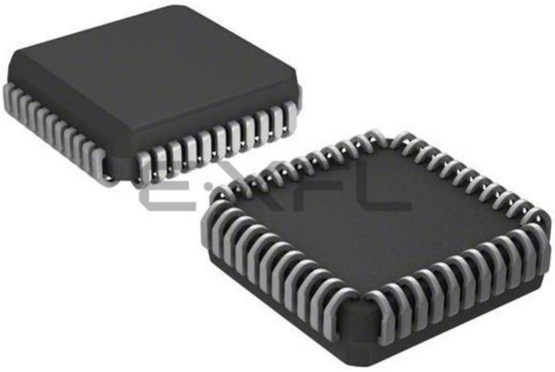


What is "Embedded - Microcontrollers"?

"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	40MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	33
Program Memory Size	32KB (16K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC (16.59x16.59)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18c452-i-l

NOTES:

NOTES:

7.6 INT0 Interrupt

External interrupts on the RB0/INT0, RB1/INT1 and RB2/INT2 pins are edge triggered: either rising, if the corresponding INTEDGx bit is set in the INTCON2 register, or falling, if the INTEDGx bit is clear. When a valid edge appears on the RBx/INTx pin, the corresponding flag bit INTxF is set. This interrupt can be disabled by clearing the corresponding enable bit INTxE. Flag bit INTxF must be cleared in software in the Interrupt Service Routine before re-enabling the interrupt. All external interrupts (INT0, INT1 and INT2) can wake-up the processor from SLEEP, if bit INTxE was set prior to going into SLEEP. If the global interrupt enable bit GIE set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the interrupt priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high priority interrupt source.

7.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow (FFh → 00h) in the TMR0 register will set flag bit TMR0IF. In 16-bit mode, an overflow (FFFFh → 0000h) in the TMR0H:TMR0L registers will set flag bit TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit T0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit TMR0IP (INTCON2<2>). See Section 8.0 for further details on the Timer0 module.

7.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB Interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

7.9 Context Saving During Interrupts

During an interrupt, the return PC value is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (see Section 4.3), the user may need to save the WREG, STATUS and BSR registers in software. Depending on the user's application, other registers may also need to be saved. Example 7-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 7-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF   W_TEMP                 ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP    ; STATUS_TEMP located anywhere
MOVFF   BSR, BSR_TEMP          ; BSR located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR          ; Restore BSR
MOVF    W_TEMP, W              ; Restore WREG
MOVFF   STATUS_TEMP, STATUS    ; Restore STATUS
```

NOTES:

8.2 PORTB, TRISB and LATB Registers

PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

Note:	On a Power-on Reset, these pins are configured as digital inputs.

The Data Latch register (LATB) is also memory mapped. Read-modify-write operations on the LATB register reads and writes the latched output value for PORTB.

EXAMPLE 8-2: INITIALIZING PORTB

