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	20MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	22
Program Memory Size	7KB (4K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	192 x 8
Voltage - Supply (Vcc/Vdd)	4V ~ 5.5V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16c63a-20e-ss

Tip #13.2 Reading a Sensor With Higher Accuracy – Charge Balancing Method

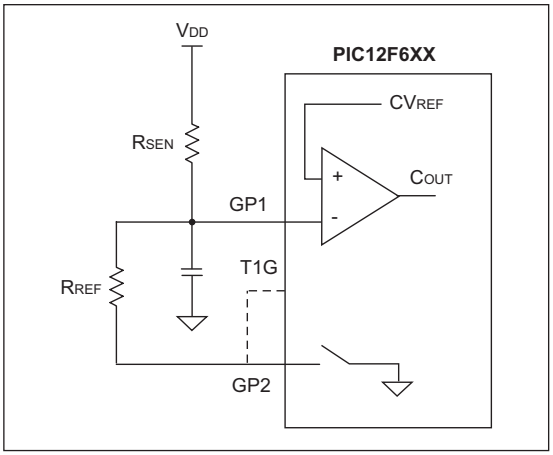
1. Sensor charges a capacitor
2. Reference resistor discharges the capacitor
3. Modulate reference resistor to maintain constant average charge in the capacitor
4. Use comparator to determine modulation

To improve resolution beyond 10 or 12 bits, a technique called “Charge Balancing” can be used. The basic concept is for the MCU to maintain a constant voltage on a capacitor by either allowing the charge to build through a sensor or discharge through a reference resistor. A timer is used to sample the capacitor voltage on regular intervals until a predetermined number of samples are counted. By counting the number of times the capacitor voltage is over an arbitrary threshold, the sensor voltage is determined.

The comparator and comparator voltage reference (CV_{REF}) on the PIC12F629/675 are ideal for this application.

1. GP1 average voltage = CV_{REF}
2. Time base as sampling rate
3. At the end of each time base period:
   - If GP1 > CV_{REF}, then GP2 Output Low
   - If GP1 < CV_{REF}, then GP2 Input mode
4. Accumulate the GP2 lows over many samples
5. Number of samples determines resolution
6. Number of GP2 lows determine effective duty cycle of R_{REF}

Figure 13-3



TIP #16 Optimizing Destinations

- Destination bit determines W for F for result
- Look at data movement and restructure

Example 16-1

Example: A + B → A

```
MOVF    A,W
ADDWF   B,W
MOVWF   A
```

3 instructions

```
MOVF    B,W
ADDWF   A,F
```

2 instructions

Careful use of the destination bits in instructions can save program memory. Here, register A and register B are summed and the result is put into the A register. A destination option is available for logic and arithmetic operations. In the first example, the result of the ADDWF instruction is placed in the working register. A MOVWF instruction is used to move the result from the working register to register A. In the second example, the ADDWF instruction uses the destination bit to place the result into the A register, saving an instruction.

TIP #17 Conditional Bit Set/Clear

- To move single bit of data from REGA to REGB
- Precondition REGB bit
- Test REGA bit and fix REGB if necessary

Example 17-1

```
BTFSS   REGA,2
BCF     REGB,5
BTFSC   REGA,2
BSF     REGB,5
```

4 instructions

```
BCF     REGB,5
BTFSC   REGA,2
BSF     REGB,5
```

3 instructions

One technique for moving one bit from the REGA register to REGB is to perform bit tests. In the first example, the bit in REGA is tested using a BTFSS instruction. If the bit is clear, the BCF instruction is executed and clears the REGB bit, and if the bit is set, the instruction is skipped.The second bit test determines if the bit is set, and if so, will execute the BSF and set the REGB bit, otherwise the instruction is skipped. This sequence requires four instructions.

A more efficient technique is to assume the bit in REGA is clear, and clear the REGB bit, and test if the REGA bit is clear. If so, the assumption was correct and the BSF instruction is skipped, otherwise the REGB bit is set. The sequence in the second example uses three instructions because one bit test was not needed.

One important point is that the second example will create a two-cycle glitch if REGB is a port outputting a high. This is caused by the BCF and BTFSC instructions that will be executed regardless of the bit value in REGA.

TIP #9 Two-Speed Start-Up

Two-speed startup is a useful feature on some nanoWatt and all nanoWatt XLP devices which helps reduce power consumption by allowing the device to wake up and return to sleep faster. Using the internal oscillator, the user can execute code while waiting for the Oscillator Start-up (OST) timer to expire (LP, XT or HS modes). This feature (called “Two-Speed Start-up”) is enabled using the IESO configuration bit. A Two-Speed Start-up will clock the device from an internal RC oscillator until the OST has expired. Switching to a faster internal oscillator frequency during start-up is also possible using the OSCCON register. The example below shows several stages on how this can be achieved. The number of frequency changes is dependent upon the designer’s discretion. Assume a 20 MHz crystal (HS Mode) in the PIC16F example below.

Example:

```
TCY
(Instruction Time)      Instruction
                        ORG   0x05                ;Reset vector

125 µs @ 32 kHz         BSF   STATUS,RP0         ;bank1
125 µs @ 32 kHz         BSF   OSCCON,IRCF2      ;switch to 1 MHz

4 µs @ 1 MHz            BSF   OSCCON,IRCF1      ;switch to 4 MHz

1 µs @ 4 MHz            BSF   OSCCON,IRCF0      ;switch to 8 MHz

500 ns                  application code
500 ns                  application code
…                       ….
..                      …
(eventually OST expires, 20 MHz crystal clocks the device)

200 ns                  application code
…                       ….
..                      …
```

TIP #10 Clock Switching

Some nanoWatt devices and all nanoWatt XLP devices have multiple internal and external clock sources, as well as logic to allow switching between the available clock sources as the main system clock. This allows for significant power savings by choosing different clocks for different portions of code. For example, an application can use the slower internal oscillator when executing non-critical code and then switch to a fast high-accuracy oscillator for time or frequency sensitive code. Clock switching allows much more flexible applications than being stuck with a single clock source. Clock switching sequences vary by device family, so refer to device data sheets or Family Reference Manuals for the specific clock switching sequences.

