

Momentary Pushbutton Switch Implementation with PIC Microcontroller Application Note

Cammi Stewart

ECE 480 – Spring 2007

March 30, 2007

Executive Summary

This application note explains both the hardware as well as the software behind using a momentary pushbutton switch with a PIC (peripheral interface controller). The hardware portion details the inner workings of a basic pushbutton switch and how it makes contacts with a circuit. The software portion details how to handle user input in the PIC programming when the button is pressed.

Keywords

Switch, pushbutton, actuator, PIC, bounce, debounce

Introduction

Switches can be used for a variety of applications but their basic purpose is simple. Switches, denoted by the name, are used to make, break, or change connections in a circuit. There are many different kinds of switches. Some switches, such as sliders, when toggled, will remain in that position until further intervention (Figure 1). These are used to select between multiple contact points to close the circuit. Other switches are only temporary when pressed. A momentary pushbutton switch contains a spring that returns the actuator to the original position. This type of switch can be used when a user wants to use the same switch for multiple triggers.



Figure 1 – Example of different switch types: slider switches

Objective

This application note explains how a momentary pushbutton switch works and how to implement it with a PIC microprocessor.

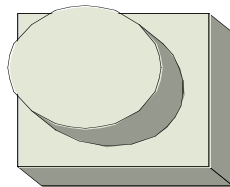


Figure 2 – Pushbutton Switch

Hardware

There are two types of momentary pushbutton switches. When a momentary-on switch is pressed the circuit is closed; it then opens again when the button is released. Momentary-off switches act in a

very similar way except the circuit is closed until the button is pressed, at which time it opens. The following wiring diagram(Figure 4) and sample code use a momentary-off pushbutton but similar logic can be applied to a momentary-on button.



Figure 3 – Pushbutton Switches

Wiring a pushbutton is very simple. There are only two contact pins. The pushbutton is connected between two desired contact points. In order to toggle a port bit when interfacing with a PIC microcontroller one button pin will be connected to the voltage source and the other will be connected to the PIC input pin as well as the ground. The wiring diagram is shown below.

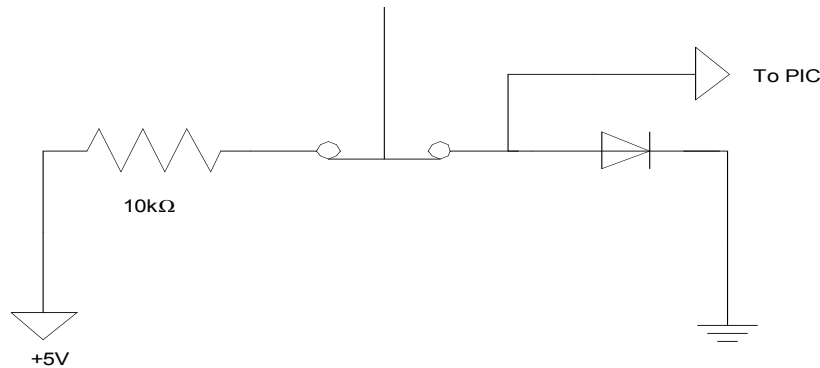


Figure 4 – Wiring diagram using momentary-off pushbutton switch

When the button is pressed a “bounce” (also known as a contact bounce or chatter) occurs. The bounce is a problem that happens due to the materials and construction of the button itself. When the

contacts are made the current rapidly pulses and oscillates like a sinusoid before it finally sticks at the proper logical value (Figure 5). This is a problem because the PIC may receive multiple triggers when the button is only pressed once.