```
CLRF   PORTB    ; Initialize PORTB by
                ; clearing output
                ; data latches
CLRF   LATB     ; Alternate method
                ; to clear output
                ; data latches
MOVLW 0xCF      ; Value used to
                ; initialize data
                ; direction
MOVWF TRISB     ; Set RB<3:0> as inputs
                ; RB<5:4> as outputs
                ; RB<7:6> as inputs
```

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit $\overline{\text{RBPU}}$ (INTCON2<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of the PORTB pins, RB7:RB4, have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB7:RB4 pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB7:RB4) are compared with the old value latched on the last read of PORTB. The “mismatch” outputs of RB7:RB4 are OR’ed together to generate the RB Port Change Interrupt with flag bit RBIF (INTCON<0>).

This interrupt can wake the device from SLEEP. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of PORTB (except with the `MOVFF` instruction). This will end the mismatch condition.

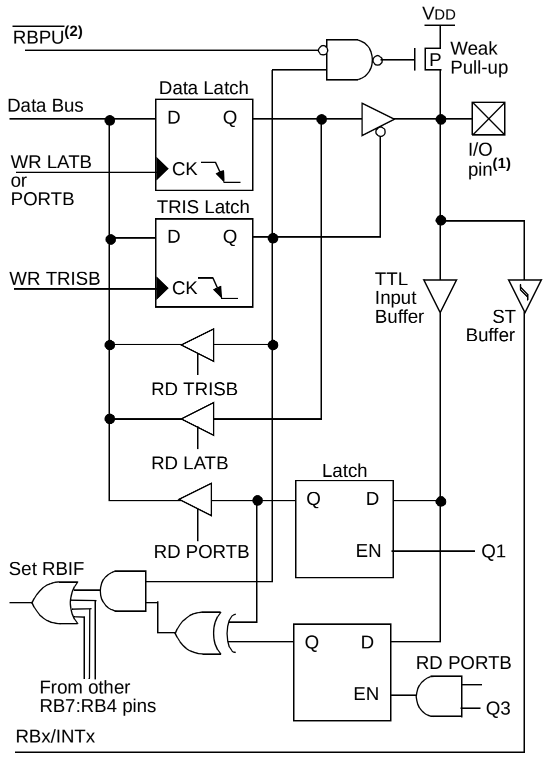
b) Clear flag bit RBIF.

A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.

The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

RB3 can be configured by the configuration bit CCP2MX as the alternate peripheral pin for the CCP2 module (CCP2MX = ‘0’).

FIGURE 8-4: BLOCK DIAGRAM OF RB7:RB4 PINS



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

2: To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the $\overline{\text{RBPU}}$ bit (INTCON2<7>).

TABLE 8-5: PORTC FUNCTIONS

Name	Bit#	Buffer Type	Function
RC0/T1OSO/T1CKI	bit0	ST	Input/output port pin or Timer1 oscillator output/Timer1 clock input.
RC1/T1OSI/CCP2	bit1	ST	Input/output port pin, Timer1 oscillator input, or Capture2 input/ Compare2 output/PWM output when CCP2MX configuration bit is disabled.
RC2/CCP1	bit2	ST	Input/output port pin or Capture1 input/Compare1 output/ PWM1 output.
RC3/SCK/SCL	bit3	ST	RC3 can also be the synchronous serial clock for both SPI and I^2C modes.
RC4/SDI/SDA	bit4	ST	RC4 can also be the SPI Data In (SPI mode) or Data I/O (I^2C mode).
RC5/SDO	bit5	ST	Input/output port pin or Synchronous Serial Port Data output.
RC6/TX/CK	bit6	ST	Input/output port pin, Addressable USART Asynchronous Transmit, or Addressable USART Synchronous Clock.
RC7/RX/DT	bit7	ST	Input/output port pin, Addressable USART Asynchronous Receive, or Addressable USART Synchronous Data.

Legend: ST = Schmitt Trigger input

TABLE 8-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

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
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	uuuu uuuu
LATC	LATC Data Output Register								xxxx xxxx	uuuu uuuu
TRISC	PORTC Data Direction Register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged

11.0 TIMER2 MODULE

The Timer2 module timer has the following features:

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

Timer2 has a control register shown in Register 11-1. Timer2 can be shut-off by clearing control bit TMR2ON (T2CON<2>) to minimize power consumption. Figure 11-1 is a simplified block diagram of the Timer2 module. Register 11-1 shows the Timer2 control register. The prescaler and postscaler selection of Timer2 are controlled by this register.

11.1 Timer2 Operation

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

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

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

TMR2 is not cleared when T2CON is written.

REGISTER 11-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

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

bit 6-3 **TOUTPS3:TOUTPS0**: Timer2 Output Postscale Select bits

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

bit 2 **TMR2ON**: Timer2 On bit

`1` = Timer2 is on
`0` = Timer2 is off

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

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

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

13.5.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

TABLE 13-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	**2.44 kHz**	**9.77 kHz**	**39.06 kHz**	**156.25 kHz**	**312.50 kHz**	**416.67 kHz**
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	14	12	10	8	7	6.58

TABLE 13-5: REGISTERS ASSOCIATED WITH 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
PIR1	PSPIF[1]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE[1]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
IPR1	PSPIP[1]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	0000 0000	0000 0000
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
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
CCPR1L	Capture/Compare/PWM Register1 (LSB)								xxxx xxxx	uuuu uuuu
CCPR1H	Capture/Compare/PWM Register1 (MSB)								xxxx xxxx	uuuu uuuu
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	--00 0000
CCPR2L	Capture/Compare/PWM Register2 (LSB)								xxxx xxxx	uuuu uuuu
CCPR2H	Capture/Compare/PWM Register2 (MSB)								xxxx xxxx	uuuu uuuu
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	--00 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.

Note 1: The PSPIF, PSPIE and PSPIP bits are reserved on the PIC18C2X2 devices. Always maintain these bits clear.

14.4.10 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge sequence enable bit, ACKEN (SSPCON2<4>). When this bit is set, the SCL pin is pulled low and the contents of the Acknowledge data bit is presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The baud rate generator then counts for one rollover period (TBRG) and the SCL pin is de-asserted (pulled high). When the SCL pin is sampled high (clock arbitration), the baud rate generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the baud rate generator is turned off and the MSSP module then goes into IDLE mode (Figure 14-20).

14.4.10.1 WCOL Status Flag

If the user writes the SSPBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn’t occur).

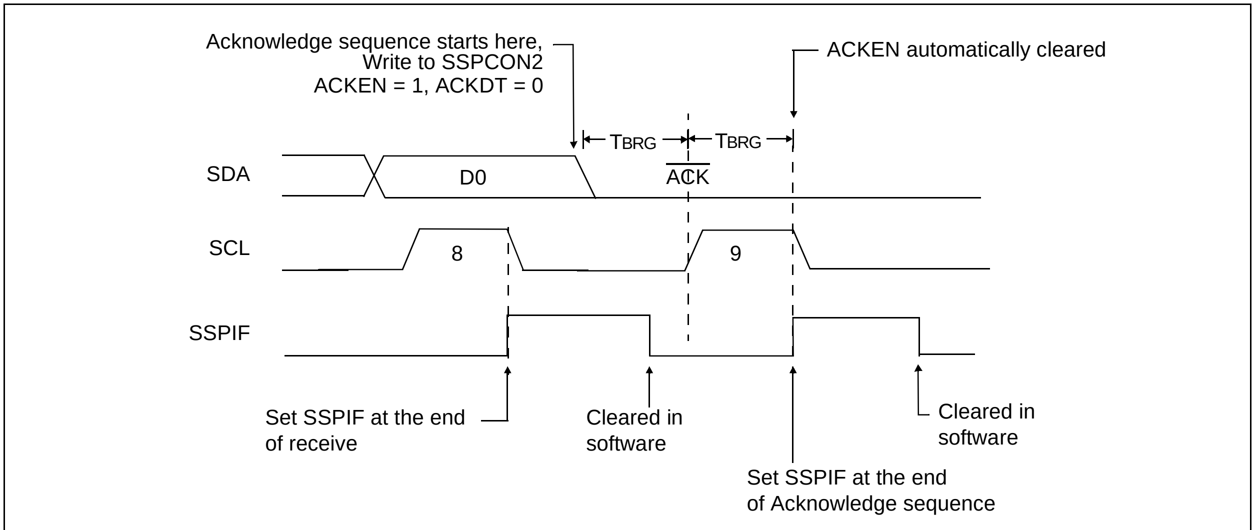
14.4.11 STOP CONDITION TIMING

A STOP bit is asserted on the SDA pin at the end of a receive/transmit by setting the STOP sequence enable bit, PEN (SSPCON2<2>). At the end of a receive/transmit, the SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the baud rate generator is reloaded and counts down to 0. When the baud rate generator times out, the SCL pin will be brought high, and one TBRG (baud rate generator rollover count) later, the SDA pin will be de-asserted. When the SDA pin is sampled high while SCL is high, the P bit (SSPSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPIF bit is set (Figure 14-21).

14.4.11.1 WCOL Status Flag

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

FIGURE 14-20: ACKNOWLEDGE SEQUENCE WAVEFORM


Acknowledge sequence starts here,
Write to SSPCON2
ACKEN = 1, ACKDT = 0
ACKEN automatically cleared
TBRG
TBRG
SDA
D0
$\overline{\text{ACK}}$
SCL
8
9
SSPIF
Set SSPIF at the end of receive
Cleared in software
Set SSPIF at the end of Acknowledge sequence
Cleared in software

Note: TBRG = one baud rate generator period.

BCF — Bit Clear f

Syntax: [*label*] BCF f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: 0 → f<b>

Status Affected: None

Encoding:

1001	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is cleared. 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 register 'f'

Example:

```
BCF     FLAG_REG,  7, 0
```

Before Instruction

```
FLAG_REG = 0xC7
```

After Instruction

```
FLAG_REG = 0x47
```

BN — Branch if Negative

Syntax: [*label*] BN n

Operands: $-128 \le n \le 127$

Operation: if negative bit is ’1’
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0110	nnnn	nnnn

Description: If the Negative bit is ’1’, then the program will branch.
The 2’s complement number ’2n’ is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC+2+2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:

```
HERE      BN   Jump
```

Before Instruction

```
PC          =   address (HERE)
```

After Instruction

```
If Negative =   1;
    PC      =   address (Jump)
If Negative =   0;
    PC      =   address (HERE+2)
```

IORLW Inclusive OR literal with WREG

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (WREG) .OR. k → WREG

Status Affected: N,Z

Encoding:

0000	1001	kkkk	kkkk

Description: The contents of WREG are OR'ed with the eight-bit literal 'k'. The result is placed in WREG.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to WREG

Example: `IORLW 0x35`

Before Instruction

```
WREG    =   0x9A
```

After Instruction

```
WREG    =   0xBF
```

IORWF Inclusive OR WREG with f

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

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

Operation: (WREG) .OR. (f) → dest

Status Affected: N,Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR WREG with register 'f'. If 'd' is 0, the result is placed in WREG. If 'd' is 1, the result is placed back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `IORWF  RESULT, 0, 1`

Before Instruction

```
RESULT  =   0x13
WREG    =   0x91
```

After Instruction

```
RESULT  =   0x13
WREG    =   0x93
```

RRNCF Rotate Right f (no carry)

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

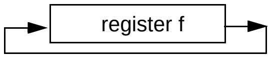
Operation: (f<n>) → dest<n-1>,
(f<0>) → dest<7>

Status Affected: N,Z

Encoding:

0100	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is 0, the result is placed in WREG. If 'd' is 1, the result is placed back in register 'f' (default). If ’a’ is 0, the Access Bank will be selected, overriding the BSR value. If ’a’ 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: `RRNCF REG, 1, 0`

Before Instruction
```
REG     =   1101 0111
```
After Instruction
```
REG     =   1110 1011
```

Example 2: `RRNCF REG, 0, 0`

Before Instruction
```
WREG    =   ?
REG     =   1101 0111
```
After Instruction
```
WREG    =   1110 1011
REG     =   1101 0111
```

SETF Set f

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

Operands: 0 ≤ f ≤ 255
a ∈ [0,1]

Operation: FFh → f

Status Affected: None

Encoding:

0110	100a	ffff	ffff

Description: The contents of the specified register are set to FFh. 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 register 'f'

Example: `SETF REG,1`

Before Instruction
```
REG         =   0x5A
```
After Instruction
```
REG         =   0xFF
```

SUBLW Subtract WREG from literal

Syntax: [*label*] SUBLW k

Operands: 0 ≤ k ≤ 255

Operation: k – (WREG) → WREG

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

Encoding:

0000	1000	kkkk	kkkk

Description: WREG is subtracted from the eight-bit literal 'k'. The result is placed in WREG.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to WREG

Example 1: `SUBLW 0x02`

Before Instruction

```
WREG    =   1
C       =   ?
```

After Instruction

```
WREG    =   1
C       =   1      ; result is positive
Z       =   0
N       =   0
```

Example 2: `SUBLW 0x02`

Before Instruction

```
WREG    =   2
C       =   ?
```

After Instruction

```
WREG    =   0
C       =   1      ; result is zero
Z       =   1
N       =   0
```

Example 3: `SUBLW 0x02`

Before Instruction

```
WREG    =   3
C       =   ?
```

After Instruction

```
WREG    =   FF   ; (2's complement)
C       =   0    ; result is negative
Z       =   0
N       =   1
```

SUBWF Subtract WREG from f

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

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) – (WREG) → dest

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

Encoding:

0101	11da	ffff	ffff

Description: Subtract WREG from register 'f' (2's complement method). If 'd' is 0, the result is stored in WREG. If 'd' is 1, the result is stored back in register 'f' (default). If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' is 1, then the bank will be selected as per the BSR value (default).

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example 1: `SUBWF REG, 1, 0`

Before Instruction

```
REG     =   3
WREG    =   2
C       =   ?
```

After Instruction

```
REG     =   1
WREG    =   2
C       =   1      ; result is positive
Z       =   0
N       =   0
```

Example 2: `SUBWF REG, 0, 0`

Before Instruction