TIP #11 Use Internal RC Oscillators

If frequency precision better than ±5% is not required, it is best to utilize the internal RC oscillators inside all nanoWatt and nanoWatt XLP devices. The internal RC oscillators have better frequency stability than external RC oscillators, and consume less power than external crystal oscillators. Additionally, the internal clock can be configured for many frequency ranges using the internal PLL module to increase frequency and the postscaler to reduce it. All these options can be configured in firmware.

ECCP/CCP Register Listing

	Capture Mode	Compare Mode	PWM Mode
CCPxCON	Select mode	Select mode	Select mode, LSB of duty cycle
CCPRxL	Timer1 capture (LSB)	Timer1 compare (LSB)	MSB of duty cycle
CCPRxH	Timer1 capture (MSB)	Timer1 compare (MSB)	N/A
TRISx	Set CCPx pin to input	Set CCPx pin to output	Set CCPx pin(s) to output(s)
T1CON	Timer1 on, prescaler	Timer1 on, prescaler	N/A
T2CON	N/A	N/A	Timer2 on, prescaler
PR2	N/A	N/A	Timer2 period
PIE1	Timer1 interrupt enable	Timer1 interrupt enable	Timer2 interrupt enable
PIR1	Timer1 interrupt flag	Timer1 interrupt flag	Timer2 interrupt flag
INTCON	Global/ peripheral interrupt enable	Global/ peripheral interrupt enable	Global/ peripheral interrupt enable
PWM1CON(1)	N/A	N/A	Set dead band, auto-restart control
ECCPAS(1)	N/A	N/A	Auto-shutdown control

Note 1: Only on ECCP module.

CAPTURE TIPS ‘N TRICKS

In Capture mode, the 16-bit value of Timer1 is captured in CCPRxH:CCPRxL when an event occurs on pin CCPx. An event is defined as one of the following and is configured by CCPxCON<3:0>:

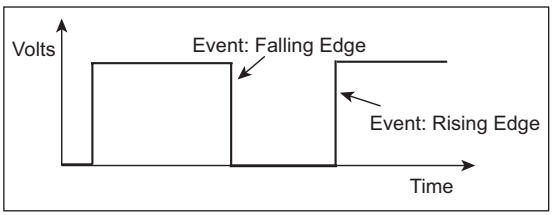
- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

“When Would I Use Capture Mode?”

Capture mode is used to measure the length of time elapsed between two events. An event, in general, is either the rising or falling edge of a signal (see Figure 1 “Defining Events”).

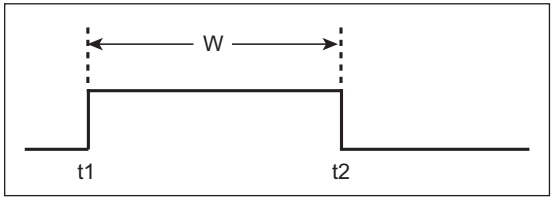
An example of an application where Capture mode is useful is reading an accelerometer. Accelerometers typically vary the duty cycle of a square wave in proportion to the acceleration acting on a system. By configuring the CCP module in Capture mode, the PIC microcontroller can measure the duty cycle of the accelerometer with little intervention on the part of the microcontroller firmware. Tip #4 goes into more detail about measuring duty cycle by configuring the CCP module in Capture mode.

Figure 1: Defining Events



TIP #3 Measuring Pulse Width

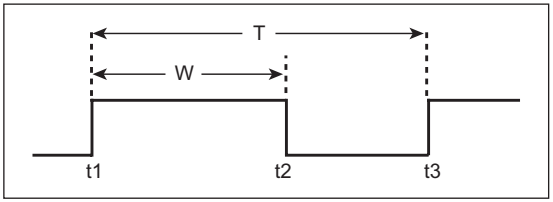
Figure 3-1: Pulse Width



1. Configure control bits CCPxM3:CCPxM0 (CCPxCON<3:0>) to capture every rising edge of the waveform.
2. Configure Timer1 prescaler so that Timer1 will run WMAX without overflowing.
3. Enable the CCP interrupt (CCPxIE bit).
4. When CCP interrupt occurs, save the captured timer value (t1) and reconfigure control bits to capture every falling edge.
5. When CCP interrupt occurs again, subtract saved value (t1) from current captured value (t2) – this result is the pulse width (W).
6. Reconfigure control bits to capture the next rising edge and start process all over again (repeat steps 3 through 6).

TIP #4 Measuring Duty Cycle

Figure 4-1: Duty Cycle



The duty cycle of a waveform is the ratio between the width of a pulse (W) and the period (T). Acceleration sensors, for example, vary the duty cycle of their outputs based on the acceleration acting on a system. The CCP module, configured in Capture mode, can be used to measure the duty cycle of these types of sensors. Here's how:

1. Configure control bits CCPxM3:CCPxM0 (CCPxCON<3:0>) to capture every rising edge of the waveform.
2. Configure Timer1 prescaler so that Timer1 will run TMAX(1) without overflowing.
3. Enable the CCP interrupt (CCPxIE bit).
4. When CCP interrupt occurs, save the captured timer value (t1) and reconfigure control bits to capture every falling edge.

Note 1: TMAX is the maximum pulse period that will occur.

5. When the CCP interrupt occurs again, subtract saved value (t1) from current captured value (t2) – this result is the pulse width (W).
6. Reconfigure control bits to capture the next rising edge.
7. When the CCP interrupt occurs, subtract saved value (t1) from the current captured value (t3) – this is the period (T) of the waveform.
8. Divide T by W – this result is the Duty Cycle.
9. Repeat steps 4 through 8.

