Lab 12: Microcontroller-based System for Real-life Application

Name: Shahmir Mustafa Chaudhry ID: 08400 Section: T1

Objective

To design and implement a Microcontroller-based System for real-life application using TM4C123 TivaC.

In-Lab

Task 0: Devise Real-life Application.
Task 1: Selection of Sensing Modules.
Task 2: Selection of Input and Output Modules.
Task 3: Controlling and Communicating your System with Mobile Phone.
Task 4: [BONUS] Incorporating Interrupts to Optimize System.
Task 5: Detailed Flow Diagram of Functional System.

TivaC LaunchPad with Energia

In Energia IDE, unlike Kiel uVision, we can use TivaC LaunchPad pins for various peripherals without the need to activate Ports using registers or specifying function of the pin. But it also comes with limitations of usability and programmable scope of the board. In Energia, we can refer to Pins of TivaC directly using numeric digit like 1,2,3.. and so on. Pin map for the EK-TM4C123GXL LaunchPad is given in Fig. 1 with Black Columns under J1, J2, J3 and J4.

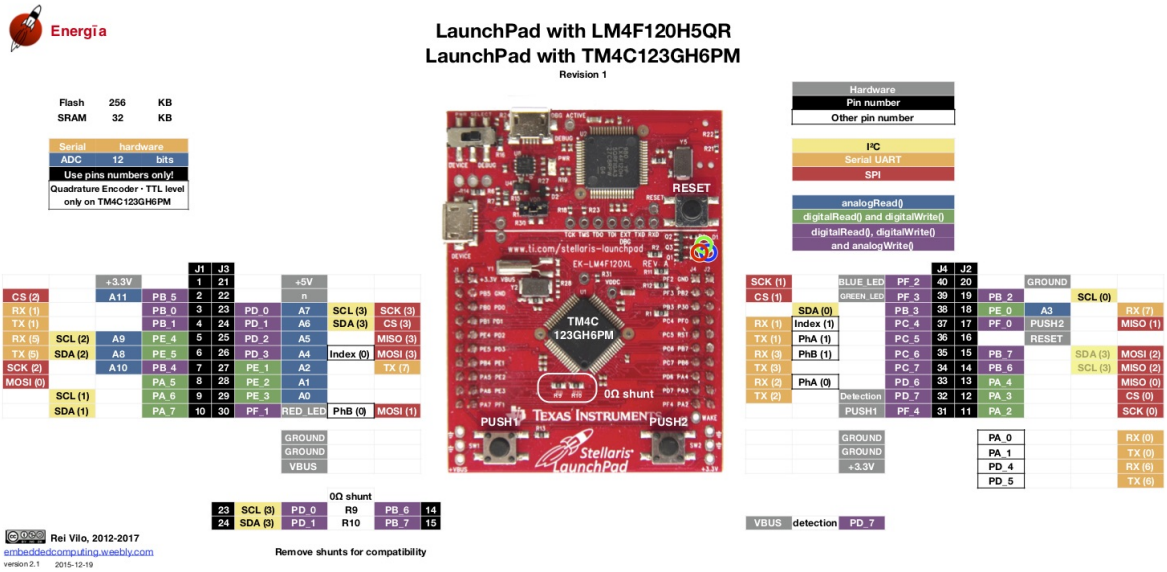


Figure 1: TivaC Pin Distribution with Energia

In-Lab Tasks

Task 0: Devise Real-life Application

In this task, you need to come up with a real-life application for which you would want to design a microcontroller-based controlling system mechanism. Following are some of the potential themes for your application:

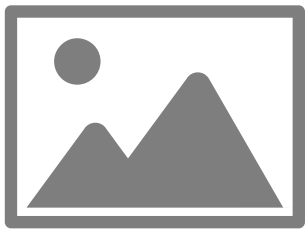
- PID based Motor Control System
- Automatic Washing Machine Motor Control System
- Bidirectional Visitor Counter
- Digital Alarm Clock
- Digital Energy Meter LCD Display
- Digital Frequency Meter with LCD Display
- Digital Thermostat with LCD Temperature Display
- Humidity and Temperature Monitoring System
- Auto Intensity Control of Street Lights
- Password Based Door Lock System
- Ultrasonic Rangefinder Alarm System
- Temperature Controlled DC Fan
- Sun Tracking Solar Panel
- Keyless Door Lock System with Servo Control
- Water Flowrate Measurement System
- Home Appliances Automation System

You can either select one of the above theme or come up with your own application that can be achieved with the microcontroller. You have to ensure the selected application must have certain sensing elements, input modules and output modules as well in addition with some processing involved with data collection and representation.