```
REG     =   2
WREG    =   2
C       =   ?
```

After Instruction

```
REG     =   2
WREG    =   0
C       =   1      ; result is zero
Z       =   1
N       =   0
```

Example 3: `SUBWF REG, 1, 0`

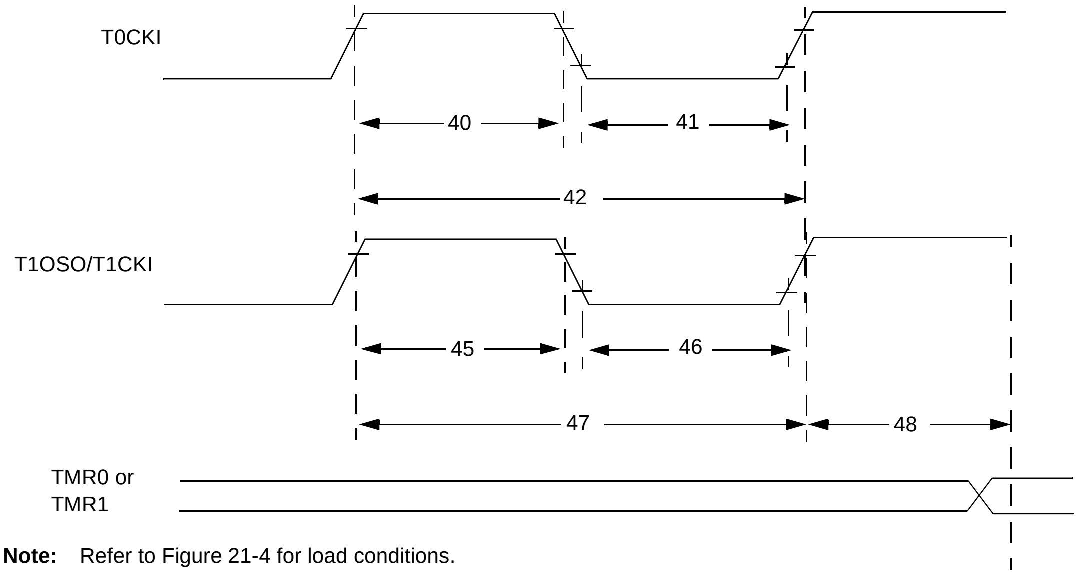
Before Instruction

