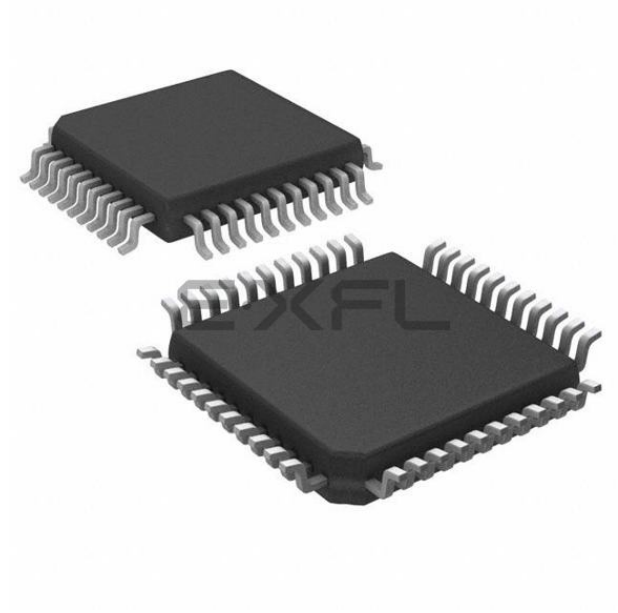


Welcome to E-XFL.COM

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	4MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	33
Program Memory Size	14KB (8K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	368 x 8
Voltage - Supply (Vcc/Vdd)	4V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-QFP
Supplier Device Package	44-MQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16c67-04i-pq

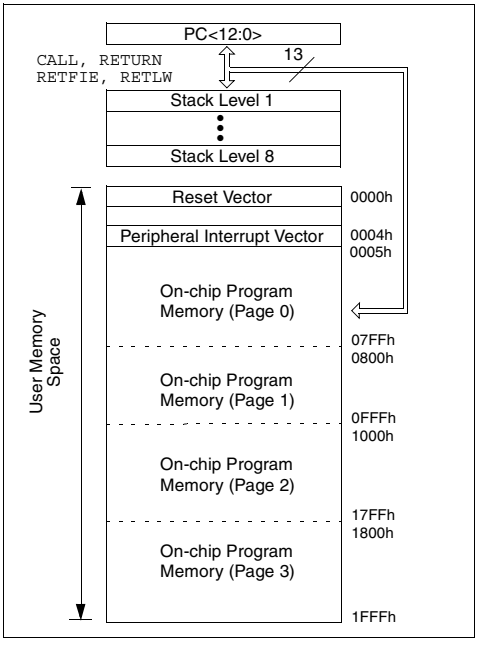
TABLE 1-1: PIC16C6X FAMILY OF DEVICES

		PIC16C61	PIC16C62A	PIC16CR62	PIC16C63	PIC16CR63
Clock	Maximum Frequency of Operation (MHz)	20	20	20	20	20
Memory	EPROM Program Memory (x14 words)	1K	2K	—	4K	—
	ROM Program Memory (x14 words)	—	—	2K	—	4K
	Data Memory (bytes)	36	128	128	192	192
Peripherals	Timer Module(s)	TMR0	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2
	Capture/Compare/ PWM Module(s)	—	1	1	2	2
	Serial Port(s) (SPI/I^2C, USART)	—	SPI/I^2C	SPI/I^2C	SPI/I^2C, USART	SPI/I^2C USART
	Parallel Slave Port	—	—	—	—	—
Features	Interrupt Sources	3	7	7	10	10
	I/O Pins	13	22	22	22	22
	Voltage Range (Volts)	3.0-6.0	2.5-6.0	2.5-6.0	2.5-6.0	2.5-6.0
	In-Circuit Serial Programming	Yes	Yes	Yes	Yes	Yes
	Brown-out Reset	—	Yes	Yes	Yes	Yes
	Packages	18-pin DIP, SO	28-pin SDIP, SOIC, SSOP	28-pin SDIP, SOIC, SSOP	28-pin SDIP, SOIC	28-pin SDIP, SOIC

		PIC16C64A	PIC16CR64	PIC16C65A	PIC16CR65	PIC16C66	PIC16C67
Clock	Maximum Frequency of Operation (MHz)	20	20	20	20	20	20
Memory	EPROM Program Memory (x14 words)	2K	—	4K	—	8K	8K
	ROM Program Memory (x14 words)	—	2K	—	4K	—	—
	Data Memory (bytes)	128	128	192	192	368	368
Peripherals	Timer Module(s)	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2
	Capture/Compare/PWM Module(s)	1	1	2	2	2	2
	Serial Port(s) (SPI/I^2C, USART)	SPI/I^2C	SPI/I^2C	SPI/I^2C, USART	SPI/I^2C, USART	SPI/I^2C, USART	SPI/I^2C, USART
	Parallel Slave Port	Yes	Yes	Yes	Yes	—	Yes
Features	Interrupt Sources	8	8	11	11	10	11
	I/O Pins	33	33	33	33	22	33
	Voltage Range (Volts)	2.5-6.0	2.5-6.0	2.5-6.0	2.5-6.0	2.5-6.0	2.5-6.0
	In-Circuit Serial Programming	Yes	Yes	Yes	Yes	Yes	Yes
	Brown-out Reset	Yes	Yes	Yes	Yes	Yes	Yes
	Packages	40-pin DIP; 44-pin PLCC, MQFP, TQFP	40-pin DIP; 44-pin PLCC, MQFP, TQFP	40-pin DIP; 44-pin PLCC, MQFP, TQFP	40-pin DIP; 44-pin PLCC, MQFP, TQFP	28-pin SDIP, SOIC	40-pin DIP; 44-pin PLCC, MQFP, TQFP

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability. All PIC16C6X Family devices use serial programming with clock pin RB6 and data pin RB7.

FIGURE 4-4: PIC16C66/67 PROGRAM MEMORY MAP AND STACK



4.2 Data Memory Organization

Applicable Devices													
61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

The data memory is partitioned into multiple banks which contain the General Purpose Registers and the Special Function Registers. Bits RP1 and RP0 are the bank select bits.

