

Intro til Arduino & Arduino Robot

Lasse Matias Hadberg
TEC Ballerup - Telegrafvej 9.

Indhold

| | |
|---|----|
| Arduino projektbeskrivelse | 3 |
| Installationsvejledning | 3 |
| Ting du skal bruge: | 3 |
| Arduino med tilhørende udstyr | 3 |
| PC med tilhørende skærm, mus & tastatur | 4 |
| Internet | 4 |
| Dokumentation..... | 4 |
| Projekt 1 – LED flashing | 4 |
| Del 1 | 4 |
| Del 2 | 4 |
| Del 3 | 5 |
| Projekt 2 – Spaceship Interface | 5 |
| Fysiske opsætningskrav..... | 5 |
| Kildekode: | 5 |
| Projekt 3 – Love-O-Meter..... | 6 |
| Fysiske opsætningskrav..... | 6 |
| Kildekode: | 7 |
| Projekt 4 – Colour Mixing Lamps..... | 8 |
| Fysiske opsætningskrav..... | 8 |
| Kildekode: | 9 |
| Projekt 5 – Mood cue..... | 11 |
| Fysiske opsætningskrav..... | 11 |
| Kildekode: | 11 |
| Projekt 6 – Light Theremin..... | 12 |
| Fysiske opsætningskrav..... | 12 |
| Kildekode: | 12 |
| Projekt 7 – Keyboard Instrument | 14 |
| Fysiske opsætningskrav..... | 14 |
| Serial outputs | 14 |
| Kildekode: | 14 |
| Projekt 8 – Digital Hourglass | 16 |
| Fysiske opsætningskrav..... | 16 |
| Kildekode: | 16 |
| Projekt 11 – Crystal Ball..... | 18 |
| Fysiske opsætningskrav..... | 18 |
| Kildekode: | 18 |

| | |
|--------------------------------|----|
| Øvelse 1. – Trafiklys | 20 |
| Fysiske opsætningskrav..... | 20 |
| Kildekode: | 20 |
| Øvelse 2. - Robot (Del 1)..... | 22 |

Arduino projektbeskrivelse

Dette projekt er en sammenfletning af tre delprojekter. Dels introøvelser med Arduino startkit, øvelse med opbyggelse af trafiklys, refleksion omhandlende robotteknologiens brug historisk samt fremadrettet & programmering af en robot vha. Arduino.

Installationsvejledning

Ting du skal bruge:

Arduino med tilhørende udstyr

Se Arduino startkit hos Arduinos egen webshop.

(<https://store.arduino.cc/>)

Da Arduino både er open-source hardware & software, er der produceret en lang række tilsvarende produkter. Disse *kan* også være gyldige til din use-case. Som et alternativt til det officielle Arduinoprodukt kan jeg anbefale én af følgende kits da disse indeholder en lang række udstyr til både begynder- og rutinerede niveauer af brug.

(<https://arduinoshoppen.dk/produkt/arduino-learning-starter-kit-inkl-uno-r3/> Eller <https://arduinoshoppen.dk/produkt/arduino-beginner-starter-kit-inkl-uno-r3/>)

Andre populære breadboards inkludere men er ikke begrænset til:

- Sparkfun Redboard Artemis
- Silicon Labs Wonder Gecko
- Adafruit Feather Huzzah

Udstyr du skal være opmærksom på, at dit kit indeholder:

- Breadboard
- Leds in varied colours (Vær opmærksom på, at disse kan have påloddede resistors)
- Resistors (100, 220, 1k, 2k, 10k, 1m)
- Capacitors (10nf, 100nf)
- Cables
- Jumper Wires
- Pushbuttons
- Potentiometers
- Battery Snap
- DC Motor
- LCD
- Piezo
- Phototransistor
- Servo Motor
- Temperature Sensor
- Tilt Sensor
- Transistor
- USB Cable
- Evt. skruetrækker til at samle breadboardet.

Efter installation kan en 7-12 volt strømforsyning bruges til at drive Arduinoen.

PC med tilhørende skærm, mus & tastatur

Denne skal køre Arduino IDE som bruges til at skrive kode i, samt at flashe din software til dit breadboard. En let tilgængelig USB-port er at foretrække da flash processen gøres over et USB-kabel.

Internet

Jeg vil stærkt foreslå en internetforbindelse da Arduino har en stor- og utroligt engageret brugergruppe. Disse har gennem tiden delt et hav af ressourcer der ligger frit tilgængeligt for alle.

Dokumentation

Introduktionsforløbet består af en gennemgang af "Arduino Projects Book" samt udførslen af opgaverne deri.

Projekt 1 – LED flashing

Første projekt består af 3 delopgaver. Disse er rent mekaniske og har ikke krævet nogen kode. Nedenfor illustreres de dele der skal bruges til hvert setup samt resultaterne fra disse.

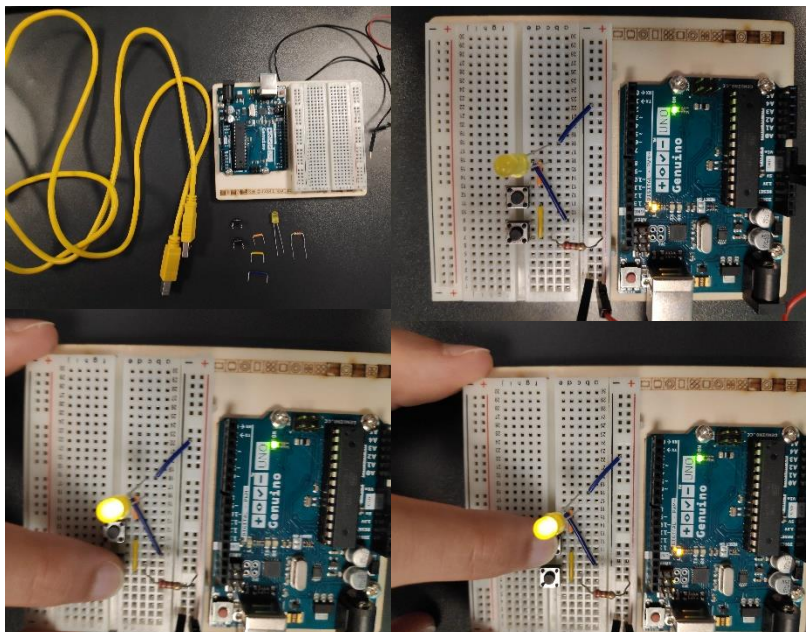
Del 1

I første delopgave skal én diode lyse op når knappen holdes inde. Dette muliggøres da Pushknappen i trykket tilstand færdiggøre det kredsløb der er op



