**/\*------------------8051-LED BANK---------------------\*/**

#include<reg51.h> //header file included

void delay(unsigned int time); //delay function declaration

void main()

{

unsigned int i;

while(1) //infinite loop for continuous saw tooth waveform generation

{

P2 = 0x55;

delay(100);//delay function call

P2 = 0xAA;

delay(100);//delay function call

}

}

void delay(unsigned int time) //for delay generation

{

unsigned int i,j;

for(i=0;i<time;i++)

for(j=0;j<1275;j++);

}

**/\*------------------8051-SEVEN SEGMENT---------------------\*/**

#include<reg51.h>

void delay(long int x)

{

int i,j;

for(i=0; i<x; i++)

{

}

}

unsigned char arr[10] = {0xF6,0x90,0xE5,0xB5,0x93,0x37,0x77,0x94,0xF7,0xB7};

void main()

{

while(1)

{

int i;

for(i=0; i<=9; i++)

{

P0 = ~arr[i];

delay(5000);

}

}

}

**/\*------------------8051-STEPPER MOTOR---------------------\*/**

#include<reg51.h>

unsigned int STEP[] = {8,4,2,1};

void delayms(unsigned long x)

{

unsigned int i;

for(i=0;i<=x;i++);

}

void main(void)

{ unsigned char k;

while(1)

{ for(k=0; k<3;k++)

{

P1= ~STEP[k];

delayms(5);

}

}

}

**/\*------------------8051-LCD---------------------\*/**

/\*

Interfacing Pins :

P0.0 thru P0.7 interfaced to D0 thru D7.

P3.2 ----> LCD\_RS (Register Select).

P3.3 ----> LCD\_EN (Enable).

-----------------------------------------------------------------------------

Jumper Settings:

J1:2-3 J2:1-2 J3:2-3 J4:1-2 J5:1-2 J6:1-2

=============================================================================

\*/

#include<REG51.h>

//#define Enable 0x08

//#define RS 0x04

sbit en = P3^3;

sbit rs = P3^2;

void DelayMs(delay)

{

int i,j;

for(i=0;i<delay;i++)

{

for(j=0;j<100;j++);

}

}

void write\_lcd\_data(value)

{

/\* data for lcd \*/

P0 = value;

rs = 1;

en = 1;

DelayMs(10);

en = 0;

}

void write\_lcd\_command(value)

{

/\* command for lcd \*/

P0 = value;

rs = 0;

en = 1;

DelayMs(10);

en = 0;

}

void main(void)

{

P0 = 0x00;

P3 = 0x00;

while(1)

{

write\_lcd\_command(0x38); //function set

DelayMs(100);

write\_lcd\_command(0x0E); //display off

DelayMs(100);

write\_lcd\_command(0x01); //display clear

DelayMs(100);

write\_lcd\_command(0x06); //entry mode set

DelayMs(100);

write\_lcd\_command(0x0F); //display on

DelayMs(100);

write\_lcd\_command(0x80); //set address counter value

DelayMs(100);

write\_lcd\_data('M');

DelayMs(100);

write\_lcd\_data('I');

DelayMs(100);

write\_lcd\_data('C');

DelayMs(100);

write\_lcd\_data('R');

DelayMs(100);

write\_lcd\_data('O');

DelayMs(100);

write\_lcd\_data('C');

DelayMs(100);

write\_lcd\_data('O');

DelayMs(100);

write\_lcd\_data('N');

DelayMs(100);

write\_lcd\_data('T');

DelayMs(100);

write\_lcd\_data('R');

DelayMs(100);

write\_lcd\_data('O');

DelayMs(100);

write\_lcd\_data('L');

DelayMs(100);

write\_lcd\_data('L');

DelayMs(100);

write\_lcd\_data('E');

DelayMs(100);

write\_lcd\_data('R');

DelayMs(100);

}

}

**/\*------------------MSP-LED---------------------\*/**

/\* Main.c file generated by New Project wizard

\*

\* Created: Thu Sep 28 2023

\* Processor: MSP430G2553

\* Compiler: GCC for MSP430

\*/

#include <MSP430.h>

int main(void)

{

WDTCTL = WDTPW | WDTHOLD; // Stop the Watchdog Timer

P1DIR |= BIT0; // Set P1.0 as an output

P1OUT &= ~BIT0; // Turn off the LED initially

int i;

while(1)

{

P1OUT ^= BIT0; // Toggle the LED (on/off)

for (i=0;i<1000;i++); // Delay for a while (adjust this value for desired LED blink rate)

}

}

**/\*------------------MSP-LED-BUTTON---------------------\*/**

#include <msp430.h>

#define LED1\_PIN BIT0 // P3.0 is the LED1 pin

#define LED2\_PIN BIT1 // P3.1 is the LED2 pin

#define BUTTON\_PIN BIT0 // P1.3 is the button pin

void main(void)