RP1:RP0 (STATUS<6:5>)
= 00 → Bank0
= 01 → Bank1
= 10 → Bank2
= 11 → Bank3

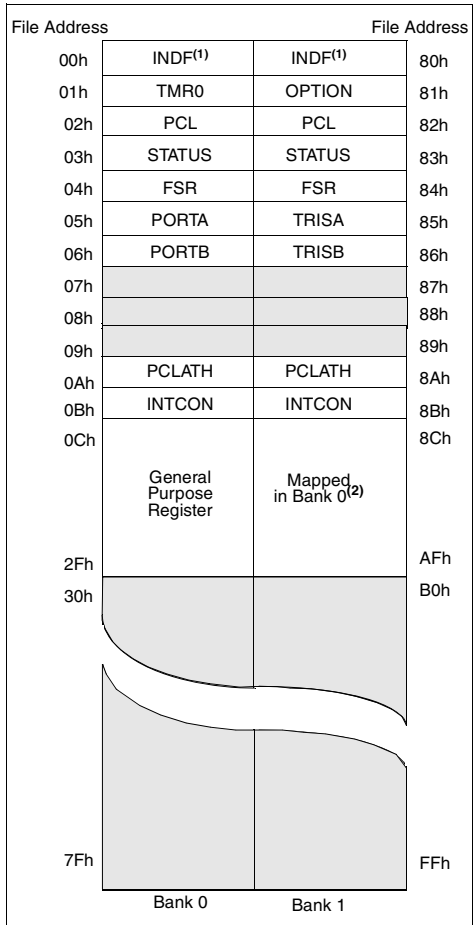
Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers, implemented as static RAM. All implemented banks contain special function registers. Some “high use” special function registers from one bank may be mirrored in another bank for code reduction and quicker access.

4.2.1 GENERAL PURPOSE REGISTERS

These registers are accessed either directly or indirectly through the File Select Register (FSR) (Section 4.5).

For the PIC16C61, general purpose register locations 8Ch-AFh of Bank 1 are not physically implemented. These locations are mapped into 0Ch-2Fh of Bank 0.

FIGURE 4-5: PIC16C61 REGISTER FILE MAP



Unimplemented data memory location; read as '0'.
Note 1: Not a physical register.
2: These locations are unimplemented in Bank 1. Any access to these locations will access the corresponding Bank 0 register.

Example 4-1 shows the calling of a subroutine in page 1 of the program memory. This example assumes that the PCLATH is saved and restored by the interrupt service routine (if interrupts are used).

EXAMPLE 4-1: CALL OF A SUBROUTINE IN PAGE 1 FROM PAGE 0

```
ORG 0x500
BSF     PCLATH,3  ;Select page 1 (800h-FFFh)
BCF     PCLATH,4  ;Only on >4K devices
CALL    SUB1_P1   ;Call subroutine in
        :         ;page 1 (800h-FFFh)
        :
        :
ORG 0x900
SUB1_P1:          ;called subroutine
        :         ;page 1 (800h-FFFh)
        :
RETURN            ;return to Call subroutine
                  ;in page 0 (000h-7FFh)
```

4.5 Indirect Addressing, INDF and FSR Registers

Applicable Devices													
61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

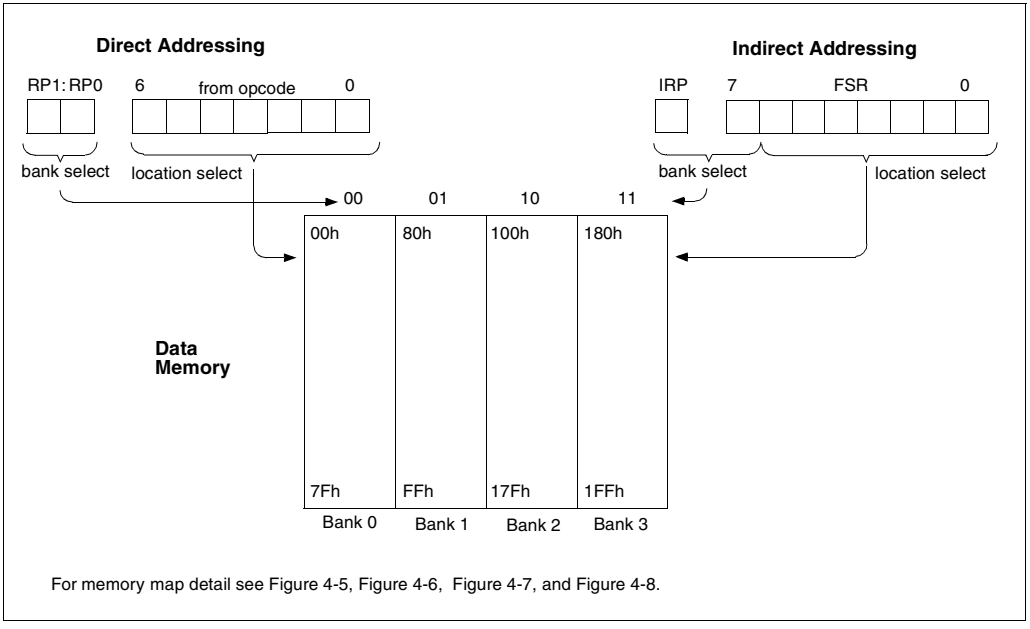
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself indirectly (FSR = '0') will produce 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-25.

A simple program to clear RAM location 20h-2Fh using indirect addressing is shown in Example 4-2.

EXAMPLE 4-2: INDIRECT ADDRESSING

```
          movlw  0x20    ;initialize pointer
          movwf  FSR     ;  to RAM
NEXT      clrf   INDF    ;clear INDF register
          incf   FSR,F   ;inc pointer
          btfss  FSR,4   ;all done?
          goto   NEXT    ;NO, clear next
CONTINUE
          :              ;YES, continue
```

FIGURE 4-25: DIRECT/INDIRECT ADDRESSING



For memory map detail see Figure 4-5, Figure 4-6, Figure 4-7, and Figure 4-8.