Write name and briefly discuss the selected application system and justification for choosing that application.

PID Control

Provide abstract block diagram for your proposed system.



Task 1: Selection of Sensing Modules

For this task, you need to identify and select the relevant sensing modules for your application. Following are some of the examples of sensing modules that you can use in accordance with selected application:

- Light Sensor: LDR, IR, Color Sensor
- Sound Sensor: Microphone, Ultrasonic, Motion PIR
- Climate Sensor: Thermocouple, Thermistor, Humidity, Rain
- Position Sensor: Potentiometer, Encoder, Gyroscope, Accelerator
- Magnetic Sensor: Magnetometer

For each of the selected input sensing modules, you have to study, analyze and discuss the following next along with building circuit and coding:

- Number and names of selected sensor and the purpose they serve for your application
- Power requirements, operating voltage and currents from datasheet
- Digital or analog range of the sensing element from datasheet
- Number of pins used by the sensor and their functionality from datasheet
- Required communication protocol and baud rates selection
- Logical working of sensing module i.e., how does it trigger the microcontroller and output device?
- Integrate the modules with TivaC and write a functional code for all the selected components using Energia or Keil.

```
#include "TM4C123GH6PM.h"
#include <stdio.h>
#include "UART.h"

/* Function prototypes */
void Timer0ACapture_init(void);  // Sensor 1 (Timer0A)
void Timer0BCapture_init(void);  // Sensor 2 (Timer0B)
void Sensor2_Trigger_Init(void); // Initialize PA7 for Sensor 2 trigger
void Sensor1_Trigger_Init(void); // Initialize PA4 for Sensor 1 trigger
void Delay_MicroSecond(int time);
void UART5_init(void);
void UART5_Transmitter(unsigned char data);
void printstring(char *str);
void Trigger_Sensors(void);

/* Global variables */
volatile uint32_t distance1; /* Distance from Sensor 1 */
volatile uint32_t distance2; /* Distance from Sensor 2 */
static volatile uint32_t distances[2]; // Distance array to store measurements
char mesg1[50];          /* String format of distance1 */
char mesg2[50];          /* String format of distance2 */

/* Function to compute and return distances */
volatile uint32_t* get_distance(void)
```

```
#include <stdio.h>
#include "UART.h"
#include "infrared.h"

// Initialize PF4 for IR sensor input
void IR_Sensor_Init(void) {
    SYSCTL->RCGCGPIO |= 0x20;  // Enable clock for PORTF (bit 5)
    GPIOA->DIR &= ~IR_SENSOR_PIN;      // Set PF4 as input
    GPIOA->DEN |= IR_SENSOR_PIN;       // Enable PF4 as a digital pin
}

// Read PA2 to check for obstacle presence
int IR_Detect(void) {
    return (GPIOA->DATA & IR_SENSOR_PIN) ? 1 : 0;
}



#include "UART.h"
#include "TM4C123GH6PM.h"

/* UART initialization */
void UART5_init(void)
{
    SYSCTL->RCGCUART |= 0x20; /* Enable clock to UART5 */
    SYSCTL->RCGCGPIO |= 0x10; /* Enable clock to PORTE for PE4/Rx and PE5/Tx */

    UART5->CTL = 0;         /* Disable UART5 */
    UART5->IBRD = 104;      /* 9600 baud rate integer part */
    UART5->FBRD = 11;       /* 9600 baud rate fractional part */
    UART5->LCRH = 0x60;     /* 8-bit, no parity, 1-stop bit */
    UART5->CTL = 0x301;     /* Enable UART5, Tx, and Rx */

    GPIOE->DEN = 0x30;      /* Enable digital PE4, PE5 */
    GPIOE->AFSEL |= 0x30;   /* Alternate function for PE4, PE5 */
    GPIOE->PCTL |= 0x00110000; /* Configure PE4, PE5 for UART */
}

void UART5_Transmitter(unsigned char data)
{
    while ((UART5->FR & (1 << 5)) != 0)
        ; /* Wait until Tx buffer is not full */
    UART5->DR = data;        /* Transmit data */
}

void printstring(char *str)
{
    while (*str)
    {
        UART5_Transmitter(*(str++));
    }
}
```