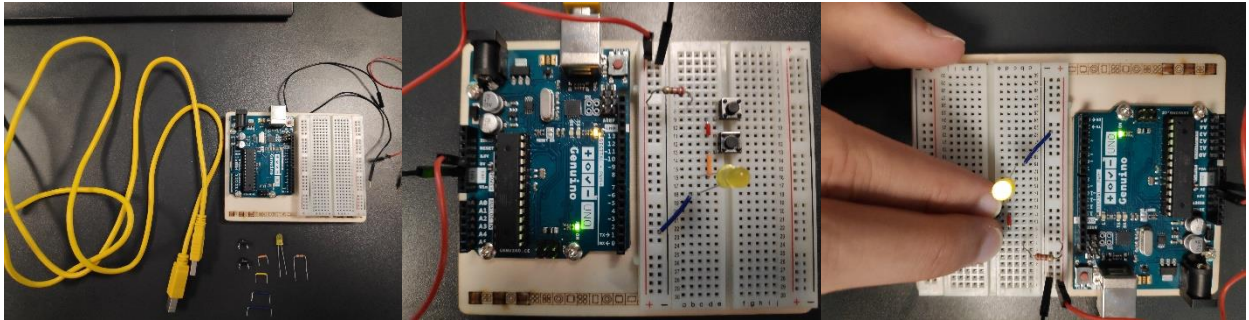
Del 2

Anden delopgave skal dioden lyse op når én af de to pushknapper trykkes på. Samme metode som ovenfor anvendes her.



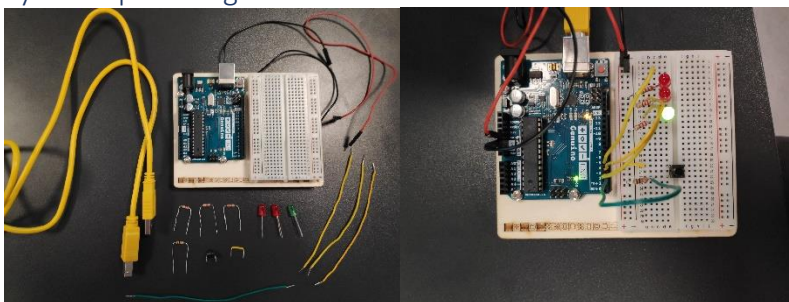
Del 3

I tredje delopgave gentager vi metoden anvendt ovenfor. Her skal *begge* taster være holdt nede for at kredsløbet færdiggøres.



Projekt 2 – Spaceship Interface

Fysiske opsætningskrav



Andet projekt indebære en mindre diodekonstruktion der skal kunne understøtte den skrevne kode. Arduinoen looper igennem at slukke og tænde bestemte dioder for at give disse en ”cascading” effekt.

Kildekode:

```
int switchState = 0;

void setup() {
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(2, INPUT);
}

void loop() {
  switchState = digitalRead(2);
  if (switchState == LOW){ // If the button ISN'T pressed.
    digitalWrite(3, HIGH); // Green LED
    digitalWrite(4, LOW); // Red LED
    digitalWrite(5, LOW); // Red LED
  }
```



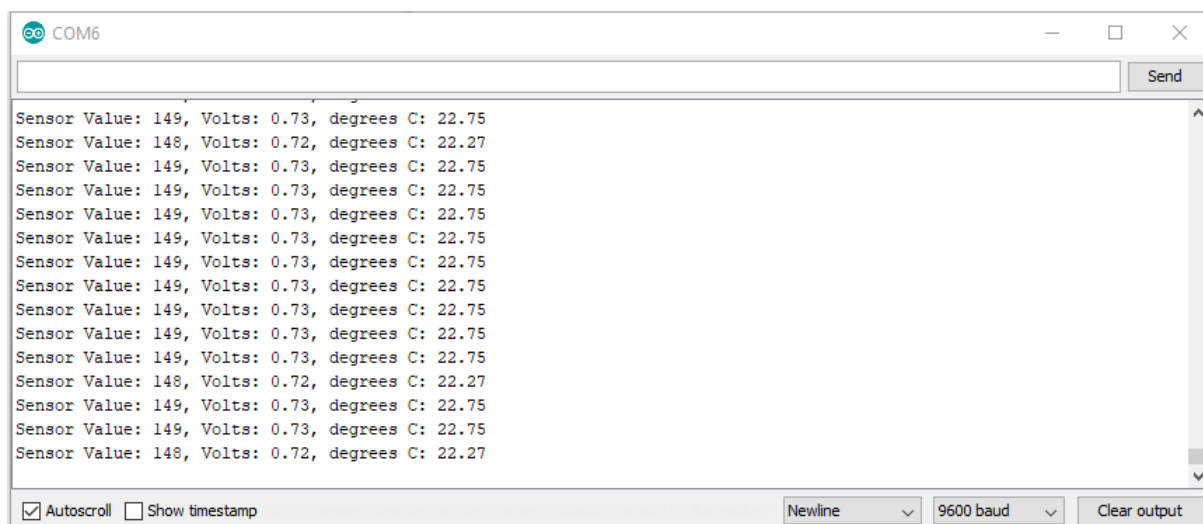
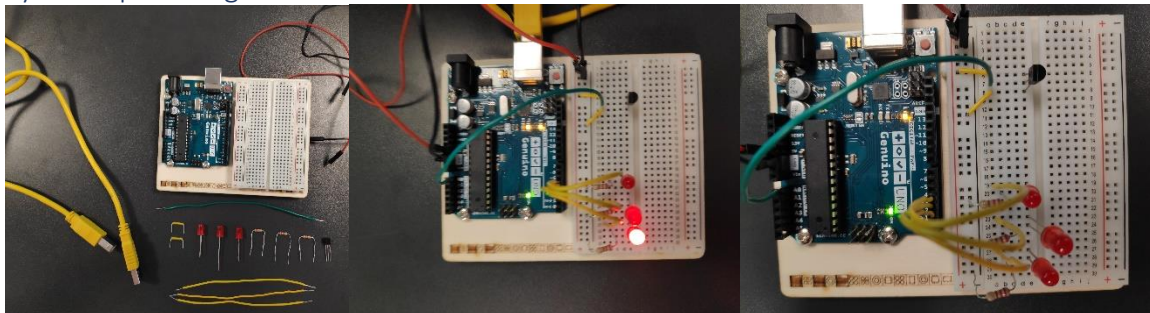
```

else { //If the button IS pressed.
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);
    delay(250); // Waits one fourth of a second.
    digitalWrite(4, HIGH);
    digitalWrite(5, LOW);
    delay(250);
}
}