NOTES:

NOTES:

FIGURE 7-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2

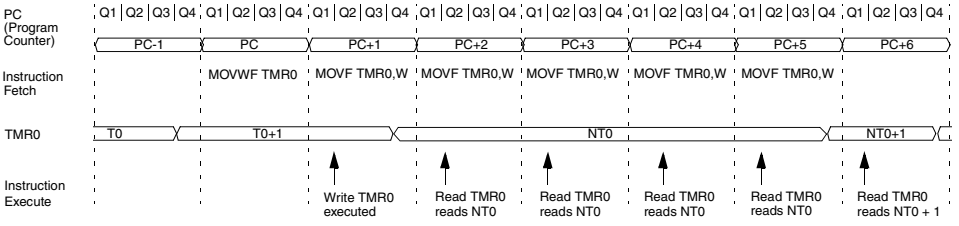
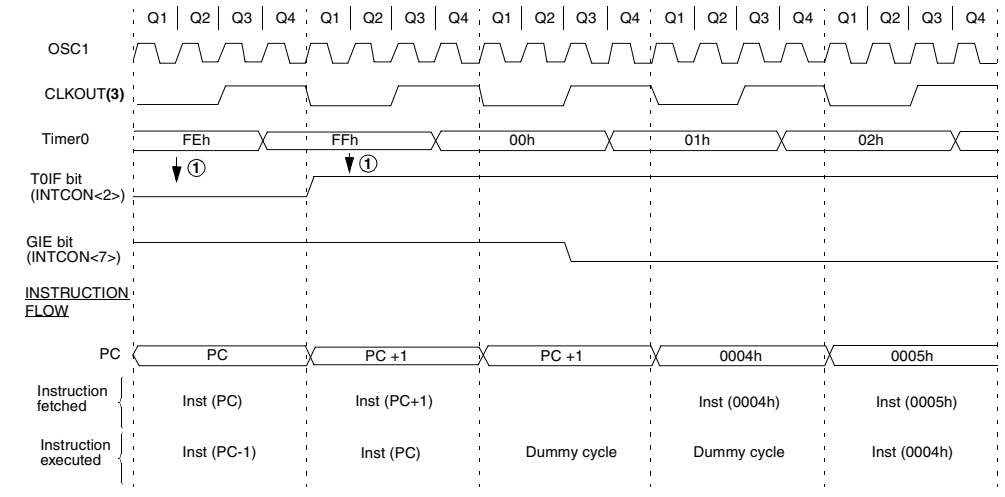


FIGURE 7-4: TMR0 INTERRUPT TIMING



Note 1: Interrupt flag bit T0IF is sampled here (every Q1).
2: Interrupt latency = 4Tcy where Tcy = instruction cycle time.
3: CLKOUT is available only in RC oscillator mode.

8.0 TIMER1 MODULE

Applicable Devices													
61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

Timer1 is a 16-bit timer/counter consisting of two 8-bit registers (TMR1H and TMR1L) which are readable and writable. Register TMR1 (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The TMR1 Interrupt, if enabled, is generated on overflow which is latched in interrupt flag bit TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing the TMR1 interrupt enable bit TMR1IE (PIE1<0>).

Timer1 can operate in one of two modes:

- As a timer
- As a counter

The operating mode is determined by clock select bit, TMR1CS (T1CON<1>) (Figure 8-2).

In timer mode, Timer1 increments every instruction cycle. In counter mode, it increments on every rising edge of the external clock input.

Timer1 can be enabled/disabled by setting/clearing control bit TMR1ON (T1CON<0>).

Timer1 also has an internal “reset input”. This reset can be generated by CCP1 or CCP2 (Capture/Compare/PWM) module. See Section 10.0 for details. Figure 8-1 shows the Timer1 control register.

For the PIC16C62A/R62/63/R63/64A/R64/65A/R65/R66/67, when the Timer1 oscillator is enabled (T1OSCEN is set), the RC1 and RC0 pins become inputs. That is, the TRISC<1:0> value is ignored.

For the PIC16C62/64/65, when the Timer1 oscillator is enabled (T1OSCEN is set), RC1 pin becomes an input, however the RC0 pin will have to be configured as an input by setting the TRISC<0> bit.

The Timer1 module also has a software programmable prescaler.

FIGURE 8-1: T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit7							bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as ‘0’
- n = Value at POR reset

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

bit 5-4: **T1CKPS1:T1CKPS0**: Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3: **T1OSCEN**: Timer1 Oscillator Enable Control bit
1 = Oscillator is enabled
0 = Oscillator is shut off
Note: The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2: **$\overline{\text{T1SYNC}}$**: Timer1 External Clock Input Synchronization Control bit

TMR1CS = 1
1 = Do not synchronize external clock input
0 = Synchronize external clock input

TMR1CS = 0
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1: **TMR1CS**: Timer1 Clock Source Select bit
1 = External clock from T1OSI (on the rising edge) (See pinouts for pin with T1OSI function)
0 = Internal clock (Fosc/4)

bit 0: **TMR1ON**: Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

FIGURE 12-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D
bit7							bit0

R	= Readable bit
W	= Writable bit
U	= Unimplemented bit, read as '0'
- n	= Value at POR reset
x	= unknown

bit 7: **SPEN**: Serial Port Enable bit
(Configures RC7/RX/DT and RC6/TX/CK pins as serial port pins when bits TRISC<7:6> are set)
1 = Serial port enabled
0 = Serial port disabled

bit 6: **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5: **SREN**: Single Receive Enable bit

Asynchronous mode
Don't care

Synchronous mode - master
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.

Synchronous mode - slave
Unused in this mode

bit 4: **CREN**: Continuous Receive Enable bit

Asynchronous mode
1 = Enables continuous receive
0 = Disables continuous receive

Synchronous mode
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

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

bit 2: **FERR**: Framing Error bit
1 = Framing error (Can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1: **OERR**: Overrun Error bit
1 = Overrun error (Can be cleared by clearing bit CREN)
0 = No overrun error

bit 0: **RX9D**: 9th bit of received data (Can be parity bit)

13.8 Power-down Mode (SLEEP)

Applicable Devices													
61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

Power-down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, status bit $\overline{PD}$ (STATUS<3>) is cleared, status bit $\overline{TO}$ (STATUS<4>) is set, and the oscillator driver is turned off. The I/O ports maintain the status they had before the SLEEP instruction was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, place all I/O pins at either VDD, or VSS, ensure no external circuitry is drawing current from the I/O pin, and disable external clocks. Pull all I/O pins, that are hi-impedance inputs, high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should also be at VDD or VSS for lowest current consumption. The contribution from on-chip pull-ups on PORTB should be considered.

The $\overline{MCLR}$/VPP pin must be at a logic high level (VIHMC).