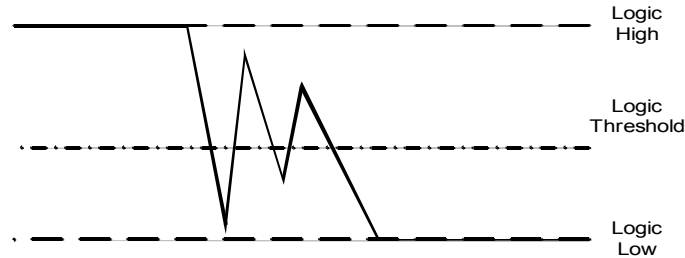


Figure 5 – Button bounce¹

There are two methods for correction, called debouncing. The first approach is to add additional hardware to the circuit. Inserting a capacitor across the switch and a pull-up resistor will be sufficient. When the switch is closed, the capacitor is discharged and has no effect. When the switch is opened the capacitor takes time to recharge so the bounce has minimal effect on the input to the PIC. This approach is easier to implement but costs more due to the additional hardware components.

The other method to account for bounce uses a software approach. After the initial button press a delay is triggered in the code sufficiently long to keep the bounce from being detected. This approach is cheaper but the delay must be determined based on the switch model for optimal operation.

Software

There are multiple ways to implement a switch in the PIC programming. An inline method using a loop is adequate for simple applications. Typically a continuous loop runs until the button is pressed and then the intended action is performed. The logic is very simple as outlined below.

//Loop Code

```
PORTBbits.RB7 = 1; //PIC input pin
while(1)
{
    if(PORTBbits.RB7 == 0){
        //delay set time accounts for bounce
        //perform action here - 1st button press
    }
    if(PORTBbits.RB7 == 0){
        //delay set time accounts for bounce
        //perform action here - 2nd button press
    }
    if(PORTBbits.RB7 == 0){
        //delay set time accounts for bounce
        //perform action here - 3rd button press
    }
    //if another button press, recycle
}
```

Another, probably more efficient, way to program the PIC to handle a button is to use an interrupt. When the button is pressed, the interrupt handles the actions to be performed then jumps back to the main code. The following code was used to change between three modes on a particular device when the button is pressed. This assumes the user accounts for bounce using the hardware method.

//Interrupt Code

```
/* setup the mode switching button interrupt */
ConfigINT1(RISING_EDGE_INT & EXT_INT_ENABLE &
EXT_INT_PRI_3);

/* This interrupt is for the mode selection button */
void __attribute__((__interrupt__)) _INT1Interrupt(void) {
switch (active_mode) {
case PEAK_MODE:
    active_mode = AVERAGE_MODE;
    *active_mode_str = "AVERAGE";
    // display_number(current_peak); //send peak value to
    //Display
    break;
case AVERAGE_MODE:
    active_mode = INTEGRATED_MODE;
    *active_mode_str = "INTEGR.";
    break;
}
```

```

case INTEGRATED_MODE:
    active_mode = RAW_MODE;
    *active_mode_str = "RAW";
    break;
case RAW_MODE:
    active_mode = PEAK_MODE;
    *active_mode_str = "PEAK";
    break;
default:
    active_mode = PEAK_MODE;
    *active_mode_str = "PEAK";
    }
BUTTON_INTERRUPT_ENABLE = 1;
BUTTON_INTERRUPT_FLAG = 0;    //Clear the INTO
interrupt flag or else
                                //the CPU will keep
//vectoring back to the ISR
}

```

Results

After reading this application note the user should be able to properly implement a momentary pushbutton with a PIC microprocessor and account for bouncing.

Conclusion

There are many different types of switches and innumerable applications for them. Simple yet useful, pushbuttons allow a user to interact with and make suitable changes to a circuit while it is running.

Recommendations

This article explains how to use a simple momentary pushbutton switch. There are many other types of switches that can be used depending on the application. It is recommended to research what type of switch will work best for the intended use. Keep the button as simple as possible for the best user interface.

References

1. "Myke Predkos' PICMicro Button Debounce Article." Reynolds Electronics. 1999. 29 Mar. 2007 <<http://www.rentron.com/Myke6.htm>>.
2. "dsPIC30F Family Reference Manual." Microchip Technology, Inc, 2006. 15 Jan. 2007 <<http://ww1.microchip.com/downloads/en/DeviceDoc/70139E.pdf>>.
3. Matic, Nebojsa. PIC microcontrollers, for beginners too. mikro Elektronika, 2003. 29 Mar. 2007 <http://www.mikroe.com/en/books/picbook/0_Uvod.htm>.