Welcome to E-XFL.COM

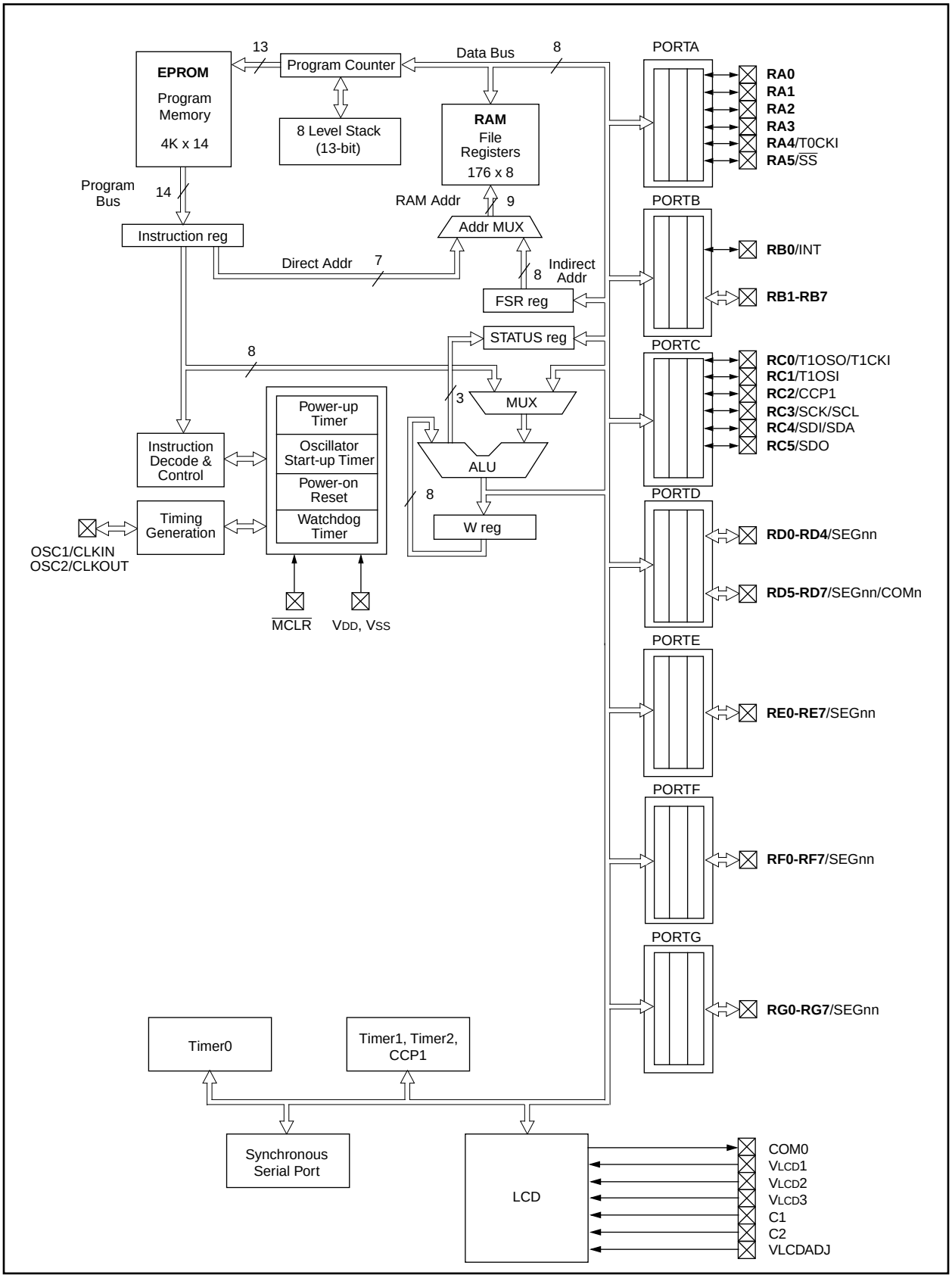
What is "Embedded - Microcontrollers"?

"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
Peripherals	LCD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	7KB (4K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	176 x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 6V
Data Converters	A/D 5x8b
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	68-PLCC (24.23x24.23)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lc924-04-l

FIGURE 3-1: PIC16C923 BLOCK DIAGRAM



Example 4-1 shows the calling of a subroutine in page 1 of the program memory. This example assumes that 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)
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

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

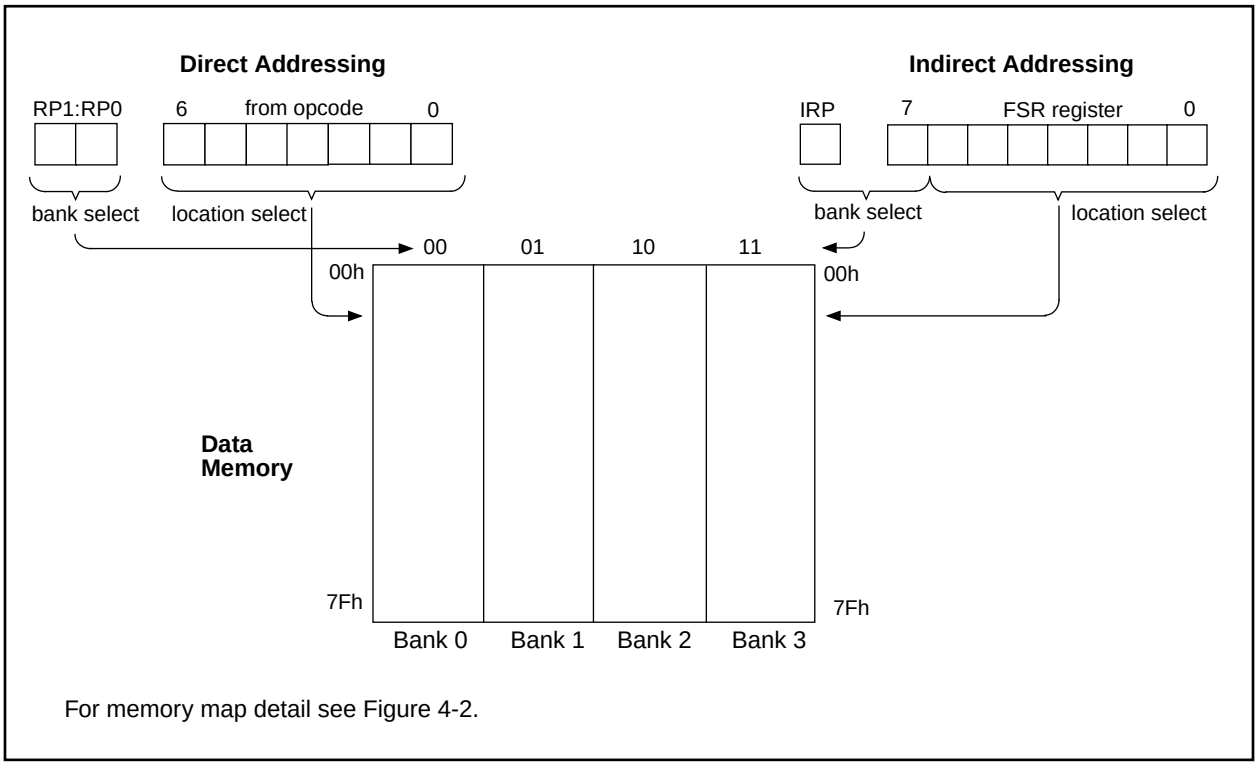
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses the register pointed to by the File Select Register (FSR). Reading the INDF register itself indirectly (FSR = '0') will produce 00h. Writing to the INDF register indirectly results in 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 4-10.

A simple program to clear RAM locations 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 4-10: DIRECT/INDIRECT ADDRESSING



For memory map detail see Figure 4-2.

7.2 Using Timer0 with an External Clock

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (TOSC). Also, there is a delay in the actual incrementing of Timer0 after synchronization.

