```
/* ************************************************************************
 * Main Program For Receiver build into Clarion Front Panel               *
 *                                                                        *
 * Clarion/VDO Infra Red Receiver                                         *
 * This project involves a PIC and an infra red receiver to be built into *
 * into the removable front panel of the radio. The receiver will be com- *
 * patible with Clarion Auto Hifi Remotes etc. the RCB-130.               *
 * Only six buttons will be implemented. These are the functions from the *
 * Toyota Steering Wheel Remote Controller. (Toyota Avensis)              *
 *                                                                        *
 * ---------------------------------------------------------------------- *
 * Copyright (C) 2004 KOCH Engineering                                    *
 * Henrik J. Koch                                                         *
 * email:   mail@koch-engineering.com                                     *
 * web:     http://www.koch-engineering.com                               *
 * (MPLAB-IDE 6.43)                                                       *
 * Version 1.0  Marts 14. 2004                                            *
 ************************************************************************ */
#include <pic.h>    // __CONFIG
#include <stdio.h>  // printf, sprintf
#include <conio.h>
#include <string.h> // strcpy,
#include <stdlib.h> // ui8toa,

#include "receiver.h"   // BYTE
#include "delay.h"      // DelayUs, DelayMs
#include "lcd.h"        // 2x16 line LCD display
#ifdef DEBUG
#include "sci.h"        // serial communication
#endif
#include "infrared.h"       //
#include "hal.h"

__CONFIG(XT & WDTDIS & PWRTDIS & BORDIS & LVPDIS & DUNPROT & WRTEN & UNPROTECT);

#define _XTAL_FREQ = 4MHZ

//#define DEBUG

    bank2 BYTE      ism_state;
    bank2 BYTE      osm_state;

    bit             NewCodeReceived;    // status indicator for when all 32 bits a
    bit             KeyPressed;         // status indicator for key pressed
#ifdef DEBUG
    bit             DebugPrint;
#endif

    bank1 char      line1[16];
    bank1 char      line2[16];
    bank1 int       led_on_cnt;         // counter for output pin

    bank2 WORD      code_common;        //
    bank2 WORD      code_separate;

#ifdef DEBUG
    bank2 WORD      capture_value[15];
#endif
    bank2 BYTE      printout_count;     // used for debug print out
    bank2 WORD      output_timer_value;
    bank1 BYTE      ReadKeyCopy;

// button text table
bank2 char ir_function_text[6] [9] = {
    "FUNC    ",
    "UP      ",
    "BACKWARD",
    "FORWARD ",
    "DOWN    ",
    "MUTE    "};

void main ()
{
```

```
        RBIE =
75      // Timer 1 set-up. Used with pre-scale 4 and timer value 250
        // which then wraps around every 1 ms.
        T1CKPS1 = 0;    // prescale 0   (Fosc/4)
        T1CKPS0 = 0;    // -//-
        TMR1CS = 0;     // Timer 1 clock source internal clock (Fosc/4)
80      T1SYNC = 0;     // Synchronize external clock input

        T1OSCEN = 0;    // Timer 1 oscillator inverter disabled to save power
        TMR1L = 0;      // Clear Timer 1 counter Low part
        TMR1H = 0;      // Clear Timer 1 counter High part
85      TMR1IE = 0;     // Disable Timer 1 interupt
        TMR1ON = 1;     // Timer 1 started

        // CCP1 Capture/Compare/PWM module 1
        set_CCP1_mode(CAPTURE_FALLING_EDGE);
90      peripheral_interrupt_enable();
        peripheral_interrupt_setup(CAPTURE_COMPARE_PWM1_ENABLE);

        // CCP2 Capture/Compare/PWM Module 2
        set_CCP2_mode(CCP_OFF);
95
        lcd_init();
        lcd_clear();

    // ############################################################################
100
    #ifdef DEBUG
        DebugPrint = FALSE;
    #endif

105     NewCodeReceived = FALSE;

        strcpy(line1, "KOCH Engineering");
        strcpy(line2, "IR Receiver     ");
        lcd_puts(line1);