TIP #5 Measuring RPM Using an Encoder

Revolutions Per Minute (RPM), or how fast something turns, can be sensed in a variety of ways. Two of the most common sensors used to determine RPM are optical encoders and Hall effect sensors. Optical encoders detect the presence of light shining through a slotted wheel mounted to a turning shaft (see Figure 5-1.) As the shaft turns, the slots in the wheel pass by the eye of the optical encoder. Typically, an infrared source on the other side of the wheel emits light that is seen by the optical encoder through slots in the wheel. Hall effect sensors work by sensing the position of the magnets in an electric motor, or by sensing a permanent magnet mounted to a rotating object (see Figure 5-2). These sensors output one or more pulses per revolution (depending on the sensor).

Figure 5-1: Optical Encoder

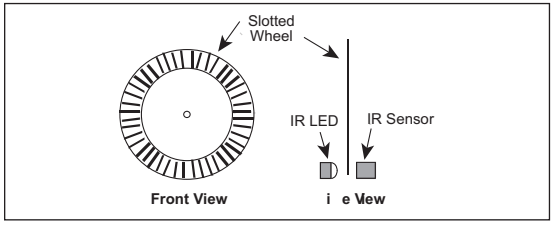
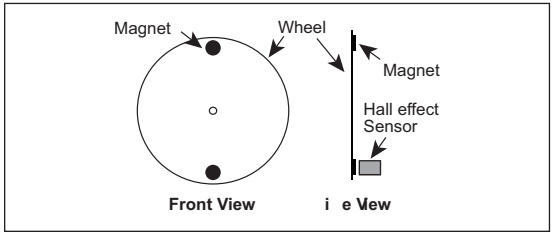


Figure 5-2: Hall Effect Sensor



In Figure 5-3 and Figure 5-4, the waveform is high when light is passing through a slot in the encoder wheel and shining on the optical sensor. In the case of a Hall effect sensor, the high corresponds to the time that the magnet is in front of the sensor. These figures show the difference in the waveforms for varying RPMs. Notice that as RPM increases, the period (T) and pulse width (W) becomes smaller. Both period and pulse width are proportional to RPM. However, since the period is the greater of the two intervals, it is good practice to measure the period so that the RPM reading from the sensor will have the best resolution. See Tip #1 for measuring period. The technique for measuring period with averaging described in Tip #2 is useful for measuring high RPMs.

Figure 5-3: Low RPM

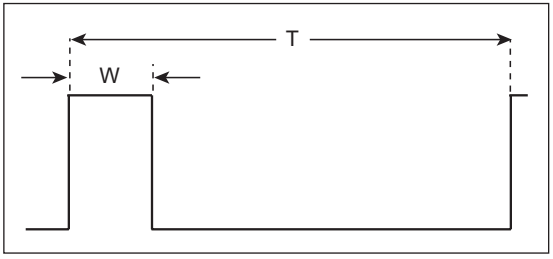
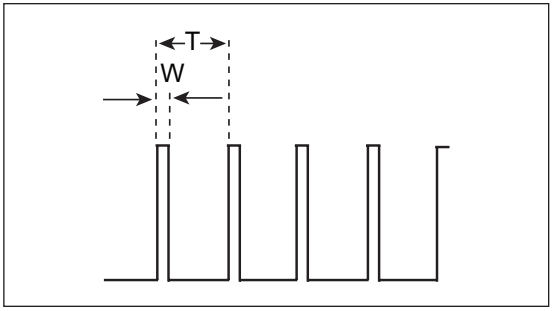


Figure 5-4: High RPM



PWM TIPS ‘N TRICKS

The ECCP and CCP modules produce a 10-bit resolution Pulse-Width Modulated (PWM) waveform on the CCPx pin. The ECCP module is capable of transmitting a PWM signal on one of four pins, designated P1A through P1D. The PWM modes available on the ECCP module are:

- Single output (P1A only)
- Half-bridge output (P1A and P1B only)
- Full-bridge output forward
- Full-bridge output reverse

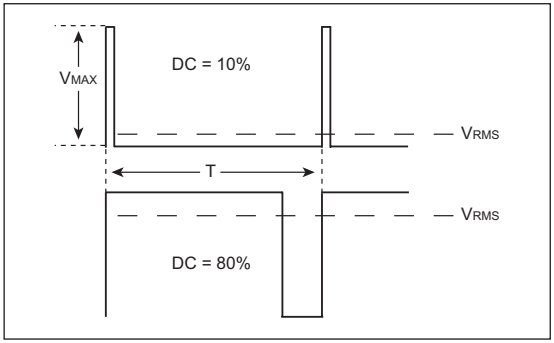
One of the following configurations must be chosen when using the ECCP module in PWM Full-bridge mode:

- P1A, P1C active-high; P1B, P1D active-high
- P1A, P1C active-high; P1B, P1D active-low
- P1A, P1C active-low; P1B, P1D active-high
- P1A, P1C active-low; P1B, P1D active-low

“Why Would I Use PWM Mode?”

As the next set of Tips ‘n Tricks demonstrate, Pulse-Width Modulation (PWM) can be used to accomplish a variety of tasks from dimming LEDs to controlling the speed of a brushed DC electric motor. All these applications are based on one basic principle of PWM signals – as the duty cycle of a PWM signal increases, the average voltage and power provided by the PWM increases. Not only does it increase with duty cycle, but it increases linearly. The following figure illustrates this point more clearly. Notice that the RMS and maximum voltage are functions of the duty cycle (DC) in the following Figure 12-3.

Figure 12-3: Duty Cycle Relation to V_{RMS}



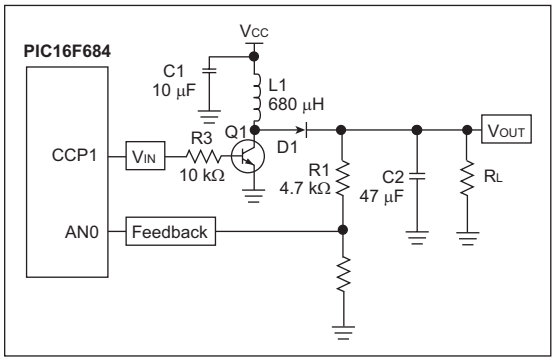
Equation 12-1 shows the relation between V_{RMS} and V_{MAX}.