7.2.1 EXTERNAL CLOCK SYNCHRONIZATION

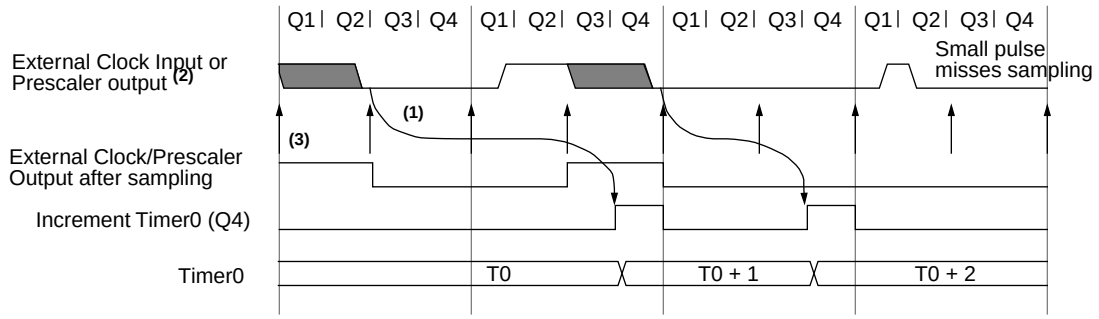
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 7-5). Therefore, it is necessary for T0CKI to be high for at least 2Tosc (and a small RC delay of 20 ns) and low for at least 2Tosc (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

When a prescaler is used, the external clock input is divided by the asynchronous ripple-counter type prescaler so that the prescaler output is symmetrical. For the external clock to meet the sampling requirement, the ripple-counter must be taken into account. Therefore, it is necessary for T0CKI to have a period of at least 4Tosc (and a small RC delay of 40 ns) divided by the prescaler value. The only requirement on T0CKI high and low time is that they do not violate the minimum pulse width requirement of 10 ns. Refer to parameters 40, 41 and 42 in the electrical specification of the desired device.

7.2.2 TMR0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the Timer0 module is actually incremented. Figure 7-5 shows the delay from the external clock edge to the timer incrementing.

FIGURE 7-5: TIMER0 TIMING WITH EXTERNAL CLOCK



Note 1: Delay from clock input change to Timer0 increment is 3Tosc to 7Tosc. (Duration of Q = Tosc). Therefore, the error in measuring the interval between two edges on Timer0 input = ±4Tosc max.
2: External clock if no prescaler selected, Prescaler output otherwise.
3: The arrows indicate the points in time where sampling occurs.

7.3 Prescaler

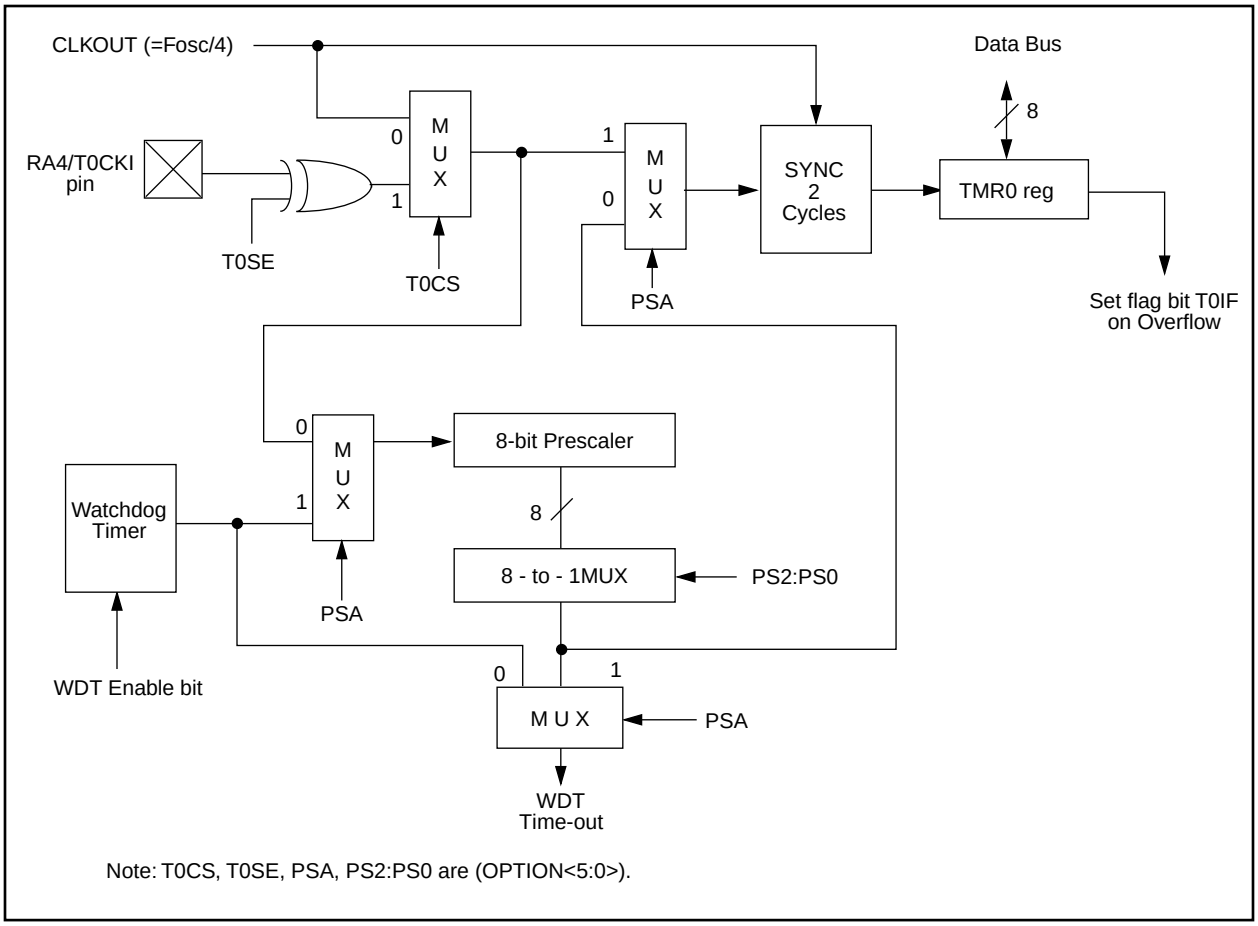
An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer (Figure 7-6). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet. Note that the prescaler may be used by either the Timer0 module or the WDT but not both. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer, and vice-versa.

The PSA and PS2:PS0 bits (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g. `CLRF 1, MOVWF 1, BSF 1,x`....etc.) will clear the prescaler count. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler count along with the Watchdog Timer. The prescaler is not readable or writable.

Note:	Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count, but will not change the prescaler assignment.

FIGURE 7-6: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER



Note: T0CS, T0SE, PSA, PS2:PS0 are (OPTION<5:0>).

7.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control, i.e., it can be changed “on the fly” during program execution.

Note: To avoid an unintended device RESET, the following instruction sequence (shown in Example 7-1) must be executed when changing the prescaler assignment from Timer0 to the WDT. This precaution must be followed even if the WDT is disabled.

EXAMPLE 7-1: CHANGING PRESCALER (TIMER0→WDT)

Lines 2 and 3 do NOT have to be included if the final desired prescale value is other than 1:1. If 1:1 is final desired value, then a temporary prescale value is set in lines 2 and 3 and the final prescale value will be set in lines 10 and 11.

```
1)  BSF    STATUS, RP0   ;Select Bank1
2)  MOVLW  b'xx0x0xxx'   ;Select clock source and prescale value of
3)  MOVWF  OPTION_REG    ;other than 1:1
4)  BCF    STATUS, RP0   ;Select Bank0
5)  CLRF   TMR0          ;Clear TMR0 and prescaler
6)  BSF    STATUS, RP1   ;Select Bank1
7)  MOVLW  b'xxxx1xxx'   ;Select WDT, do not change prescale value
8)  MOVWF  OPTION_REG    ;
9)  CLRWDT               ;Clears WDT and prescaler
10) MOVLW  b'xxxx1xxx'   ;Select new prescale value and WDT
11) MOVWF  OPTION_REG    ;
12) BCF    STATUS, RP0   ;Select Bank0
```

To change prescaler from the WDT to the Timer0 module use the precaution shown in Example 7-2.

EXAMPLE 7-2: CHANGING PRESCALER (WDT→TIMER0)

```
CLRWDT              ;Clear WDT and prescaler
BSF     STATUS, RP0 ;Select Bank1
MOVLW   b'xxxx0xxx' ;Select TMR0, new prescale value and
MOVWF   OPTION_REG  ;clock source
BCF     STATUS, RP0 ;Select Bank0
```

TABLE 7-1: REGISTERS ASSOCIATED WITH TIMER0

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets
01h, 101h	TMR0	Timer0 module’s register								xxxx xxxx	uuuu uuuu
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
81h, 181h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	—	—	PORTA Data Direction Control Register						--11 1111	--11 1111

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by Timer0.

