**Template for Arduino and Joystick Shield and Motor Activity**

**Statement Given:**

**Task A:**

**Using Joystick Shield**

1. Test Up , Down , Left , Right Buttons with LED output
2. Test 2-Axis joystick values with Intensity of LED

**Task B:**

1. Control Positioning of Servo Motor changing from 20 to 180 degree and Vice Versa

Try changing values

1. Try Running DC Motor in Forward and Reverse Direction

Control Speed of DC Motor

**Evaluation Criteria:**

1. Connections as per task given,

2. Code for Arduino for the Task.

3. Successful execution of the activities.

**Performance-15 Marks : Joystick tasks = 5 marks , Servo Motor : 5 marks DC Motor using TinkerCAD = 5 marks**

**Submission-10 Marks**

**Team**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr No** | **Roll No** | **Name** | **Work Done** |
| 1 | 16010123011 | Aaryan Dubey | Connections + Coding |
| 2 | 16010123012 | Aaryan Sharma | Connections + Coding |
| 3 | 16010123013 | Aayush Hardas | Connections + Coding |
| 4 | 16010123014 | Aayush Sawant | Connections + Coding |
| 5 | 16010123015 | Abdullah Qureshi | Connections + Coding |

**Fill your details as per following points**

* **Code for the task given**

**1.A.**

include <nRF24L01.h>

#include <RF24.h>

#include <SPI.h>

#define CE\_PIN 9

#define CSN\_PIN 10

int up\_button = 2; // Boton Amarillo - A

int down\_button = 4; // Boton Amarillo - C

int left\_button = 5; // Boton Azul - D

int right\_button = 3; // Boton Azul - B

int start\_button = 6; // Boton F

int select\_button = 7; // Boton E

int analog\_button = 8; //

int x\_axis = A0;

int y\_axis = A1;

int buttons[]={up\_button, down\_button,left\_button,

right\_button,start\_button,select\_button,analog\_button};

const uint64\_t pipe = 0xE8E8F0F0E1LL;

RF24 radio(CE\_PIN,CSN\_PIN);

char msg[20] = "";

void setup(){

for(int i; i <7 ; i++)

{

pinMode(buttons[i],INPUT);

digitalWrite(buttons[i],HIGH);

}

Serial.begin(9600);

radio.begin();

radio.openWritingPipe(pipe);

}

void loop(){

if(digitalRead(up\_button)==LOW)

{

char msg[]="up";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("UP Button Pressed");

}

if(digitalRead(down\_button)==LOW)

{

char msg[]="down";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("Down Button Pressed");

}

if(digitalRead(left\_button)==LOW)

{

char msg[]="left";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("Left Button Pressed");

}

if(digitalRead(right\_button)==LOW)

{

char msg[]="right";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("Right Button Pressed");

}

if(digitalRead(start\_button)==LOW)

{

char msg[]="start";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("Start Button Pressed");

}

if(digitalRead(select\_button)==LOW)

{

char msg[]="select";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("Select Button Pressed");

}

if(digitalRead(analog\_button)==LOW)

{

char msg[]="analgobut";

radio.write(&msg,sizeof(msg));

delay(100);

Serial.println("Analog Button Pressed");

}

Serial.print("\n X = "),Serial.print(analogRead(x\_axis)),Serial.print(" \n Y = "), Serial.print(analogRead(y\_axis));

Serial.print(" ");

delay(1000);

}

**B.**

#include <Servo.h>

Servo servo;

int angle = 10;

void setup() {

servo.attach(5);

servo.write(angle);

}

void loop()

{

// scan from 0 to 180 degrees

for(angle = 20; angle < 130; angle++)

{

servo.write(angle);

delay(15);

}

// now scan back from 180 to 0 degrees

for(angle = 130; angle > 10; angle--)

{

servo.write(angle);

delay(15);

}

}

**2.**

void setup()

{

pinMode(11, OUTPUT);// En of Motor 1

pinMode(7,OUTPUT); // In1 of Motor 1

pinMode(6,OUTPUT);// In 2 of Motor 1

}

void fwdrev() // function reverses direction of motor

{

digitalWrite(7,LOW);// In1 = 0, FWD

digitalWrite(6,HIGH);//In2 =1

digitalWrite(11,HIGH);

delay(5000); // Wait for 5 second(s)

digitalWrite(7, LOW);// In1 = 0, Stop

digitalWrite(6,LOW);// In2 = 0

delay(5000); // Wait for 1000 millisecond(s)

digitalWrite(7, HIGH);//In1 =1, REV

digitalWrite(6,LOW);//In2 =0

delay(5000);

digitalWrite(7, LOW);// In1 = 0, Stop

digitalWrite(6,LOW);// In2 = 0

}

void speed () // function for 2 speed settings of DC Motor

{

digitalWrite(7, LOW);

digitalWrite(6,HIGH);

analogWrite(11,50); // Slow Speed

delay(25000); // Wait for 1000 millisecond(s)

analogWrite(11,200);// High Speed

digitalWrite(7, LOW);

digitalWrite(6,HIGH);

delay(25000); // Wait for 1000 millisecond(s)

}

void loop()

{

fwdrev();

speed();

}

* **Photo and Video of Actual implementation**

[**https://drive.google.com/drive/folders/16qrmkVsXrSr-wmseMYh1v6T9v4Dg7dvC?usp=sharing**](https://drive.google.com/drive/folders/16qrmkVsXrSr-wmseMYh1v6T9v4Dg7dvC?usp=sharing)

* **Observations, reflection on activity**

Designing a Motar circuit in Tinkercad and building a working model and then coding it is an excellent hands-on activity. It helps students in practical applications. Coding the joystick shield enables real-time interaction with your projects. As you move the joystick, the code interprets those movements and translates them into actions.Writing code for a joystick shield involves creating the programming logic that determines how the input from the joystick is processed. This could include mapping joystick movements to specific functions or controlling other components based on the input.When coding for a joystick shield, you often use event-driven programming, where actions are triggered by specific events, such as joystick movement or button presses. This approach can make the code more responsive and modular.The process of coding for a joystick shield is a valuable learning journey. It allows you to delve into topics such as analog sensor readings, conditional statements, and event-driven programming, contributing to your overall programming and electronics knowledge.Projects involving joystick control of servo motors provide an excellent learning opportunity, as they combine elements of hardware interfacing, coding, and real-time control.In summary, coding for a joystick shield adds a layer of complexity and creativity to your projects. It's an opportunity to apply programming concepts in a hands-on, interactive context and create projects that respond dynamically to user input.