INTRODUCTION

It is the responsibility of engineers to get the most performance and capability out of the resources available. In this application note, we present ideas that have helped previous designs reduce size and cost. This application note looks at different ways of:

• saving input pins
• driving multiple LEDs
• saving output pins
• maximizing pin usage
• power saving tricks

SAVING INPUT PINS

Multiple Switches on a Single Input

Some designs connect one switch to a single input while others use keypad style multiplexing to get multiple switches on fewer inputs. However, some designs are able to get many switches on one input. This technique requires you to change the switch problem from digital-to-analog. Once this change has been made, we can measure the analog signals with the digital microcontroller.

ANALOG-TO-DIGITAL CONVERTER (ADC)

If your microcontroller has an extra ADC input available, the ADC can be used to detect one or more switch closings by using a few resistors. See Figure 1.

By using a switched resistor network, the value of a voltage divider can be changed. The voltage is then measured with the ADC. Using simple code, the switch value is determined. One downside of this design is detecting combinations of switches. To detect combinations, the resistors have to be chosen so that the voltage resulting from a switch combination is unique.

COMPARATOR AND TIMING

A comparator can be used to detect multiple switches on a single line. If the resistor values are selected appropriately then the internal voltage reference could be adjusted to find the pressed switches. By making a small change to the circuit, two additional methods become possible. We can add enough circuitry to use the comparator as an ADC and then use the method presented above, or we can change the circuit slightly and charge a capacitor. See Figure 2.
As Figure 2 shows, R4 has been changed to a capacitor (C1). Instead of measuring voltage, we are now measuring the time it takes the capacitor to charge. Again, care must be taken to ensure that the capacitor charge time is reasonable. If the timing is too fast, the microcontroller could miss the event. If it is too slow, it will take all day to process the switch inputs. The resistor values should be chosen to provide enough gap in charge times so that each switch or switch combination is unique. If the design does not allow simultaneous switch presses, the resistor selection is greatly simplified.

Using a single comparator input to detect the switch presses requires that the input pin be changed to an output. This allows the capacitor to be pulled to ground. Afterward, the pin is changed back into the single comparator input to let the capacitor charge up. When the capacitor voltage reaches the comparator threshold voltage, the state of the comparator will change. The software will measure the time it takes the capacitor to charge to the threshold from the time the pin direction is changed.

**DIGITAL THRESHOLDS AND TIMING**

The digital method is similar to the comparator method. In fact, the circuitry is the same. Instead of using a comparator threshold to detect the charge of the capacitor, the V\text{IH} threshold is used. Since the V\text{IH} threshold of a Schmitt trigger input is larger than the TTL input, it provides a larger input detection range. Therefore, it is best to use a Schmitt trigger input to detect the capacitor charge. The downside to using a digital input is the current consumption is higher. The higher current consumption is caused by placing intermediate analog voltages on a CMOS gate. V\text{IH} charges between parts so calibration may be required.

**Keypads to a Single Input**

Keypads are commonly used to input numeric data into many applications. Therefore, PORTB on the PICmicro microcontroller includes special features to simplify reading the keypad. While keypads have benefits, they require many pins in order to be functional. A typical 4x4 keypad requires eight I/O pins. Figure 3 illustrates a resistor network, which changes a 4x4 keypad into a switchable resistor matrix. The resulting resistance is read by an ADC or a capacitor charge timer, which determines the key that is pressed. This is practical only when one key is pressed at a time.
Driving two LEDs with one pin is simple. A bit more complex, driving 6 LEDs with 3 I/O pins is shown in Figure 5.

**FIGURE 5:** 6 LEDS ON 3 I/O PINS

![Diagram of 6 LEDs on 3 I/O pins]

If this concept is extended further, four I/O pins can be used to drive 12 LEDs as in Figure 6.