8.5 Resetting Timer1 using the CCP Trigger Output

If the CCP1 module is configured in compare mode to generate a “special event trigger" (CCP1M3:CCP1M0 = 1011), this signal will reset Timer1.

Note:	The special event trigger from the CCP1 module will not set interrupt flag bit TMR1IF (PIR1<0>).

Timer1 must be configured for either timer or synchronized counter mode to take advantage of this feature. If Timer1 is running in asynchronous counter mode, this reset operation may not work.

In the event that a write to Timer1 coincides with a special event trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L registers pair effectively becomes the period register for Timer1.

8.6 Resetting of Timer1 Register Pair (TMR1H:TMR1L)

TMR1H and TMR1L registers are not reset on a POR or any other reset except by the CCP1 special event trigger.

T1CON register is reset to 00h on a Power-on Reset. In any other reset, the register is unaffected.

8.7 Timer1 Prescaler

The prescaler counter is cleared on writes to the TMR1H or TMR1L registers.

TABLE 8-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	LCDIF	ADIF(1)	—	—	SSPIF	CCP1IF	TMR2IF	TMR1IF	00-- 0000	00-- 0000
8Ch	PIE1	LCDIE	ADIE(1)	—	—	SSPIE	CCP1IE	TMR2IE	TMR1IE	00-- 0000	00-- 0000
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 register								xxxx xxxx	uuuu uuuu
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 register								xxxx xxxx	uuuu uuuu
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	--00 0000	--uu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented read as '0'. Shaded cells are not used by theTimer1 module.
Note 1: Bits ADIE and ADIF are reserved on the PIC16C923, always maintain these bits clear.

Figure 11-13 and Figure 11-14 show Master-transmitter and Master-receiver data transfer sequences.

When a master does not wish to relinquish the bus (by generating a STOP condition), a repeated START condition (Sr) must be generated. This condition is identical to the start condition (SDA goes high-to-low while SCL is high), but occurs after a data transfer acknowledge pulse (not the bus-free state). This allows a master to send "commands" to the slave and then receive the requested information or to address a different slave device. This sequence is shown in Figure 11-15.