Task 2: Selection of Input and Output Modules

For this task, you need to identify and select the relevant input/output modules and actuators. Following are some of the examples of output modules and actuators that you can use in accordance with selected application:

- Input: 4x3 Keypad, Buttons Keypad, Joystick, Push Button, Switches
- Output: LCD Display, 7-Segment Display, Speaker, Buzzers, LEDs, LED Bar
- Actuators: DC Motor, Servo Motor, Stepper Motor, Motor Drivers, Water Pump, Relays, Optocouplers

For each of the selected input/output modules and actuators, you have to study, analyze and discuss the following next along with building circuit and coding:

- Number and names of selected output modules and actuators and the purpose they serve for your application
- Power requirements, operating voltage and currents from datasheet
- Digital or analog range of the output module from datasheet
- Number of pins used by the module and their functionality from datasheet
- Required communication protocol and baud rates selection
- Logical working of output modules and actuators i.e., how is it affected by the input trigger and microcontroller processing?
- Integrate the modules with TivaC and write a functional code for all the selected components using Energia or Keil.

Motors for movement. PWM. Pin config

```
#include "TM4C123GH6PM.h"
void Delay_ms(int time_ms);
//int duty_cycle_A = 4999;
//int duty_cycle_B = 4999;
void PWM_init(void);
void Turn_Clockwise_A(void);
void Turn_AntiClockwise_A(void);
void Turn_Clockwise_B(void);
void Turn_AntiClockwise_B(void);


// Motor A - Clockwise
void Turn_Clockwise_A(void)
{
    SYSCTL->RCGCGPIO |= (1 << 0);         // Enable clock to PORTA
    GPIOA->DIR |= (1 << 2) | (1 << 3);    // Set PA2 and PA3 as outputs
    GPIOA->DEN |= (1 << 2) | (1 << 3);    // Enable digital functionality for PA2, PA3
    GPIOA->DATA |= (1 << 2);              // Set PA2 (IN1) HIGH
    GPIOA->DATA &= ~(1 << 3);             // Set PA3 (IN2) LOW
}

// Motor A - Counterclockwise
void Turn_AntiClockwise_A(void)
{
    SYSCTL->RCGCGPIO |= (1 << 0);         // Enable clock to PORTA
    GPIOA->DIR |= (1 << 2) | (1 << 3);    // Set PA2 and PA3 as outputs
    GPIOA->DEN |= (1 << 2) | (1 << 3);    // Enable digital functionality for PA2, PA3
    GPIOA->DATA &= ~(1 << 2);             // Set PA2 (IN1) LOW
    GPIOA->DATA |= (1 << 3);              // Set PA3 (IN2) HIGH
}

// Motor B - Clockwise
void Turn_Clockwise_B(void)
{
    SYSCTL->RCGCGPIO |= (1 << 0);         // Enable clock to PORTA
    GPIOA->DIR |= (1 << 5) | (1 << 6);    // Set PA5 and PA6 as outputs
    GPIOA->DEN |= (1 << 5) | (1 << 6);    // Enable digital functio nality for PA5, PA6
    GPIOA->DATA |= (1 << 5);              // Set PA5 (IN3) HIGH
    GPIOA->DATA &= ~(1 << 6);             // Set PA6 (IN4) LOW
}

// Motor B - Counterclockwise
void Turn_AntiClockwise_B(void)
{
    SYSCTL->RCGCGPIO |= (1 << 0);         // Enable clock to PORTA
    GPIOA->DIR |= (1 << 5) | (1 << 6);    // Set PA5 and PA6 as outputs
    GPIOA->DEN |= (1 << 5) | (1 << 6);    // Enable digital functionality for PA5, PA6
    GPIOA->DATA &= ~(1 << 5);             // Set PA5 (IN3) LOW
    GPIOA->DATA |= (1 << 6);              // Set PA6 (IN4) HIGH
}

void PWM_init(void)
{
    SYSCTL->RCGCPWM |= (1 << 1);          // Enable clock to PWM1 module
```

Task 3: Controlling and Communicating your System with Mobile Phone

In this task, you will use you smart phone to control the designed system. It's upto you to use Bluetooth based communication or any other identified module to interface your mobile with TivaC. Explain the implemented communication interface between system and mobile along with ensuring that your mobile you should be able to achieve the following:

- Start/Stop/Reset the system while its running
- Mobile should display some information/data if requested from system