Equation 12-1: Relation Between V_{RMS} and V_{MAX}

$$V_{RMS} = DC \times V_{MAX}$$

TIP #17 Boost Power Supply

Figure 17-1: Boost Power Supply Circuit



Hardware

Pulse-width modulation plays a key role in boost power supply design. Figure 17-1 shows a typical boost circuit. The circuit works by Q1 grounding the inductor (L1) during the high phase of the PWM signal generated by CCP1. This causes an increasing current to flow through L1 while V_{CC} is applied. During the low phase of the PWM signal, the energy stored in L1 flows through D1 to the storage capacitor (C2) and the load. V_{OUT} is related to V_{IN} by Equation 17-1.

Note: Technical Brief TB053 "*Generating High Voltage Using the PIC16C781/782*" provides details on boost power supply design.

The first parameter to determine is the duty cycle based upon the input and output voltages. See Equation 17-1.

Equation 17-1

$$\frac{V_{OUT}}{V_{IN}} = \frac{1}{1 - D}$$

Next, the value of the inductor is chosen based on the maximum current required by the load, the switching frequency and the duty cycle. A function for inductance in terms of load current is given by Equation 17-2, where T is the PWM period, D is the duty cycle, and I_{OUT} is the maximum load current.

Equation 17-2

$$L = \frac{V_{IN}\,(1 - D)\,DT}{2\,I_{OUT}}$$

The value for L is chosen arbitrarily to satisfy this equation given I_{OUT}, a maximum duty cycle of 75% and a PWM frequency in the 10 kHz to 100 kHz range.

Using the value chosen for L, the ripple current is calculated using Equation 17-3.

Equation 17-3

$$I_{RIPPLE} = \frac{V_{IN}\,DT}{L}$$

I_{RIPPLE} can not exceed the saturation current for the inductor. If the value for L does produce a ripple current greater than I_{SAT}, a bigger inductor is needed.

Note: All equations above assume a discontinuous current mode.

Firmware

The PWM duty cycle is varied by the microcontroller in order to maintain a stable output voltage over fluctuating load conditions. A firmware implemented PID control loop is used to regulate the duty cycle. Feedback from the boost power supply circuit provides the input to the PID control.

Note: Application Note AN258 "*Low Cost USB Microcontroller Programmer*" provides details on firmware-based PID control.

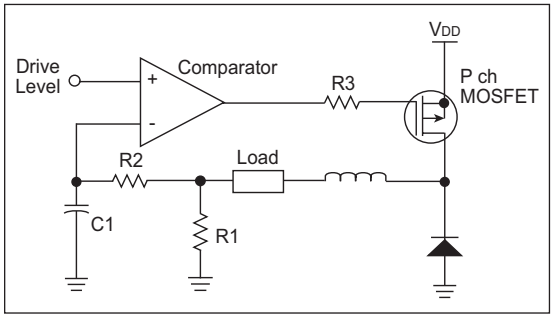
NOTES:

TIP #13 PWM High-Current Driver

This tip combines a comparator with a MOSFET transistor and an inductor to create a switch mode high-current driver circuit. (See Figure 13-1).

The operation of the circuit begins with the MOSFET off and no current flowing in the inductor and load. With the sense voltage across R1 equal to zero and a DC voltage present at the drive level input, the output of the comparator goes low. The low output turns on the MOSFET and a ramping current builds through the MOSFET, inductor, load and R1.

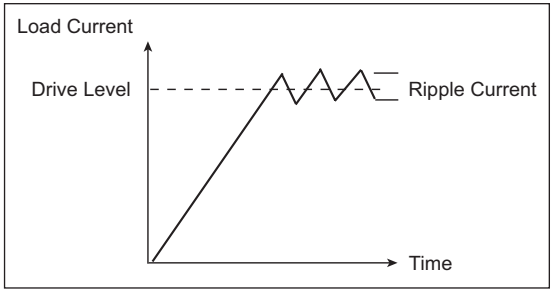
Figure 13-1: High Current Driver



When the current ramps high enough to generate a voltage across R1 equal to the drive level, the comparator output goes high turning off the MOSFET. The voltage at the junction of the MOSFET and the inductor then drops until D1 forward biases. The current continues ramping down from its peak level toward zero. When the voltage across the sense resistor R1 drops below the drive level, the comparator output goes low, the MOSFET turns on, and the cycle starts over.

R2 and C1 form a time delay network that limits the switching speed of the driver and causes it to slightly overshoot and undershoot the drive level when operating. The limit is necessary to keep the switching speed low, so the MOSFET switches efficiently. If R2 and C1 were not present, the system would run at a speed set by the comparator propagation delay and the switching speed of the MOSFET. At that speed, the switching time of the MOSFET would be a significant portion of the switching time and the switching efficiency of the MOSFET would be too low.

Figure 13-1: Current Through the Load



To design a PWM high current driver, first determine a switching speed (F_{SWX}) that is appropriate for the system. Next, choose a MOSFET and D1 capable of handling the load current requirements. Then choose values for R2 and C1 using Equation 13-1.

Equation 13-1

$$F_{SWX} = \frac{2}{R2 * C1}$$

Next determine the maximum ripple current that the load will tolerate, and calculate the required inductance value for L1 using Equation 13-2.

Equation 13-2

$$L = \frac{V_{DD} - V_{LOAD}}{I_{RIPPLE} * F_{SWX} * 2}$$

Finally, choose a value for R1 that will produce a feedback ripple voltage of 100 mV for the maximum ripple current I_{RIPPLE}.