FIGURE 11-13: MASTER-TRANSMITTER SEQUENCE

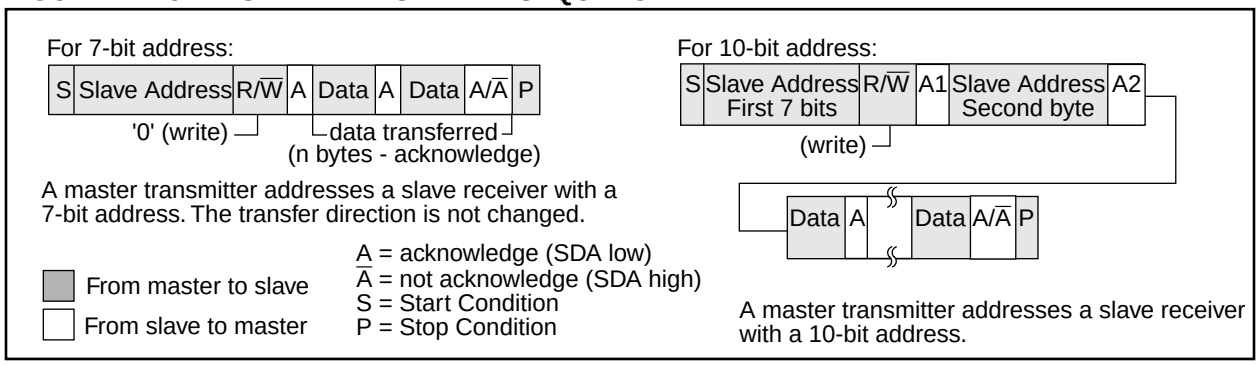


FIGURE 11-14: MASTER-RECEIVER SEQUENCE

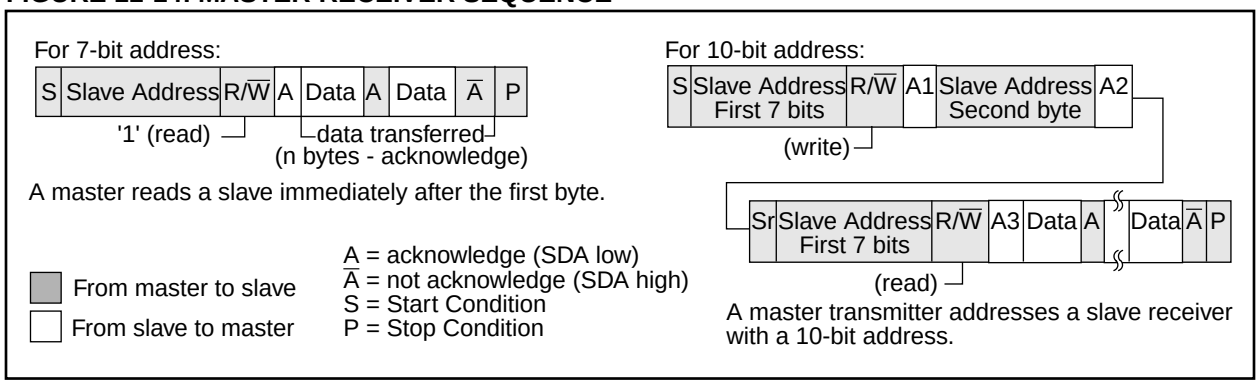


FIGURE 11-15: COMBINED FORMAT

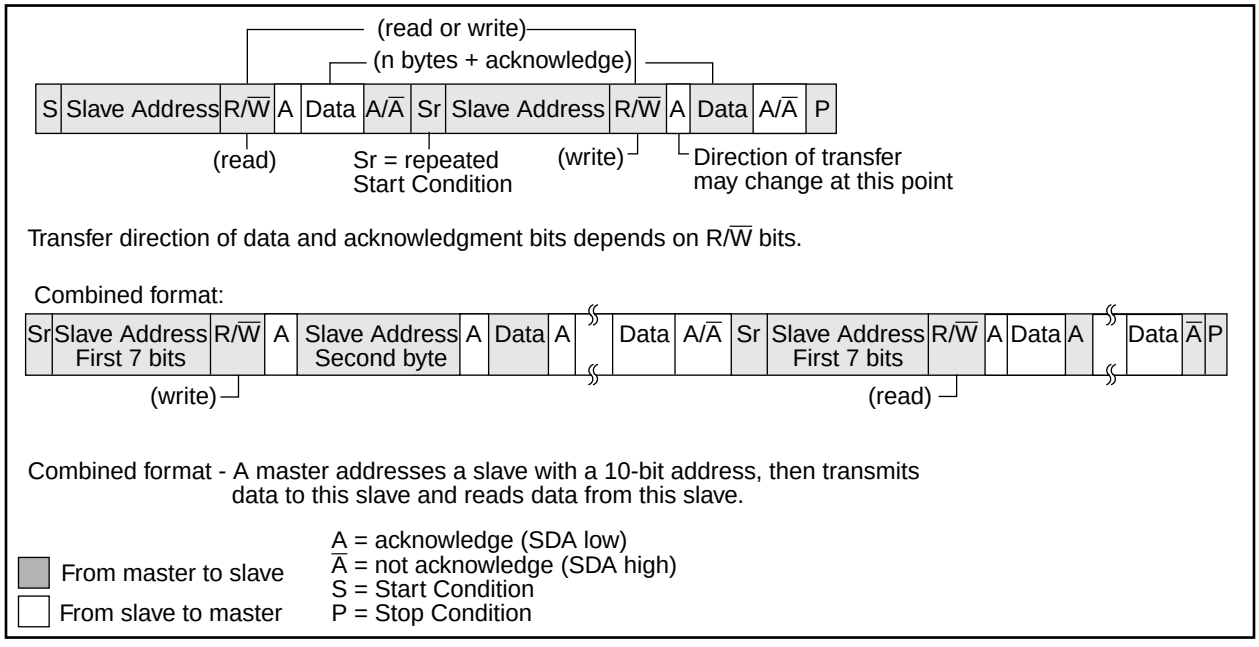


FIGURE 11-21: OPERATION OF THE I²C MODULE IN IDLE_MODE, RCV_MODE OR XMIT_MODE

```
IDLE_MODE (7-bit):
if (Addr_match)                 {       Set interrupt;
                                        if (R/W = 1)    {       Send ACK = 0;
                                                                set XMIT_MODE;
                                                        }
                                        else if (R/W = 0) set RCV_MODE;
                                }
RCV_MODE:
if ((SSPBUF=Full) OR (SSPOV = 1))
        {       Set SSPOV;
                Do not acknowledge;
        }
else    {       transfer SSPSR → SSPBUF;
                send ACK = 0;
        }
Receive 8-bits in SSPSR;
Set interrupt;
XMIT_MODE:
While ((SSPBUF = Empty) AND (CKP=0)) Hold SCL Low;
Send byte;
Set interrupt;
if ( ACK Received = 1)          {       End of transmission;
                                        Go back to IDLE_MODE;
                                }
else if ( ACK Received = 0)   Go back to XMIT_MODE;
IDLE_MODE (10-Bit):
If (High_byte_addr_match AND (R/W = 0))
        {       PRIOR_ADDR_MATCH = FALSE;
                Set interrupt;
                if ((SSPBUF = Full) OR ((SSPOV = 1))
                        {       Set SSPOV;
                                Do not acknowledge;
                        }
                else    {       Set UA = 1;
                                Send ACK = 0;
                                While (SSPADD not updated) Hold SCL low;
                                Clear UA = 0;
                                Receive Low_addr_byte;
                                Set interrupt;
                                Set UA = 1;
                                If (Low_byte_addr_match)
                                        {       PRIOR_ADDR_MATCH = TRUE;
                                                Send ACK = 0;
                                                while (SSPADD not updated) Hold SCL low;
                                                Clear UA = 0;
                                                Set RCV_MODE;
                                        }
                        }
        }
else if (High_byte_addr_match AND (R/W = 1)
        {       if (PRIOR_ADDR_MATCH)
                        {       send ACK = 0;
                                set XMIT_MODE;
                        }
        else PRIOR_ADDR_MATCH = FALSE;
        }
```

12.8 Use of the CCP Trigger

An A/D conversion can be started by the "special event trigger" of the CCP1 module. This requires that the CCP1M3:CCP1M0 bits (CCP1CON<3:0>) be programmed as 1011 and that the A/D module is enabled (ADON bit is set). When the trigger occurs, the GO/$\overline{\text{DONE}}$ bit will be set, starting the A/D conversion, and the Timer1 counter will be reset to zero. Timer1 is reset to automatically repeat the A/D acquisition period with minimal software overhead (moving the ADRES to the desired location). The appropriate analog input channel must be selected and the minimum acquisition done before the "special event trigger" sets the GO/$\overline{\text{DONE}}$ bit (starts a conversion).

If the A/D module is not enabled (ADON is cleared), then the "special event trigger" will be ignored by the A/D module, but will still reset the Timer1 counter.

12.9 Connection Considerations

If the input voltage exceeds the rail values (V_{SS} or V_{DD}) by greater than 0.2V, then the accuracy of the conversion is out of specification.

An external RC filter is sometimes added for anti-aliasing of the input signal. The R component should be selected to ensure that the total source impedance is kept under the 10 kΩ recommended specification. Any external components connected (via hi-impedance) to an analog input pin (capacitor, zener diode, etc.) should have very little leakage current at the pin.

12.10 Transfer Function

The ideal transfer function of the A/D converter is as follows: the first transition occurs when the analog input voltage (V_{AIN}) is Analog V_{REF} / 256 (Figure 12-5).

FIGURE 12-5: A/D TRANSFER FUNCTION

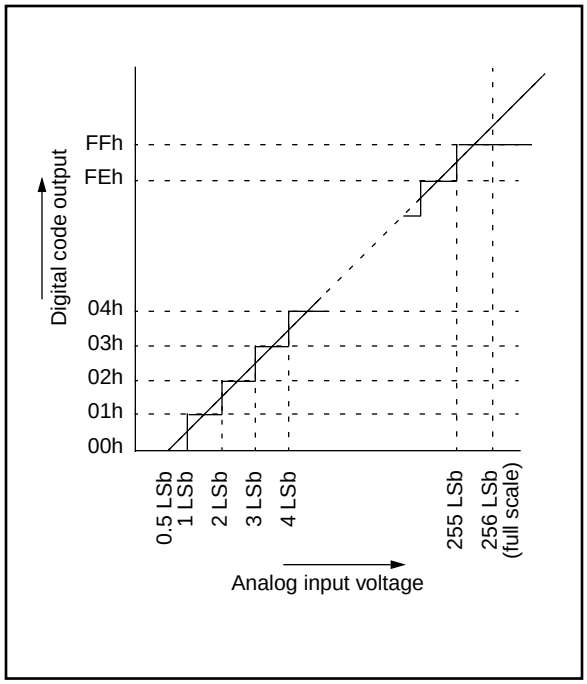


FIGURE 13-2: LCD MODULE BLOCK DIAGRAM

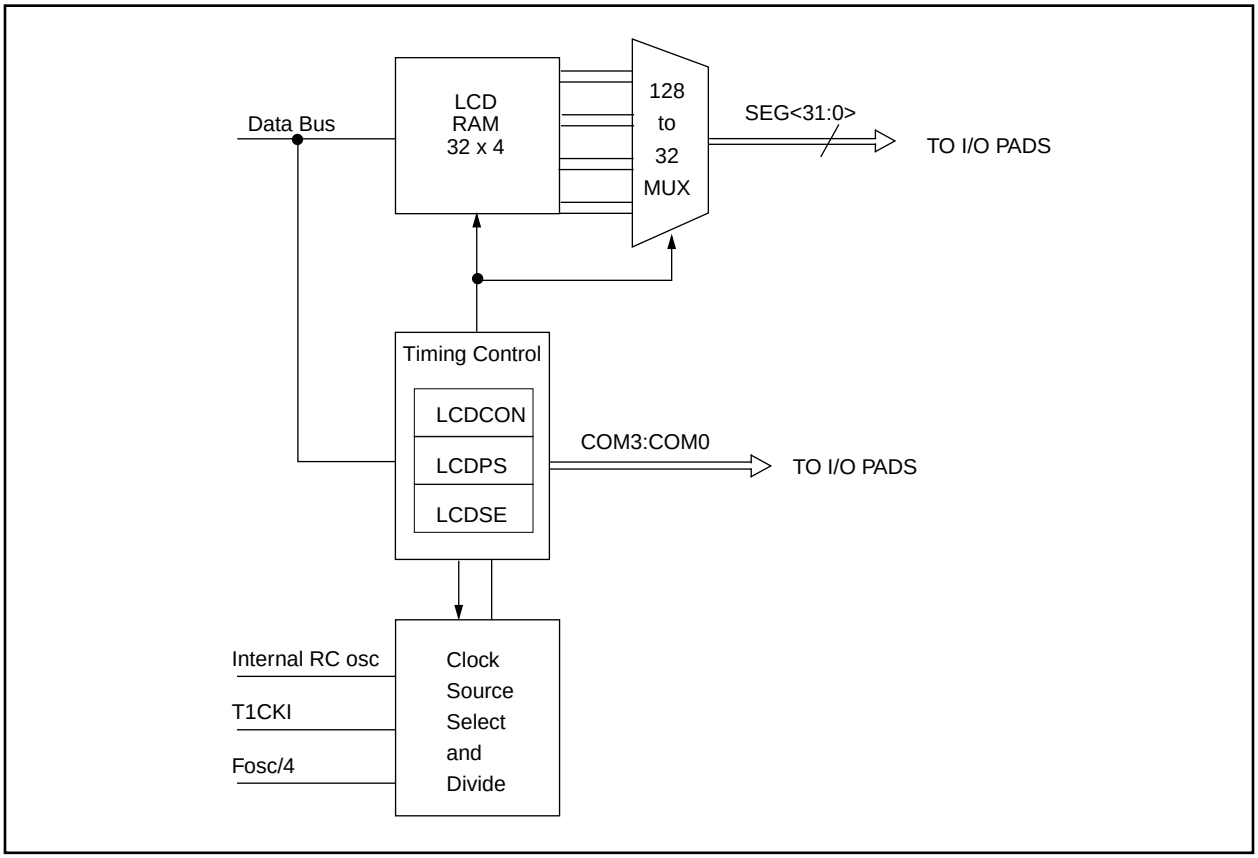


FIGURE 13-3: LCDPS REGISTER (ADDRESS 10Eh)