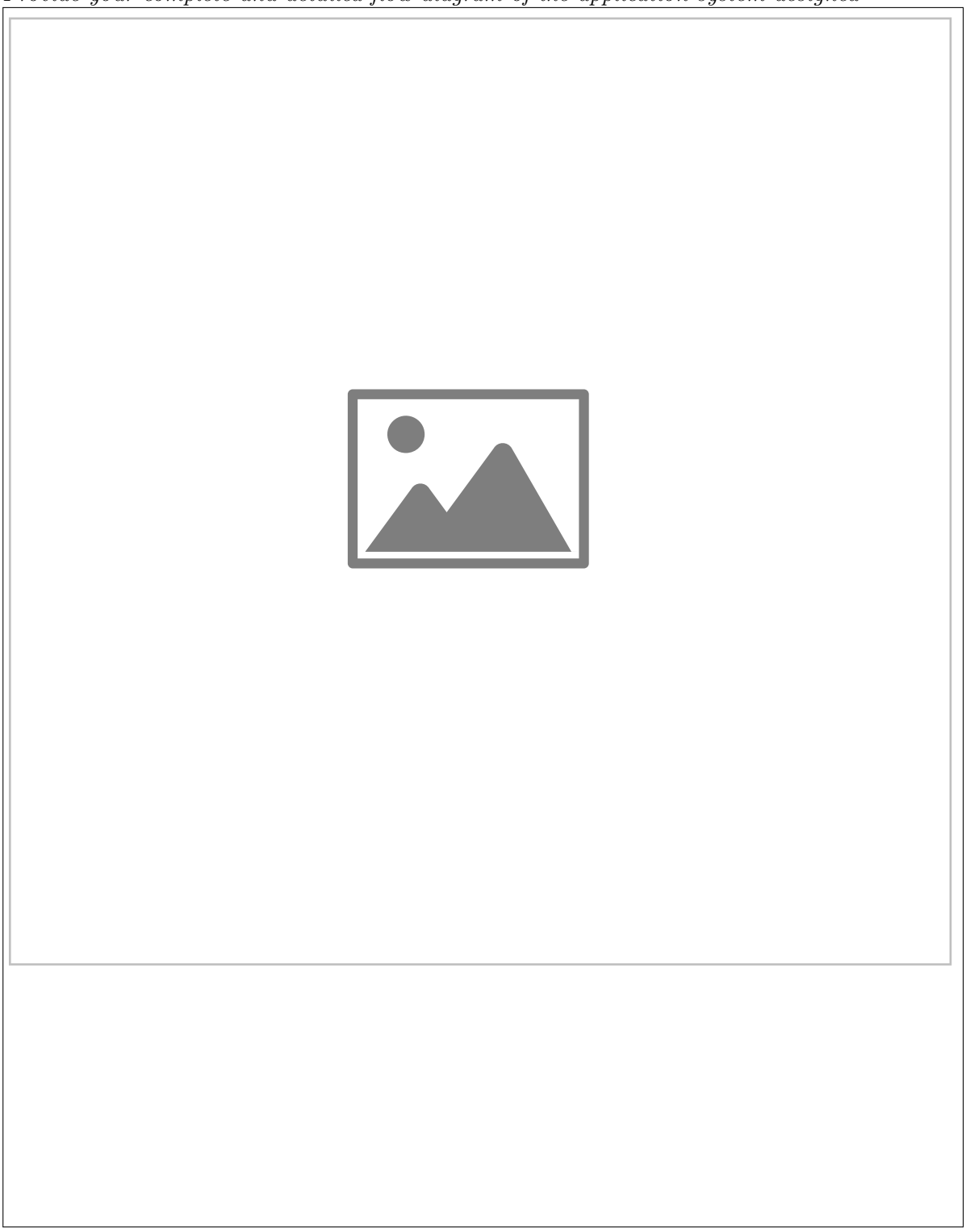
```
#include "UART.h"
#include "TM4C123GH6PM.h"

/* UART initialization */
void UART5_init(void)
{
   SYSCTL->RCGCUART |= 0x20; /* Enable clock to UART5 */
   SYSCTL->RCGCGPIO |= 0x10; /* Enable clock to PORTE for PE4/Rx and PE5/Tx */

   UART5->CTL = 0;         /* Disable UART5 */
   UART5->IBRD = 104;      /* 9600 baud rate integer part */
   UART5->FBRD = 11;       /* 9600 baud rate fractional part */
   UART5->LCRH = 0x60;     /* 8-bit, no parity, 1-stop bit */
   UART5->CTL = 0x301;     /* Enable UART5, Tx, and Rx */

   GPIOE->DEN = 0x30;      /* Enable digital PE4, PE5 */
   GPIOE->AFSEL |= 0x30;   /* Alternate function for PE4, PE5 */
   GPIOE->PCTL |= 0x00110000; /* Configure PE4, PE5 for UART */
}

void UART5_Transmitter(unsigned char data)
{
   while ((UART5->FR & (1 << 5)) != 0)
      ; /* Wait until Tx buffer is not full */
   UART5->DR = data;        /* Transmit data */
}

void printstring(char *str)
{
   while (*str)
   {
      UART5_Transmitter(*(str++));
   }
}
```