```
REG     =   1
WREG    =   2
C       =   ?
```

After Instruction

```
REG     =   FFh    ;(2's complement)
WREG    =   2
C       =   0      ; result is negative
Z       =   0
N       =   1
```

FIGURE 21-9: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

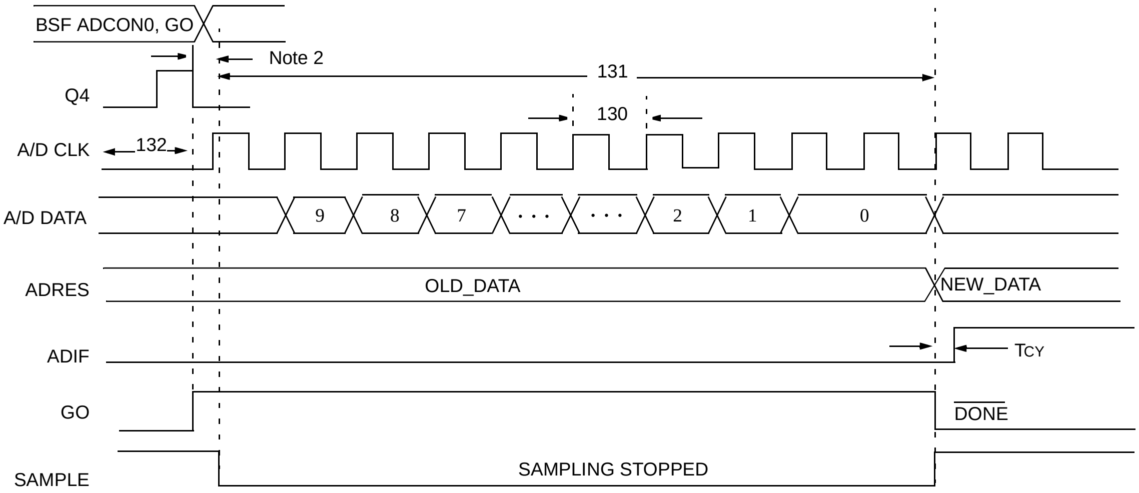


Note: Refer to Figure 21-4 for load conditions.

TABLE 21-8: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Symbol	Characteristic			Min	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width		No Prescaler	0.5TCY + 20	—	ns	
				With Prescaler	10	—	ns	
41	Tt0L	T0CKI Low Pulse Width		No Prescaler	0.5TCY + 20	—	ns	
				With Prescaler	10	—	ns	
42	Tt0P	T0CKI Period		No Prescaler	TCY + 10	—	ns	
				With Prescaler	Greater of: 20 ns or $\frac{T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 2, 4,..., 256)
45	Tt1H	T1CKI High Time	Synchronous, no prescaler		0.5TCY + 20	—	ns	
			Synchronous, with prescaler	PIC18**C**XXX	10	—	ns	
				PIC18**LC**XXX	25	—	ns	
			Asynchronous	PIC18**C**XXX	30	—	ns	
				PIC18**LC**XXX	40	—	ns	
46	Tt1L	T1CKI Low Time	Synchronous, no prescaler		0.5TCY + 20	—	ns	
			Synchronous, with prescaler	PIC18**C**XXX	15	—	ns	
				PIC18**LC**XXX	30	—	ns	
			Asynchronous	PIC18**C**XXX	30	—	ns	
				PIC18**LC**XXX	40	—	ns	
47	Tt1P	T1CKI input period	Synchronous		Greater of: 20 ns or $\frac{T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous		60	—	ns	
	Ft1	T1CKI oscillator input frequency range			DC	50	kHz	
48	Tcke2tmrI	Delay from external T1CKI clock edge to timer increment			2TOSC	7TOSC	—	

FIGURE 21-22: A/D CONVERSION TIMING



Note 1: If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the `SLEEP` instruction to be executed.

2: This is a minimal RC delay (typically 100 ns), which also disconnects the holding capacitor from the analog input.

TABLE 21-22: A/D CONVERSION REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
130	TAD	A/D clock period	PIC18**C**XXX	1.6	20(5)	μs	TOSC based, VREF ≥ 3.0V
			PIC18**LC**XXX	3.0	20(5)	μs	TOSC based, VREF full range
			PIC18**C**XXX	2.0	6.0	μs	A/D RC mode
			PIC18**LC**XXX	3.0	9.0	μs	A/D RC mode