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	LP3	LP2	LP1	LP0
bit7							bit0

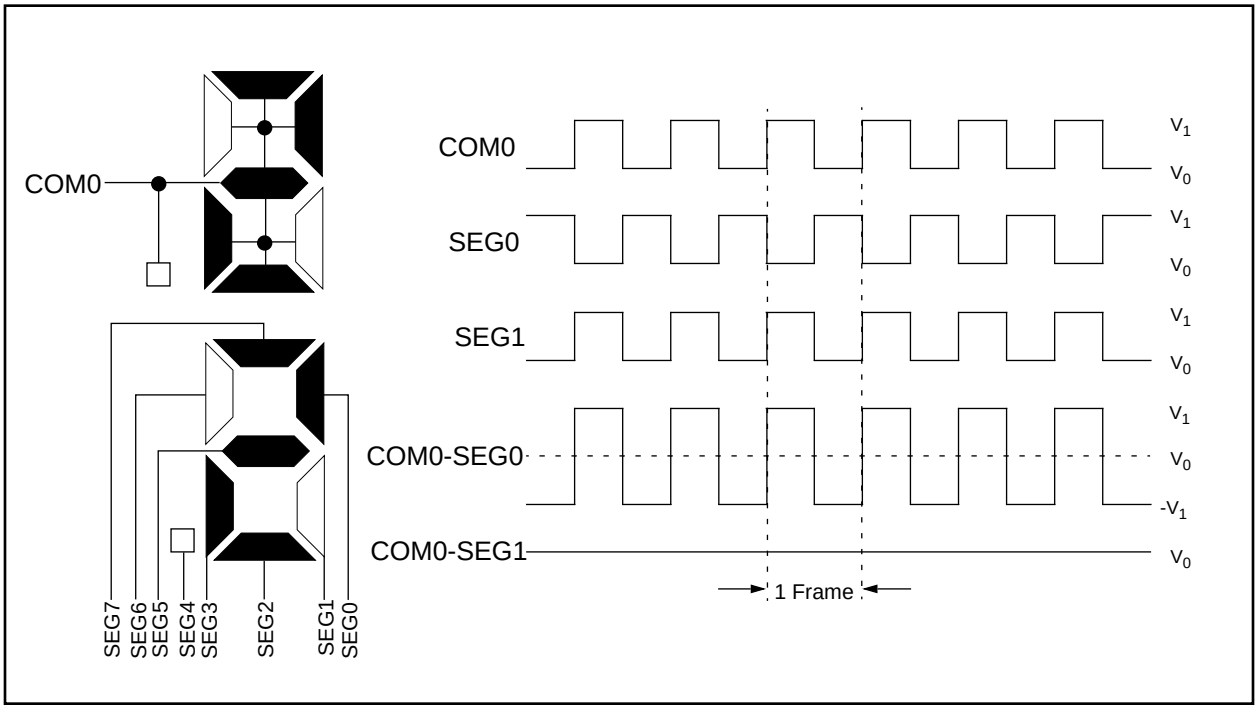
R =Readable bit
W =Writable bit
U =Unimplemented bit, Read as '0'
-n =Value at POR reset

bit 7-4: **Unimplemented,** read as '0'

bit 3-0: **LP3:LP0**: Frame Clock Prescale Selection bits

LMUX1:LMUX0	Multiplex	Frame Frequency =
00	Static	Clock source / (128 * (LP3:LP0 + 1))
01	1/2	Clock source / (128 * (LP3:LP0 + 1))
10	1/3	Clock source / (96 * (LP3:LP0 + 1))
11	1/4	Clock source / (128 * (LP3:LP0 + 1))

FIGURE 13-4: WAVEFORMS IN STATIC DRIVE



13.1.2 MULTIPLEX TIMING GENERATION

The timing generation circuitry will generate 1 to 4 common clocks based on the display mode selected. The mode is specified by bits LMUX1:LMUX0 (LCDCON<1:0>). Table 13-1 shows the formulas for calculating the frame frequency.

TABLE 13-1: FRAME FREQUENCY FORMULAS

Multiplex	Frame Frequency =
Static	Clock source / (128 * (LP3:LP0 + 1))
1/2	Clock source / (128 * (LP3:LP0 + 1))
1/3	Clock source / (96 * (LP3:LP0 + 1))
1/4	Clock source / (128 * (LP3:LP0 + 1))

TABLE 13-2: APPROX. FRAME FREQ IN Hz USING TIMER1 @ 32.768 kHz OR Fosc @ 8 MHz

LP3:LP0	Static	1/2	1/3	1/4
2	85	85	114	85
3	64	64	85	64
4	51	51	68	51
5	43	43	57	43
6	37	37	49	37
7	32	32	43	32

TABLE 13-3: APPROX. FRAME FREQ IN Hz USING INTERNAL RC OSC @ 14 kHz

LP3:LP0	Static	1/2	1/3	1/4
0	109	109	146	109
1	55	55	73	55
2	36	36	49	36
3	27	27	36	27