```

Projekt 3 – Love-O-Meter

Fysiske opsætningskrav



Tredje projekt er opbygget af en række dioder, der hver især skal illustrere omgivelsestemperaturen. Ved under 20c grader, vil ingen dioder være tændt. Den første, anden og tredje diode vil tænde ved henholdsvis 20, 24- og 26 grader. For at dette kan lade sig gøre, skal vi benytte os af den analoge voltage temperaturmåleren outputter. Denne konverteres af Arduinoens analog-to-digital (ADC) converter. Denne strømstyrke kan omregnes til en temperatur i grader som vi derved kan udskrive til vores brug, som i dette tilfælde er et termometer.

Kildekode:

```
const int sensorPin = A0; //Creates constants.

const float baselineTemp = 20.0;

void setup() {
  Serial.begin(9600); // Opens Serial port connection.
  for(int pinNumber = 2; pinNumber<5; pinNumber++){
    pinMode(pinNumber,OUTPUT);
    digitalWrite(pinNumber,LOW);
  }
}

void loop() {
  int sensorVal = analogRead(sensorPin);
  Serial.print("Sensor Value: ");
  Serial.print(sensorVal);
  float voltage = (sensorVal/1024.0) * 5.0;
  Serial.print(", Volts: ");
  Serial.print(voltage);
  Serial.print(", degrees C: ");
  // We convert voltage to temperature in degrees.
  float temperature = (voltage - .5) * 100;
  Serial.println(temperature);
  if(temperature < baselineTemp){ //The diodes will turn on and off in accordance with the
    temperature readings.
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  else if(temperature >= baselineTemp &&
    temperature < baselineTemp+4){
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
}
```



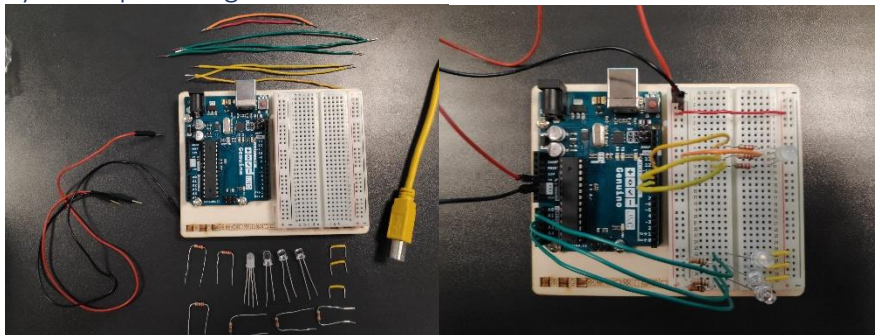
```

    else if(temperature >= baselineTemp+4 &&
temperature < baselineTemp+6){
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
}
    else if(temperature >= baselineTemp+6){
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
}
    delay(100);
}

```

Projekt 4 – Colour Mixing Lamps

Fysiske opsætningskrav

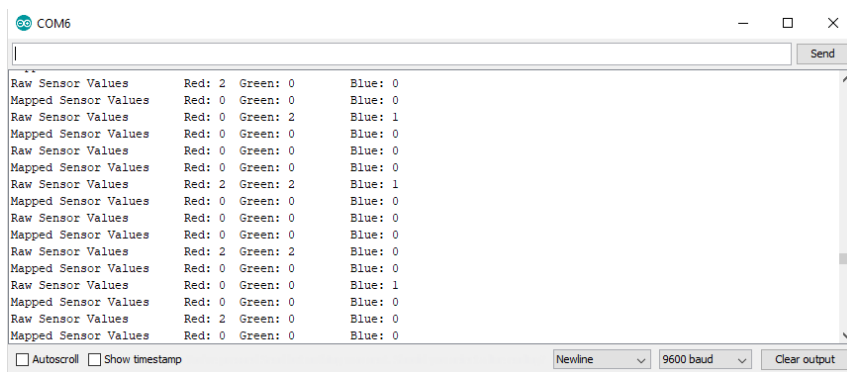


OBS. Første illustration viser en transparent diode i stedet for en tredje temperatursensor. Dette er rettet op på i den fysiske opsætning.

