```
//***********************************************************************
//
// Name:         Module 1 - Keypad Project
// Filename:     keypad.c
// Author(s):    Timothy Caro
//               Peter Carcaterra
//
// Modified by: Ken Short (1/13/07)
//               Ben Jarnagin (PORTC substitution, delay resync to 16MHz, tbl)
//
// Description: Configures ports and interrupt. When keypad interrupt occurs,
//               key matirx is scanned and is encoded using a table lookup. The
//               green LED is flashed the number of times equal to the
//               keycode. If the keycode is 0, the red LED is flashed
//               once. The keypad is connected to PORTC while the LEDS are
//               connected to PORTD. See diagram in laboratory description.
//
//***********************************************************************

//Definitions

#define debug   //this can be uncommented to remove delays for simulation

#include <iom128.h>         //Atmega128 definitions
#include <intrinsics.h>     //Intrinsic functions.
#include <avr_macros.h>     //Useful macros.
/*
 * Port pin numbers for columns and rows of the keypad and for LEDs
 */
//PORT Pin Definitions.
#define COL1  7   //pin definitions for PortB
#define COL2  6
#define COL3  5
#define COL4  4
#define ROW1  3
#define ROW2  2
#define ROW3  1
#define ROW4  0

#define INT0  0   //pin definitions for PortD
#define LED1  1
#define LED2  2
/*
 * Function declarations.
 */
void flash_leds(char num);
void check_release(void);

/*
 * Lookup table declaration
 */
const char tbl[16] = {1, 2, 3, 15, 4, 5, 6, 14, 7, 8, 9, 13, 10, 0, 11, 12};
```

```
//*********************************************************************************
//Code

/*
 * Interrupt service routine
 */
#pragma vector=INT0_vect          //Declare Vector location.
__interrupt void ISR_INT0(void)   //Declare Interrupt Function
{
  char keycode;                   //Holds keycode. (UNSIGNED by default)

//Note: TESTBIT returns 0 if bit is not set and a non-zero number otherwise.

  if(!TESTBIT(PINC,ROW1))         //Find Row of pressed key.
    keycode = 0;
  else if(!TESTBIT(PINC,ROW2))
    keycode = 4;
  else if(!TESTBIT(PINC,ROW3))
    keycode = 8;
  else if(!TESTBIT(PINC,ROW4))
    keycode = 12;

  DDRC = 0x0F;                    //Reconfigure PORTC for Columns.
  PORTC = 0xF0;

#ifndef debug
  __delay_cycles(4096);            //Let PORTC settle.
#endif

  if(!TESTBIT(PINC,COL1))         //Find Column.
    keycode += 0;
  else if(!TESTBIT(PINC,COL2))
    keycode += 1;
  else if(!TESTBIT(PINC,COL3))
    keycode += 2;
  else if(!TESTBIT(PINC,COL4))
    keycode += 3;

  DDRC = 0xF0;                    //Reconfigure PORTC for Rows.
  PORTC = 0x0F;

  flash_leds(tbl[keycode]);       // Fix keycode and Flash LEDS.
  check_release();                //Wait for keypad release.
}

/*
 * Main entry point for program.
 */
int main(void)
{
  DDRD = 0xFE;                    //Initialize PORTD.
  PORTD = 0x07;
```

```
  DDRC = 0xF0;                    //Initialize PORTC.
  PORTC = 0x0F;

  MCUCR = 0x30;                   //Config interrupts and sleep mode.
  EIMSK = 0x01;                   //Enable interrupt masks.
  __enable_interrupt();           //Enable global interrupts.

  return 0;                       //Return
}

/*
 *  Flashes LEDS set number of times.
 */
void flash_leds(char num)
{
  if(num == 0)                    //If num is 0 flash LED1 (red LED) once.
  {
    CLEARBIT(PORTD,LED1);
#ifndef debug
    __delay_cycles(4000000);       //Delay (.25secs) / (1 / 16MHz) cycles.
#endif
    SETBIT(PORTD, LED1);
  }
  else                            //IF num is > 0 flash LED2 (green LED)
                                  //that many times.
  {
    for(char i = 0; i < num; i++)
    {
      CLEARBIT(PORTD,LED2);
#ifndef debug
      __delay_cycles(4000000);     //Delay (.25secs) / (1 / 16MHz) cycles.
#endif
      SETBIT(PORTD, LED2);
#ifndef debug
      __delay_cycles(4000000);     //Delay (.25secs) / (1 / 16MHz) cycles.
#endif
    }
  }
  return;
}

/*
 *  Check keypad is released and not bouncing.
 */
void check_release(void)
{
#ifndef debug
  while(!TESTBIT(PIND,INT0));     //Check that keypad key is released.

  __delay_cycles(800000);          //Delay (.05secs) / (1 / 16MHz) cycles.

  while(!TESTBIT(PIND,INT0));     //Check that key has stopped bouncing.
#endif
```

```
  return;
}
```