Example:

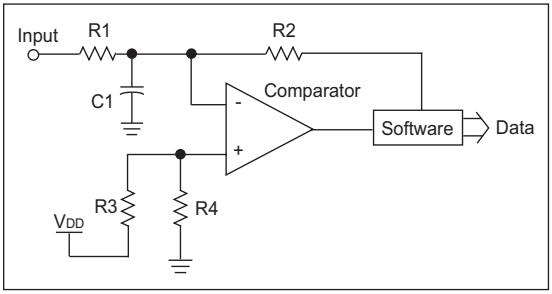
- F_{SWX} = 10 kHz, R2 = 22k, C1 = .01 µF
- I_{RIPPLE} = 100 mA, V_{DD} = 12V, V_L = 3.5V
- L = 4.25 mH

TIP #14 Delta-Sigma ADC

This tip describes the creation of a hardware/software-based Delta-Sigma ADC. A Delta-Sigma ADC is based on a Delta-Sigma modulator composed of an integrator, a comparator, a clock sampler and a 1-bit DAC output. In this example, the integrator is formed by R1 and C1. The comparator is an on-chip voltage comparator. The clock sampler is implemented in software and the 1-bit DAC output is a single I/O pin. The DAC output feeds back into the integrator through R2.

Resistors R3 and R4 form a $V_{DD}/2$ reference for the circuit (see Figure 14-1).

Figure 14-1: Delta-Sigma Modulator



In operation, the feedback output from the software is a time sampled copy of the comparator output. In normal operation, the modulator output generates a PWM signal which is inversely proportional to the input voltage. As the input voltage increases, the PWM signal will drop in duty cycle to compensate. As the input decreases, the duty cycle rises.

To perform an A-to-D conversion, the duty cycle must be integrated over time, digitally, to integrate the duty cycle to a binary value. The software starts two counters. The first counts the total number of samples in the conversion and the second counts the number of samples that were low. The ratio of the two counts is equal to the ratio of the input voltage over V_{DD}.

> **Note:** This assumes that R1 and R2 are equal and R3 is equal to R4. If R1 and R2 are not equal, then the input voltage is also scaled by the ratio of R2 over R1, and R3 must still be equal to R4.

For a more complete description of the operation of a Delta-Sigma ADC and example firmware, see Application Note AN700 “*Make A Delta-Sigma Converter Using a Microcontroller’s Analog Comparator Module*.”

Example:

- R3 = R4 = 10 kHz
- R1 = R2 = 5.1k
- C1 = 1000 pF

TIP #18 Logic: OR/NOR Gate

This tip shows the use of the comparator to implement an OR gate, and its complement, the NOR gate.

Resistors R1 and R2 drive the non-inverting input of the comparator with 1/3 V_{DD}. Resistors R3 and R4 average the voltages of the inputs A and B at the inverting input. If either A or B is high, the average voltage is 1/2 V_{DD} and the output of the comparator is high. Only if both A and B are low does the average voltage at the non-inverting input drop below 1/3 the supply voltage, causing the comparator output to go low. The operation of the NOR gate is identical to the OR gate, except the output is inverted due to the swap of the inverting and non-inverting inputs.

Note: Typical propagation delay for the circuit is 250-350 ns using the typical on-chip comparator peripheral of a microcontroller. Delay measurements were made with 10k resistance values.

While the circuit is fairly simple, there are a few requirements for correct operation:

1. The inputs A and B must drive from ground to V_{DD} for the circuit to operate properly.
2. The combination of R1 and R2 will draw current constantly, so they must be kept large to minimize current draw.
3. All resistances on the inverting input react with the input capacitance of the comparator, so the speed of the gate will be affected by the source resistance of A and B, as well as the size of resistors R3 and R4.
4. Resistor R1 must be 2 x R2.
5. Resistor R3 must be equal to R4.

Figure 18-1: OR Gate

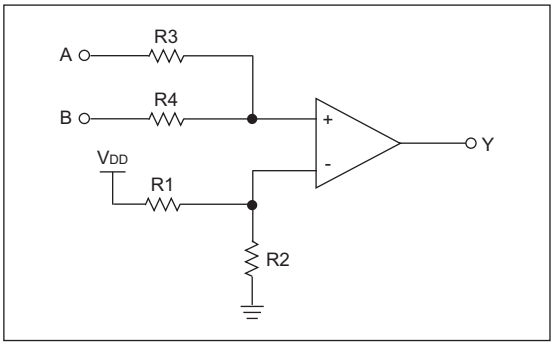
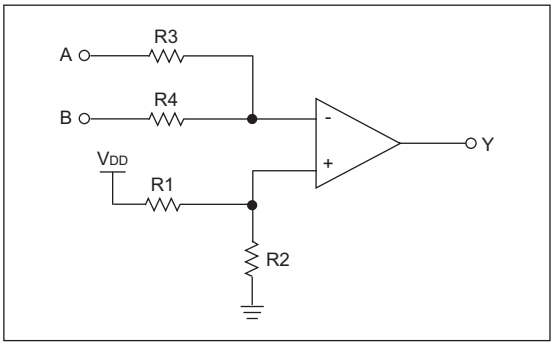


Figure 18-2: NOR Gate



Example:

- V_{DD} = 5V, R3 = R4 = 10k
- R1 = 10k, R2 = 5.1k

TIP #19 Logic: XOR/XNOR Gate

This tip shows the use of the comparator to implement an XOR gate and its complement the XNOR gate.

The operation is best described in three sections:

1. Both A and B inputs are low
   With both inputs low, the inverting input is held at .7V and the non-inverting is held at ground. This combination results in a low output.
2. Both A and B inputs are high
   With both inputs high, the inverting input is pulled up to V_{DD} and the non-inverting input is equal to 2/3 V_{DD} (the average of V_{DD} inputs and G_{ND}). This combination also results in a low output.
3. Input A or B is high
   With one input high and one low, The inverting input is held at .7V and the non-inverting input is equal to 1/3 V_{DD} (the average of a V_{DD} input and G_{ND}). This combination results in a high output.

Note: Typical propagation delay for the circuit is 250-350 ns using the typical on-chip comparator peripheral of a microcontroller. Delay measurements were made with 10k resistance values.

While the circuit is fairly simple, there are a few requirements for correct operation:

1. The inputs A and B must drive from ground to V_{DD} for the circuit to operate properly.
2. All resistances on the both inputs react with the input capacitance of the comparator, so the speed of the gate will be affected by the source resistance of A and B, as well as, the size of resistors R1, R2, R3 and R4.
3. Resistor R1, R2 and R3 must be equal.
4. Resistor R4 must be small enough to produce a 1.0V, or lower, voltage drop across D1 and D2.