FIGURE 14-8: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 1

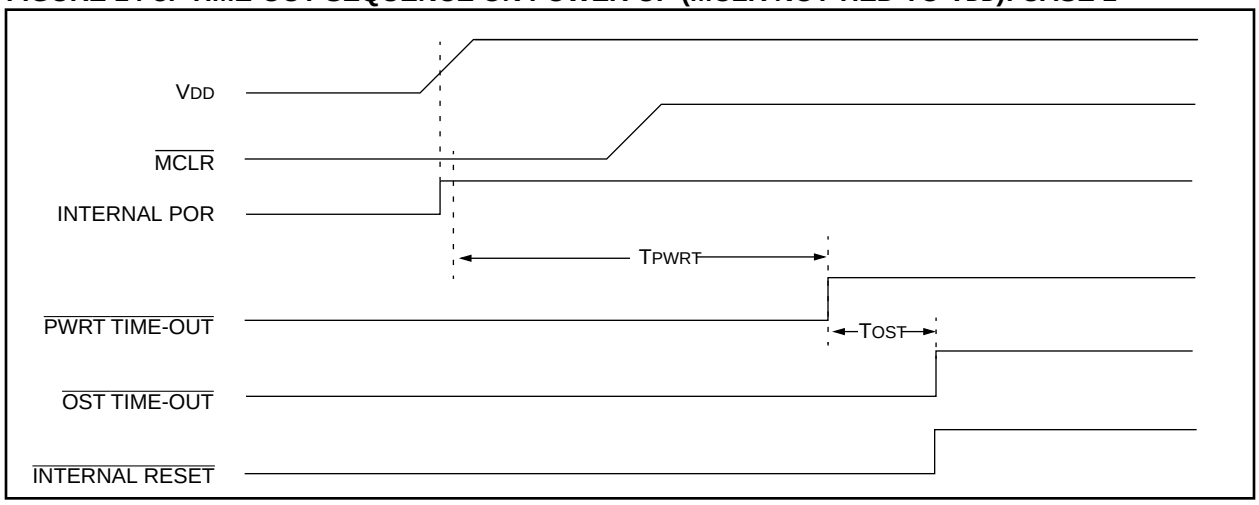


FIGURE 14-9: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 2

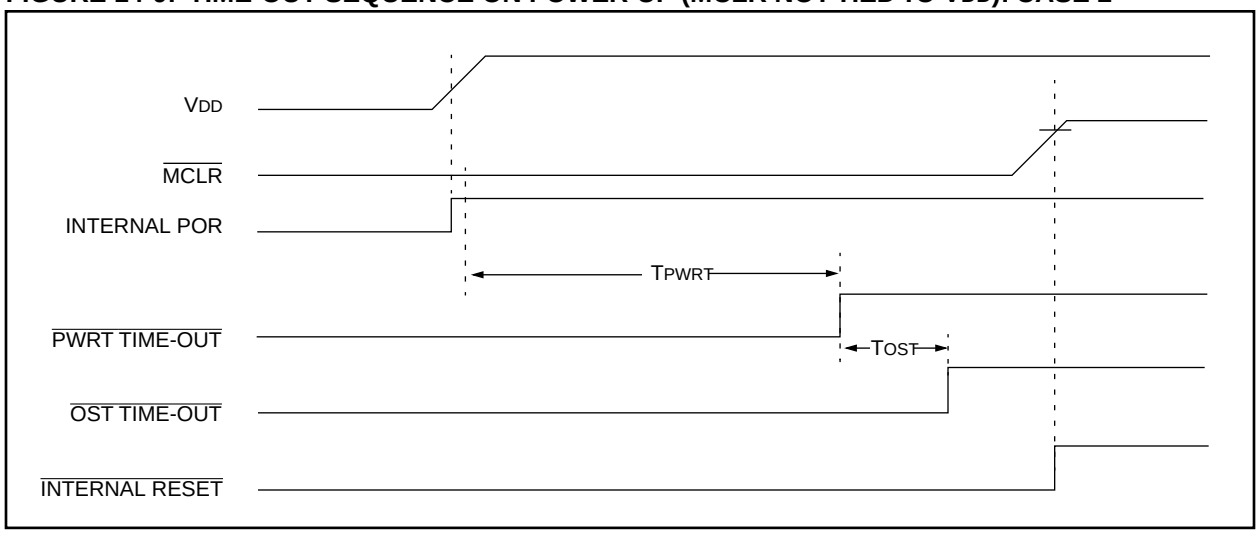
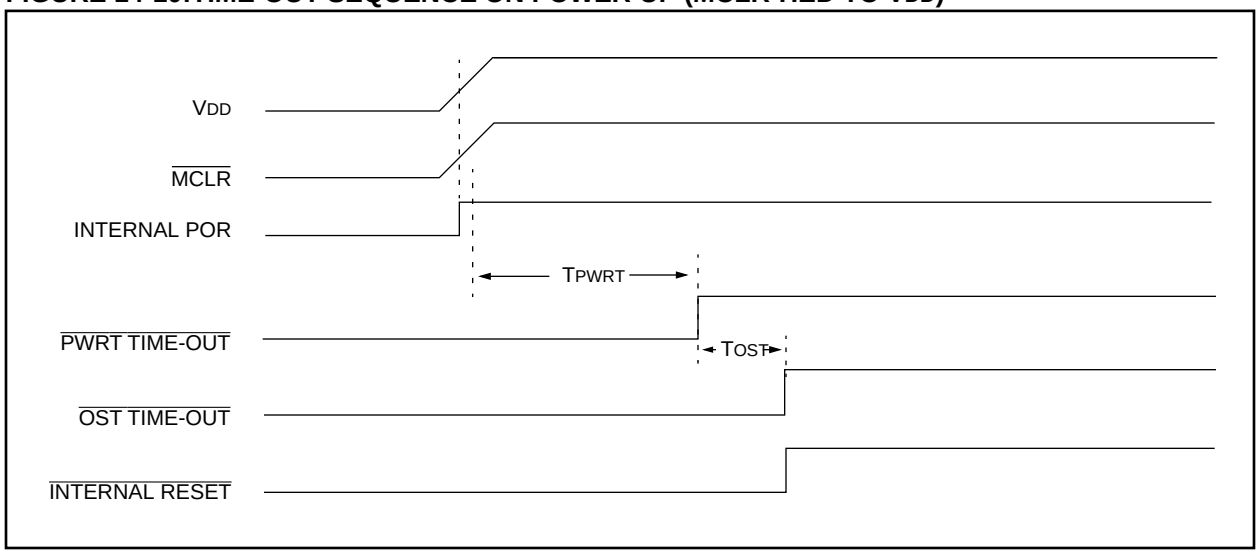


FIGURE 14-10:TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO V_{DD})



CLRF **Clear f**

Syntax: [*label*] CLRF f

Operands: $0 \le f \le 127$

Operation: $00h \rightarrow (f)$
$1 \rightarrow Z$

Status Affected: Z

Encoding:

00	0001	1fff	ffff

Description: The contents of register 'f' are cleared and the Z bit is set.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example

```
CLRF    FLAG_REG
```

Before Instruction
FLAG_REG = 0x5A
After Instruction
FLAG_REG = 0x00
Z = 1

CLRW **Clear W**

Syntax: [*label*] CLRW

Operands: None

Operation: $00h \rightarrow (W)$
$1 \rightarrow Z$

Status Affected: Z

Encoding:

00	0001	0xxx	xxxx

Description: W register is cleared. Zero bit (Z) is set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No-Operation	Process data	Write to W

Example

```
CLRW
```

Before Instruction
W = 0x5A
After Instruction
W = 0x00
Z = 1

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f,d

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

Operation: (f) - (W) → (destination)

Status Affected: C, DC, Z

Encoding:

00	0010	dfff	ffff

Description: Subtract (2's complement method) W register from register 'f'. If 'd' is 0 the result is stored 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 1: SUBWF REG1,1

Before Instruction
REG1 = 3
W = 2
C = ?
Z = ?

After Instruction
REG1 = 1
W = 2
C = 1; result is positive
Z = 0

Example 2: Before Instruction
REG1 = 2
W = 2
C = ?
Z = ?

After Instruction
REG1 = 0
W = 2
C = 1; result is zero
Z = 1

Example 3: Before Instruction
REG1 = 1
W = 2
C = ?
Z = ?

After Instruction
REG1 = 0xFF
W = 2
C = 0; result is negative
Z = 0

SWAPF **Swap Nibbles in f**

Syntax: [*label*] SWAPF f,d

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

Operation: (f<3:0>) → (destination<7:4>),
(f<7:4>) → (destination<3:0>)

Status Affected: None

Encoding:

00	1110	dfff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0 the result is placed in W register. If 'd' is 1 the result is placed in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example SWAPF REG, 0

Before Instruction
REG1 = 0xA5

After Instruction
REG1 = 0xA5
W = 0x5A

TRIS **Load TRIS Register**

Syntax: [*label*] TRIS f

Operands: $5 \le f \le 7$

Operation: (W) → TRIS register f;

Status Affected: None

Encoding:

00	0000	0110	0fff

Description: The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.

Words: 1

Cycles: 1

Example

To maintain upward compatibility with future PIC16CXX products, do not use this instruction.