**FIGURE 6:** 12 LEDS ON 4 I/O PINS

![Diagram of 12 LEDs on 4 I/O pins]

Although this is a very interesting concept, there are potential problems with this approach. Additional software is required to drive this display. If the display is illuminating more than one LED at a time, then significant amounts of software time can be consumed multiplexing the LEDs to appear to be lit simultaneously. If the application is driving a dot matrix graphics display where each LED is one pixel, then the display will get dimmer as the LED count goes up. This is due to the amount of time spent switching one LED on at a time. The duty cycle and intensity is reduced. However, if the application’s purpose is to light only 1 LED at a time, there is no reduction in intensity and this concept works well.

As mentioned earlier, driving this display requires some work. Each LED requires two outputs to be active, one high and one low. The rest of the pins are inputs so that no current flows through them. To light more than one LED at a time, the inputs and outputs must be re-configured often enough to appear that both LEDs are active at the same time. It is best to keep each LED on for the same amount of time and reserve time slots in the multiplexing for the LEDs that are not lit. If the multiplexing only cycles between active LEDs and ignores the inactive LEDs, the light intensity will decrease as additional LEDs are lit.

**Driving Multiple Seven-segment Displays with 8 I/O Pins**

Seven-segment displays are one of the most often used numeric (and sometimes alphanumeric) displays. Like the keypads, seven-segment displays require many output pins. They require output pins for each segment and may require output pins to control the common cathode (or anode) of each digit. In the case of a single digit display, pins are only required for the segments as the common cathode (or anode) can be tied to ground (or power). Which begs the question, how to drive multiple digits with only a small handful of pins.

If you only need to drive 2 digits, it is possible to use eight pins by mixing common cathode and common anode type displays. See Figure 7.

**FIGURE 7:** 2 DIGITS ON 8 I/O PINS

![Diagram of 2 digits on 8 I/O pins]

One method of driving two displays is to use two seven-segment display driver chips. These chips translate four bits of data into the correct segments pattern. Of course, if you need more than two digits, you will need additional help.

---

*Note:* Pins A - G are Digital I/O.
Another technique is to use an octal latch, such as 74F573, to be the digit selection. See Figure 8.

To use the octal latch described in Figure 8, the latch must be loaded with the correct digit. The procedure is as follows:

1. Raise LE and set <A-G> to the desired digit pattern. <A> = 1 and <B-G> = 0 will enable G1 or Digit 1.
2. Lower LE and set <A-G> to display the correct numeral. <A-F> = 1 and <G> = 0 will display '0' on Digit 1.
3. Delay long enough for the reader to notice. With 3 ms per digit and 21 ms for all 7 digits, the result is a 47 Hz refresh rate.
4. Repeat.

SAVING OUTPUT PINS

Using Opto-isolators as Output Expansion

Earlier, a method was described to drive many LEDs with just a few I/O pins. This same technique can be used with opto-isolators to drive a large number of outputs with just a few I/O pins.

When D1 is activated, Q1 allows current to flow. This will reduce the voltage on C1 and activate the load. When D1 is deactivated, Q1 releases C1 and the voltage on C1 begin to rise. C1 will rise until M1 turns on and the load is deactivated. R1 and R2 slow the turn-on of M1 and provide the resistive component of a RC time constant. The time to turn off depends on the gate threshold of M1 and the RC time constant of (R1 + R2) x C. R2 should be chosen to limit the turn-on rate of M1. R1 is required to limit the turn off rate. C1 is added to further reduce the turn off rate and reduce the variation due to different gate capacitance. For example:

\[ R1 = R2 = 1K \]
\[ C1 = 1\mu F \]
\[ V_{\text{SUPPLY}} = 12\text{V} \]
\[ V_{\text{GTH}} = -5\text{V} \]

When M1 is on, \( V_{\text{GATE}} \) is 6V.

\[ V_{C1} = 0\text{V} \]

To turn M1 off, \( V_{\text{GATE}} \) must reach 5V so:

\[ V_{C1} \geq 2\text{V} \]
Turn off time is the time it takes the capacitor to charge to 2V, solving:

$$V_T = V_{\text{SUPPLY}} \left( 1 - C^{\frac{-T}{RC}} \right) \text{ for } T$$

$$T = -RC \cdot L_N \left( 1 - \frac{V_T}{V_{\text{SUPPLY}}} \right)$$

$$R = R1 + R2 = 2000$$

$$V_T = 2$$

$$T = 365 \mu s$$

Refresh the output every 300 \(\mu s\) and the load will remain active. Ignore the output and the load will turn off.