13.8.1 WAKE-UP FROM SLEEP

The device can wake from SLEEP through one of the following events:

1. External reset input on $\overline{MCLR}$/VPP pin.
2. Watchdog Timer Wake-up (if WDT was enabled).
3. Interrupt from RB0/INT pin, RB port change, or some peripheral interrupts.

External $\overline{MCLR}$ Reset will cause a device reset. All other events are considered a continuation of program execution and cause a “wake-up”. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of device reset. The $\overline{PD}$ bit, which is set on power-up is cleared when SLEEP is invoked. The $\overline{TO}$ bit is cleared if WDT time-out occurred (and caused wake-up).

The following peripheral interrupts can wake the device from SLEEP:

1. TMR1 interrupt. Timer1 must be operating as an asynchronous counter.
2. SSP (Start/Stop) bit detect interrupt.
3. SSP transmit or receive in slave mode (SPI/I^2C).
4. CCP capture mode interrupt.
5. Parallel Slave Port read or write.
6. USART TX or RX (synchronous slave mode).

Other peripherals can not generate interrupts since during SLEEP, no on-chip Q clocks are present.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

13.8.2 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a SLEEP instruction, the SLEEP instruction will complete as a NOP. Therefore, the WDT and WDT postscaler will not be cleared, the $\overline{TO}$ bit will not be set and $\overline{PD}$ bits will not be cleared.
- If the interrupt occurs **during or after** the execution of a SLEEP instruction, the device will immediately wake up from sleep. The SLEEP instruction will be completely executed before the wake-up. Therefore, the WDT and WDT postscaler will be cleared, the $\overline{TO}$ bit will be set and the $\overline{PD}$ bit will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

GOTO **Unconditional Branch**

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 2047$

Operation: $k \rightarrow PC<10:0>$
$PCLATH<4:3> \rightarrow PC<12:11>$

Status Affected: None

Encoding:

10	1kkk	kkkk	kkkk

Description: GOTO is an unconditional branch. The eleven bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

	Q1	Q2	Q3	Q4
1st Cycle	Decode	Read literal 'k'	Process data	Write to PC
2nd Cycle	No-Operation	No-Operation	No-Operation	No-Operation

Example GOTO THERE

After Instruction
PC = Address THERE

INCF **Increment f**

Syntax: [*label*] INCF f,d

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) + 1 → (destination)

Status Affected: Z

Encoding:

00	1010	dfff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example INCF CNT, 1

Before Instruction
CNT = 0xFF
Z = 0
After Instruction
CNT = 0x00
Z = 1

IORWF **Inclusive OR W with f**

Syntax: [*label*] IORWF f,d

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (W) .OR. (f) → (destination)

Status Affected: Z

Encoding:

00	0100	dfff	ffff

Description: Inclusive OR the W register with register 'f'. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example
```
IORWF   RESULT, 0
```
Before Instruction
RESULT = 0x13
W = 0x91
After Instruction
RESULT = 0x13
W = 0x93
Z = 1

MOVF **Move f**

Syntax: [*label*] MOVF f,d

Operands: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f) → (destination)

Status Affected: Z

Encoding:

00	1000	dfff	ffff

Description: The contents of register f is moved to a destination dependant upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example
```
MOVF    FSR, 0
```
After Instruction
W = value in FSR register
Z = 1

MOVLW **Move Literal to W**

Syntax: [*label*] MOVLW k

Operands: $0 \le k \le 255$

Operation: k → (W)

Status Affected: None

Encoding:

11	00xx	kkkk	kkkk

Description: The eight bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W

Example
```
MOVLW   0x5A
```
After Instruction
W = 0x5A

MOVWF **Move W to f**

Syntax: [*label*] MOVWF f

Operands: $0 \le f \le 127$

Operation: (W) → (f)

Status Affected: None

Encoding:

00	0000	1fff	ffff

Description: Move data from W register to register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example
```
MOVWF   OPTION_REG
```
Before Instruction
OPTION = 0xFF
W = 0x4F
After Instruction
OPTION = 0x4F
W = 0x4F

RLF **Rotate Left f through Carry**

Syntax: [*label*] RLF f,d

Operands: $0 \le f \le 127$
$d \in [0,1]$

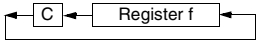
Operation: See description below

Status Affected: C

Encoding:

00	1101	dfff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is stored back in register 'f'.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example
```
RLF     REG1,0
```
Before Instruction
```
REG1  =  1110 0110
C     =  0
```
After Instruction
```
REG1  =  1110 0110
W     =  1100 1100
C     =  1
```

RRF **Rotate Right f through Carry**

Syntax: [*label*] RRF f,d

Operands: $0 \le f \le 127$
$d \in [0,1]$

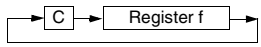
Operation: See description below

Status Affected: C

Encoding:

00	1100	dfff	ffff

Description: The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.



Words: 1

Cycles: 1

Q Cycle Activity:

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

Example
```
RRF     REG1,0
```
Before Instruction
```
REG1  =  1110 0110
C     =  0
```
After Instruction
```
REG1  =  1110 0110
W     =  0111 0011
C     =  0
```