FIGURE 17-2: LOAD CONDITIONS

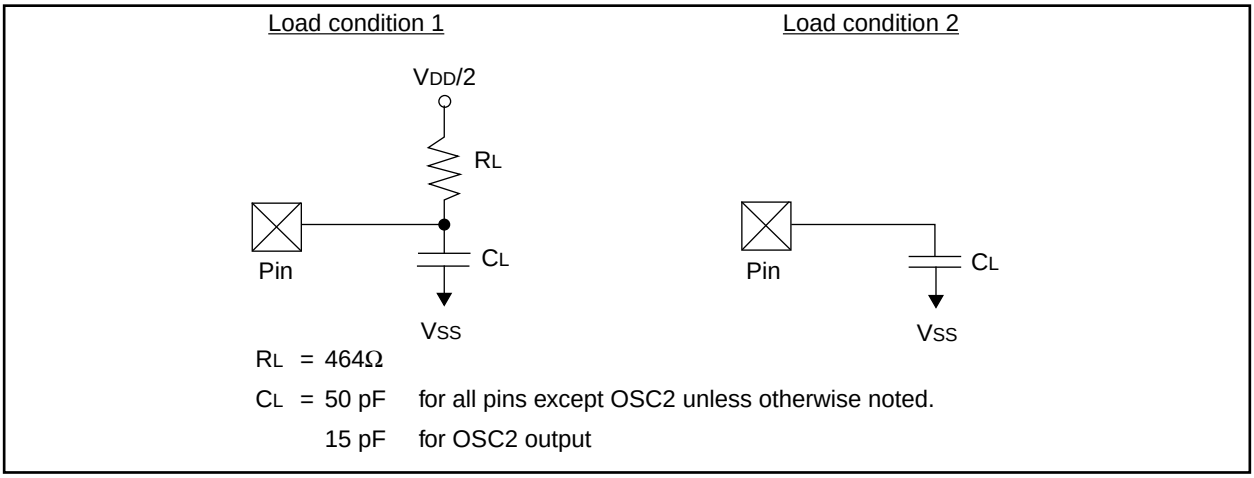
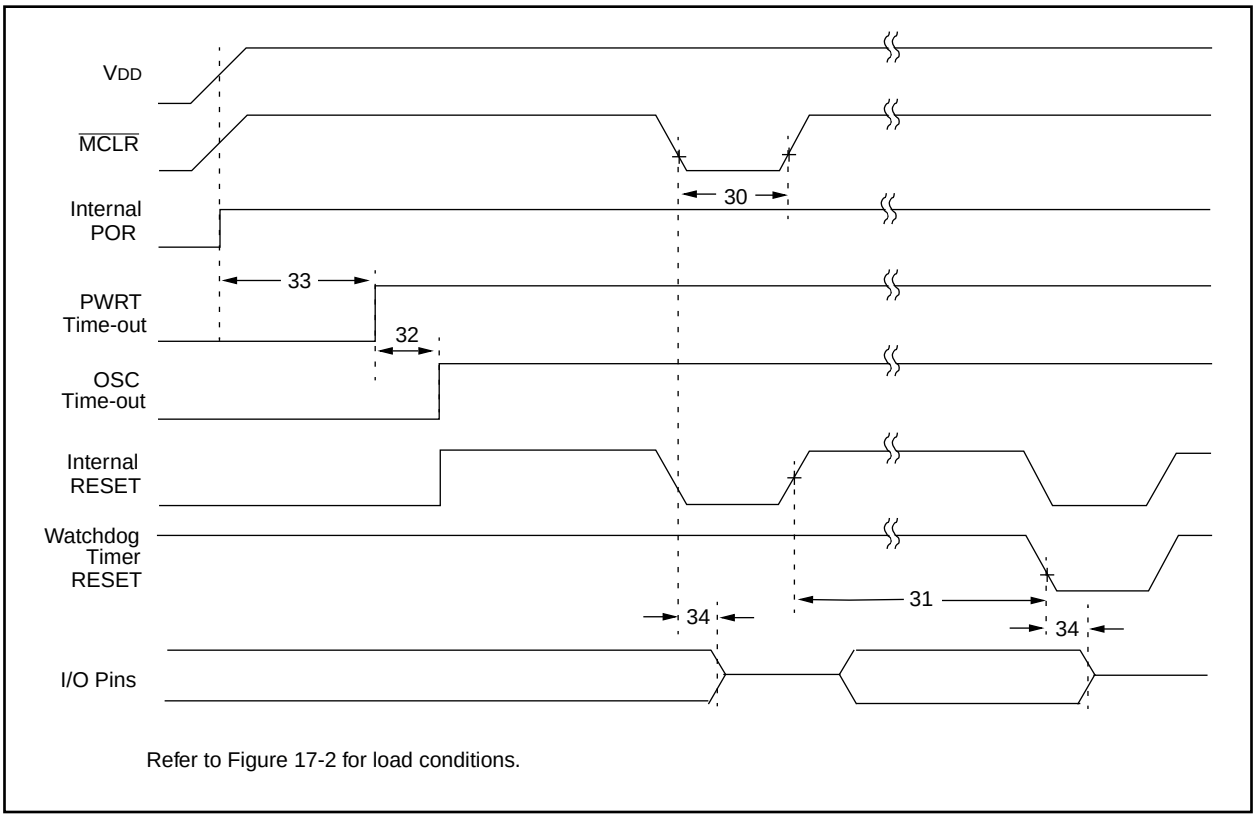


FIGURE 17-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING



Refer to Figure 17-2 for load conditions.

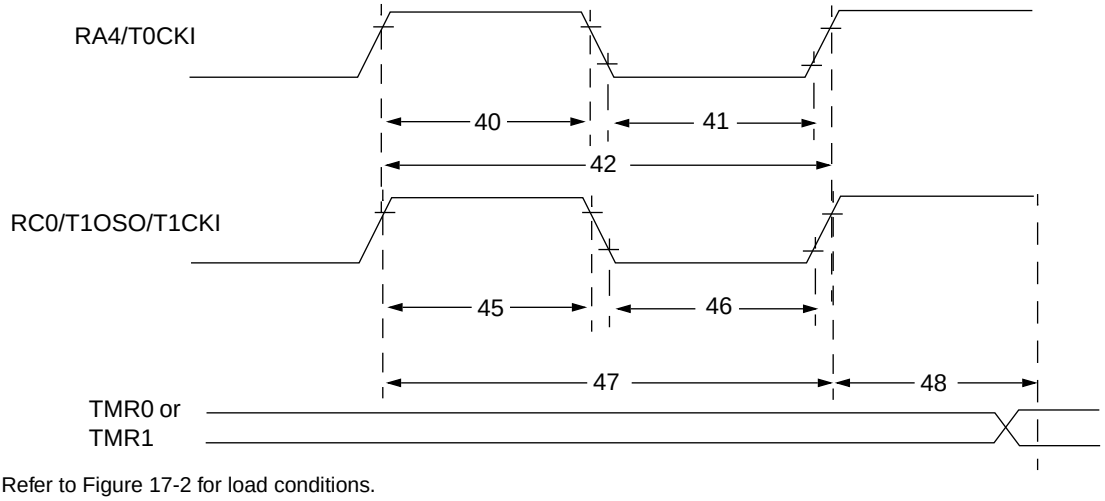
TABLE 17-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	TmcL	$\overline{MCLR}$ Pulse Width (low)	2	—	—	µs	
31*	Twdt	Watchdog Timer Time-out Period (No Prescaler)	7	18	33	ms	VDD = 5V, -40˚C to +85˚C
32	Tost	Oscillation Start-up Timer Period	—	1024TOSC	—	—	TOSC = OSC1 period
33*	Tpwrt	Power-up Timer Period	28	72	132	ms	VDD = 5V, -40˚C to +85˚C
34	TIOZ	I/O Hi-impedance from $\overline{MCLR}$ Low or Watchdog Timer Reset	—	—	2.1	µs	

* 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.

FIGURE 17-6: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS



Refer to Figure 17-2 for load conditions.

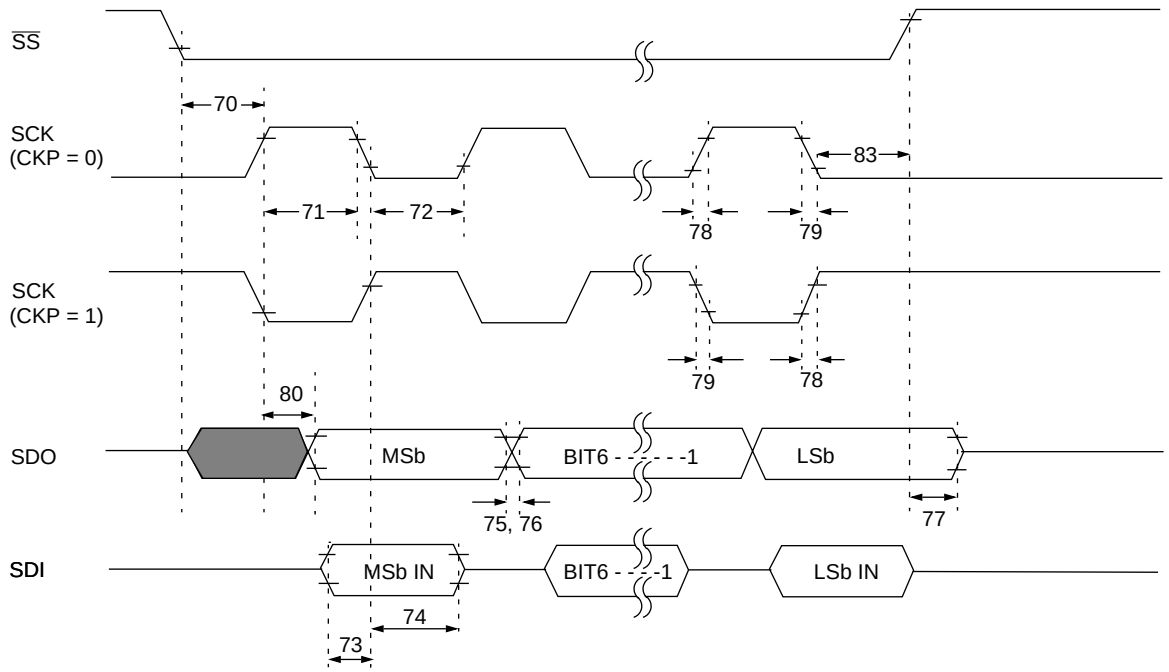
TABLE 17-7: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

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**923/924	15	—	—	ns	
				PIC16**LC**923/924	25	—	—	ns	
			Asynchronous	PIC16**C**923/924	30	—	—	ns	
				PIC16**LC**923/924	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**923/924	15	—	—	ns	
				PIC16**LC**923/924	25	—	—	ns	
			Asynchronous	PIC16**C**923/924	30	—	—	ns	
				PIC16**LC**923/924	50	—	—	ns	
47*	Tt1P	T1CKI input period	Synchronous	PIC16**C**923/924	Greater of: 30 OR TCY + 40 N	—	—	ns	N = prescale value (1, 2, 4, 8)
				PIC16**LC**923/924	Greater of: 50 OR TCY + 40 N				N = prescale value (1, 2, 4, 8)
			Asynchronous	PIC16**C**923/924	60	—	—	ns	
				PIC16**LC**923/924	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.