Task 4: [BONUS] Incorporating Interrupts to Optimize System

This task serve as bonus, in which you are required to implement an internal or external interrupt. It's upto you to decide how to generate the external interrupt either using push button, keypad, switch etc. You are required to use this internal/external interrupt to generate the output instead of continuously reading the sensor. Discuss your implementation and approach below.

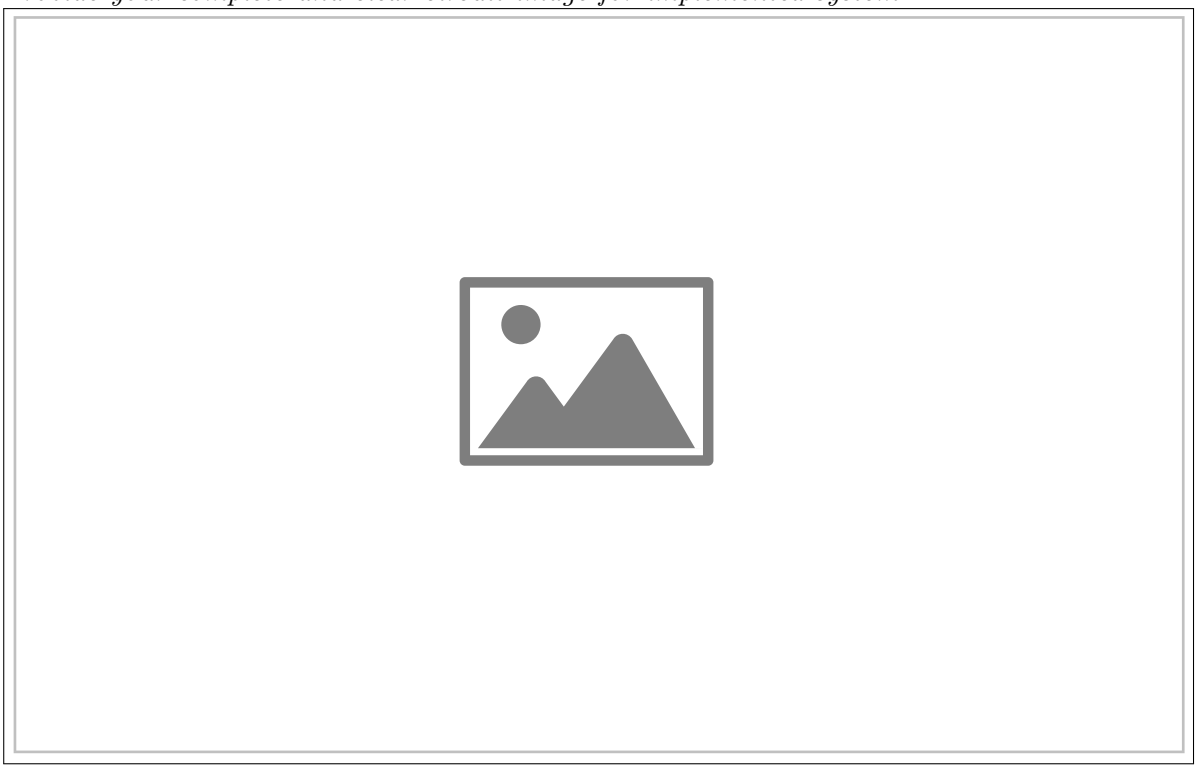
Task 5: Detailed Flow Diagram of Functional System

Provide your complete and detailed flow diagram of the application system designed