I fjerde projekt skabes en lyssensor der er i stand til at måle rummet *ambient lighting*. Denne belysning kan måles på ved hjælp af phototransistore. Vha. disse transistores enkelte output kan vi gennem den 4-benede RGB led dynamisk ændre lys-outputtet. For at kunne benytte *analogWrite* til at give et output dividere vi sensorerne rå værdier med 4, for at få disse værdier ned fra mellem 0-1023 til en værdi mellem 0-255.

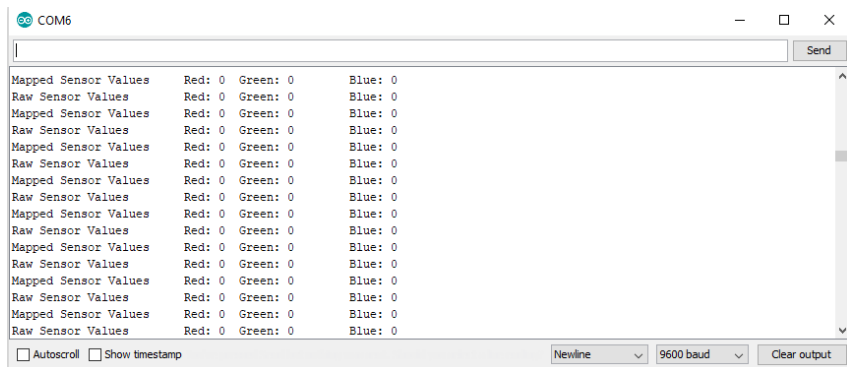
Nedenfor ses serialoutputtet fra 3 vidt forskellige læsninger.

Med baggrundsbelysning:



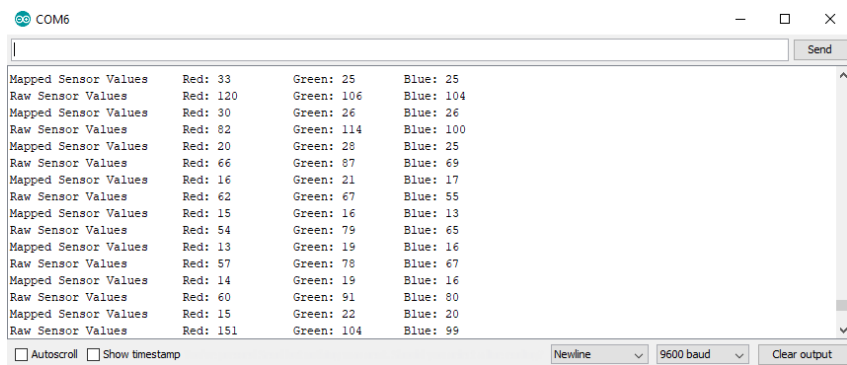
```
COM6
Raw Sensor Values Red: 2 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 2 Blue: 1
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 2 Green: 2 Blue: 1
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 2 Green: 2 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 1
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 2 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
```

Helt uden lys:



```
COM6
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
Mapped Sensor Values Red: 0 Green: 0 Blue: 0
Raw Sensor Values Red: 0 Green: 0 Blue: 0
```

Med kraftigt lys fra lommelygte:



```
COM6
Mapped Sensor Values Red: 33 Green: 25 Blue: 25
Raw Sensor Values Red: 120 Green: 106 Blue: 104
Mapped Sensor Values Red: 30 Green: 26 Blue: 26
Raw Sensor Values Red: 82 Green: 114 Blue: 100
Mapped Sensor Values Red: 20 Green: 28 Blue: 25
Raw Sensor Values Red: 66 Green: 87 Blue: 69
Mapped Sensor Values Red: 16 Green: 21 Blue: 17
Raw Sensor Values Red: 62 Green: 67 Blue: 55
Mapped Sensor Values Red: 15 Green: 16 Blue: 13
Raw Sensor Values Red: 54 Green: 79 Blue: 65
Mapped Sensor Values Red: 13 Green: 19 Blue: 16
Raw Sensor Values Red: 57 Green: 78 Blue: 67
Mapped Sensor Values Red: 14 Green: 19 Blue: 16
Raw Sensor Values Red: 60 Green: 91 Blue: 80
Mapped Sensor Values Red: 15 Green: 22 Blue: 20
Raw Sensor Values Red: 151 Green: 104 Blue: 99
```

Kildekode:

//These constants are used to track the colourpairs.

```
const int greenLEDPin = 9;
```

```
const int greenSensorPin = A1;
```

```
const int redLEDPin = 11;
```

```
const int redSensorPin = A0;
```

```
const int blueLEDPin = 10;
```

```
const int blueSensorPin = A2;
```

//We'll use these integers to save and use the data readings from the sensors.

```

int redValue = 0;
int greenValue = 0;
int blueValue = 0;
int redSensorValue = 0;
int greenSensorValue = 0;
int blueSensorValue = 0;

void setup() {
  Serial.begin(9600); //Opens the serial connection
  pinMode(greenLEDPin,OUTPUT);
  pinMode(redLEDPin,OUTPUT);
  pinMode(blueLEDPin,OUTPUT);
}

void loop() {
  //We continuously read from the pins and save these values to e.g 'redSensorValue'
  redSensorValue = analogRead(redSensorPin);
  delay(5);
  greenSensorValue = analogRead(greenSensorPin);
  delay(5);
  blueSensorValue = analogRead(blueSensorPin);

  //We print the raw values of the readings.
  Serial.print("Raw Sensor Values \t Red: ");
  Serial.print(redSensorValue);
  Serial.print("\t Green: ");
  Serial.print(greenSensorValue);
  Serial.print("\t Blue: ");
  Serial.print(blueSensorValue);
  redValue = redSensorValue/4;
  greenValue = greenSensorValue/4;
  blueValue = blueSensorValue/4;

  //We print the mapped values. These values are one fourth of the raw value in order to use
  'analogWrite'.
  Serial.print("\nMapped Sensor Values \t Red: ");

