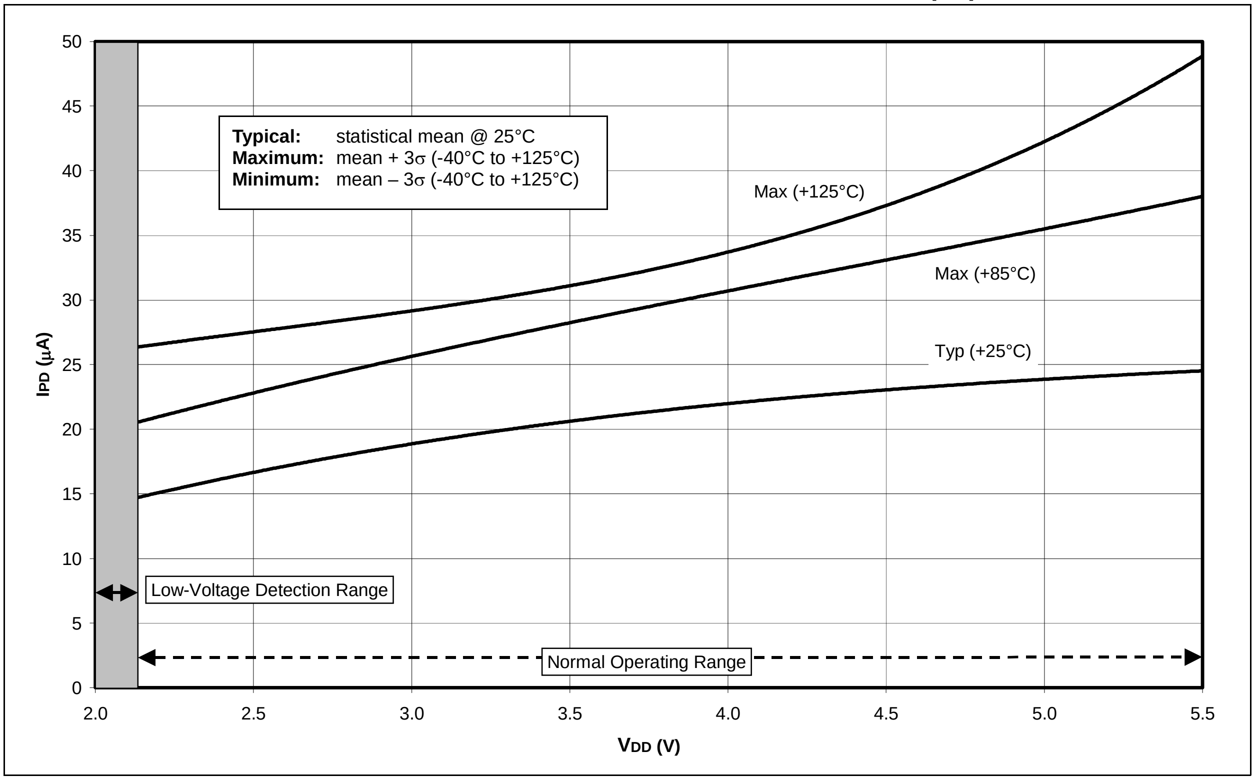
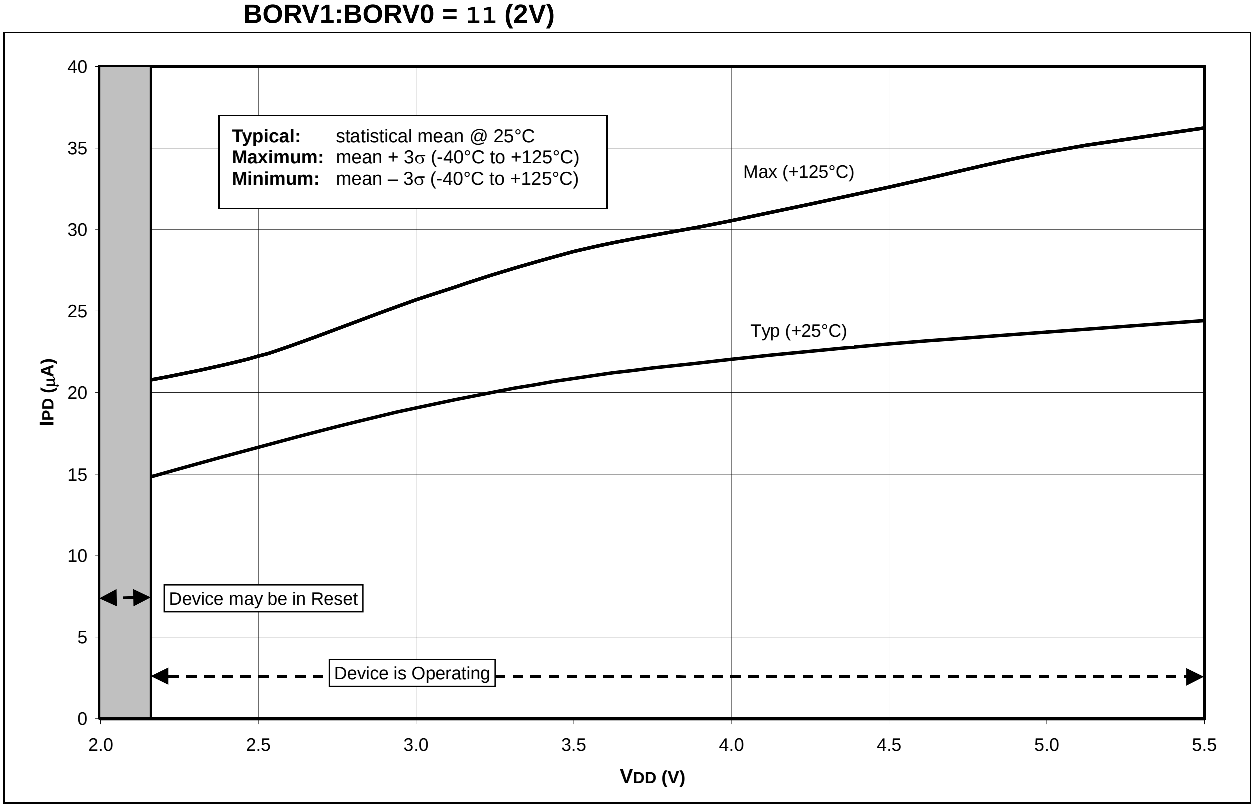


