```
/*****************************************************************
 *                                                               *
 *  Project: Channel Selector                                    *
 *   Source: Channel_Selector_v1.c                               *
 *   Author: Mike McLaren, K8LH                                  *
 *     Date: 16-Feb-10                                           *
 *  Revised: 16-Feb-10                                           *
 *                                                               *
 *  OEM 16F886 2-Digit 01-99 Channel Selector                    *
 *                                                               *
 *                                                               *
 *      IDE: MPLAB 8.14 (tabs = 4)                               *
 *     Lang: SourceBoost BoostC v6.95, Lite/Free version         *
 *                                                               *
 *                                                               *
 *****************************************************************/

#include <system.h>

#pragma DATA _CONFIG1, _LVP_OFF & _MCLRE_OFF & _WDT_OFF & _INTOSCIO
#pragma DATA _CONFIG2, _BOR40V

//--< function prototypes >-----------------------------------------
//--< typedef and defines >-----------------------------------------

typedef unsigned char u08;
typedef unsigned int u16;

#define r08 const rom unsigned char

#define upswitch flags.7        // debounced "up" switch flag
#define dnswitch flags.6        // debounced "dn" switch flag
#define polarity flags.0        // eprom "polarity" jumper

//--< variables >---------------------------------------------------

u08 swnew = 0;                  // switch sample var'
u08 swold = 0;                  // switch state latch
u08 flags = 0;                  // switch press flags

u08 channel = 1;                // packed BCD channel, 0x01..0x99

r08 segdata[] = { 0b00111111,   // "0"   -|-|F|E|D|C|B|A
                  0b00000110,   // "1"   -|-|-|-|-|C|B|-
                  0b01011011,   // "2"   -|G|-|E|D|-|B|A
                  0b01001111,   // "3"   -|G|-|-|D|C|B|A
                  0b01100110,   // "4"   -|G|F|-|-|C|B|-
                  0b01101101,   // "5"   -|G|F|-|D|C|-|A
                  0b01111101,   // "6"   -|G|F|E|D|C|-|A
                  0b00000111,   // "7"   -|-|-|-|-|C|B|A
                  0b01111111,   // "8"   -|G|F|E|D|C|B|A
                  0b01101111 }; // "9"   -|G|F|-|D|C|B|A


//--< main >--------------------------------------------------------

void main()
{
  ansel = 0;                    // ANS0..ANS7 ADC inputs off
  anselh = 0;                   // ANS8..ANS13 ADC inputs off
  osccon = 0b01110000;          // set INTOSC to 8 MHz
  while(!osccon.HTS);           // wait 'til oscillator stable
  trisc = 0b00000000;           // set all PORTC pins to outputs
  trisb = 0b00000000;           // set all PORTB pins to outputs
  trisa = 0b11000001;           // RA7, RA6, and RA1 = inputs
  porta = 0b00000000;           // set all output latches to '0'
  portb = 0b00000000;           // set all output latches to '0'
  porta = 0b00000000;           // set all output latches to '0'

//  setup Timer 2 for 8 msec interrupts (8 MHz clock)

  tmr2 = 0;                     // clear Timer 2 register
  t2con = 0b00011110;           // '0-------' unimplemented bit
                                // '-0011---' TOUTPS<3:0>, postscale 4
                                // '-----1--' TMR2ON, turn Timer 2 on
                                // '------10' T2CKPS<1:0>, prescale 16
  pr2 = 250-1;                  // 250 x 32-usec 'ticks' = 8 msecs
  pir1 = 0;                     // clear peripheral interrupt flags
  pie1.TMR2IE = 1;              // set Timer 2 interrupt enable bit
  intcon = 0b11000000;          // '1-------' GIE, enable global ints
                                // '-1------' PEIE, enable peripheral ints
                                // '--0-----' TOIE, TMR0 ints disabled
                                // '---0----' INTE, off
                                // '----0---' GPIE, IOC disabled
```

```
                                // '-----OOO' TOIF/INTF/GPIF flags

//  main program loop

  while(1)
  { if(upswitch)                // if "up" switch pulse
    { upswitch = 0;             // clear the switch flag and
      if(channel < 0x99)        //   if less than upper limit 0x99
      { asm                     //     do packed BCD increment
        { movf  _channel,W      ;
          addlw 7               ; bcd increment + bcd adjust
          btfss _status,DC      ; adjust required, yes, skip, else
          addlw 0xFA            ; undo adjust (add -6)
          movwf _channel        ; update "channel" var'
        }                       //
      }                         //
    }
    if(dnswitch)                // if "dn" switch pulse
    { dnswitch = 0;             //   clear the switch flag and
      if(channel > 0x01)        //   if greater than lower limit 0x01
      { asm                     //     do packed BCD decrement
        { movf  _channel,W      ;
          addlw 0xFF            ; bcd decrement
          btfss _status,DC      ; bcd adjust required? no, skip, else
          addlw 0xFA            ; do bcd adjust (add -6)
          movwf _channel        ; update "channel" var'
        }                       //
      }                         //
    }
    if(polarity)                // if polarity jumper off (hi)
      portc = channel;          //   normal eprom address data
    else                        // if polarity jumper on (lo)
      portc = ~channel;         //   invert eprom address data
  }
}

/*************************************************************************
 *  ISR - refresh display and switch flags (62.5 Hz refresh rate)   *
 *************************************************************************/

void interrupt()                // 8-msec timer 2 interrupts
{ u08 index;                    // isr work variable
  pir1.TMR2IF = 0;              // clear timer 2 interrupt flag
  portb &= 0x80;                // blank the display
  portb ^= 0x80;                // toggle digit select pin (RB7)
  if(portb.7)                   // if 'tens' digit selected
    index = channel >> 4;       //   use upper nybble as index
  else                          // if 'ones' digit selected
    index = channel & 15;       //   use lower nybble as index
  portb |= segdata[index];      // display new digit

  swnew = ~porta;               // sample active low switches
  swnew &= 0b11000001;          // on RA7, RA6, and RA0 pins
  swnew ^= swold;               // changes, hi or lo
  swold ^= swnew;               // update switch state latch
  swnew &= swold;               // filter out "new lo" bits
  flags |= swnew;               // save "new hi" flag bits
}



/*
Note: The inline assembler statements can be replaced with C for readability
e.g. on line 92, replace (or just note the equivalence) */
        asm
        { movf  _channel,W      ;
          addlw 7               ; bcd increment + bcd adjust
          btfss _status,DC      ; adjust required, yes, skip, else
          addlw 0xFA            ; undo adjust (add -6)
          movwf _channel        ; update "channel" var'
        }                       //

//will have the same function as:
        if(channel & 0x0F == 9) // lower digit about to overflow
            channel += 7;       // adjust and increment
        else
            channel++;          // just increment

```