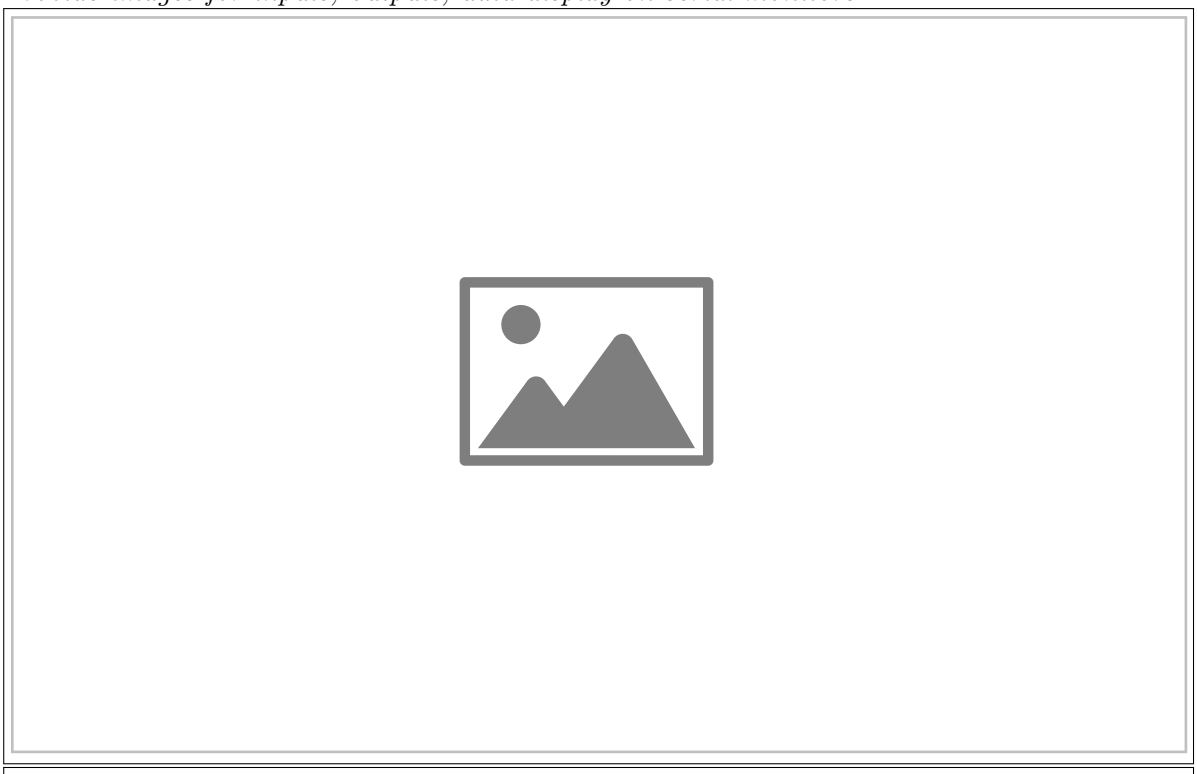
Provide your complete code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