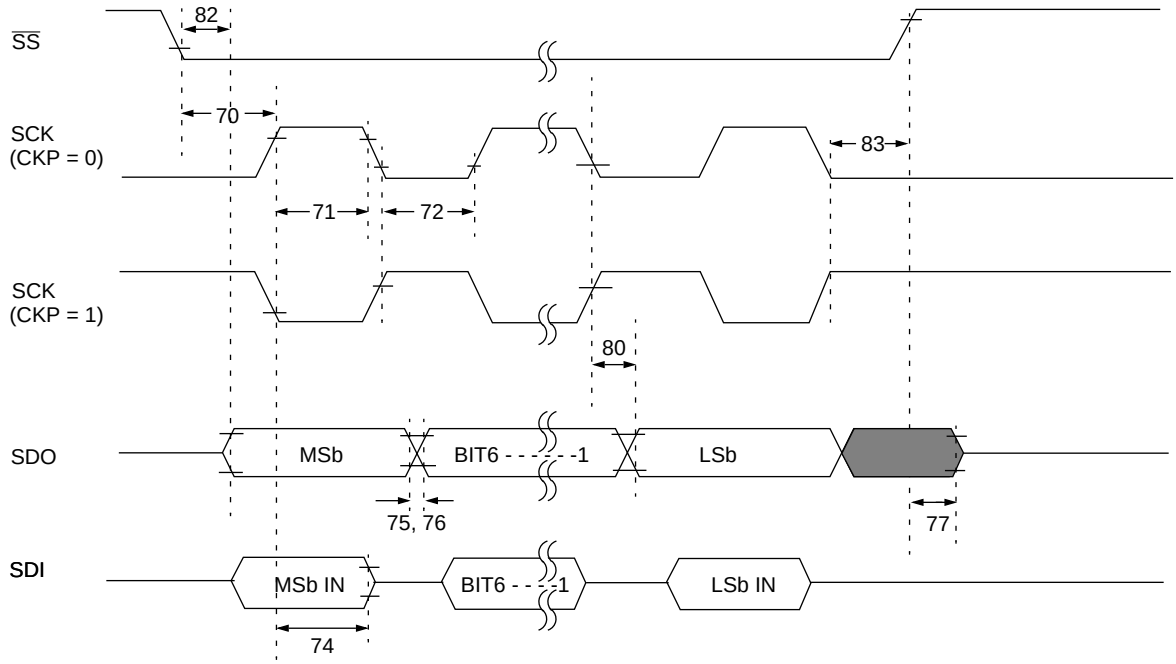
† Data in "Typ" column is at 5V, 25˚C unless otherwise stated. These parameters are for design guidance only and are not tested.

FIGURE 17-10:SPI SLAVE MODE TIMING (CKE = 0)



Refer to Figure 17-2 for load conditions.

FIGURE 17-11:SPI SLAVE MODE TIMING (CKE = 1)



Refer to Figure 17-2 for load conditions.

FIGURE 18-18:TYPICAL IDD vs. FREQUENCY (RC MODE @ 300 pF, 25°C)

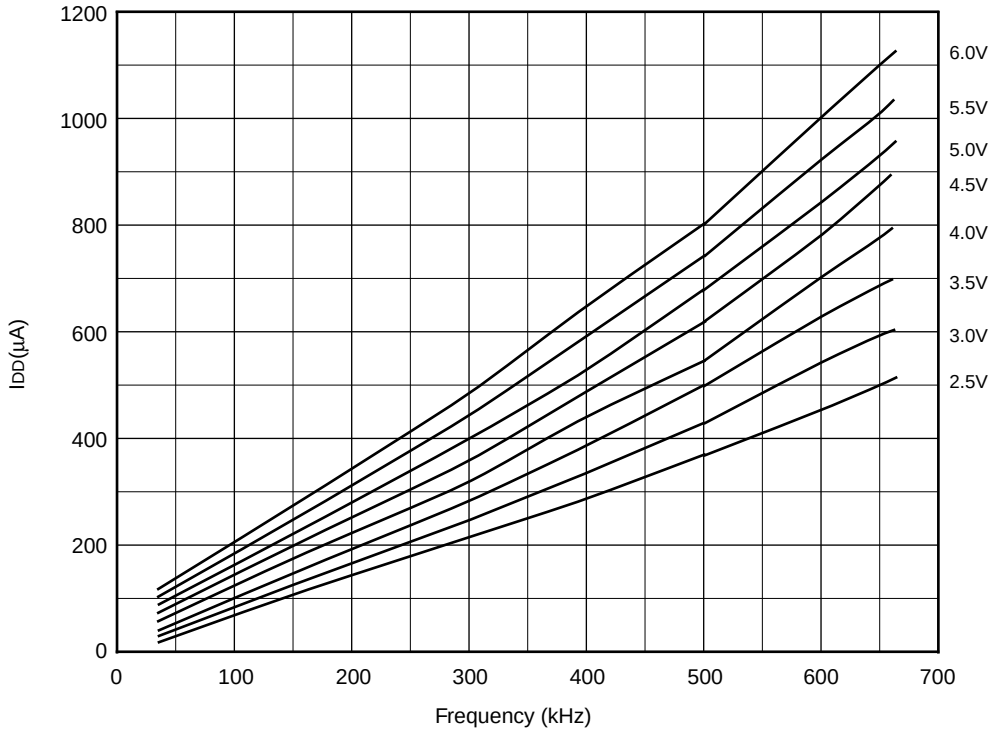
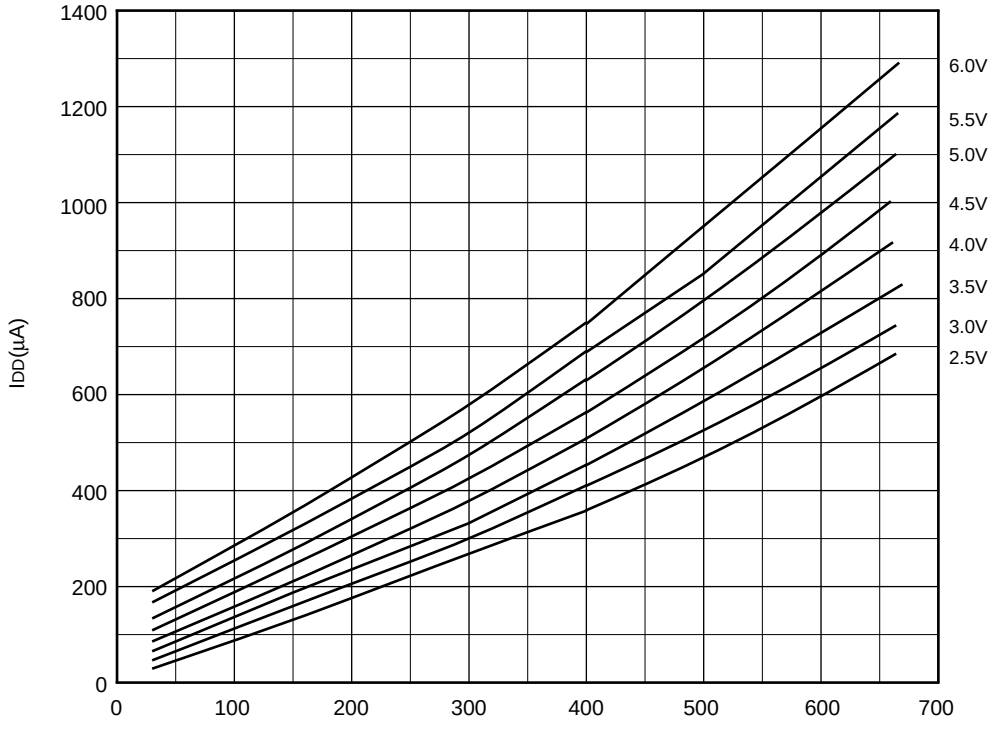


FIGURE 18-19:MAXIMUM IDD vs. FREQUENCY (RC MODE @ 300 pF, -40°C TO +85°C)



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