Programming Microcontrollers - Homework

Thomas BALZAN - Joachim SONIGO - Grégoire DOUILLARD-JACQ

April 2023

1 Introduction

In this practical work we used an arduino nano card on site and then continued this work with a simulation on tinkercad. The goal of this project is to use analogue pins and I/O devices in a way that allows us to change the state of certain pins in function of physical changes.

2 Exercise: Ultrasonic Sensor

2.1

The value 0.01723 is obtained through the conversion of the speed of sound (343 m/s). To get the speed in microseconds, we need to divide the speed of sound by 2 to account for the time the wave takes to make the trip and come back to the sensor. With this we have:
distance in centimeters = (time in microseconds / 2) * 34300 cm/s, therefore :
distance in centimeters = time in microseconds * 0.01723

2.2 Questions 2

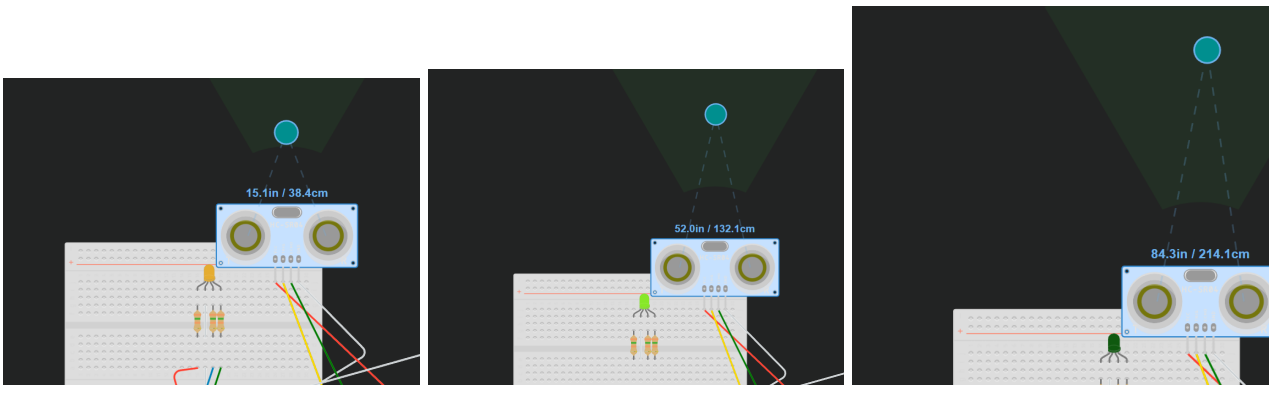
```
// Arduino code
//
int trigPin = 3;
int echoPin = 2;
int ledPin = 11;
long lecture_echo;
long cm;

long readdistance(int trigPin, int echoPin)
{
  pinMode(trigPin, OUTPUT);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(5);

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  pinMode(echoPin, INPUT);
  return pulseIn(echoPin, HIGH);
}

void setup()
{
  pinMode(trigPin, OUTPUT);
  digitalWrite(trigPin, LOW);
  pinMode(echoPin, INPUT);

  Serial.begin(9600);
}

void loop()
{
```

```
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  cm = 0.01723*readdistance(trigPin, echoPin);
  Serial.print("Distance in cm :");
  Serial.println(cm);

  if (cm > 50 && cm <= 250)
    analogWrite(ledPin, 200-0.85*cm);
  else
    digitalWrite(ledPin, HIGH);

  Serial.print("Distance in cm :");
  Serial.println(readdistance(trigPin, echoPin));
  delay(100);
}
```

2.3 Questions 4

```
int trigPin = 3;
int echoPin = 2;
int redPin = 11;
int greenPin = 9;
int bluePin = 10;
int buzzerPin= 8;

long read_echo;
long cm;

long readdistance(int triggerPin, int echoPin)
{
 pinMode(trigPin, OUTPUT);
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);

 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);

 pinMode(echoPin, INPUT);
 return pulseIn(echoPin, HIGH);
}


void setup()
{
  Serial.begin(9600);

  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);

  digitalWrite(redPin, LOW);
  digitalWrite(greenPin, LOW);
  digitalWrite(bluePin, LOW);
  digitalWrite(buzzerPin, LOW);

  delay(2000);
}

void loop()
```

```
{
  cm = 0.01723*readdistance(trigPin, echoPin);
  Serial.print("Distance in cm");
  Serial.println(cm);

  if (cm <= 15)
  {
    digitalWrite(greenPin, LOW);
    digitalWrite(redPin, LOW);
      digitalWrite(bluePin, HIGH);
      digitalWrite(buzzerPin, HIGH);
      delay(250);
      digitalWrite(bluePin, LOW);
      digitalWrite(buzzerPin, LOW);
  }
  if (15 < cm && cm <= 150)
  {
    analogWrite(greenPin, 1.275*cm);
    analogWrite(redPin, 190-1.275*cm);
      analogWrite(bluePin, 0);
      digitalWrite(buzzerPin, LOW);
  }
  if (150 < cm && cm <= 300)
  {
      digitalWrite(greenPin, HIGH);
      digitalWrite(redPin, LOW);
      digitalWrite(bluePin, LOW);
      digitalWrite(buzzerPin, LOW);
  }
  if (cm > 300)
  {
    digitalWrite(greenPin, LOW);
    digitalWrite(redPin, LOW);
      digitalWrite(bluePin, LOW);
      digitalWrite(buzzerPin, LOW);
  }

  delay(70);
}
```