131	TCNV	Conversion time (not including acquisition time) **(Note 1)**		11	12	TAD	
132	TACQ	Acquisition time **(Note 3)**		15 10	— —	μs μs	-40°C ≤ Temp ≤ 125°C 0°C ≤ Temp ≤ 125°C
135	TSWC	Switching Time from convert → sample		—	**(Note 4)**		
136	TAMP	Amplifier settling time **(Note 2)**		1	—	μs	This may be used if the “new” input voltage has not changed by more than 1 LSb (i.e., 5 mV @ 5.12V) from the last sampled voltage (as stated on CHOLD).

Note 1: ADRES register may be read on the following TCY cycle.

2: See Section 16.0 for minimum conditions, when input voltage has changed more than 1 LSb.

3: The time for the holding capacitor to acquire the “New” input voltage, when the voltage changes full scale after the conversion (AVDD to AVSS, or AVSS to AVDD). The source impedance (*RS*) on the input channels is 50 Ω.

4: On the next Q4 cycle of the device clock.

5: The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.

22.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES

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

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. 'Max' or 'min' represents (mean + 3σ) or (mean - 3σ) respectively, where σ is standard deviation, over the whole temperature range.

FIGURE 22-1: TYPICAL IDD vs. FOSC OVER VDD (HS MODE)

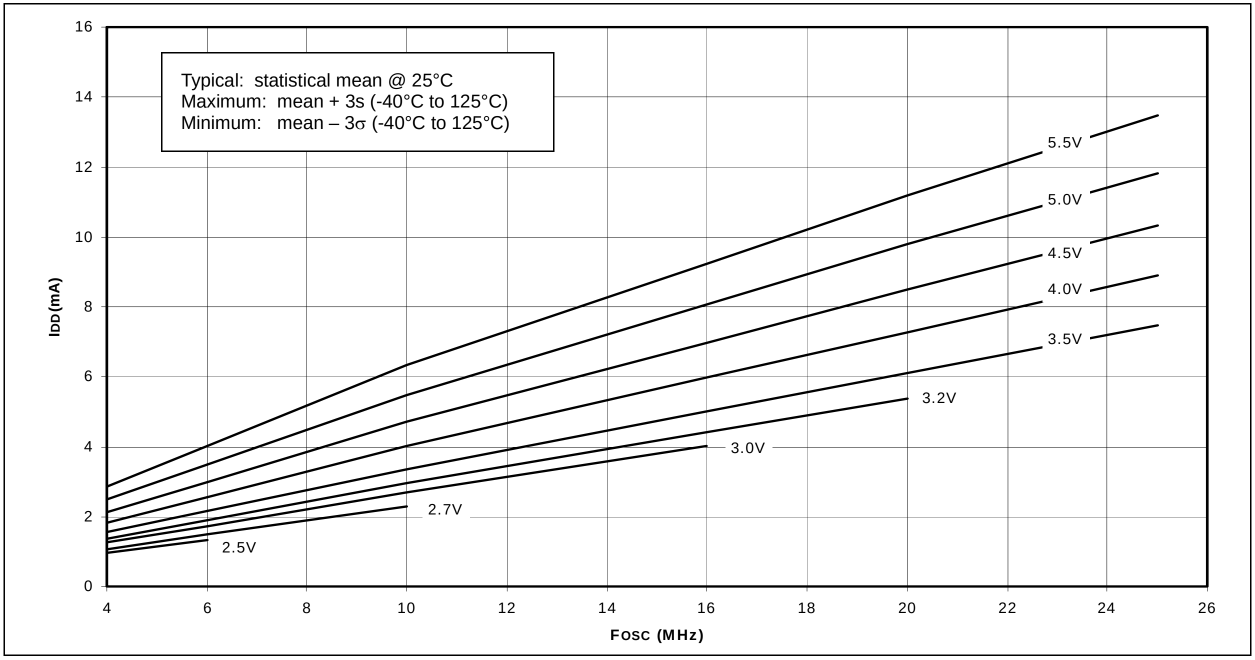


FIGURE 22-2: MAXIMUM IDD vs. FOSC OVER VDD (HS MODE)

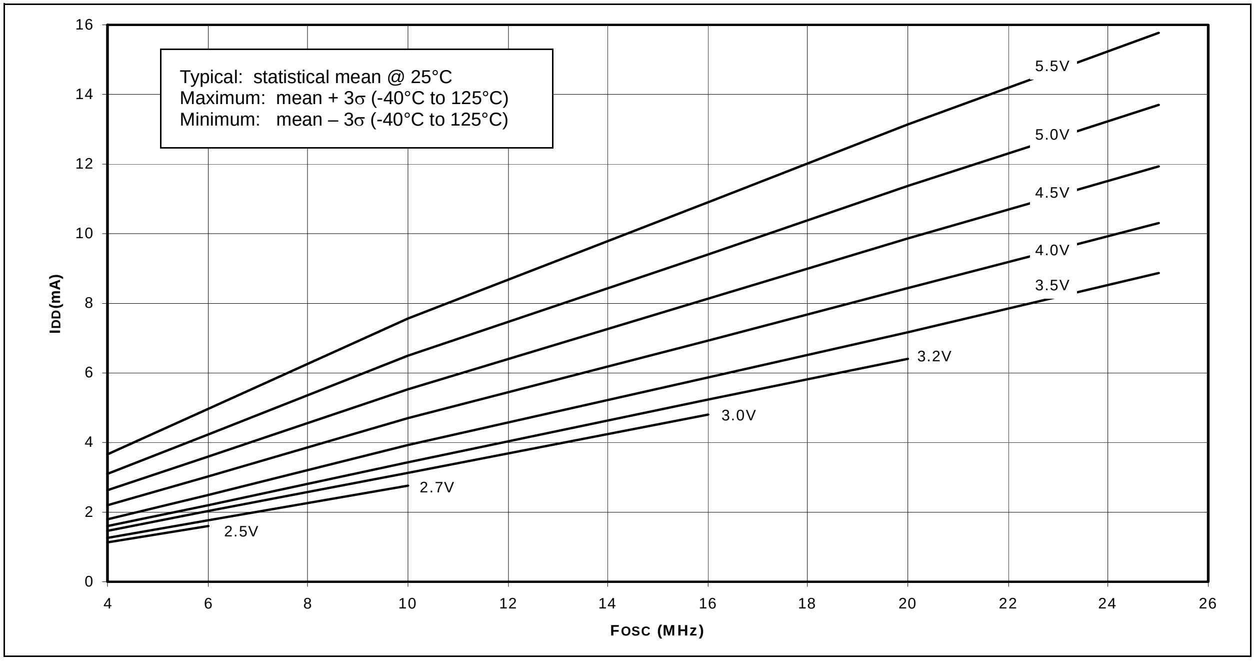


FIGURE 22-19: ΔILVD vs. VDD OVER TEMPERATURE (LVD ENABLED, VLVD = 4.5V - 4.78V)

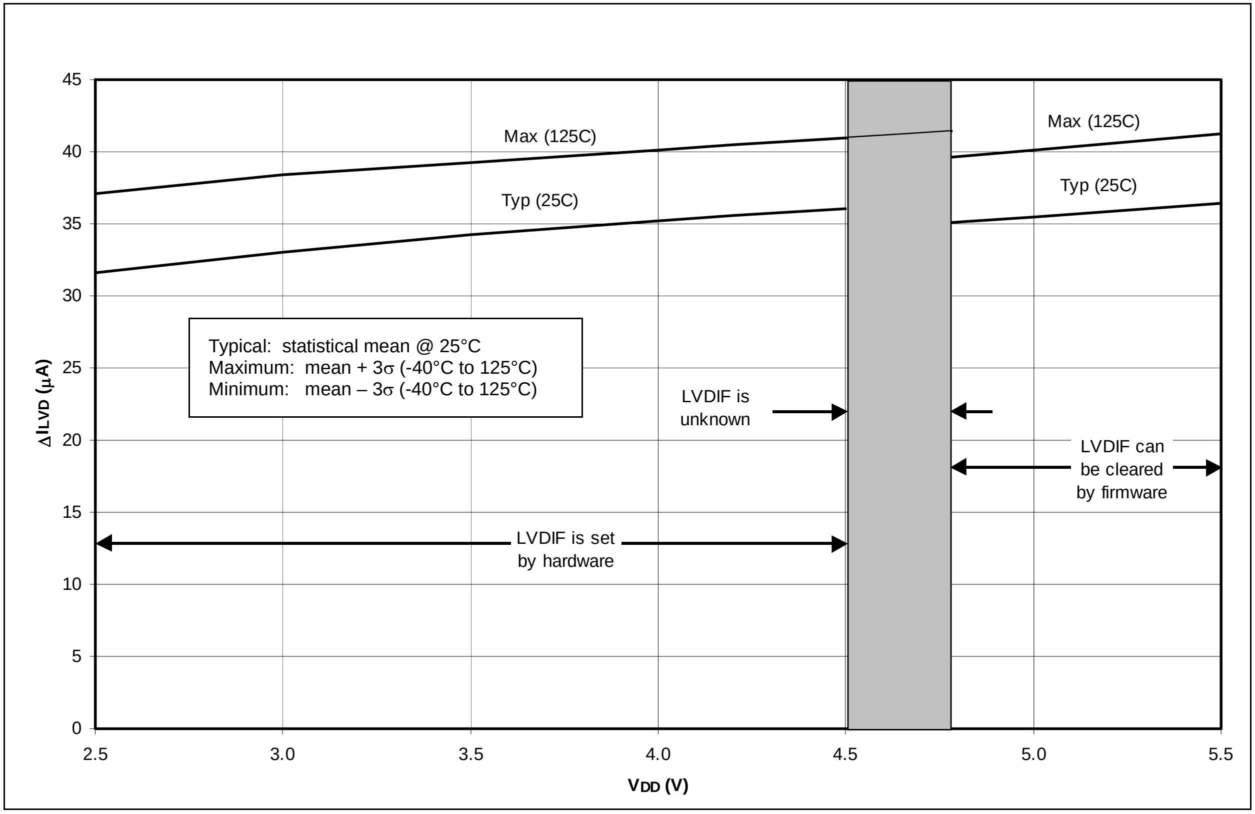
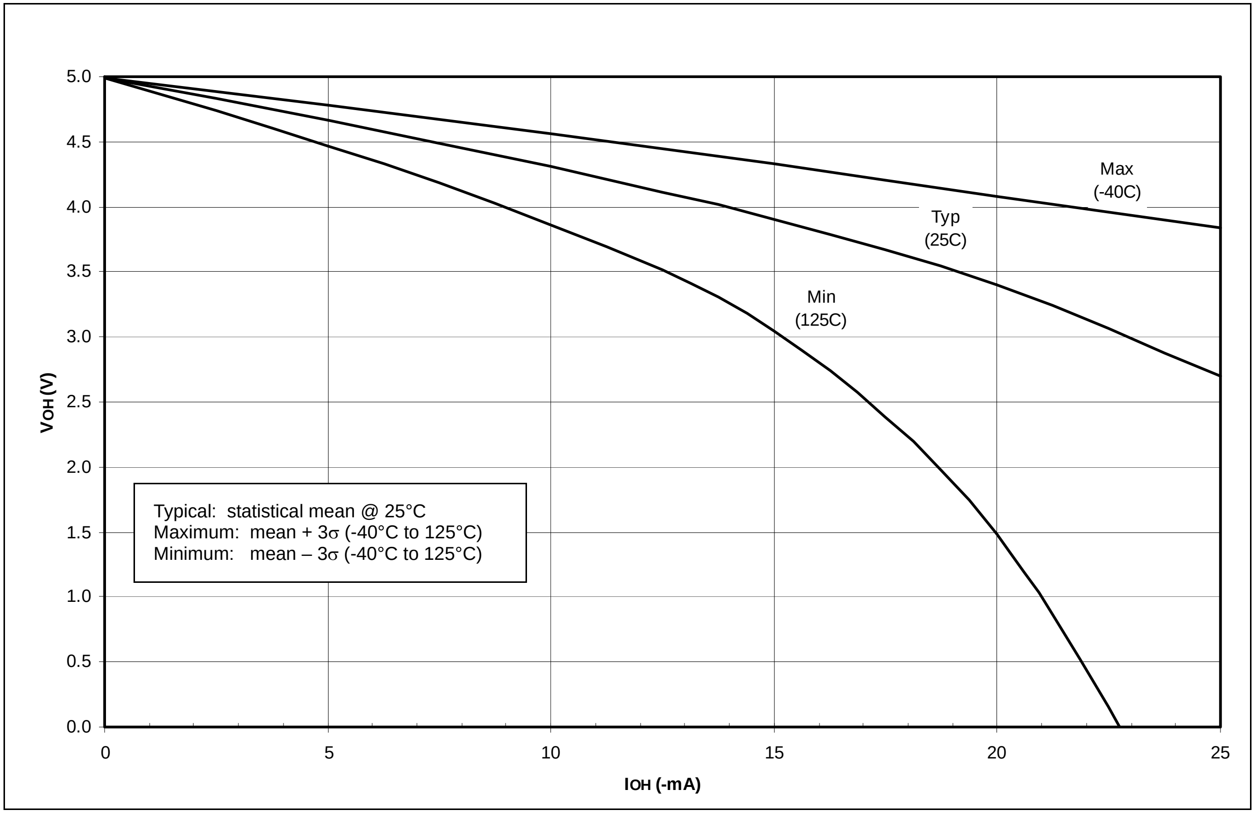
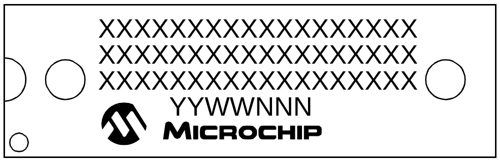


FIGURE 22-20: TYPICAL, MINIMUM AND MAXIMUM VOH vs. IOH (VDD = 5V, -40°C TO +125°C)

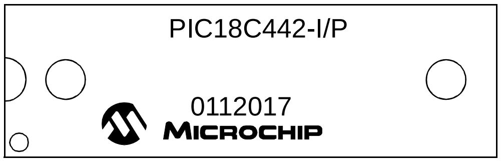


Package Marking Information (Cont'd)

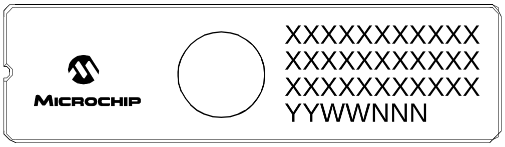
40-Lead PDIP



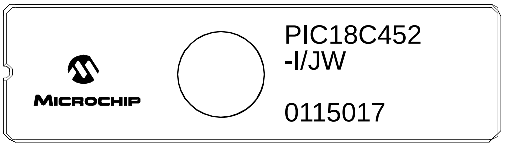
Example



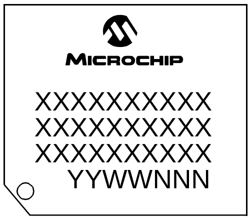
28- and 40-Lead JW (CERDIP)



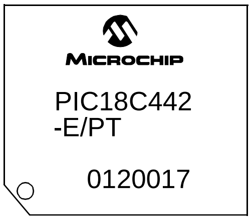
Example



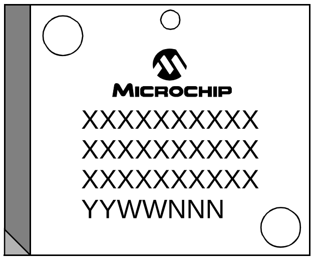
44-Lead TQFP