Applicable Devices	61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

FIGURE 16-2: TYPICAL RC OSCILLATOR FREQUENCY VS. VDD

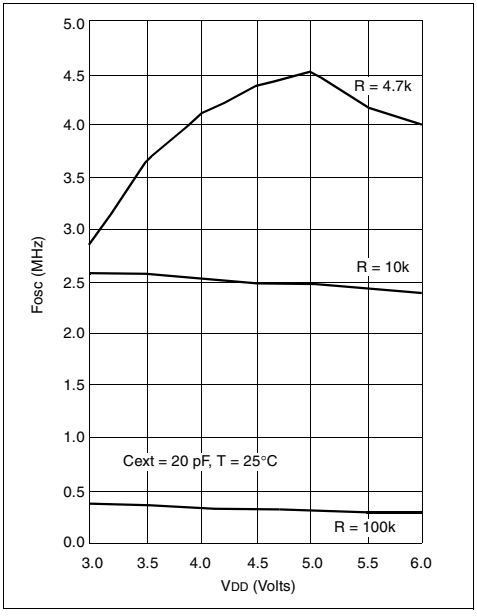


FIGURE 16-3: TYPICAL RC OSCILLATOR FREQUENCY VS. VDD

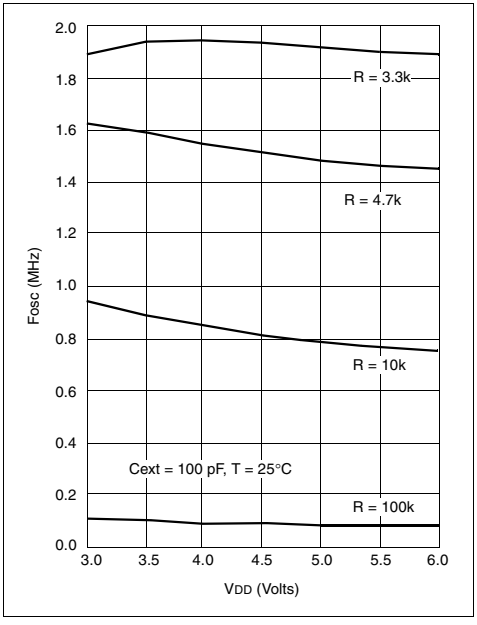


FIGURE 16-4: TYPICAL RC OSCILLATOR FREQUENCY VS. VDD

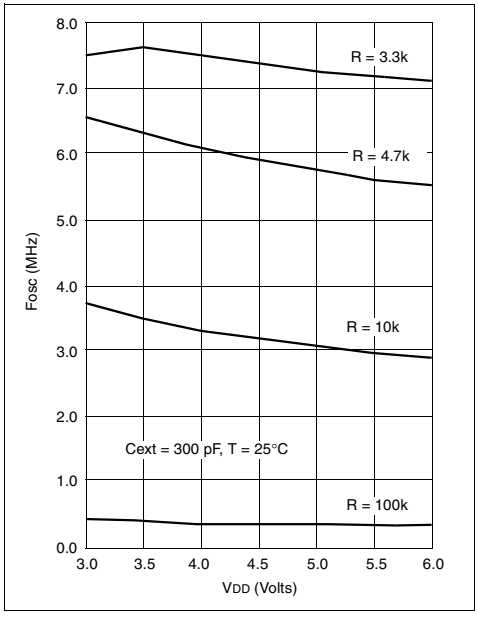
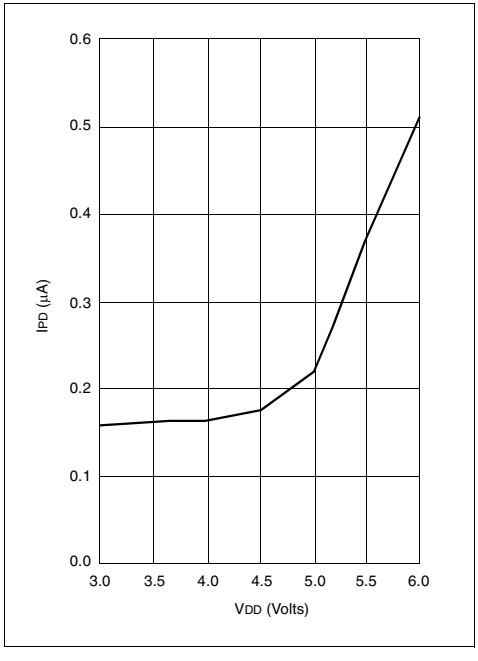


FIGURE 16-5: TYPICAL IPD VS. VDD WATCHDOG TIMER DISABLED 25°C



Data based on matrix samples. See first page of this section for details.

Applicable Devices	61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

17.5 Timing Diagrams and Specifications

FIGURE 17-2: EXTERNAL CLOCK TIMING

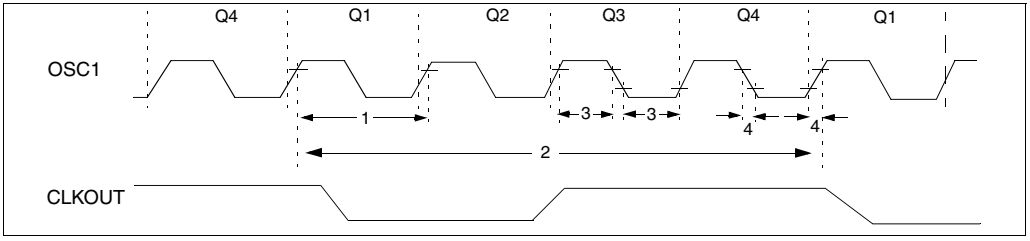


TABLE 17-2: EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fosc	**External CLKIN Frequency (Note 1)**	DC	—	4	MHz	XT and RC osc mode
			DC	—	4	MHz	HS osc mode (-04)
			DC	—	10	MHz	HS osc mode (-10)
			DC	—	20	MHz	HS osc mode (-20)
			DC	—	200	kHz	LP osc mode
		Oscillator Frequency (Note 1)	DC	—	4	MHz	RC osc mode
			0.1	—	4	MHz	XT osc mode
			4	—	20	MHz	HS osc mode
			5	—	200	kHz	LP osc mode
1	Tosc	**External CLKIN Period (Note 1)**	250	—	—	ns	XT and RC osc mode
			250	—	—	ns	HS osc mode (-04)
			100	—	—	ns	HS osc mode (-10)
			50	—	—	ns	HS osc mode (-20)
			5	—	—	µs	LP osc mode
		Oscillator Period (Note 1)	250	—	—	ns	RC osc mode
			250	—	10,000	ns	XT osc mode
			250	—	250	ns	HS osc mode (-04)
			100	—	250	ns	HS osc mode (-10)
			50	—	1,000	ns	HS osc mode (-20)
			5	—	—	µs	LP osc mode
2	TCY	**Instruction Cycle Time (Note 1)**	200	TCY	DC	ns	TCY = 4/FOSC
3	TosL, TosH	**External Clock in (OSC1) High or Low Time**	100	—	—	ns	XT oscillator
			2.5	—	—	µs	LP oscillator
			15	—	—	ns	HS oscillator
4	TosR, TosF	**External Clock in (OSC1) Rise or Fall Time**	—	—	25	ns	XT oscillator
			—	—	50	ns	LP oscillator
			—	—	15	ns	HS oscillator

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min." values with an external clock applied to the OSC1/CLKIN pin. When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices.