```

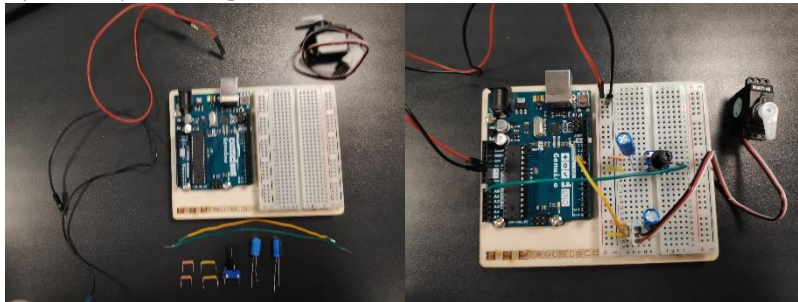
```

Serial.print(redValue);
Serial.print("\t Green: ");
Serial.print(greenValue);
Serial.print("\t Blue: ");
Serial.print(blueValue );
Serial.print("\n");
analogWrite(redLEDPin, redValue);
analogWrite(greenLEDPin, greenValue);
analogWrite(blueLEDPin, blueValue);
}

```

Projekt 5 – Mood cue

Fysiske opsætningskrav



Femte projekt benytter et potentiometer samt en servomotor til at ”angive dit humør”. Ved at dreje på potentiometeret kan vi styre strømstyrken der føres igennem denne. Dette output kan vi bruge til at styre servomotorens arms bevægelser.

Kildekode:

```

#include <Servo.h> //Imports the Servo library

Servo myServo; //Named instance of the Servo library

//Create integers for later use.

int const potPin = A0;
int potVal;
int angle;

void setup() {
    myServo.attach(9); //Tells the Arduino which pin the servo is attached to.
    Serial.begin(9600); //Opens the serial port
}

void loop() {
    potVal = analogRead(potPin);

```

```

Serial.print("potVal: ");
Serial.print(potVal);

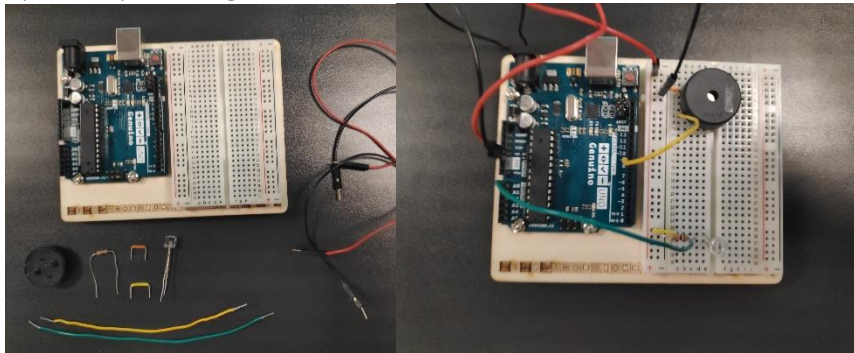
angle = map(potVal, 0, 1023, 0, 179); // 'Map' changes the numerical values from 0-1023 to 0-179.

Serial.print(", angle: ");
Serial.println(angle);
myServo.write(angle);
delay(15);
}

```

Projekt 6 – Light Theremin

Fysiske opsætningskrav



For at generere lyd gennem vores *Piezo* benytter vi en *photoresistor*. Denne måler lys omkring den, som vi bruger til at generere en frekvens mellem 50 og 5000. Denne frekvens vil derefter blive afspillet gennem *Piezoen*.

Kildekode:

```

//Creates integers for later use along with 'sensorValue' to hold the analogRead value.
int sensorValue;
int sensorLow = 1023;
int sensorHigh = 0;

const int ledPin = 13; //This is used as an indicator to show that the initial calibration is complete.

void setup() {
  //We change the pinMode of ledPin to OUTPUT and turn the led on.
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);

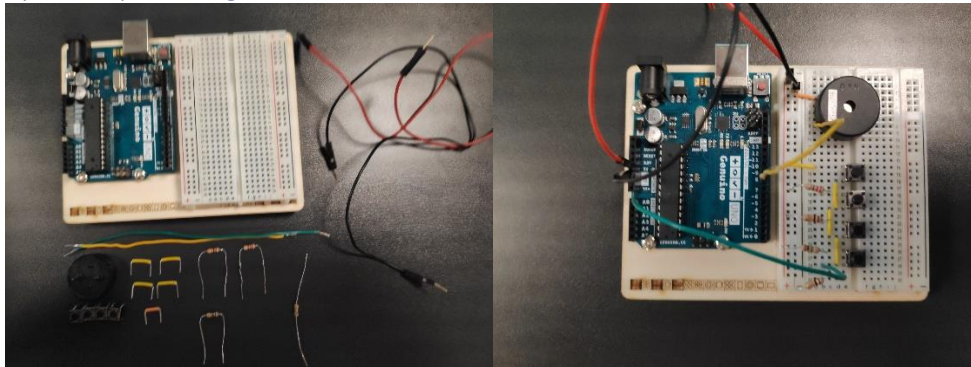
  // 'millis' reports how long the Arduino has been running since the last reset.
  while (millis() < 5000) {
    sensorValue = analogRead(A0);

```

```
    if (sensorValue > sensorHigh) {  
        sensorHigh = sensorValue;  
    }  
    if (sensorValue < sensorLow) {  
        sensorLow = sensorValue;  
    }  
}  
digitalWrite(ledPin, LOW);  
}  
void loop() {  
    sensorValue = analogRead(A0); //Reads the value in A0 and saves it to sensorValue.  
    //The pitch is mapped to sensorValue. sensorLow and sensorHigh are the bounds of the last  
    input here(50 and 5000)  
    //This is going to be the range of frequency the Arduino will generate.  
    int pitch =  
        map(sensorValue,sensorLow,sensorHigh, 50, 4000);  
    //tone will now play a sound piezo in accordance with the pitchvalue generated earlier.  
    tone(8,pitch,20);  
    delay(10);  
}
```