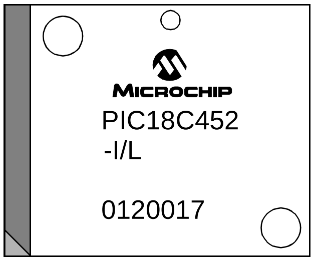
Example



44-Lead PLCC

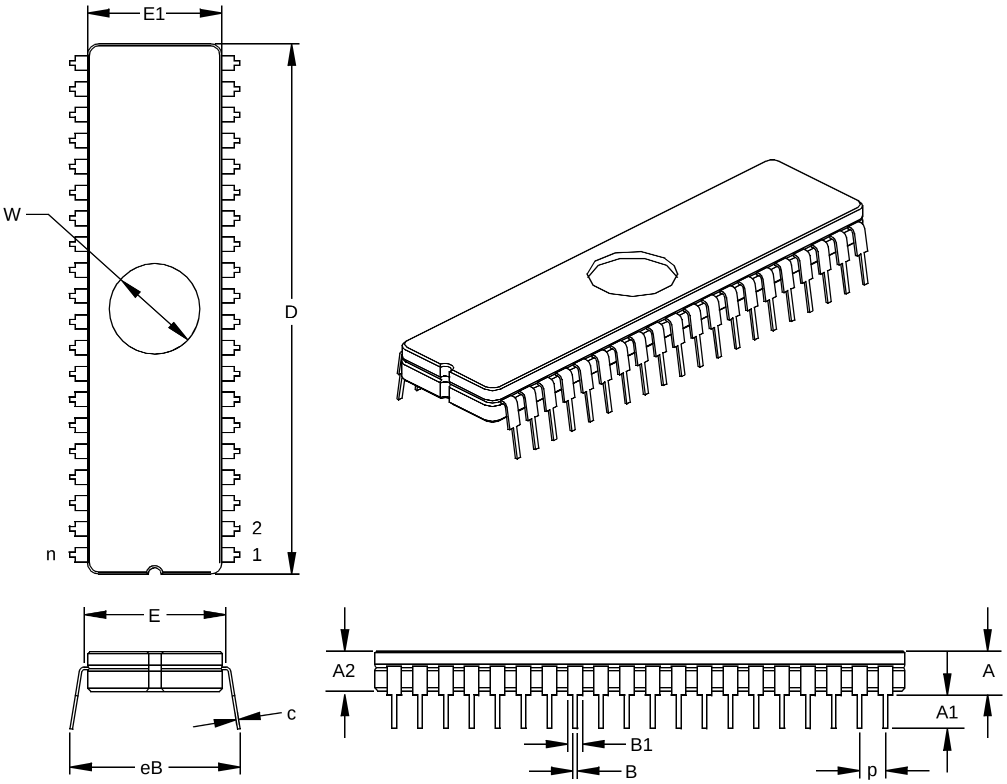


Example



40-Lead Ceramic Dual In-line with Window (JW) – 600 mil (CERDIP)

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



		INCHES*			MILLIMETERS		
	Units						
	Dimension Limits	MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		40			40	
Pitch	p		.100			2.54	
Top to Seating Plane	A	.185	.205	.225	4.70	5.21	5.72
Ceramic Package Height	A2	.155	.160	.165	3.94	4.06	4.19
Standoff	A1	.030	.045	.060	0.76	1.14	1.52
Shoulder to Shoulder Width	E	.595	.600	.625	15.11	15.24	15.88
Ceramic Pkg. Width	E1	.514	.520	.526	13.06	13.21	13.36
Overall Length	D	2.040	2.050	2.060	51.82	52.07	52.32
Tip to Seating Plane	L	.135	.140	.145	3.43	3.56	3.68
Lead Thickness	c	.008	.011	.014	0.20	0.28	0.36
Upper Lead Width	B1	.050	.053	.055	1.27	1.33	1.40
Lower Lead Width	B	.016	.020	.023	0.41	0.51	0.58
Overall Row Spacing §	eB	.610	.660	.710	15.49	16.76	18.03
Window Diameter	W	.340	.350	.360	8.64	8.89	9.14

* Controlling Parameter
§ Significant Characteristic
JEDEC Equivalent: MO-103
Drawing No. C04-014

- Code Examples
  - 16 x 16 Signed Multiply Routine 62
  - 16 x 16 Unsigned Multiply Routine 62
  - 8 x 8 Signed Multiply Routine 61
  - 8 x 8 Unsigned Multiply Routine 61
  - Changing Between Capture Prescalers 109
  - Fast Register Stack 39
  - Initializing PORTA 77
  - Initializing PORTB 80
  - Initializing PORTC 83
  - Initializing PORTD 85
  - Initializing PORTE 87
  - Loading the SSPBUF Register 122
  - Saving STATUS, WREG and BSR Registers
    - in RAM 75
- Code Protection 179, 186
- COMF 204
- Compare (CCP Module) 110
  - Associated Registers 111
  - Block Diagram 110
  - CCP Pin Configuration 110
  - CCPR1H:CCPR1L Registers 110
  - Software Interrupt 110
  - Special Event Trigger 99, 105, 110, 171
  - Timer1 Mode Selection 110
- Configuration Bits 179
- Context Saving During Interrupts 75
  - Example Code 75
- Conversion Considerations 288
- CPFSEQ 204
- CPFSGT 205
- CPFSLT 205

D

- Data Memory 42
  - General Purpose Registers 42
  - Special Function Registers 42
- DAW 206
- DC Characteristics 237, 240