Applicable Devices	61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

18.4 Timing Parameter Symbology

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST (I^2C specifications only)
4. Ts (I^2C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

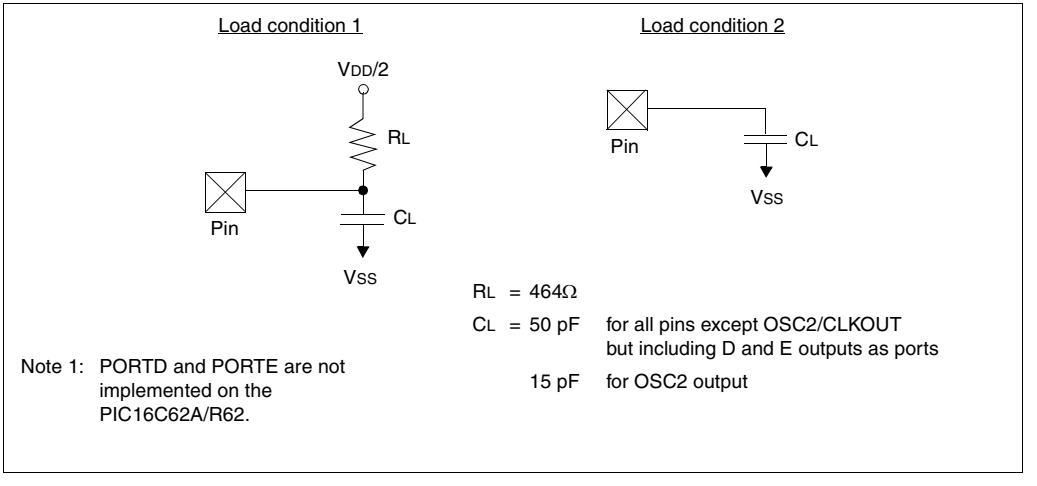
Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-impedance
I^2C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I^2C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	STOP condition
STA	START condition		

FIGURE 18-1: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



Note 1: PORTD and PORTE are not implemented on the PIC16C62A/R62.

Applicable Devices	61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

20.4 Timing Parameter Symbology

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST (I^2C specifications only)
4. Ts (I^2C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

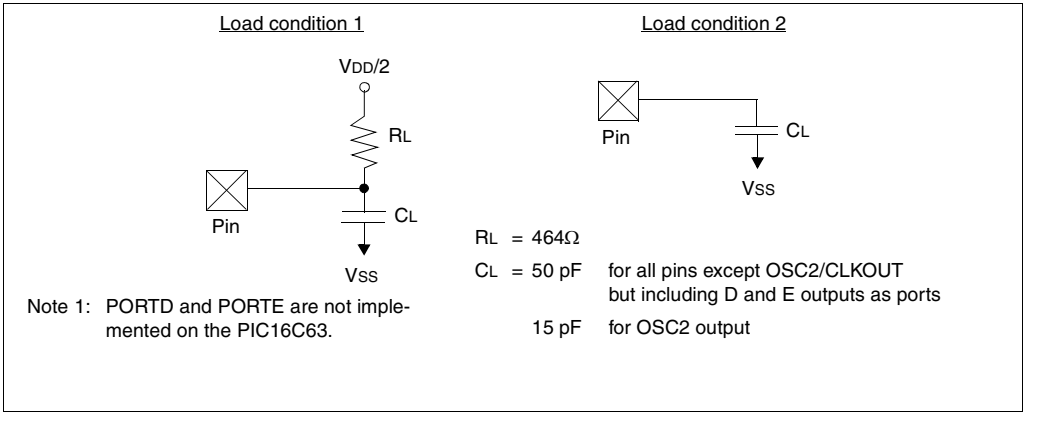
Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-impedance
I^2C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I^2C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	STOP condition
STA	START condition		

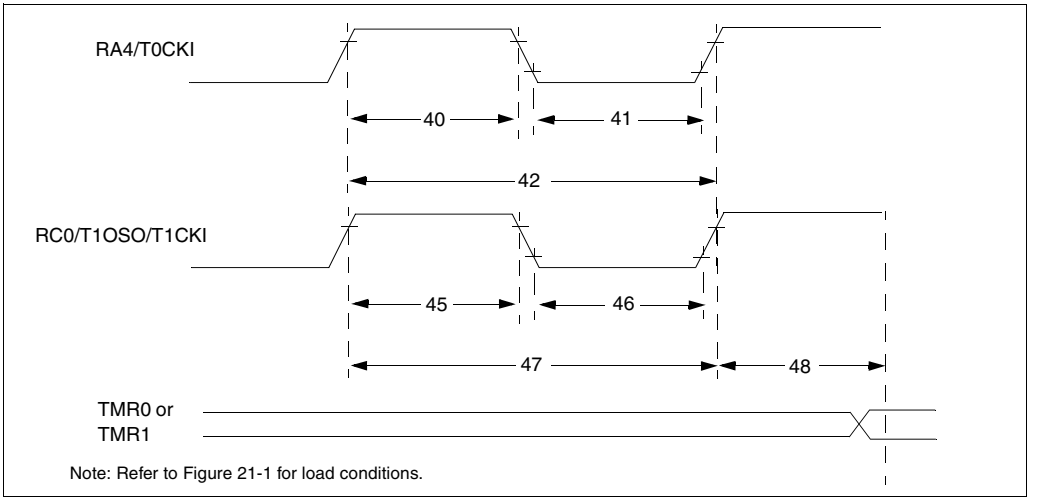
FIGURE 20-1: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



Note 1: PORTD and PORTE are not implemented on the PIC16C63.