{

WDTCTL = WDTPW | WDTHOLD; // Stop the watchdog timer

P3DIR |= LED1\_PIN + LED2\_PIN ; // Set LED1 pin as an output

P3OUT &= ~LED1\_PIN + LED2\_PIN; // Initialize LED1 pin low and LED2 pin High

P1DIR &= ~BUTTON\_PIN; // Set button pin as an input

//P1REN |= BUTTON\_PIN; // Enable internal resistor for button

//P1OUT |= BUTTON\_PIN; // Set the resistor as a pull-up resistor

while (1)

{

if ((P1IN & BUTTON\_PIN)==0x01) // Check if the button is pressed

{

P3OUT ^= LED1\_PIN; // Toggle the LED

\_\_delay\_cycles(100); // Delay for button debounce

}

else

{

P3OUT ^= LED2\_PIN; // Toggle the LED

\_\_delay\_cycles(100); // Delay for button debounce

}

}

}

**/\*------------------MSP-DC MOTOR---------------------\*/**

#include <msp430.h>

#define IN1\_PIN BIT0 // GPIO pin for IN1

#define IN2\_PIN BIT1 // GPIO pin for IN2

#define EN1\_PIN BIT2 // GPIO pin for EN1 (PWM)

void main(void)

{

WDTCTL = WDTPW | WDTHOLD; // Stop the watchdog timer

P1DIR |= IN1\_PIN | IN2\_PIN | EN1\_PIN; // Set pins as outputs

P1OUT &= ~(IN1\_PIN | IN2\_PIN); // Initialize IN1 and IN2 low

//P1SEL |= EN1\_PIN; // Enable PWM on EN1 pin

P1OUT |= EN1\_PIN; // Initialize EN1 high

int i;

while (1)

{

// Forward rotation

P1OUT |= IN1\_PIN;

P1OUT &= ~IN2\_PIN;

// Control motor speed by changing TACCR1 value

for(i=0;i<10000;i++); // Delay for a while

// Reverse rotation

P1OUT &= ~IN1\_PIN;

P1OUT |= IN2\_PIN;

// Control motor speed by changing TACCR1 value

for(i=0;i<10000;i++); // Delay for a while

}

}

**/\*------------------MSP-IR SENSOR---------------------\*/**

/\* Main.c file generated by New Project wizard

\*

\* Created: Thu Sep 28 2023

\* Processor: MSP430G2553

\* Compiler: GCC for MSP430

\*/

#include <msp430.h>

#define IR\_SENSOR\_PIN BIT0 // Input from IR sensor (adjust as needed)

#define BUZZER\_PIN BIT1 // Output to control buzzer (adjust as needed)

void main(void)

{

WDTCTL = WDTPW | WDTHOLD; // Stop the Watchdog Timer

P1DIR &= ~IR\_SENSOR\_PIN; // Set IR sensor pin as input

P1REN |= IR\_SENSOR\_PIN; // Enable pull-up resistor on IR sensor pin

P1OUT |= IR\_SENSOR\_PIN; // Set pull-up resistor

P1DIR |= BUZZER\_PIN; // Set buzzer pin as output

P1OUT &= ~BUZZER\_PIN; // Initially turn off the buzzer

int i;

while (1)

{

if ((P1IN & IR\_SENSOR\_PIN)) // Check if IR sensor output is High (object detected)

{

P1OUT |= BUZZER\_PIN; // Turn on the buzzer

for(i=0;i<100;i++); // Delay to avoid rapid toggling (adjust as needed)

}

else

{

P1OUT &= ~BUZZER\_PIN; // Turn off the buzzer

}

}

}

**/\*------------------MSP-LED-PWM---------------------\*/**

#include <msp430.h>

#define GREEN BIT6 // Green LED -> P1.6

/\*\*

\* @brief

\* These settings are w.r.t enabling TIMER0 on Lunchbox

\*\*/

void register\_settings\_for\_TIMER0()

{

P1DIR |= GREEN; // Green LED -> Output

P1SEL |= GREEN; // Green LED -> Select Timer Output

CCR0 = 255; // Set Timer0 PWM Period

CCTL1 = OUTMOD\_7; // Set TA0.1 Waveform Mode - Clear on Compare, Set on Overflow

CCR1 = 0; // Set TA0.1 PWM duty cycle

CCTL0 = CCIE; // CCR0 Enable Interrupt

TACTL = TASSEL\_2 + MC\_1; // Timer Clock -> SMCLK, Mode -> Up Count

}

/\*\*

\* @brief

\* Entry point for the code

\*\*/

void main(void) {

WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer

register\_settings\_for\_TIMER0();

\_\_bis\_SR\_register(GIE); // Enable CPU Interrupt

while(1)

{

}

}

/\*\*

\* @brief

\* Entry point for TIMER0\_interrupt vector

\*\*/

#pragma vector = TIMER0\_A0\_VECTOR

\_\_interrupt void Timer\_A(void)

{

CCR1 = CCR1 + 1; // Increment CCR1

if(CCR1 == 256)

{

CCR1 = 0;

}

}