Figure 19-1: XOR Gate

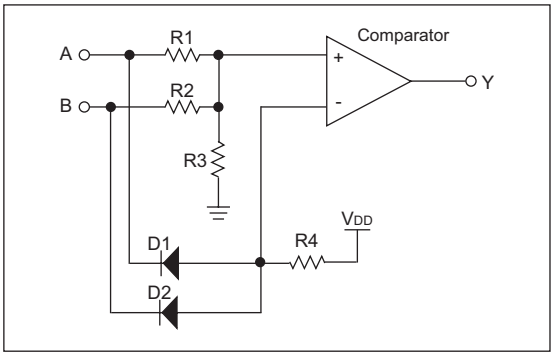
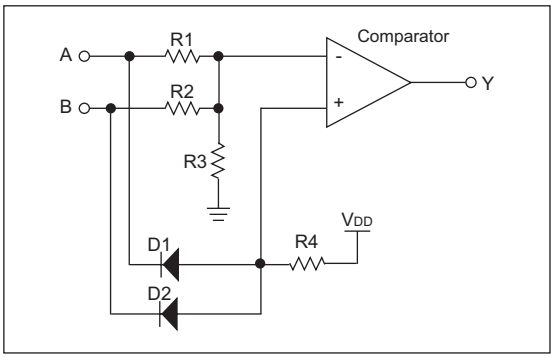


Figure 19-2: XNOR Gate



Example:

- D1 = D2, = 1N4148
- R4 = 10k, R1 = R2 = R3 = 5.1k

CHAPTER 6
LCD PIC® Microcontroller Tips ‘n Tricks

Table Of Contents

TIPS ‘N TRICKS INTRODUCTION

Using an LCD PIC® MCU for any embedded application can provide the benefits of system control and human interface via an LCD. Design practices for LCD applications can be further enhanced through the implementation of these suggested “Tips ‘n Tricks”.

This booklet describes many basic circuits and software building blocks commonly used for driving LCD displays. The booklet also provides references to Microchip application notes that describe many LCD concepts in more detail.

TIP #1 Typical Ordering Considerations and Procedures for Custom Liquid Displays

1. Consider what useful information needs to be displayed on the custom LCD and the combination of alphanumeric and custom icons that will be necessary.
2. Understand the environment in which the LCD will be required to operate. Operating voltage and temperature can heavily influence the contrast of the LCD and potentially limit the type of LCD that can be used.
3. Determine the number of segments necessary to achieve the desired display on the LCD and reference the PIC Microcontroller LCD matrix for the appropriate LCD PIC microcontroller.
4. Create a sketch/mechanical print and written description of the custom LCD and understand the pinout of the LCD. (Pinout definition is best left to the glass manufacturer due to the constraints of routing the common and segment electrodes in two dimensions.)
5. Send the proposed LCD sketch and description for a written quotation to at least 3 vendors to determine pricing, scheduling and quality concerns.
   a) Take into account total NRE cost, price per unit, as well as any setup fees.
   b) Allow a minimum of two weeks for formal mechanical drawings and pin assignments and revised counter drawings.
6. Request a minimal initial prototype LCD build to ensure proper LCD development and ensure proper functionality within the target application.
   a) Allow typically 4-6 weeks for initial LCD prototype delivery upon final approval of mechanical drawings and pin assignments.
7. Upon receipt of prototype LCD, confirm functionality before giving final approval and beginning production of LCD.

Note: Be sure to maintain good records by keeping copies of all materials transferred between both parties, such as initial sketches, drawings, pinouts, etc.

TIP #2 LCD PIC® MCU Segment/ Pixel Table

Table 2-1: Segment Matrix Table

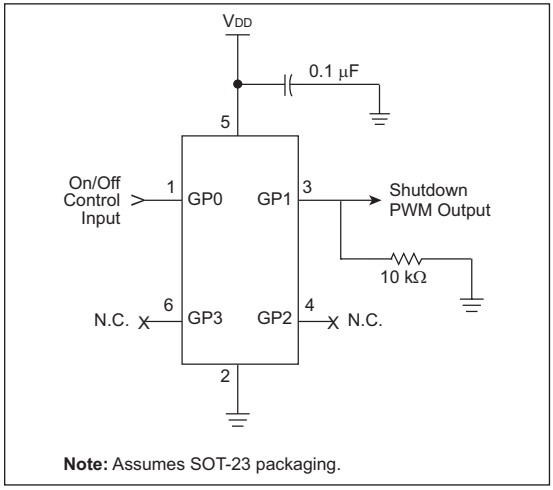
Multiplex Commons	Maximum Number of Segments/Pixels					Bias
	PIC16F913/ 916	PIC16F914/ 917	PIC16F946	PIC18F6X90 (PIC18F6XJ90)	PIC18F8X90 (PIC18F8XJ90)	
Static (COM0)	15	24	42	32/ (33)	48	Static
1/2 (COM1: COM0)	30	48	84	64/ (66)	96	1/2 or 1/3
1/3 (COM2: COM0)	45	72	126	96/ (99)	144	1/2 or 1/3
1/4 (COM3: COM0)	60	96	168	128/ (132)	192	1/3

This Segment Matrix table shows that Microchip's 80-pin LCD devices can drive up to 4 commons and 48 segments (192 pixels), 64-pin devices can drive up to 33 segments (132 pixels), 40/44 pin devices can drive up to 24 segments (96 pixels) and 28-pin devices can drive 15 segments (60 segments).

TIP #1 Soft-Start Using a PIC10F200

Almost all power supply controllers are equipped with shutdown inputs that can be used to disable the MOSFET driver outputs. Using Pulse-Width Modulation (PWM), the amount of time the power supply is allowed to operate can be slowly incremented to allow the output voltage to slowly rise from 0% to 100%.