**FIGURE 9: OPTO-ISOLATOR CONTROLLED LOAD W/ AUTOMATIC TURN OFF**

This method may not work for all applications. However, Figure 10 shows how seven-segment displays can use this technique to multiplex the digits.
FIGURE 10: 12 DIGIT MULTIPLEXING WITH OPTO-ISOLATORS
Using PICmicro as Output Expansion

Many ideas for saving I/O pins require more cost. The previous example would add opto-isolators, which would increase cost. It is also possible to use discrete logic chips to add functionality and decrease pin usage. This works well, but often takes more than one. Consider using a second PICmicro microcontroller instead of additional logic chips. In many cases, small PICmicro microcontrollers are less expensive and take less board space than several logic ICs. The benefit is its function. It can be specific to the application, which simplifies the code. A PICmicro microcontroller can easily be programmed to cycle through multiple seven-segment display digits at the command of one input pin. The multiplexer can be reset to the first digit by a longer pulse.

This application would take a few logic chips but it only requires one PICmicro microcontroller. By using two small PICmicro microcontrollers, it may be easier to fit the design into a small irregular space, where a single high-pin count device would not fit.

FIGURE 11: PICMICRO AS I/O EXPANDER

Note: Pins A - G are Digital I/O.
MAXIMIZING PIN USAGE

Sharing I/O Pins with LCD’s

HD44780A type LCD’s may be the most popular alphanumeric display used with embedded systems. They are simple to interface and are quite flexible in their output. The only downside is they use 6 to 11 I/O pins. It would be more useful if the pins could accomplish additional features. With proper technique, it is possible to reuse the LCD control pins. The LCD latches data while the E pin is high. As long as the E pin is low, the remaining pins can be used for any function. Although, there are a few issues:

1. If the application is an output, it must accept the pin transitions while the pin is driving the LCD;
2. If the application is an input, it must not drive the pin harder than the LCD.

Of course, these two issues can be reduced if the port is gated with a bus transceiver and the E clock is used to enable or disable the link between the LCD Port pins and the application.

FIGURE 12: SHARING I/O PINS WITH LCD CONTROLLER
POWER SAVING TRICKS

Variable Frequency RC Oscillator

The external RC oscillator is inexpensive and easy to use. Unfortunately, it only has one setting. By using a free I/O pin, a second resistor or capacitor can be added to the circuit to shift the frequency. By doing this, you have a multi-speed RC oscillator creating low speed for low current and high speed for quick math.

![VARIABLE SPEED RC OSCILLATOR](image1.png)

RC Watchdog Wake-up

The watchdog timer is very useful for protecting your code or waking the microcontroller up out of SLEEP mode. However, for some applications it consumes too much power. If your microcontroller has a comparator, use a resistor and capacitor on the comparator to provide the wake-up. Simply wire the R and C to the comparator input, program the threshold voltage, and enable the interrupt. Then, zero the capacitor and go into SLEEP mode. The capacitor will charge, reach the threshold, cause an interrupt, and wake-up your program. See procedure steps below. The RC network can be designed to use very little current.

WAKE-UP PROCEDURE

1. Set comparator input to a digital output.
2. Write a zero to clear the capacitor.
3. Set the comparator input back to a comparator input.
4. Read the comparator state flag to clear it.
5. Enable the comparator interrupt.
6. Set to SLEEP.

![LOW CURRENT WAKE-UP](image2.png)

CONCLUSION

This document is filled with ideas that have been successfully used to reduce cost or simplify designs. This is not a comprehensive list of cost saving ideas and these ideas may not work for all situations. Hopefully, these ideas have sparked your imagination to the possibilities of alternate ways to use a PICmicro microcontroller.

REFERENCE DOCUMENTS

Complementary LED Drive Technical Brief, TB029; Microchip Technology Inc.
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip’s Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.