As we can see in the screenshots, the light of the RGB led gradually switches from green to red as we get closer to the sensor.
Also, the previously given code can be translated into arduino assembly language :

```
#include <avr/io.h>
#include <util/delay.h>

int trigPin = 3;
int echoPin = 2;
int ledPin = 11;
long lecture_echo;
long cm;

long readdistance(int trigPin, int echoPin) {
    DDRD |= (1 << trigPin);
    PORTD &= ~(1 << trigPin);
    _delay_us(5);
    PORTD |= (1 << trigPin);
    _delay_us(10);
    PORTD &= ~(1 << trigPin);
    DDRD &= ~(1 << echoPin);
    return pulseIn(echoPin, HIGH);
}

void setup() {
    DDRD |= (1 << trigPin);
    PORTD &= ~(1 << trigPin);
    DDRD &= ~(1 << echoPin);
    Serial.begin(9600);
}

void loop() {
    PORTD |= (1 << trigPin);
    _delay_us(10);
    PORTD &= ~(1 << trigPin);
    cm = 0.01723 * readdistance(trigPin, echoPin);
    Serial.print("Distance in cm :");
    Serial.println(cm);
    if (cm > 50 && cm <= 250) {
        analogWrite(ledPin, 200 - 0.85 * cm);
    } else {
        digitalWrite(ledPin, HIGH);
    }
    Serial.print("Distance in cm :");
    Serial.println(readdistance(trigPin, echoPin));
    delay(100);
}
```

2.4 Question 5

Here is the code translated with registers instead of using pinMode(), digitalWrite() and analogWrite() functions.

```
// C++ code
//
int trigPin = 3;
int echoPin = 2;
int redPin = 11;
int greenPin = 9;
int bluePin = 10;
int buzzerPin= 8;

long read_echo;
long cm;

long readdistance(int triggerPin, int echoPin)
{
 DDRD|=(1<<4);
```

```
 //digitalWrite(trigPin, LOW);
 PORTD &=~(1<<4);
 delayMicroseconds(2);

 //digitalWrite(trigPin, HIGH);
 PORTD |=(1<<4);
 delayMicroseconds(10);
 //digitalWrite(trigPin, LOW);
 PORTD &=~(1<<4);

 DDRD &=~(1<<5);
 return pulseIn(echoPin, HIGH);
}

void setup()
{
  Serial.begin(9600);
  DDRB|=(1<<2);
  DDRB|=(1<<4);
  DDRB|=(1<<3);
  DDRB|=(1<<5);

  //digitalWrite(redPin, LOW);
  PORTB &=~(1<<2);
  //digitalWrite(greenPin, LOW);
  PORTB &=~(1<<4);
  //digitalWrite(bluePin, LOW);
  PORTD &=~(1<<3);
  //digitalWrite(buzzerPin, LOW);
  PORTB &=~(1<<5);
  delay(2000);
}

void loop()
{
  cm = 0.01723*readdistance(trigPin, echoPin);
  Serial.print("Distance in cm :");
  Serial.println(cm);

  if (cm <= 15)
  {
    //digitalWrite(greenPin, LOW);
    PORTB &=~(1<<4);
    //digitalWrite(redPin, LOW);
    PORTB &=~(1<<2);
    //digitalWrite(bluePin, HIGH);
    PORTB |=(1<<3);
    //digitalWrite(buzzerPin, HIGH);
    PORTB |=(1<<5);
    delay(250);
    //digitalWrite(bluePin, LOW);
    PORTB &=~(1<<3);
    //digitalWrite(buzzerPin, LOW);
    PORTB &=~(1<<5);
  }
  if (15 < cm && cm <= 150)
  {
    //analogWrite(greenPin, 1.275*cm);
    OCR1A = 1.275*cm;
    //analogWrite(redPin, 190-1.275*cm);
    OCR2A = (190-1.275*cm);
    //analogWrite(bluePin, 0);
    OCR1B = 0;
    //digitalWrite(buzzerPin, LOW);
```

```
    PORTB &=~(1<<5);
  }
  if (150 < cm && cm <= 300)
  {
      //digitalWrite(greenPin, HIGH);
            PORTB |=(1<<4);
      //digitalWrite(redPin, LOW);
            PORTB &=~(1<<2);
      //digitalWrite(bluePin, LOW);
            PORTB &=~(1<<3);
      //digitalWrite(buzzerPin, LOW);
            PORTB &=~(1<<5);
  }
  if (cm > 300)
  {
    //digitalWrite(greenPin, LOW);
    PORTB &=~(1<<4);
    //digitalWrite(redPin, LOW);
    PORTB &=~(1<<2);
    //digitalWrite(bluePin, LOW);
    PORTB &=~(1<<3);
    //digitalWrite(buzzerPin, LOW);
    PORTB &=~(1<<5);
  }
  delay(70);
}
```

We used registers OCR1A, OCR2A and OCR1B to replace the analogWrite() functions, respectively for greenpin, redpin and bluepin by looking at the ARDUINO's datasheet.