//Task 1 - Output Digital Signal

int outputPin = 9; // Set output pin

int state = 0; // Set initial state

void setup() {

pinMode(outputPin, OUTPUT); // Set output pin as output

pinMode(inputPin, INPUT); // Set input pin as input

pinMode(LED\_PIN, OUTPUT);

Serial.begin(9600);// Initialize serial communication

}

void loop() {

if (state == 0) {

digitalWrite(outputPin, HIGH);

delayMicroseconds(200);

state = 1;

} else if (state == 1) {

digitalWrite(outputPin, LOW);

delayMicroseconds(50);

state = 2;

} else if (state == 2) {

digitalWrite(outputPin, HIGH);

delayMicroseconds(50);

state = 3;

} else if (state == 3) {

digitalWrite(outputPin, LOW);

delayMicroseconds(50);

state = 4;

} else if (state == 4) {

digitalWrite(outputPin, LOW);

delayMicroseconds(50);

state = 5;

} else if (state == 5) {

digitalWrite(outputPin, HIGH);

delayMicroseconds(50);

state = 6;

} else if (state == 6) {

digitalWrite(outputPin, HIGH);

delayMicroseconds(30);

state = 7;

} else if (state == 7) {

digitalWrite(outputPin, LOW);

delayMicroseconds(50);

state = 8;

} else if (state == 8) {

digitalWrite(outputPin, HIGH);

delayMicroseconds(50);

state = 0;

}

delayMicroseconds(3700); // Wait for the remaining time in the 4ms period

}

//Task 2 - Measure Frequency of Square Wave Signal

int inputPin = 2; // Set input pin

int frequency = 0; // Set initial frequency

void loop() {

int period = pulseIn(inputPin, HIGH); // Measure period of input signal

frequency = (int) (1000.0 / period); // Convert period to frequency in Hz

frequency = constrain(frequency, 0, 99); // Scale frequency between 0 and 99

Serial.print(frequency);

Serial.print(",");

Serial.println(0); // Output frequency value and 0 (since it's Task 2)

delay(20); // Wait for 20ms before next measurement

}

//Task 3 - Measure Frequency of Second Square Wave Signal

int inputPin = 3; // Set input pin

int frequency = 0; // Set initial frequency

void loop() {

int period = pulseIn(inputPin, HIGH); // Measure period of input signal

frequency = (int) (1000.0 / period); // Convert period to frequency in Hz

frequency = constrain(frequency, 0, 99); // Scale frequency between 0 and 99

Serial.print(0);

Serial.print(",");

Serial.print(frequency); // Output 0 and frequency value (since it's Task 3)

Serial.println();

delay(8); // Wait for 8ms before next measurement

}

//Task 4 - Read Analogue Input and Compute Filtered Analogue Value

// Pin definitions

#define ANALOG\_IN A0

#define LED\_PIN 13

// Variables

float last\_four\_readings[4];

int current\_reading\_index = 0;

float average\_analog\_in = 0.0;

float max\_range = 3.3;

int half\_max\_range = (int)(max\_range / 2.0 \* 1023.0 / max\_range); // Map to 0-1023 range

int frequency\_task\_2 = 0;

int frequency\_task\_3 = 0;

unsigned long last\_serial\_output\_time = 0;

void loop() {

// Read the analog input

int raw\_analog\_in = analogRead(ANALOG\_IN);

float analog\_in\_voltage = raw\_analog\_in / 1023.0 \* max\_range;

// Filter the analog input by averaging the last 4 readings

last\_four\_readings[current\_reading\_index] = analog\_in\_voltage;

current\_reading\_index = (current\_reading\_index + 1) % 4;

average\_analog\_in = (last\_four\_readings[0] + last\_four\_readings[1] + last\_four\_readings[2] + last\_four\_readings[3]) / 4.0;

// Check if the filtered value is greater than half of the maximum range

if (average\_analog\_in > (float)half\_max\_range) {

digitalWrite(LED\_PIN, HIGH);

} else {

digitalWrite(LED\_PIN, LOW);

}

// Compute the frequency for Task 2 and Task 3

frequency\_task\_2 = (int)(1.0 / (20.0 / 1000.0));

frequency\_task\_3 = (int)(1.0 / (average\_analog\_in / max\_range \* (20.0 / 1000.0)));

// Scale the frequencies and bound them between 0 and 99

frequency\_task\_2 = map(frequency\_task\_2, 0, 333, 0, 99);

frequency\_task\_3 = map(frequency\_task\_3, 0, 500, 0, 99);

// Log the information to the serial port every 100ms

if (millis() - last\_serial\_output\_time >= 100) {

Serial.print(frequency\_task\_2);

Serial.print(",");

Serial.println(frequency\_task\_3);

last\_serial\_output\_time = millis();

}

// Wait for 20ms before the next reading

delay(20);

}