- DECF 206
- DECFSNZ 207
- DECFSZ 207
- Device Differences 287
- Direct Addressing 51

E

- Electrical Characteristics 235
- Errata 5

F

- Firmware Instructions 187

G

- General Call Address Sequence 133
- General Call Address Support 133
- GOTO 208

I

- I/O Ports 77
- I^{2}C ($\overline{\text{SSP}}$ Module) 128
  - $\overline{\text{ACK}}$ Pulse 128, 129
  - Addressing 129
  - Block Diagram 128
  - Read/Write Bit Information (R/$\overline{\text{W}}$ Bit) 129
  - Reception 129
  - Serial Clock (RC3/SCK/SCL) 129
  - Slave Mode 128
  - Timing Diagram, Data 257
  - Timing Diagram, START/STOP Bits 256
  - Transmission 129
- I^{2}C Master Mode Reception 139
- I^{2}C Master Mode Repeated START Condition 138
- I^{2}C Module
  - Acknowledge Sequence Timing 142
  - Baud Rate Generator
  - Block Diagram
    - Baud Rate Generator 136
  - BRG Reset Due to SDA Collision 146
  - BRG Timing 136
  - Bus Collision
    - Acknowledge 144
    - Repeated START Condition 147
    - Repeated START Condition Timing
      - (Case 1) 147
    - Repeated START Condition Timing
      - (Case 2) 147
    - START Condition 145
    - START Condition Timing 145, 146
    - STOP Condition 148
    - STOP Condition Timing (Case 1) 148
    - STOP Condition Timing (Case 2) 148
    - Transmit Timing 144
  - Bus Collision Timing 144
  - Clock Arbitration 143
  - Clock Arbitration Timing (Master Transmit) 143
  - General Call Address Support 133
  - Master Mode 7-bit Reception Timing 141
  - Master Mode Operation 135
  - Master Mode START Condition 137
  - Master Mode Transmission 139
  - Master Mode Transmit Sequence 135
  - Multi-Master Mode 144
  - Repeat START Condition Timing 138
  - STOP Condition Receive or Transmit Timing 143
  - STOP Condition Timing 142
  - Waveforms for 7-bit Reception 130
  - Waveforms for 7-bit Transmission 130
- ICEPIC In-Circuit Emulator 230
- ID Locations 179, 186
- INCF 208
- INCFSZ 209
- In-Circuit Serial Programming (ICSP) 179, 186
- Indirect Addressing 51
  - FSR Register 50
- INFSNZ 209
- Instruction Cycle 39
- Instruction Flow/Pipelining 40
- Instruction Format 189