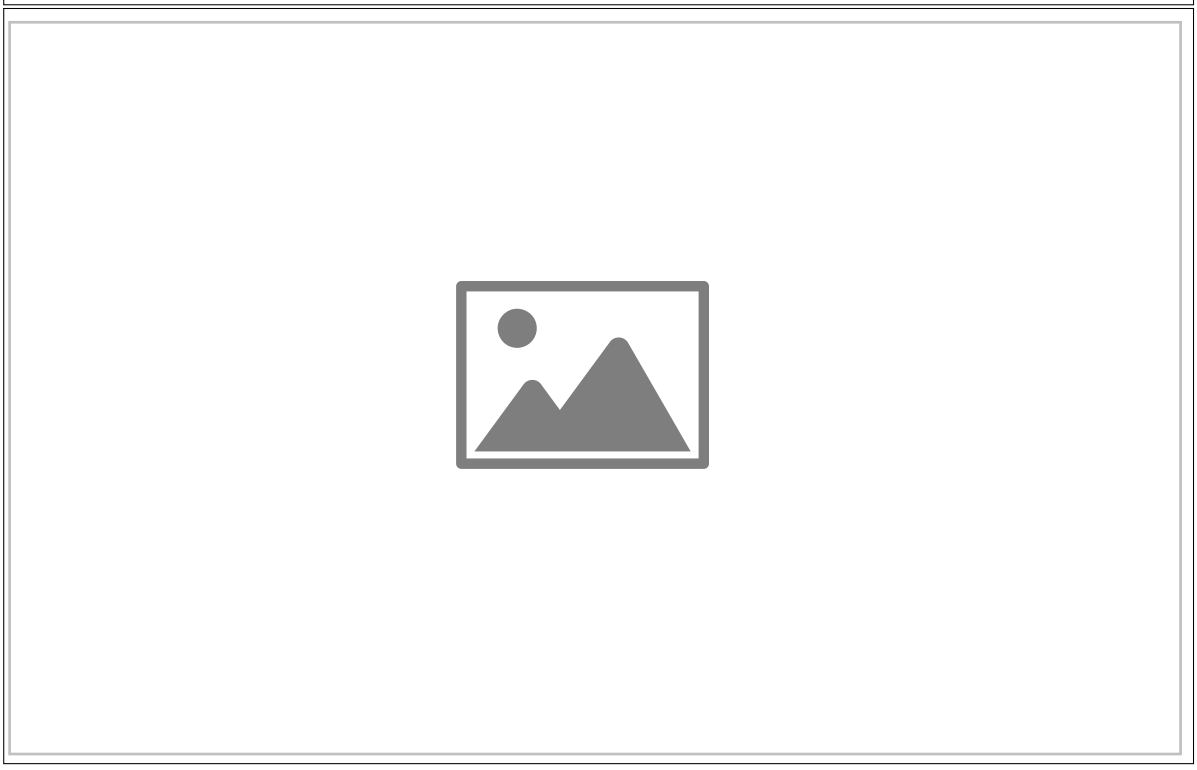
Provide your complete and clear circuit image for implemented system



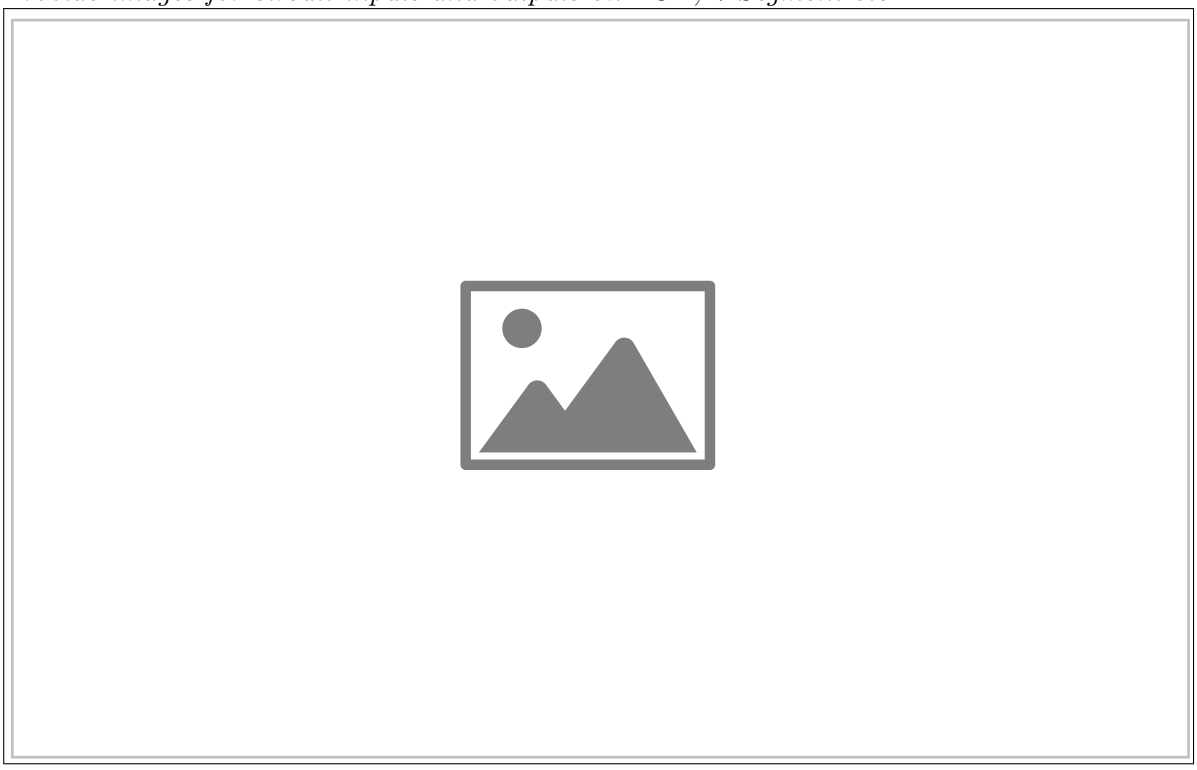
Attach link for video demonstration of your functional system designed (Youtube Link or OneDrive)