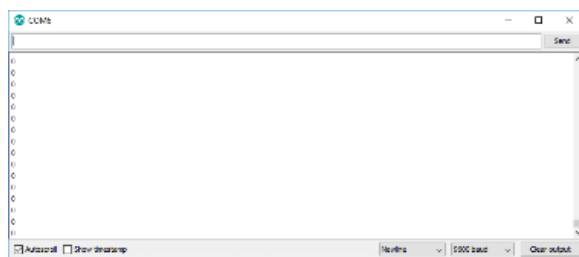
Projekt 7 – Keyboard Instrument

Fysiske opsætningskrav

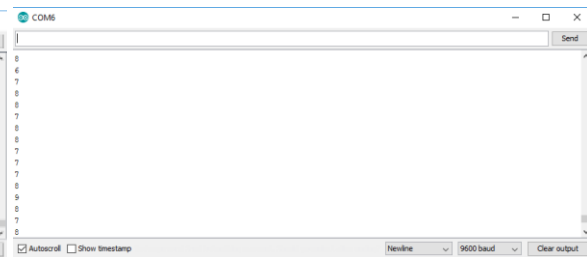


Serial outputs

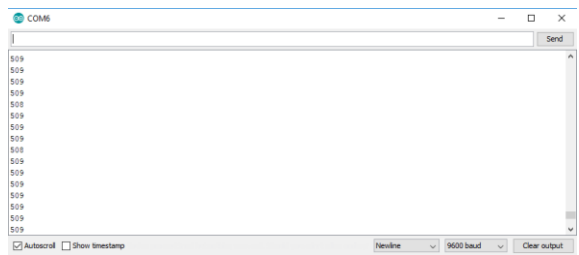
Knap 1



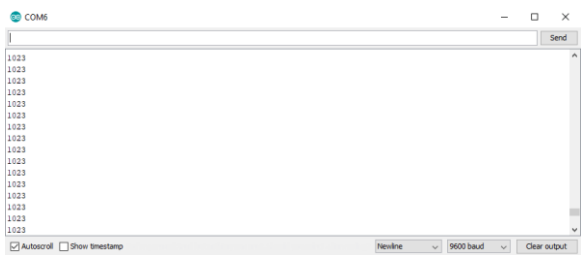
Knap 2



Knap 3



Knap 4



Dette projekt omhandler kreationen af et keyboard. Dette mål nås ved hjælp af 4 switches, vores ”tangenter” samt en *Piezo* der afspiller en lyd i en valgt frekvens.

For at undgå at en tone fortsætter efter den har været holdt nede, er der inkluderet en *else* med ingen tone. Der vil altså efter frigivelse af en tangent blive slukket for *piezoen*. *Piezoen* afspiller en bestemt frekvens med afsæt i de værdier vi har givet i selve main loopet.

Kildekode:

```
//The instructional text suggested these two lines to create an array of 6 elements and assign one of them the value of //2. I've found this to be redundant as the device functions without it.
```

```
//int buttons[6]; //We create an array of 6 integers.
```

```
//int buttons[0] = 2; //Gives the first element of the array the value 2.
```

```
int notes[] = {262,294,339,349}; //Set up an array of notes using four different frequencies.
```

```
void setup() {
```

```
    // put your setup code here, to run once:
```

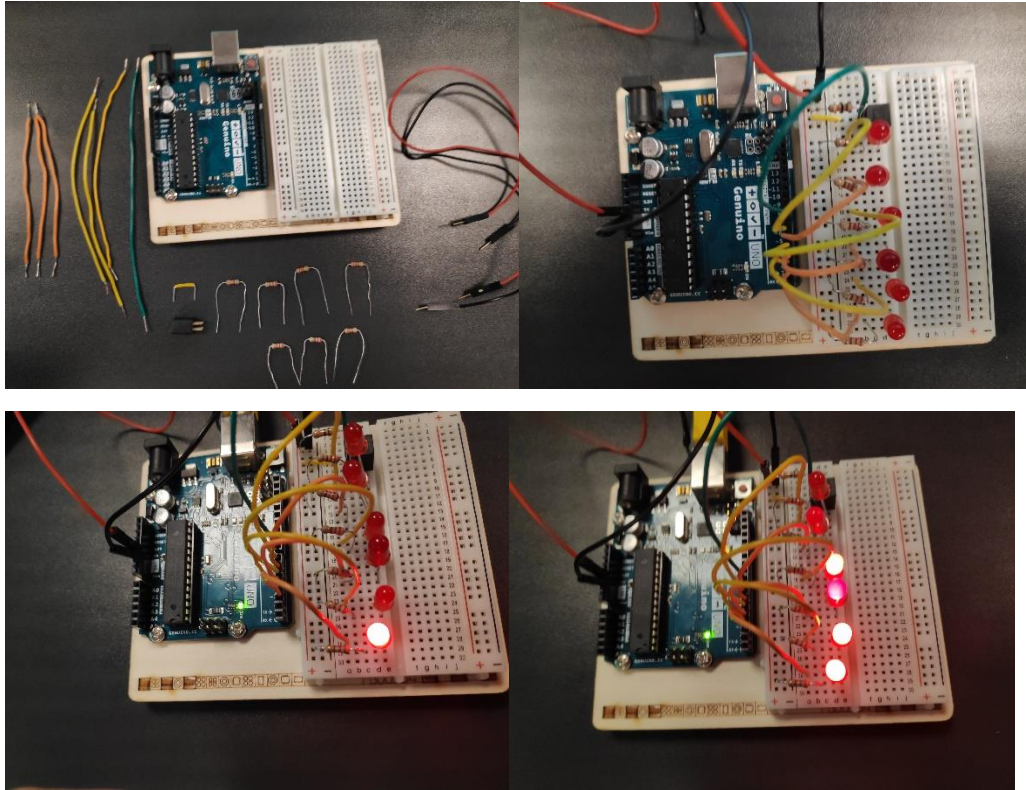
```
    Serial.begin(9600);
```



```
}  
void loop() {  
    // put your main code here, to run repeatedly:  
    int keyVal = analogRead(A0); //Declares the variable to hold the reading from A0.  
    Serial.println(keyVal);  
    if (keyVal == 1023){  
        tone(8, notes[0]);  
    }  
    else if(keyVal >= 990 && keyVal <= 1010){  
        tone(8, notes[1]);  
    }  
    else if(keyVal >= 505 && keyVal <= 515){  
        tone(8, notes[2]);  
    }  
    else if(keyVal >= 5 && keyVal <= 10){  
        tone(8, notes[3]);  
    }  
    else{  
        noTone(8); //Fills the void with no tone.  
    }  
}
```