Applicable Devices	61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

FIGURE 21-6: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-5: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Sym	Characteristic			Min	Typ†	Max	Units	Conditions
40*	Tt0H	T0CKI High Pulse Width		No Prescaler	0.5TCY + 20	—	—	ns	Must also meet parameter 42
				With Prescaler	10	—	—	ns	
41*	Tt0L	T0CKI Low Pulse Width		No Prescaler	0.5TCY + 20	—	—	ns	Must also meet parameter 42
				With Prescaler	10	—	—	ns	
42*	Tt0P	T0CKI Period		No Prescaler	TCY + 40	—	—	ns	
				With Prescaler	Greater of: 20 or TCY + 40 / N	—	—	ns	N = prescale value (2, 4, ..., 256)
45*	Tt1H	T1CKI High Time	Synchronous, Prescaler = 1		0.5TCY + 20	—	—	ns	Must also meet parameter 47
			Synchronous, Prescaler = 2,4,8	PIC16**C**6X	15	—	—	ns	
				PIC16**LC**6X	25	—	—	ns	
			Asynchronous	PIC16**C**6X	30	—	—	ns	
				PIC16**LC**6X	50	—	—	ns	
46*	Tt1L	T1CKI Low Time	Synchronous, Prescaler = 1		0.5TCY + 20	—	—	ns	Must also meet parameter 47
			Synchronous, Prescaler = 2,4,8	PIC16**C**6X	15	—	—	ns	
				PIC16**LC**6X	25	—	—	ns	
			Asynchronous	PIC16**C**6X	30	—	—	ns	
				PIC16**LC**6X	50	—	—	ns	
47*	Tt1P	T1CKI input period	Synchronous	PIC16**C**6X	Greater of: 30 OR TCY + 40 / N	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16**LC**6X	Greater of: 50 OR TCY + 40 / N				N = prescale value (1, 2, 4, 8)
			Asynchronous	PIC16**C**6X	60	—	—	ns	
				PIC16**LC**6X	100	—	—	ns	
	Ft1	Timer1 oscillator input frequency range (oscillator enabled by setting bit T1OSCEN)			DC	—	200	kHz	
48	TCKEZtmr1	Delay from external clock edge to timer increment			2Tosc	—	7Tosc	—	

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Applicable Devices	61	62	62A	R62	63	R63	64	64A	R64	65	65A	R65	66	67

23.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES FOR: PIC16C62, PIC16C62A, PIC16CR62, PIC16C63, PIC16C64, PIC16C64A, PIC16CR64, PIC16C65A, PIC16C66, PIC16C67

The graphs and tables provided in this section are for design guidance and are not tested or guaranteed.

In some graphs or tables the data presented are outside specified operating range (i.e., outside specified VDD range). This is for information only and devices are guaranteed to operate properly only within the specified range.

Note: The data presented in this section is a statistical summary of data collected on units from different lots over a period of time and matrix samples. 'Typical' represents the mean of the distribution at, 25°C, while 'max' or 'min' represents (mean +3σ) and (mean -3σ) respectively where σ is standard deviation.

FIGURE 23-1: TYPICAL IPD vs. VDD (WDT DISABLED, RC MODE)

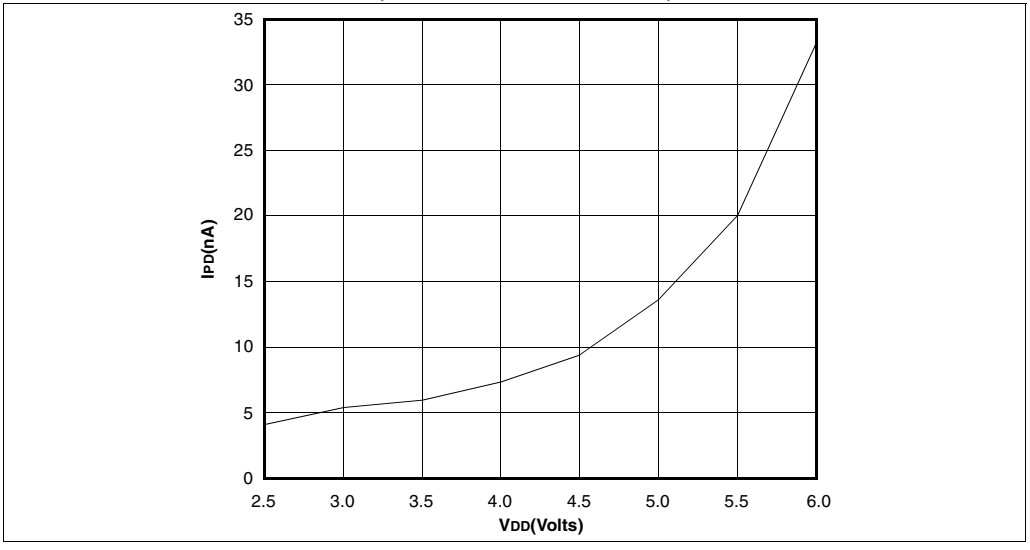


FIGURE 23-2: MAXIMUM IPD vs. VDD (WDT DISABLED, RC MODE)

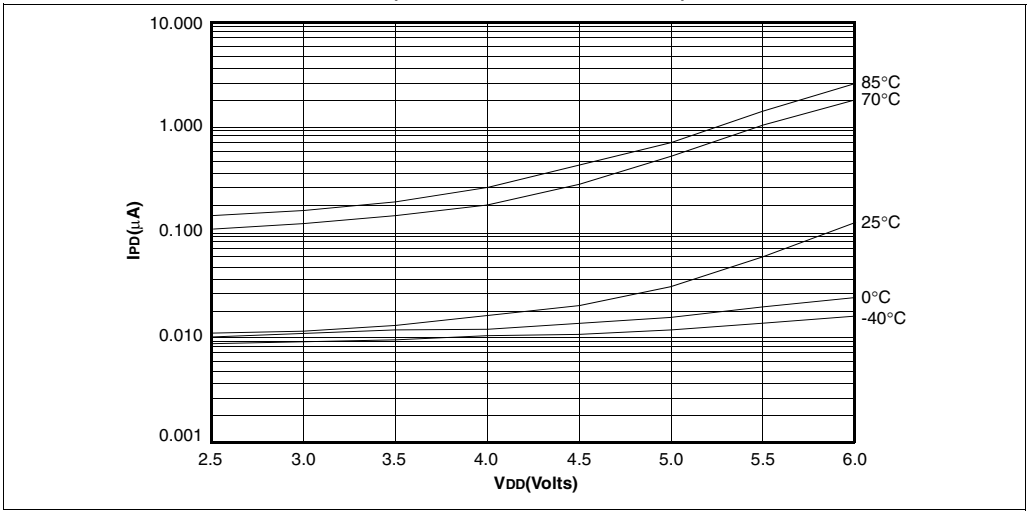


FIGURE 23-18: TYPICAL IDD vs. CAPACITANCE @ 500 kHz (RC MODE)

