#include <LPC17xx.h>

#include <stdlib.h>

#include <stdio.h>

#define RS\_CTRL 0x08000000 // P0.27, 1<<27

#define EN\_CTRL 0x10000000 // P0.28, 1<<28

#define DT\_CTRL 0x07800000 // P0.23 to P0.26 data lines, F<<23

unsigned long int temp1 = 0, temp2 = 0, i, j, r, x;

unsigned char flag1 = 0, flag2 = 0, k;

char msg1[16];

char msg2[16];

int count = 0, threshold = 15, empty = 1, full = 0;

unsigned long int init\_command[] = {0x30, 0x30, 0x30, 0x20, 0x28, 0x01, 0x06, 0x0c,0x80};

void EINT0\_IRQHandler(void); // Interrupt handler for Entry (P2.10)

void EINT1\_IRQHandler(void); // Interrupt handler for Exit (P2.11)

void lcd\_write(void);

void port\_write(void);

void delay\_lcd(unsigned int);

void lcd\_write(void){

      temp2 = temp1 & 0xf0; // Extract the 4 significant bits to get least significant digit place

      temp2 = temp2 >> 4;

      port\_write(); // Send least significant 4 bits only when it is data other than 0x30/0x20

      if (!((flag1 == 0) && ((temp1 == 0x20) || (temp1 == 0x30)))) {

            temp2 = temp1 & 0x0f;

            temp2 = temp2;

            port\_write();}

}

void port\_write(void){

      LPC\_GPIO0->FIOPIN = temp2 << 23; // Send the ASCII code

      if (flag1 == 0)

            LPC\_GPIO0->FIOCLR = RS\_CTRL; // Command mode

      else

            LPC\_GPIO0->FIOSET = RS\_CTRL; // Data mode

      LPC\_GPIO0->FIOSET = EN\_CTRL; // Send a low-to-high edge on the enable input

      for (r = 0; r < 25; r++);

      LPC\_GPIO0->FIOCLR = EN\_CTRL;

      for (r = 0; r < 30000; r++);

}

void display(){

      flag1 = 0; // Initialization commands for the LCD

      for (i = 0; i < 9; i++){

            temp1 = init\_command[i];

            lcd\_write();

      }

      flag1 = 1; // Data mode

      for (i = 0; msg1[i] != '\0'; i++){ // Display the first message

            temp1 = msg1[i];

            lcd\_write();

      }

      if (!full) { // If not full, display the car count

            flag1 = 0;

            temp1 = 0xC0; // Move the cursor to the second line

            lcd\_write();

            flag1 = 1;

            sprintf(msg2, "%d", count);

            for (i = 0; msg2[i] != '\0'; i++) {

                  temp1 = msg2[i];

                  lcd\_write();

            }

      }

}

int main(void){

      SystemInit();

      SystemCoreClockUpdate();

      sprintf(msg1, "Car count is:");

      LPC\_PINCON->PINSEL1 = 0; // Configure pin functions

      LPC\_PINCON->PINSEL4 |= (1 << 20 | 1 << 22); // Configure pins for EINT0 and EINT1

      // Configure pins for LCD control and data lines

      LPC\_GPIO0->FIODIR = DT\_CTRL | RS\_CTRL | EN\_CTRL | (0xFF << 4);

      LPC\_GPIO1->FIODIR = 0; // LCD

      LPC\_GPIO2->FIODIR = 0;

      display();

      // Configure external interrupts EINT0 and EINT1

      LPC\_SC->EXTMODE = 1 << 0 | 1 << 1; // EINT0 and EINT1 are initiated as edgesensitive

      LPC\_SC->EXTPOLAR = 0; // EINT0 and EINT1 are falling edge-sensitive

      NVIC\_EnableIRQ(EINT0\_IRQn); // Enable interrupt for Entry

      NVIC\_EnableIRQ(EINT1\_IRQn); // Enable interrupt for Exit

      while (1);

}

void EINT0\_IRQHandler(void){

      int i=0;

      LPC\_SC->EXTINT = 1 << 0; // Clear the interrupt

      for (i = 0; i < 10000; i++);

       if (LPC\_GPIO2->FIOPIN & (1 << 10)) { // Adjust according to your pin configuration

        return; // If the pin is not low, ignore the interrupt

    }

      // Increment count if not full

      if (!full){

            count=(count+1);}

      // Update flags

      if (count > 0 && empty){

            empty = 0;}

      // Display 'Full' if count exceeds the threshold

      if (count >= threshold){

            sprintf(msg1, "Full");

            full = 1;}

      // Update the LCD display

      display();

}

void EINT1\_IRQHandler(void){

      int i=0;

      LPC\_SC->EXTINT = 1 << 1; // Clear the interrupt

      for (i = 0; i < 10000; i++);

      if (LPC\_GPIO2->FIOPIN & (1 << 11)) { // Adjust according to your pin configuration

        return;} // If the pin is not low, ignore the interrupt

      // Decrement count if not empty

      if (!empty){

            count=count-1;}

      // Update flags

      if (count < threshold && full){

            sprintf(msg1, "Car count is:");

            full = 0;}

      // Update the LCD display

      display();}