FIGURE 23-32: ΔIPD BOR vs. VDD, -40°C TO +125°C SLEEP MODE, BORV1:BORV0 = 11 (2V)

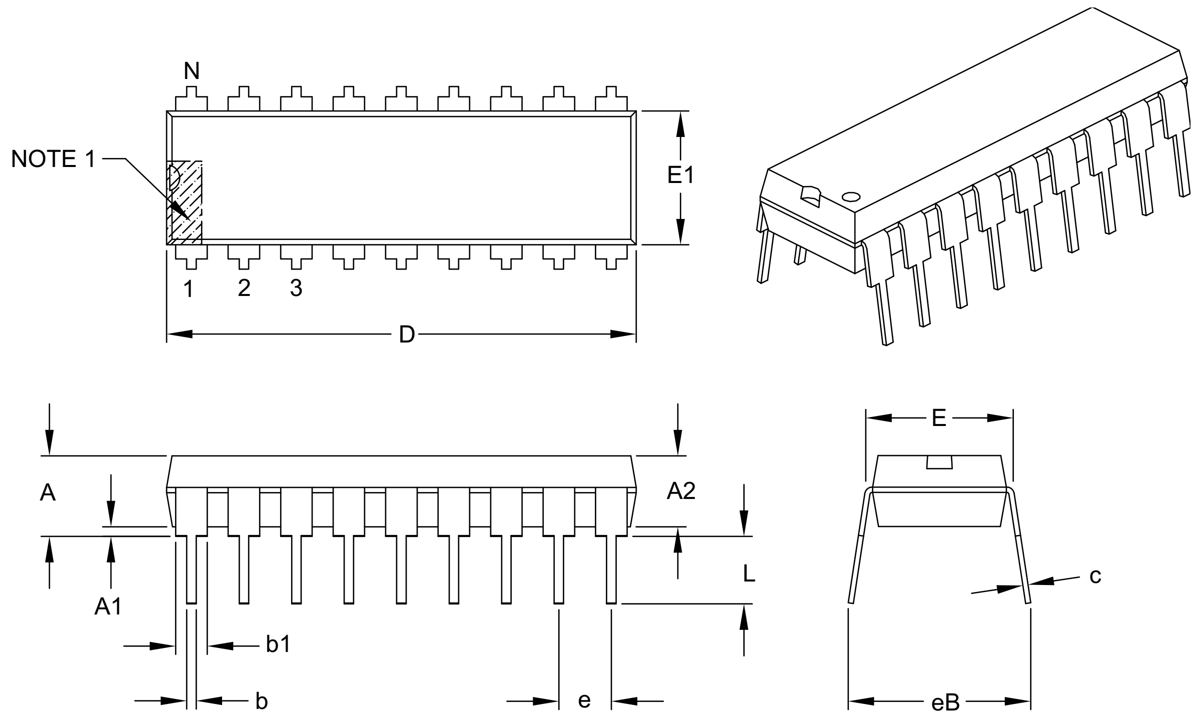


24.2 Package Details

The following sections give the technical details of the packages.

18-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

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



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		18	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.300	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.880	.900	.920
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.014
Upper Lead Width	b1	.045	.060	.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 C04-007B