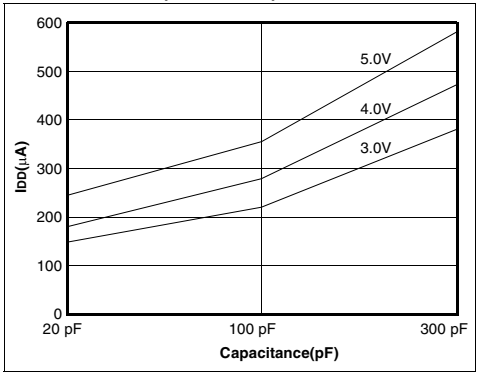


TABLE 23-1: RC OSCILLATOR FREQUENCIES

Cext	Rext	Average Fosc @ 5V, 25°C	
22 pF	5k	4.12 MHz	± 1.4%
	10k	2.35 MHz	± 1.4%
	100k	268 kHz	± 1.1%
100 pF	3.3k	1.80 MHz	± 1.0%
	5k	1.27 MHz	± 1.0%
	10k	688 kHz	± 1.2%
	100k	77.2 kHz	± 1.0%
300 pF	3.3k	707 kHz	± 1.4%
	5k	501 kHz	± 1.2%
	10k	269 kHz	± 1.6%
	100k	28.3 kHz	± 1.1%

The percentage variation indicated here is part to part variation due to normal process distribution. The variation indicated is ±3 standard deviation from average value for VDD = 5V.

FIGURE 23-19: TRANSCONDUCTANCE(gm) OF HS OSCILLATOR vs. VDD

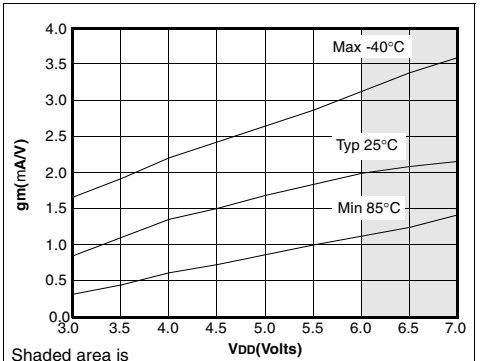


FIGURE 23-20: TRANSCONDUCTANCE(gm) OF LP OSCILLATOR vs. VDD

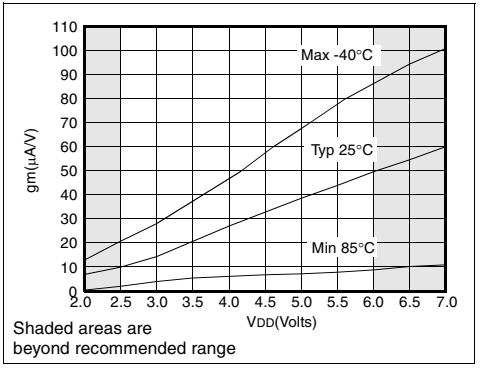
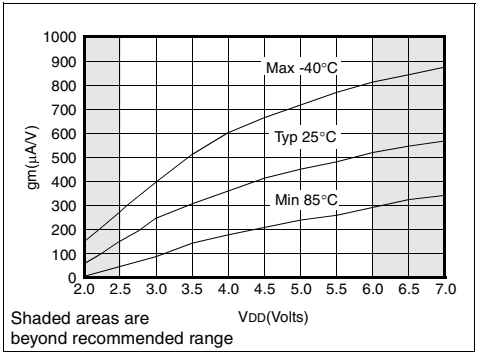


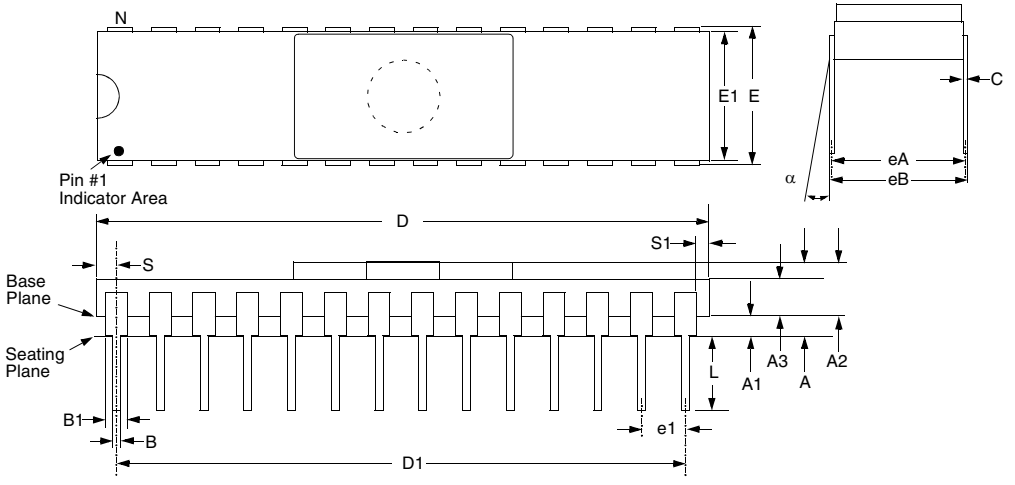
FIGURE 23-21: TRANSCONDUCTANCE(gm) OF XT OSCILLATOR vs. VDD



Data based on matrix samples. See first page of this section for details.

24.9 28-Lead Ceramic Side Brazed Dual In-Line with Window (300 mil) (JW)

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



Package Group: Ceramic Side Brazed Dual In-Line (CER)						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
α	0°	10°		0°	10°	
A	3.937	5.030		0.155	0.198	
A1	1.016	1.524		0.040	0.060	
A2	2.921	3.506		0.115	0.138	
A3	1.930	2.388		0.076	0.094	
B	0.406	0.508		0.016	0.020	
B1	1.219	1.321	**Typical**	0.048	0.052	
C	0.228	0.305	**Typical**	0.009	0.012	
D	35.204	35.916		1.386	1.414	
D1	32.893	33.147	**Reference**	1.295	1.305	
E	7.620	8.128		0.300	0.320	
E1	7.366	7.620		0.290	0.300	
e1	2.413	2.667	**Typical**	0.095	0.105	
eA	7.366	7.874	**Reference**	0.290	0.310	
eB	7.594	8.179		0.299	0.322	
L	3.302	4.064		0.130	0.160	
N	28	28		28	28	
S	1.143	1.397		0.045	0.055	
S1	0.533	0.737		0.021	0.029	