Figure 1-1: Soft-Start Circuit Schematic



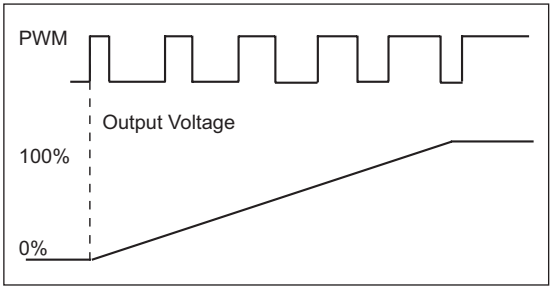
Note: Assumes SOT-23 packaging.

This technique is called soft-start and is used to prevent the large inrush currents that are associated with the start-up of a switching power supply.

GP0 on the PIC MCU is used to enable or disable the soft-start. Once enabled, the on-time of the PWM signal driving the shutdown output will increase each cycle until the power supply is fully on.

During the PIC MCU Power-on Reset, the PWM output (GP1) is initially in a high-impedance state. A pull-down resistor on the PWM output ensures the power supply will not unexpectedly begin operating.

Figure 1-2: Timing Diagram

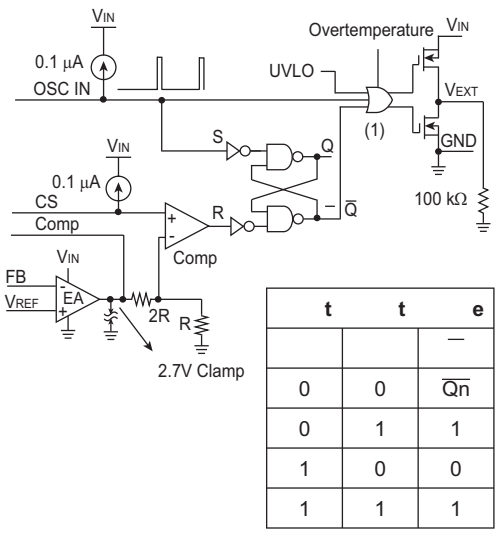


It is important to note that this type of soft-start controller can only be used for switching regulators that respond very quickly to changes on their shutdown pins (such as those that do cycle-by-cycle limiting). Some linear regulators have active-low shutdown inputs, however, these regulators do not respond fast enough to changes on their shutdown pins in order to perform soft-start.

Example software is provided for the PIC10F200 which was taken from TB081. Please refer to TB081, “*Soft-Start Controller For Switching Power Supplies*” (DS91081) for more information.

TIP #6 Current Limiting Using the MCP1630

Figure 6-1: MCP1630 High-Speed PWM



Note 1: During overtemperature, VEXT driver is high-impedance.

The block diagram for the MCP1630 high-speed PWM driver is shown in Figure 6-1. One of the features of the MCP1630 is the ability to perform current limiting. As shown in the bottom left corner of the diagram, the output of the Error Amplifier (EA) is limited by a 2.7V clamp. Therefore, regardless of the actual error, the input to the negative terminal of the comparator (labeled Comp) is limited to 2.7V ÷ 3 or 0.9V.

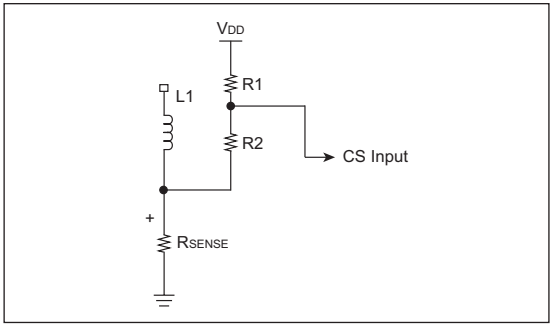
It is possible to implement the current limiting by using a single sense resistor. In this case, the maximum current would be given by Equation 6-1.

Equation 6-1

$$I_{MAX} = (0.9V) / R_{SENSE}$$

For high current applications, this method may be acceptable. When lower current limits are required, the size of the sense resistor, RSENSE, must be increased. This will cause additional power dissipation. An alternative method for lower current limits is shown in Figure 6-2.

Figure 6-2: Low Current Limits



In this case, the Current Sense (CS) input of the MCP1630 is biased upward using the R1/R2 resistor divider. The equations for the new current limit are shown in Equation 6-2.

Equation 6-2

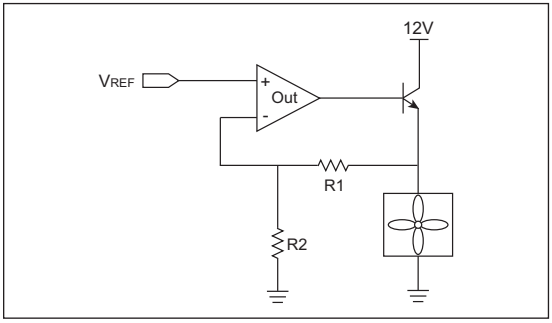
$$0.9V = \frac{(V_{DD} - I_{MAX} \bullet R_{SENSE}) \bullet R2}{R1 + R2}$$

Equation 6-2 can be solved to determine the values of R1 and R2 that provide the desired current limit.

Method 2 – Linear Control

When using PWM, the voltage will vary between a maximum and a minimum, however, is it also possible to use a linear method to control fan speed, as shown in Figure 14-4.

Figure 14-4: Linear Control Drive



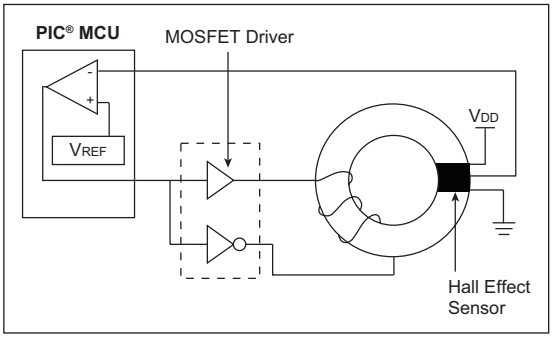
The voltage applied at the non-inverting terminal of the op amp is used to vary the voltage across the op amp. The non-inverting terminal voltage can be produced by a Digital-to-Analog Converter (DAC) or by the method shown in Tip #11.

