```
/*******************************************************************************
Project : led_but.c
Version : v1.1 for AVR-GCC
Date    : 10/30/2003
Author  : Chris Troutner
Company : MyRobot
Chip    : ATMEGA128
Platform: Model-T Prototype
Comments: This program shows how to control the LEDs and switches on the Mini-PC.


Clock frequency     : 16.0000 MHz
Memory model        : Small
Internal SRAM size  : 4096
External SRAM size  : 0
Data Stack size     : 1024
*******************************************************************************/

//Compiler Derectives

    //Includes
    #include <modelt.h>

    //Defines
    /*By defining a string constant as is done here, the string is stored in program memory rather than loaded
    into RAM at run time. The ATMega128 has 128KB of program memory and only 4KB of RAM, so it is much better
    to store strings in this way if possible. The catch is that strings stored in program memory can't be
    manipulated (efficiently) durring run time.*/
    #define SW1MSG "Switch 1"
    #define SW2MSG "Switch 2"
    #define SW3MSG "Switch 3"
    #define SW4MSG "Switch 4"

//Global Variables
    char t[16];

//Sub-Function Prototypes

//Interrupt Handlers

    INTERRUPT(SIG_INTERRUPT0)
    {
        //REMEMBER: SET UP FOR Falling EDGE
        asm volatile ("cli"); //Disable global interrupts while servicing this interrupt.
        ms_spin(20);            //Wait 20ms to let mechanical oscilations to die down.

        strcpy(t, SW3MSG);      //Send a message to the LCD
        line1(t);

        led(1,2000);            //Turn on LED1 for 2 seconds (2000 mS).

        clr_lcd();              //Clear the lcd.

        EIFR |= 0x0F;           //Clear all the interrupt flags. This instruction makes it so that you
```

```
                                //can't trigger an interrupt from another button until the first button's
                                //interrupt routine has been serviced.

        asm volatile ("sei");   //Re-enable global interrupts
    }

    INTERRUPT(SIG_INTERRUPT1)
    {
        //REMEMBER: SET UP FOR Falling EDGE
        asm volatile ("cli"); //Disable global interrupts while servicing this interrupt.
        ms_spin(20);            //Wait 20ms to let mechanical oscilations to die down.

        strcpy(t, SW4MSG);      //Send a message to the LCD
        line1(t);

        led(1,2000);            //Turn on LED1 for 2 seconds (2000 mS).

        clr_lcd();              //Clear the lcd.

        EIFR |= 0x0F;           //Clear all the interrupt flags. This instruction makes it so that you
                                //can't trigger an interrupt from another button until the first button's
                                //interrupt routine has been serviced.

        asm volatile ("sei");   //Re-enable global interrupts
    }

    INTERRUPT(SIG_INTERRUPT2)
    {
        //REMEMBER: SET UP FOR Falling EDGE
        asm volatile ("cli"); //Disable global interrupts while servicing this interrupt.
        ms_spin(20);            //Wait 20ms to let mechanical oscilations to die down.

        strcpy(t, SW2MSG);      //Send a message to the LCD
        line1(t);

        led(1,2000);            //Turn on LED1 for 2 seconds (2000 mS).

        clr_lcd();              //Clear the lcd.

        EIFR |= 0x0F;           //Clear all the interrupt flags. This instruction makes it so that you
                                //can't trigger an interrupt from another button until the first button's
                                //interrupt routine has been serviced.

        asm volatile ("sei");   //Re-enable global interrupts
    }

    INTERRUPT(SIG_INTERRUPT3)
    {
        //REMEMBER: SET UP FOR Falling EDGE
        asm volatile ("cli"); //Disable global interrupts while servicing this interrupt.
        ms_spin(20);            //Wait 20ms to let mechanical oscilations to die down.

        strcpy(t, SW1MSG);      //Send a message to the LCD
        line1(t);
```

```
109
110             led(1,2000);            //Turn on LED1 for 2 seconds (2000 mS).
111
112             clr_lcd();              //Clear the lcd.
113
114             EIFR |= 0x0F;           //Clear all the interrupt flags. This instruction makes it so that you
115                                     //can't trigger an interrupt from another button until the first button's
116                                     //interrupt routine has been serviced.
117
118             asm volatile ("sei");   //Re-enable global interrupts
119         }
120
121     //Main
122     int main(void)
123         {
124         //Local Variables
125
126         //Initialization
127             reset();
128
129             lcd_init();
130
131             //              PERIPHERIAL AND INTERRUPT INITIALIZATION
132             //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
133             //*************************************************************
134             //               External Interrupt 0-3 Initialization
135             //*************************************************************
136             EICRA=0xAA;     //Set INT0-3 to trigger by a falling edge. (pg. 86 of M128 datasheet)
137             EICRB=0x00;     //This controlls INT4-7. (pg. 87 of M128 datasheet)
138             EIMSK=0x0F;     //Enable INT0-3 interrupts (Turn them on). (pg. 88 of M128 datasheet)
139             //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
140
141
142         //Enable global interrupts (Keep this instruction at the end of initialization).
143             asm volatile ("sei");
144
145         //Main Execution Code
146             while (1)
147             {
148
149             };
150         }
151
152
153
154
```