Projekt 8 – Digital Hourglass

Fysiske opsætningskrav



I dette projekt bygges et timeglas, med 6 dioder hvoraf én af dem skal tændes hvert tiende minut. For at forskynde test-processen har jeg valgt at forkorte dette til 6 sekunder. *Tilt-switchen* vi bruger vil ved bevægelse nulstille loopet og tælle forfra. Derved vil alle dioder blive slukket.

Kildekode:

//OBS: THE 6 RED LED ARE MISSING FROM THE PICTURE!

```
const int switchPin = 8; //Declares which pin the tilt switch is connected to.
```

```
unsigned long previousTime = 0; // Will hold the time the LED was last changed.
```

```
int switchState = 0;
```

```
int prevSwitchState = 0;
```

```
int led = 2;
```

```
long interval = 6000;
```

```
void setup() {
```

```
    // put your setup code here, to run once:
```

```
    for (int x = 2; x<8;x++){
```

```
        pinMode(x, OUTPUT);
```

```
    }
```

```
    pinMode(switchPin, INPUT);
```

```

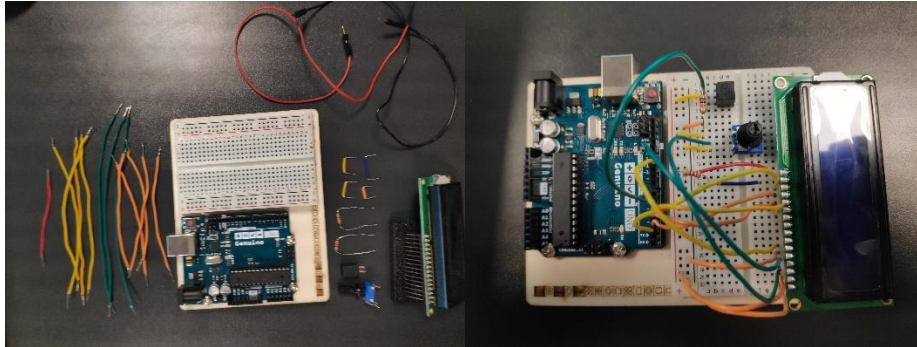
}

void loop() {
    // put your main code here, to run repeatedly:
    unsigned long currentTime = millis();
    if(currentTime - previousTime > interval) {
        previousTime = currentTime;
        digitalWrite(led, HIGH);
        led++;
        if(led == 7){
            }
        }
    switchState = digitalRead(switchPin);
    if(switchState != prevSwitchState){
        for(int x = 2; x<8;x++){
            digitalWrite(x, LOW);
        }
        led = 2;
        previousTime = currentTime;
    }
    prevSwitchState = switchState;
}

```

Projekt 11 – Crystal Ball

Fysiske opsætningskrav



Obs. Grundet ukendte fejl har jeg ikke kunne fremkalde de ønskede beskeder på LCD'et. Dette skal derfor ses som et intentionelt eksempel

I projekt elleve laves der en ”krystalkugle” som du skal kunde stille et spørgsmål og derefter ryste for at få et svar. Potentiometeret på Arduinoen vil måle bevægelser og derefter vil der blive genereret et tal mellem 1- og 8. Denne int vil blive brugt i en switchcase til at bestemme udfaldet fra denne generering.

Kildekode:

```
#include <LiquidCrystal.h>
```

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

```
const int switchPin= 6;
```

```
int switchState = 0;
```

```
int prevSwitchState = 0;
```

```
int reply;
```

```
void setup() {
```

```
    lcd.begin(16, 2);
```

```
    pinMode(switchPin,INPUT);
```

```
    lcd.print("Ask the");
```

```
    lcd.setCursor(0, 1);
```

```
    lcd.print("Crystal Ball!");
```

```
}
```

```
void loop() {
```

```
    switchState = digitalRead(switchPin);
```

```
    if(switchState != prevSwitchState) {
```

```
//Chooses a random number used to determine the outcome.
```

```
    if(switchState == LOW){
```

```
        reply = random(8);
```

```
lcd.clear();  
lcd.setCursor(0, 0);  
lcd.print("The ball says:");  
lcd.setCursor(0, 1);
```