Provide images for inputs, outputs, data display on serial monitors

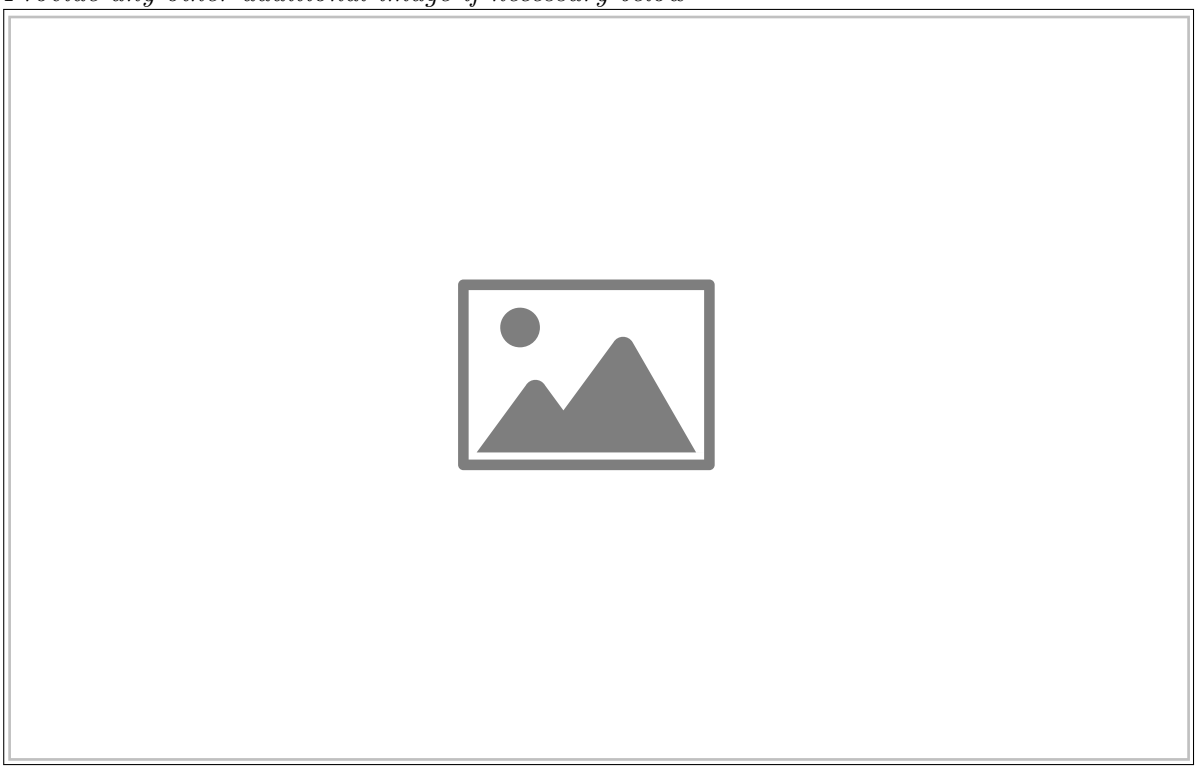




Provide images for circuit inputs and outputs on LCD, 7-Segment etc.



Provide any other additional image if necessary below



1 Assessment Rubrics

Marks distribution

		LR1	LR2	LR3	LR7	LR10	LR11	LR9
In-lab	Task 0	-	-	-	-	5	-	20
	Task 1	-	-	-	-	5	-	
	Task 2	-	-	-	-	5	-	
	Task 3	-	-	5	-	5	-	
	Task 4	-	-	-	-	5	-	
	Task 5	15	15	-	20	5	-	
Total Marks 105								

Marks obtained

		LR1	LR2	LR3	LR7	LR10	LR11	LR9
In-lab	Task 0	-	-	-	-		-	
	Task 1	-	-	-	-		-	
	Task 2	-	-	-	-		-	
	Task 3	-	-		-		-	
	Task 4	-	-	-	-		-	
	Task 5			-			-	
Marks Obtained 105								