When using this method, care must be taken to ensure that the fan voltage is not too low or the fan will stop spinning. One advantage this method has over PWM is that the tachometer output will function properly on 3-wire fans. The disadvantage, however, is that it often offers less speed control. For example, a 12V fan will not spin below 8V, so a range of only 4V is available for speed control. A 5V fan will not spin below 4V and so the control range is only 1V, which is often unacceptable. Another disadvantage is the power consumption of the circuit. The transistor will dissipate more power than the PWM method.

TIP #15 High Current Delta-Sigma Based Current Measurement Using a Slotted Ferrite and Hall Effect Device

Many current sensors rely on ferrite cores. Non-linearity in the ferrite can lead to inaccurate results, especially at high currents. One way to avoid the non-linearities is to keep the net flux in the ferrite near zero. Consider the circuit in Figure 15-1.

Figure 15-1: Hall Effect Current Measurement Schematic



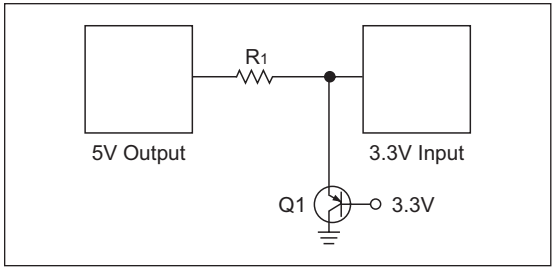
The Hall effect sensor output is proportional to the current being measured. When I_{IN} = 0 amps, the output of the sensor will be $V_{DD}/2$. A current passing through the sensor in one direction will increase the output of the sensor, and a current in the other direction will decrease the output of the sensor.

The output of the comparator is used to drive a coil of wire wound around the ferrite core. This coil of wire will be used to create flux in the opposite direction as the flux imposed in the core.

TIP #11 5V → 3.3V Active Clamp

One problem with using a diode clamp is that it injects current onto the 3.3V power supply. In designs with a high current 5V outputs, and lightly loaded 3.3V power supply rails, this injected current can float the 3.3V supply voltage above 3.3V. To prevent this problem, a transistor can be substituted which routes the excess output drive current to ground instead of the 3.3V supply. Figure 11-1 shows the resulting circuit.

Figure 11-1: Transistor Clamp

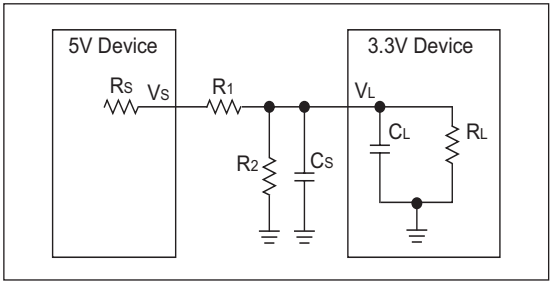


The base-emitter junction of Q1 performs the same function as the diode in a diode clamp circuit. The difference is that only a small percentage of the emitter current flows out of the base of the transistor to the 3.3V rail, the bulk of the current is routed to the collector where it passes harmlessly to ground. The ratio of base current to collector current is dictated by the current gain of the transistor, typically 10-400, depending upon which transistor is used.

TIP #12 5V → 3.3V Resistor Divider

A simple resistor divider can be used to reduce the output of a 5V device to levels appropriate for a 3.3V device input. An equivalent circuit of this interface is shown in Figure 12-1.

Figure 12-1: Resistive Interface Equivalent Circuit



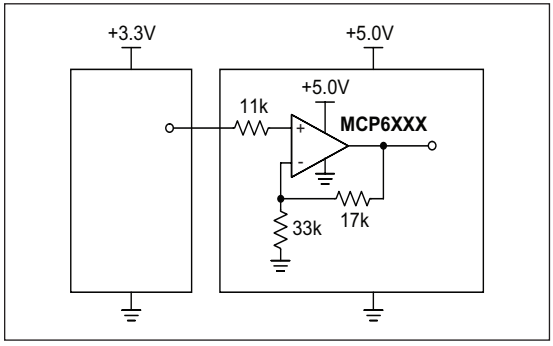
Typically, the source resistance, Rs, is very small (less than 10Ω) so its affect on R1 will be negligible provided that R1 is chosen to be much larger than Rs. At the receive end, the load resistance, R_L, is very large (greater than 500 kΩ) so its affect on R2 will be negligible provided that R2 is chosen to be much less than R_L.

There is a trade-off between power dissipation and transition times. To keep the power requirements of the interface circuit at a minimum, the series resistance of R1 and R2 should be as large as possible. However, the load capacitance, which is the combination of the stray capacitance, Cs, and the 3.3V device input capacitance, C_L, can adversely affect the rise and fall times of the input signal. Rise and fall times can be unacceptably long if R1 and R2 are too large.

TIP #14 3.3V → 5V Analog Gain Block

To scale analog voltage up when going from 3.3V supply to 5V supply. The 33 kΩ and 17 kΩ set the op amp gain so that the full scale range is used in both sides. The 11 kΩ resistor limits current back to the 3.3V circuitry.

Figure 14-1: Analog Gain Block



TIP #15 3.3V → 5V Analog Offset Block

Offsetting an analog voltage for translation between 3.3V and 5V.

Shift an analog voltage from 3.3V supply to 5V supply. The 147 kΩ and 30.1 kΩ resistors on the top right and the +5V supply voltage are equivalent to a 0.85V voltage source in series with a 25 kΩ resistor. This equivalent 25 kΩ resistance, the three 25 kΩ resistors, and the op amp form a difference amplifier with a gain of 1 V/V. The 0.85V equivalent voltage source shifts any signal seen at the input up by the same amount; signals centered at 3.3V/2 = 1.65V will also be centered at 5.0V/2 = 2.50V. The top left resistor limits current from the 5V circuitry.

Figure 15-1: Analog Offset Block