//Switchcase for the possible outputs. Outcome based on random number generated previously.

```
switch(reply){  
  case 0:  
    lcd.print("Yes");  
    break;  
  case 1:  
    lcd.print("Most likely");  
    break;  
  case 2:  
    lcd.print("Certainly");  
    break;  
  case 3:  
    lcd.print("Outlook good");  
    break;  
  case 4:  
    lcd.print("Unsure");  
    break;  
  case 5:  
    lcd.print("Ask again");  
    break;  
  case 6:  
    lcd.print("Doubtful");  
    break;  
  case 7:  
    lcd.print("No");  
    break;  
}
```

```

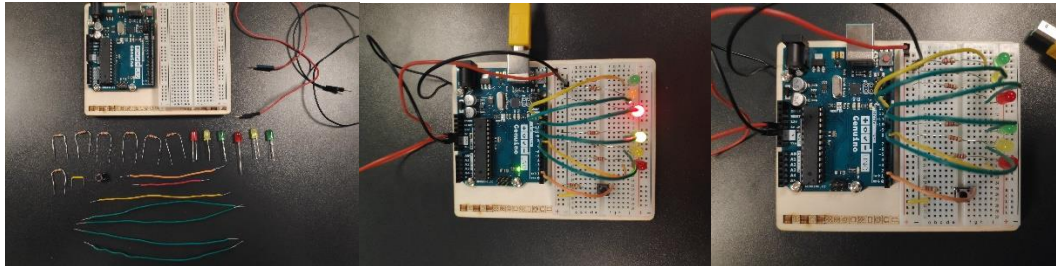
    }
}

prevSwitchState = switchState;
}

```

Øvelse 1. – Trafiklys

Fysiske opsætningskrav



I første øvelse har jeg lavet et trafiklyssystem der med to tværgående lyskasser skal kunne illustrere et lyskryds. I main loopet skifter lyssignalerne automatisk mellem grønne, gule og røde signaler. Dette loop kan breakes ved hjælp af et *attachInterrupt*. Dette kalder på metoden *changeLight()* som er en modificeret cyclus af loopet. Denne vil efter et delay på 2 sekunder begynde et skift mellem lyssignalerne. Når denne metode har kørt vil der returneres til main loopet.

Kildekode:

```
// x direction traffic light.
```

```
int L1Red = 8;
```

```
int L1Yellow = 9;
```

```
int L1Green = 10;
```

```
// y direction traffic light.
```

```
int L2Red = 11;
```

```
int L2Yellow = 12;
```

```
int L2Green = 13;
```

```
unsigned long cTime = 0;
```

```
unsigned long wait = 0;
```

```
void setup() {
```

```
  for (int i = 8; i <= 13; i++) // For-loop assigns the pins as outputs
```

```
  {
```

```
    pinMode(i, OUTPUT);
```

```
  }
```

```
  pinMode(2, INPUT); // We assign pin 2 as the input. This is the button.
```

```

attachInterrupt(0,changeLight,RISING); //This calls changeLight()
digitalWrite(L1Green, HIGH);
digitalWrite(L2Red, HIGH);
}

void loop() // This is the main loop of the traffic light.
{
    digitalWrite(L1Green, LOW);
    digitalWrite(L1Yellow, HIGH);
    delay(10000);
    digitalWrite(L1Yellow,LOW);
    digitalWrite(L1Red,HIGH);
    delay(1500);
    digitalWrite(L2Red,LOW);
    digitalWrite(L2Green, HIGH);
    delay(20000);
    digitalWrite(L2Green, LOW);
    digitalWrite(L2Yellow, HIGH);
    delay(10000);
    digitalWrite(L2Yellow,LOW);
    digitalWrite(L2Red,HIGH);
    delay(1000);
    digitalWrite(L1Red, LOW);
    digitalWrite(L1Green, HIGH);
    delay(20000);
}

void changeLight() // This is the 'button-push' scenario. Following a two second delay the
light will begin to switch. This behavior ensures that the signaling of the traffic light does not
cause instantaneous crashes.
{
    if (digitalRead(L2Green == LOW)) // The 'if' is to check if the green light is already on.
    {
        delay(2000);
        digitalWrite(L1Green, LOW);
    }
}

```



```

digitalWrite(L1Yellow, HIGH);
delay(10000);
digitalWrite(L1Yellow,LOW);
digitalWrite(L1Red,HIGH);
delay(1500);
digitalWrite(L2Red,LOW);
digitalWrite(L2Green, HIGH);
delay(20000);
digitalWrite(L2Green, LOW);
digitalWrite(L2Yellow, HIGH);
delay(10000);
digitalWrite(L2Yellow,LOW);
digitalWrite(L2Red,HIGH);
delay(1000);
digitalWrite(L1Red, LOW);
digitalWrite(L1Green, HIGH);
delay(2000);
}
}

```

Øvelse 2. - Robot (Del 1)

Robotteknologi kan- og bruges allerede i høj grad til at automatisere arbejdsprocessor. Dette medvirker blandt andet at arbejdspladser for nogle mennesker er sikrere, da farlige arbejdsopgaver nu kan udføres af maskiner. Da dette i mange tilfælde er langt mere effektivt har dette også den bivirkning at det tager jobs fra mennesker. Dette ses blandt andet hos en lang række webshops hvor en stor del af nedpakningen og afsendingen nu kan foretages af automatiserede robotter. Der bruges ligeledes et hav af robotter indenfor fødevareindustrien. Disse bruges bl.a. til at sortere fødevarer. I det private ses robotter også i øget grad, da automatiske græsslåmaskiner og støvsugere bliver lettere tilgængelig.

Jeg skynder at robotteknologi for mange programmører er- og vil blive i højere grad aktuelt. Effektive sorteringssystemer samt procesautomatisering er profitabelt i høj grad da lønninger i høj grad kan spares væk. Dette er på trods af den høje *entry price*.

Konklusion

Arbejdet med Arduino virkede i starten som en mere *labourintensive* arbejdsproces da denne tog udgangspunkt i mere elektrikerrettet viden blev præsenteret. Arbejdet vendte sig hurtigt til at foregå i Arduinos IDE hvori koden skulle skrives. Af læringsmateriale skal bogen *Arduino project book* have ros da denne gennemgående nåede et rigtig godt trin mellem kode samt beskrivelse af denne.