110     lcd_gotoXY(0,1);    // goto first position on line 2
        lcd_puts(line2);

        DelayMs(255);

115     strcpy(line2, "Ready to Capture");
        lcd_gotoXY(0,1);
        lcd_puts(line2);

    #ifdef DEBUG
120     sci_Init(19200,SCI_EIGHT);
        printf("\n\nKOCH Engineering (C) 2004\n\r");    // print to RS-232
        printf("Clarion Infra Red Interface\n\r");
    #endif

125     KeyPressed = FALSE;

        ir_init();

    #ifdef DEBUG
130     printf("\n\rReady to Capture\n\r");
    #endif

        ei();   // enable global interrupt

135     while(1)
        {


    #ifdef DEBUG
140         if (DebugPrint)
            {
                di();
                printf("\r\n----------------");

145             for (printout_count = 0; printout_count <= 14; printout_count++)
                {
```

```
                    printf("\n\rcapture_value[%d] : %d", printout_count,capture_value[
170             }
                DebugPrint = FALSE;
                ei();
            }
    #endif
175
            if (NewCodeReceived)
            {
                NewCodeReceived = FALSE;
                ReadKeyCopy = ReadKey();
180 #ifdef DEBUG
                printf("\n\r-----------------------------------------------");
                printf("\n\rCommonPart   = %x", code_common);
                printf("\n\rSeparatePart = %x", code_separate);
                printf("\n\rKeyNumber    = %d", ReadKeyCopy);
185             printf("\n\rButton       = ");
    #endif
                if (ReadKeyCopy <= 5)
                {
    #ifdef DEBUG
190                 printf(ir_function_text[ReadKeyCopy]);
    #endif
                    // clear 2.nd line
                    strcpy(line2, "                ");
                    lcd_gotoXY(0,1);
195                 lcd_puts(line2);

                    strcpy(line2, ir_function_text[ReadKeyCopy]);
                    lcd_gotoXY(0,1);
                    lcd_puts(line2);
200             }
                else
                {
    #ifdef DEBUG
                    printf("button not supported");
205 #endif
                    strcpy(line2, "key no support! ");
                    lcd_gotoXY(0,1);
                    lcd_puts(line2);
                }
210         } // if NewCodeReceived


            if (KeyPressed)
            {
215             set_output_pin(ReadKey());
                output_timer_value = OUTPUT_TIME;
                KeyPressed = FALSE;
            }

220
            if ( output_timer_value == 0)
            {
                set_output_pin(KEY_ALL_CLEAR);
                strcpy(line2, "                ");
225             lcd_gotoXY(0,1);
                lcd_puts(line2);
                LED_0 = 0;
                LED_1 = 1;
                IR_RECEIVE_BLINK = 0;   // turn off blink LED
230         }

    #ifdef DEBUG
            if (RC0 == 1)       // if button pressed
            {
235             DelayMs(10);    // check for debounce
                if (RC0 == 1)
                {
                    printf("\n\rosm_state : %d", osm_state);
                    printf("\n\rism_state : %d", ism_state);
240                 printf("\n");
                    while (RC0 == 1) // wait for release of button
```

```
                    ;
                }
            }
245 #endif

        } // while
    } // main

250 // ************************************************************
    static void interrupt
    isr(void)
    {
        if (T0IF)
255     {
            TMR0 = 256-250+13;      // gives 250 tick until next wrap around (13 is a
            T0IF = FALSE;           // clear Timer0 interrupt flag

            if (output_timer_value > 0)
260         {
                output_timer_value--;
            }

        }
265
        if (CCP1IF) // Capture Interrrupt
        {
            ir_receiver_ism();  // this state maschine clears the interrupt flag
            // interrupt flag is cleared in above function
270     }

        INTF = FALSE;     // clears interrupt flag
    }   // isr

275
```